

# Hausaufgabe 5: Form validation

Laden Sie ihre Lösung für die Hausaufgabe in einer ZIP-Datei auf ISIS bis zum 05. Februar um 23:59 Uhr hoch. Die ZIP-Datei soll dabei ausschließlich den Ordner `ha05` mit dessen Inhalt wie in der Vorgabe gegeben enthalten (Der Inhalt der Dateien muss natürlich nach Aufgabenstellung bearbeitet werden). Keine weiteren Dateien sollen hinzugefügt oder entfernt werden. Auch sollen keine Dateien oder Ordner umbenannt werden.

## Vorbereitung

The screenshot shows a web form titled "Complaint form" with the subtitle "Flight got canceled? Luggage lost? Get a refund!". The form has a yellow header bar with the text "Hausaufgabe 5" and a "Run tests" button. The form fields and their validation status are as follows:

- Flight number:** Input field with "LH0011" and a "123" character count. Below the field, it says "Flight number correct." in green.
- Email address:** Input field with "email@example.com". Below the field, it says "Please enter a valid TUB email address." in red.
- Date of flight:** Date picker showing "01.01.2022". Below the field, it says "Date of flight correct." in green.
- Time of flight:** Time picker showing "00:10". Below the field, it says "Time of flight correct." in green.
- IBAN:** Input field with "DE11 1111 1111 1111 1111 11". Below the field, it says "Please enter a valid IBAN." in red.

At the bottom right of the form is a blue button labeled "Submit complaint".

Abbildung 1: Screenshot der Vorgabe nach dem ersten Start. **Beachten Sie:** Die meisten Komponenten wurden als Bilder eingefügt, um Ihnen eine Vorstellung davon zu geben wie die Seite am Ende aussehen soll bzw. kann.

Zur Bearbeitung müssen Sie Node.js installieren. Gehen Sie dazu auf <https://nodejs.org/>, laden Sie die entsprechende Version<sup>1</sup> für Ihr Betriebssystem herunter und installieren Sie im Anschluss Node.js auf Ihrem Rechner. Danach können Sie den *Node Package Manager* (kurz: *npm*<sup>2</sup>) nutzen. Diesen benötigen wir um die Vorgabe starten zu können.

Installieren Sie zunächst die Vue.js Command-line Tools *vue-cli* mit folgendem Befehl:

```
npm install -g @vue/cli
```

Extrahieren Sie dann den Ordner `ha05/` aus der Vorgabe an einem beliebigen Ort auf Ihrem Rechner. Gehen Sie dann im Terminal in das Verzeichnis `ha05/` und führen Sie dort folgenden Befehl aus.

```
npm install
```

<sup>1</sup>idealerweise LTS-Version v18.10.0

<sup>2</sup>idealerweise v.8.1.2

Im Anschluss werden weitere Pakete installiert, die für die Ausführung der Vorgabe benötigt werden. Ist der Installationsvorgang abgeschlossen können Sie mit folgendem Befehl die Vorgabe starten.

```
npm run serve
```

Gehen Sie im Browser auf `http://localhost:8080/`. Sie sollten nun die oben abgebildete Seite (siehe Abbildung 1) sehen.

## Grober Aufbau

In dieser Hausaufgabe sollen Sie ein Online-Formular erstellen, das Nutzerinnen und Nutzern erlaubt Beschwerden an eine Fluggesellschaft zu übermitteln. Dazu müssen folgenden Informationen angegeben werden: **Flugnummer**, **E-Mail Adresse**, **Datum und Uhrzeit** des Flugs und eine **Bankverbindung (IBAN)** für eventuelle Rückerstattungen. Um sicherzustellen, dass die Nutzereingaben korrekt sind, überprüfen Sie diese vor dem Absenden auf Fehler. Machen Sie sich dazu mit *Regular Expressions*<sup>3</sup> und *Form validation in Bootstrap*<sup>4</sup> vertraut. Schauen Sie sich ebenfalls das in der Vorlesung gezeigte Beispiel nochmal an.<sup>5</sup>

Implementieren Sie Ihre Lösung hauptsächlich in der Komponente `ComplaintForm.vue`. Die Hauptkomponente `App.vue` bedarf keinerlei Änderungen. Es steht Ihnen jedoch frei weitere Subkomponenten zu implementieren.

### 5.1 Input-Felder & Utility-Klassen

Implementieren Sie zunächst ein Formular bestehend aus den 5 benötigten Input-Feldern. Vergeben Sie die folgenden IDs entsprechend, damit die automatisierten Tests erfolgreich durchlaufen können: `#flight-number`, `#email-address`, `#flight-date`, `#flight-time` und `#bank-iban`. In der Komponente `ComplaintForm.vue` sind die Datenfelder `flightNumber`, `emailAddress`, `flightDate`, `flightTime` und `bankIban` bereits vorgegeben. Sie müssen diese Felder als v-model für die entsprechenden Input-Felder nutzen, damit die automatisierten Tests durchlaufen können.

Wenn das Datenfeld `formSubmitted` auf `true` gesetzt wird, sollen alle Input-Felder auf ihre Korrektheit überprüft werden. Felder deren Eingaben valide sind erhalten die Klasse `is-valid` und die Klasse `is-invalid` im entgegengesetzten Fall.

**Tipp:** Sie können diese Aufgabe lösen und die automatisierten Tests dazu bestehen ohne bereits die Validierungsfunktionen implementiert zu haben. Nehmen Sie hierzu einfach eine leere Eingabe für den invaliden Fall an.

<sup>3</sup>siehe: [https://developer.mozilla.org/de/docs/Web/JavaScript/Guide/Regular\\_Expressions](https://developer.mozilla.org/de/docs/Web/JavaScript/Guide/Regular_Expressions)

<sup>4</sup>siehe: <https://getbootstrap.com/docs/5.0/forms/validation/>

<sup>5</sup>siehe: <https://git.tu-berlin.de/snet-edu/webtech2022-vorlesung/-/tree/main/chapter05/example11>

## 5.2 Validierung: Flugnummer

Betrachten Sie nun in `ComplaintForm.vue` die Methode `flightNumberValid()`, die einen `String` übergeben bekommt.

Eine Flugnummer besteht aus einer Kennung der Fluggesellschaft (z.B. LH für Lufthansa) und einer Zahl mit bis zu vier Stellen (z.B. 1934). Die Kennung, der sogenannte IATA-Airline-Code, besteht aus exakt zwei Zeichen (Großbuchstaben oder Ziffern). **Beachten Sie:** Ein IATA-Airline-Code kann nicht aus zwei Ziffern bestehen.

Gefolgt wird die Kennung der Fluggesellschaft von einer Zahl mit bis zu vier Stellen. Führende Nullen können weggelassen oder angegeben werden. Außerdem kann die Flugnummer mit einem Leerzeichen von der Kennung der Fluggesellschaft getrennt werden.

Die folgenden Fälle sind valide Flugnummern:

`'LH0001'`, `'LH 0001'`, `'LH001'`, `'LH01'`, `'LH1'`, `'3X1'`, `'X31'`.

Die folgenden Fälle sind **keine** gültigen Flugnummern:

`'LHA 0001'`, `'LH 00001'`, `'111'`, `'LH 1A'`, `'1h1'`.

## 5.3 Validierung: E-Mail Adresse

Betrachten Sie nun in `ComplaintForm.vue` die Methode `emailAddressValid()`, die ebenfalls einen `String` übergeben bekommt.

Diese soll überprüfen, ob eine valide **TU-Berlin E-Mail Adresse** angegeben wurde. Die Validierung einer E-Mail Adresse ist relativ komplex und in der Vorlesung haben Sie gelernt, dass man weitestgehend auf bewährte Lösungen zurückgreifen sollte. Daher wollen wir weitere Einschränkungen zur Vereinfachung definieren. Der Teil vor dem `[@]` darf nur aus Groß-, Kleinbuchstaben und Ziffern bestehen, die optional mit einem Punkt `[.]` getrennt werden können. **Hinweis:** Sonderzeichen wie `[ß, ö, ü]` sind **nicht** erlaubt. Gültige Hostnamen (der Teil nach dem `[@]`) sind `mailbox.tu-berlin.de`, `campus.tu-berlin.de` und natürlich `tu-berlin.de`.

Die folgenden Fälle sind valide TUB-Email Adressen:

`'john@tu-berlin.de'`, `'john.doe@tu-berlin.de'`, `'j.o.h.n@tu-berlin.de'`, `'jo.doe.@tu-berlin.de'`, `'john@campus.tu-berlin.de'`, `'john@mailbox.tu-berlin.de'`.

Die folgenden Fälle sind **keine** gültigen TUB-Email Adressen:

`'...@tu-berlin.de'`, `'john@example.com'`, `'john@mailbox.example.com'`, `'john doe@tu-berlin.de'`.

## 5.4 Validierung: Datum und Uhrzeit

Betrachten Sie nun in `ComplaintForm.vue` die Methoden `dateValid()` und `timeValid()`, die jeweils einen `String` übergeben bekommen.

Ein valides Datum wird in der Regel im Format `JJJJ-MM-TT` angegeben (z.B. `0001-01-01`). Ihre Funktion soll jedoch das Weglassen von führenden Nullen erlauben (also: `1-1-1`). Darüber hinaus endet die Zeitrechnung (zumindest vorerst) mit dem Jahr 9999.

Die folgenden Fälle sind valide Datumsangaben:

`'2022-01-01'`, `'2022-1-1'`, `'222-1-1'`, `'22-1-1'`, `'2-1-1'`, `'0002-01-01'`.

Die folgenden Fälle sind **keine** gültigen Datumsangaben:

`'2022-01-01'`, `'2022-13-01'`, `'2022-01-32'`.

Eine valide Uhrzeit wird in der Regel im Format `HH:MM` angegeben (z.B. `01:00`). Für die Angabe der Stunden soll Ihre Funktion wieder das Weglassen der führenden Null ermöglichen (also: `1:00`). Das soll aber nicht für die Angabe der Minuten gelten (also nicht: `1:1`).

Die folgenden Fälle sind valide Uhrzeitangaben:

`'00:00'`, `'00:10'`, `'10:00'`, `'1:00'`, `'23:00'`.

Die folgenden Fälle sind **keine** gültigen Uhrzeitangaben:

`'24:10'`, `'23:60'`, `'10:1'`.

## 5.5 Validierung: IBAN

Betrachten Sie zuletzt in `ComplaintForm.vue` die Methode `ibanValid()`, die einen `String` übergeben bekommt.

Diese soll überprüfen, ob eine valide IBAN (kurz für: *International Bank Account Number*) eingegeben wurde. In der Realität folgt diese einem relativ komplexen Format, das von Land zu Land ein wenig variiert. In Deutschland besteht die IBAN aus 22 Zeichen. Sie beginnt mit dem Ländercode `DE` gefolgt von einer zwei-stelligen Prüfsumme (eine zweistellige Zahl). Die restlichen 18 Zeichen sind ausschließlich Ziffern. Ein Leerzeichen zwischen Ländercode und Prüfsumme ist **nicht** zulässig, jedoch ein optionales Leerzeichen zwischen Prüfsumme und den restlichen 18 Zeichen, sowie ein optionales Leerzeichen nach jeweils 4 Ziffern.

Darüber hinaus sollen Sie anhand der Prüfsumme die IBAN validieren. Dazu müssen die 18 Ziffern aufsummiert und der Modulo von 100 bestimmt werden. Ist dieser identisch mit der Prüfsumme ist die IBAN gültig. Die Methode `ibanValid()` gibt also `true` zurück, wenn die IBAN im korrekten Format angegeben wurde **und** die IBAN anhand der Prüfsumme auf Gültigkeit überprüft wurde. Sonst gibt diese `false` zurück. **Hinweis:** Das tatsächliche Verfahren zur Überprüfung der Gültigkeit von IBAN ist ähnlich, aber technisch etwas aufwendiger.

Die folgenden Fälle sind valide IBAN:

`'DE18111111111111111111'`

`'DE18 1111 1111 1111 1111 11'`

`'DE18 11111111 11111111 11'`

`'DE18 1111111111111111 11'`

Die folgenden Fälle sind **keine** gültigen IBAN:

`'DE11 1111 1111 1111 1111 11'`

`'BE18 1111 1111 1111 1111 11'`

`'DE11 1111 1111 1111 1111 1111'`

`'DE11 1111 1111'`