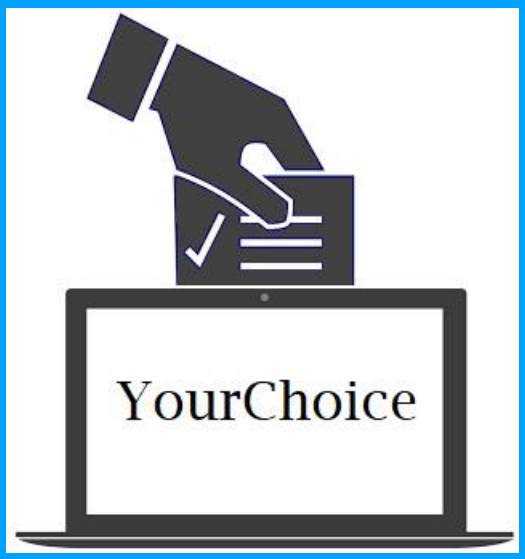
FEINDESIGN



Josua Weber

YOURCHOICE

Contents

1.0 Frontend-Modulbeschreibung

Der Einsatz von React erlaubt es die Frontend-Anwendung als System von Komponenten zu entwerfen. Die Webseite wird nicht als Sammlung von einzelnen Seiten erstellt, sondern mit Hilfe von Komponenten zusammengesetzt. Diese Bausteine der Benutzeroberfläche werden in separaten Gruppen organisiert. Der Hintergrundgedanke ist eine klare Trennung einzelner Zuständigkeiten.

Des Weiteren werden funktionale Bestandteile der Benutzeroberfläche (UI) ebenfalls in Module untergebracht. Somit können z.B. Dienste oder auch Hilfsfunktionen klar von der restlichen Anwendung getrennt werden. Dies erlaubt einen wiederverwendbaren Einsatz der Geschäftslogik ohne sie direkt in der Benutzeroberfläche integrieren zu müssen.

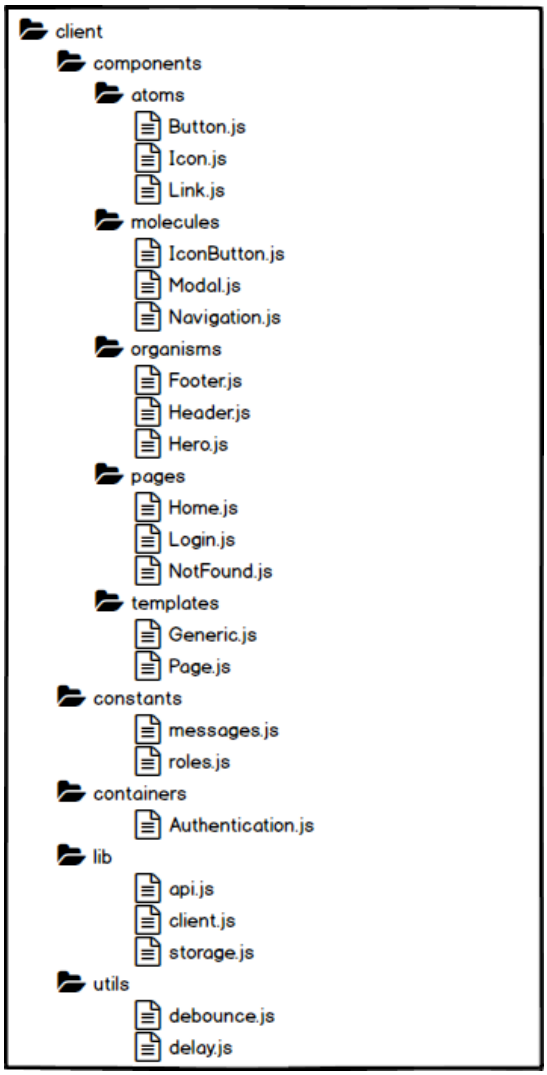


Abbildung 1: Exemplarische Datei-/Verzeichnisstruktur

1.1 “components”

React ist eine Bibliothek zur Erstellung von Benutzeroberflächen. Eine klare Struktur bzw. ein Rahmen wird dabei nicht vorgegeben. Komponenten können beliebig aufgeteilt und

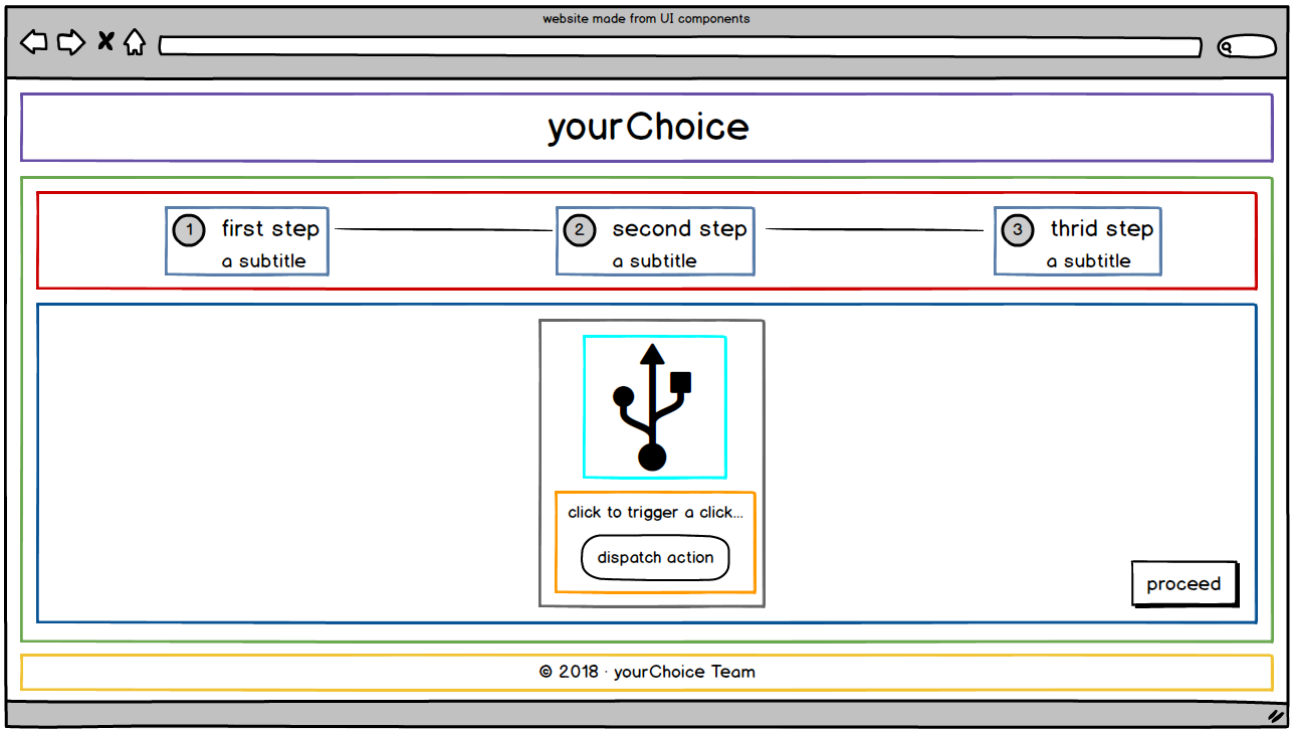
wiederverwendet werden. Die Modularisierung gibt dabei vor wie sich schlussendlich die gesamte Anwendung aus atomaren Elementen zusammensetzt.

Abbildung 2: Beispiel einer komponentenbasierten Benutzeroberfläche

1.1.1 Übersicht

Bei den hier aufgeführten Komponenten-Typen handelt es sich hauptsächlich um funktionslose Bestandteile der Benutzeroberfläche. Sprich diese sind zustandslos und somit rein zur Darstellung von Inhalten gedacht. Das Verhalten bzw. die eingehenden Benutzerinteraktionen werden von Containern separat bereitgestellt. Somit können Daten von ihrer Darstellung separiert werden.

Die nachfolgenden Bestandteile bzw. Module der Benutzeroberfläche werden in folgende Gruppen eingeteilt, wobei von oben nach unten gesehen sich die Komponenten immer weiter aus darüber liegenden Gruppen zusammensetzen. Grundsätzlich gibt es keine strikte Trennung da schlussendlich jeder Komponenten-Typ von React gleich behandelt wird. Die Strukturierung dient zur Steigerung der Modularität und der daraus folgenden Wart- und Testbarkeit. Klar getrennte und isolierte Bestandteile des UIs lassen sich geschickter verwalten.



1.1.2 “atoms”

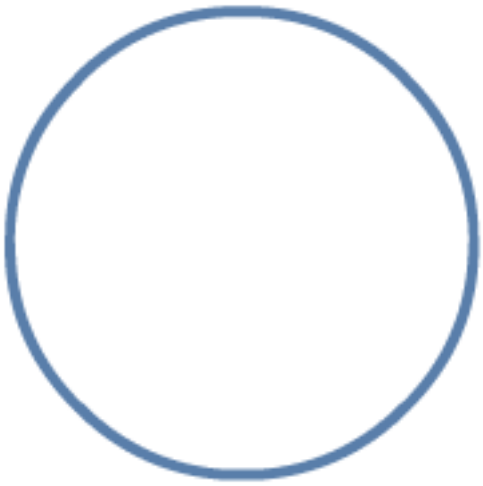


Abbildung 3: Atomare Komponente

Atome sind die kleinstmögliche Form von UI-Komponenten. Unter anderem werden einzelne native HTML-Tags, einfache React-Komponenten oder Komponenten aus Bibliotheken zu diesen atomaren Bestandteilen einer Benutzeroberfläche dazugezählt. Atome dienen einem einzelnen Zweck. Sie bilden den Grundbestandteil des UIs.

z.B. ein Hyperlink bzw. ein Verweis

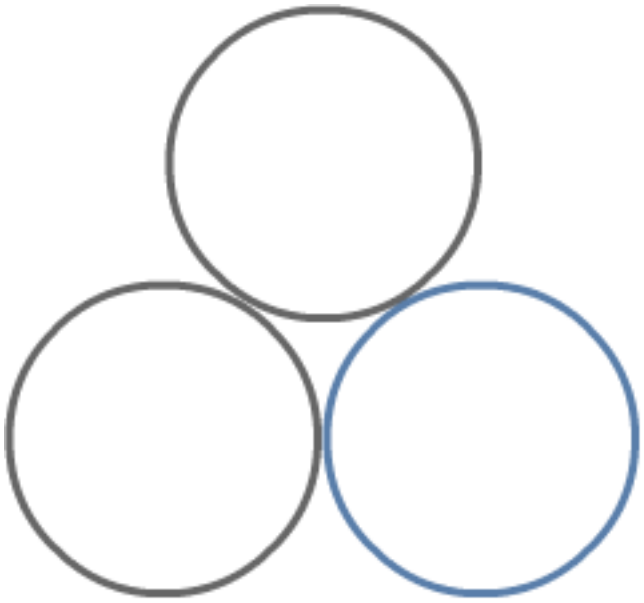
```
const Link = ({ to, ...props }) => <a href={to} {...props} />
```

1.3. “molecules”

Abbildung 4: Molekül zusammengesetzt aus Atomen

Ein Molekül setzt sich aus einer Gruppe von Atomen zusammen. Die Gruppe zusammen repräsentiert ein weiterer Baustein des UIs. Diese Gruppierung ist für Elemente gedacht die in Kombination als eine Einheit arbeiten.

z.B. eine Liste von Navigationsverweisen



```
const NavigationItem = props => (
  <li>
    <Link {...props} />
  </li>
)

const NavigationList = ({ children }) => (
  <ul>
    {children}
  </ul>
)
```

1.1.3 “organisms”

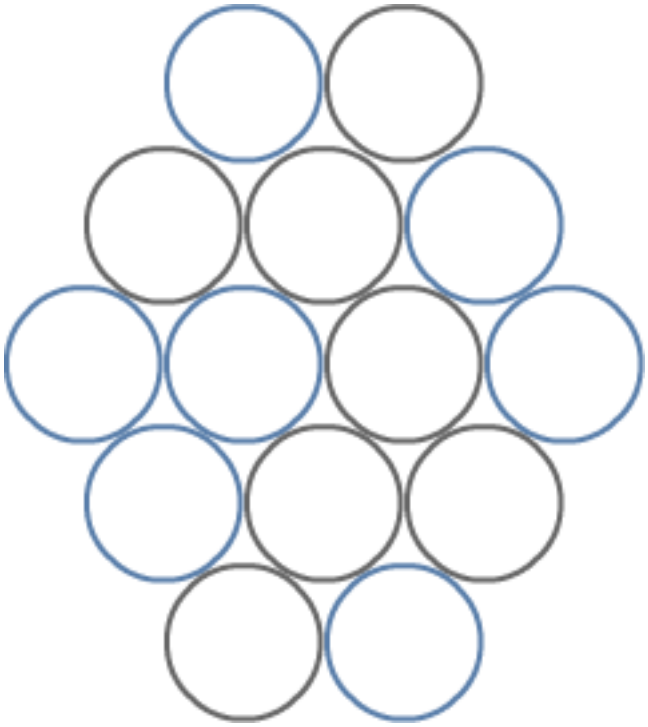


Abbildung 5: Organismus zusammengesetzt aus Molekülen und/oder Atomen

Organismen stellen die nächstgrößere Form da. Sie setzen sich aus einer Gruppe von Atomen, Molekülen und/oder anderen Organismen zusammen. Diese UI-Elemente fallen in der Regel komplex aus und stellen einen deutlichen Abschnitt der Oberfläche dar.

z.B. die Navigation einer Anwendung

```
const Navigation = props => (
  <nav {...props}>
    <NavigationList {...props}>
      <NavigationItem to=”/”      label=”Home”  />
      <NavigationItem to=”/about” label=”About” />
    </NavigationList>
  </nav>
)
```

1.1.4 “templates”

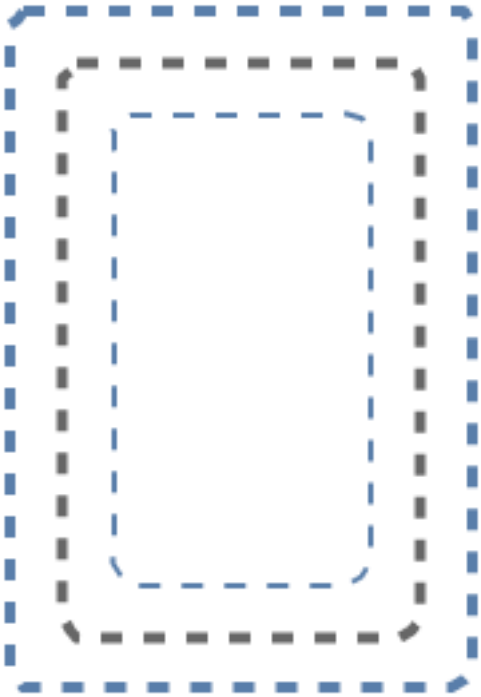


Abbildung 6: Template zur Wiederverwendung eines Seitenlayouts

Templates werden zur Erstellung von Layouts verwendet. Sie dienen als Rahmen für einzelne Seiten. So können zum Beispiel Elemente, die auf allen Seiten benötigt werden, stets gleich platziert werden. Insgesamt sind Templates der Verbund von recht abstrakten Molekülen/Organismen. Sie stellen diese Einheiten zusammen dar und repräsentieren somit das Grundgerüst einer jeden Szene.

z.B. die Standard-Vorlage einer Szene auf einer Webseite

```
const Page = ({ header, navigation, children }) => ( <main>
   {header}
   {navigation}
   {children}
</main> )
```

1.1.5 “pages”

Abbildung 7: Einzelne Seite bzw. Szene basierend auf Komponenten

Seiten oder auch Szenen setzen sich hauptsächlich, aber nicht ausschließlich, aus Organismen zusammen. Sie sind die reale Repräsentation bzw. Darstellung eines Templates.

z.B. zwei Szenen einer Webseite mit gleichem Layout

```
const HomePage = () => (
  <Page
    header={<Header />}
    navigation={<Navigation />}
  >
    <h1>Welcome!</h1>
    <h2>This is the home scene.</h2>
  </Page>
)

const AboutPage = () => (
  <Page
    header={<Header />}
    navigation={<Navigation />}
  >
    <h1>About</h1>
    <h2>This is the about scene.</h2>
  </Page>
)
```

1.2 “containers”

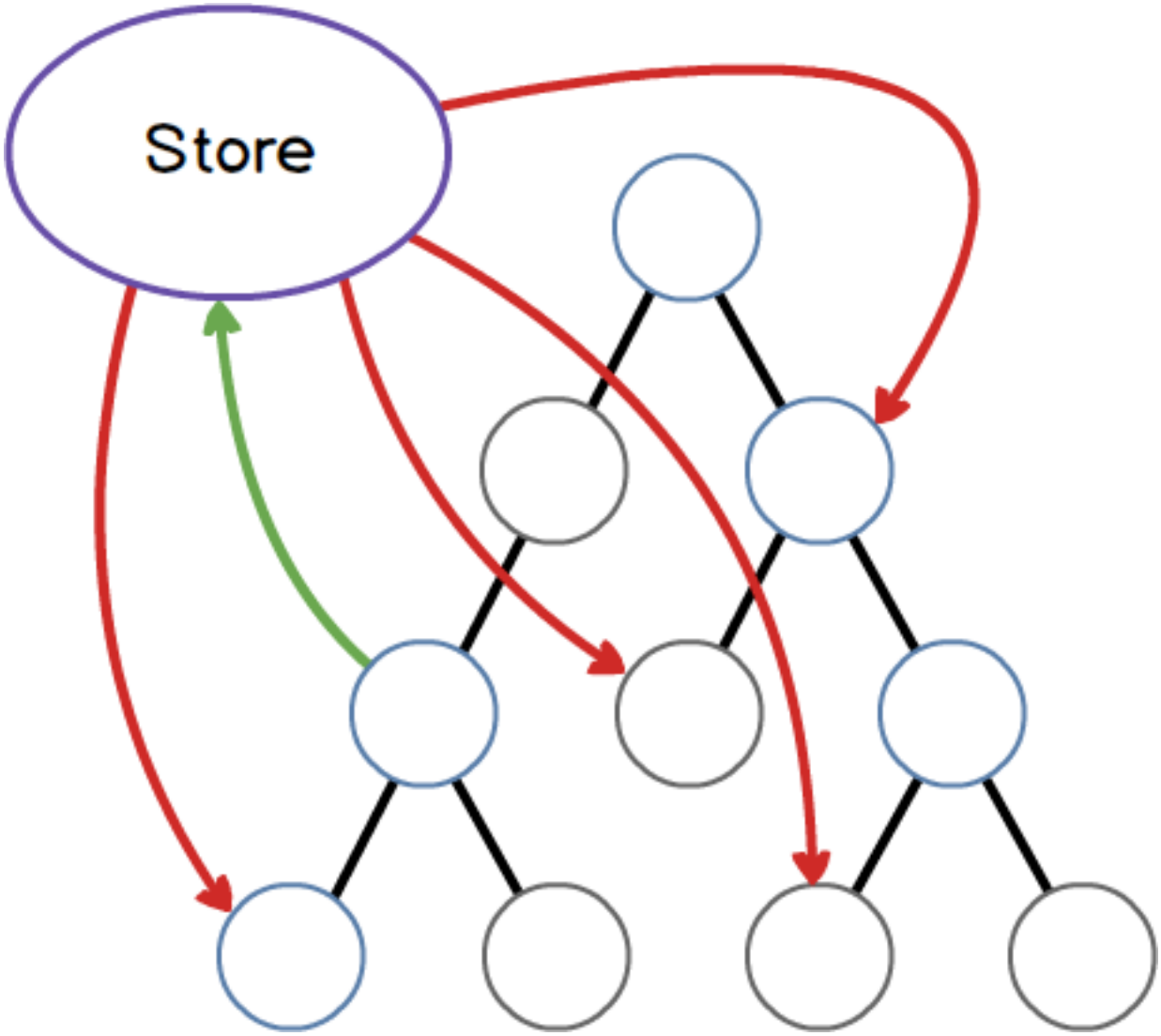


Abbildung 8: Store Komponenten-Anbindung über Container/Subscriber

Ein modularer Ansatz erfordert ebenfalls eine Auslagerung des globalen Anwendungszustands. Immerhin zeichnet React sich besonders im Neu-Rendern von Änderungen aus. Es muss also über eine zentrale Schnittstelle erfasst werden können welche Daten sich wirklich ändern. Damit kann der React-Render-Algorithmus die Änderungen im DOM feststellen und nur diese austauschen.

Ebenfalls gehört das Verhalten von Komponenten nicht zu ihrer eigentlichen Darstellung. Mit Hilfe von Containern können Daten getrennt verwaltet werden. Diese verknüpfen den globalen Zustand mit den gewählten Komponenten. So kann das Verhalten und die darzustellenden Informationen von einer zentralen Stelle an alle Unterelemente weitergereicht werden.

Der globale Zustand wird als Store bezeichnet und durch einen sogenannten Provider allen Unterkomponenten zur Verfügung gestellt. Damit der Zustand nicht an jede Komponente weitergereicht wird, muss eine Subkomponente, die auf den Store zugreifen möchte, unter Einsatz einer Subscriber-Komponente mit diesem verknüpft werden. Die Subkomponente wird in diesem Fall als Container bezeichnet und stellt der darunterliegenden Komponente Funktionen und Daten bereit.

1.3 Weitere Module

1.3.1. “constants”

Konstante Werte die anwendungsübergreifend verwendet werden, können über dieses Modul bereitgestellt werden. Allgemein sollte Code nicht unnötig wiederholt sondern wiederverwendet werden.

1.3.2. “lib”

Nicht alles an Modulen kann in Komponenten oder Containern eingeteilt werden. Untermodule in diesem Modul werden für sich gekapselt behandelt und definieren den Kern der applikationsweiten Geschäftslogik. Die einzelnen Dienste können zwischen Komponenten, Seiten und Szenen ausgetauscht werden. Sie stellen eine Brücke zwischen der Serveranwendung und dem hier beschriebenen Frontend dar. Grundsätzlich werden Seiteneffekte z.B. Netzwerkanfragen oder das Abspeichern von Cookies über solche Module ausgeführt.

1.3.3. “utils”

Kleine Helfer-Funktionen die weder in einen Dienst noch eine Bibliothek passen, werden in diesem Modul definiert. Zum Beispiel Funktionen zum Verzögern von Benutzereingaben oder dem Formatieren von Daten. Meistens können diese Einheiten nicht klar zugeordnet werden, da sie die unterschiedlichsten Aufgaben erfüllen.

Die Einarbeitungsphase dient dazu jeden einzelnen auf einen Wissensstand zu bringen um genügend Kenntnisse über Tools und Programmiersprachen zu haben um effektiv mitzuhelfen.
Es wird eine Einführungsveranstaltung stattfinden, in welcher die benötigten Tools erklärt werden. Diese werden von denen erklärt, wie sich schon mit den Tools auskennen. Nach dieser Veranstaltung sollte jeder die nötigen Kenntnisse haben die Tools zu benutzten oder auch sich leichter Weiterzubilden.
Zu der Phase gehört zudem noch die Installation der Softwares und der Tools die je nach dem mehrere Stunden dauern kann.
Außerdem sollte von jedem einzelnen der Wissensstand für das Projekt erweitert werden, so dass er die für ihn zugewiesene Aufgabe erledigen kann.

2. Modulbeschreibung Backend

2.1 Aufbau der Projektstruktur

In dieser Phase werden alle Aufgaben(-pakete) des Backends den Beteiligten zugewiesen. Hier werden alle vorhersehbaren Aufgaben zugeteilt. Die Darstellung hilft später im Projekt den Überblick zu behalten. Zudem ermöglicht es eine eindeutige Zuordnung wer was macht und vermindert so Probleme in der projektinternen Kommunikation.

2.2 Erstellung der Datenbank

Die Erstellung der Datenbank umfasst das Erstellen eines Datenbankschemas (erledigt), das Erstellen der Models und die Implementation der Query Funktionen.

2.3 Server einrichten

Um den Server einzurichten muss Laravel Homestead eingerichtet werden.

2.4 Tests erstellen

In den Tests werden folgende Sachverhalte getestet: Bundestagswahl anlegen, Europawahl anlegen, Bürgerentscheid anlegen, Wahl auswerten (Wahlleiter), Wahl bearbeiten, nur Wahlleiter kann Wahl freigeben/ablehnen, Wähler kann Stimme abgeben aber nur für die Wahl für die er Berechtigt ist, Rolle kann nur sehen was für sie bestimmt ist, Roll kann nur das bearbeiten für was sie berechtigt ist, Wähler Daten importieren, Partei importieren, Kandidaten importieren.

2.5 API (Schnittstelle Frontend)

Um die Schnittstelle zwischen Frontend und Backend zu implementieren, muss die Logik für das Auslesen der Datenbank (GET request) als auch die Logik für das Schreiben in die Datenbank (POS/PUT request) implementiert werden.

2.6 Sicherheit des Backend

Die Sicherheit des Backend ist sehr wichtig, da Datenmanipulation oder ähnliches unbedingt vermieden werden soll. Die Sicherheit wird durch diverse Methoden ermöglicht.

2.7 Statistik auswerten

Außerdem muss im Backend die Statistik der Wahlen ausgewertet werden, falls sie vom Benutzer angefordert wird. Dazu müssen einige Funktionen implementiert werden, die zum Beispiel die Wahlergebnisse zusammen fassen.

3. Schnittstellen Beschreibung Backend

Die URL fängt immer mit https://yourChoice.de/api/v1/ an (wird mit ... abgekürzt).

3.1 Login

Ein **POST** Request über folgende URL:
.../login

3.2 Logout

Ein **POST** Request über folgende URL: .../logout

3.3 Neues Objekt anlegen (speichern)

Hierfür wird ein **POST** request an das *entsprechende* Model geschickt:

.../<model_name> .../party

Sollte dies (aus irgendwelchen Gründen) nicht funktionieren, kann es auch über den expliziten Aufruf der save Funktion realisiert werden:

.../<model_name>/save

3.4 Objekt bearbeiten

Hierfür wird ein **PUT** request an das *entsprechende* Model geschickt:

.../<model_name>/<id>

3.5 Objekt auslesen

Hierfür wird ein **GET** request an das entsprechende Model geschickt:

.../<model_name>/<id>

3.6 Objekt löschen

Hierfür wird ein **DELETE** request an das entsprechende Model geschickt:

.../<model_name>/<id>

3.7 Wahl auswerten

Ein **POST** Request über folgende URL: *.../election/evaluate*

3.8 Wählen

Ein **POST** Request über folgende URL:
…/vote/

Daten:

- election_id: …
- first_vote: …
- second_vote: …
- valid: …

Oder ein **POST** Request über folgende URL:
…/election/vote

Daten:

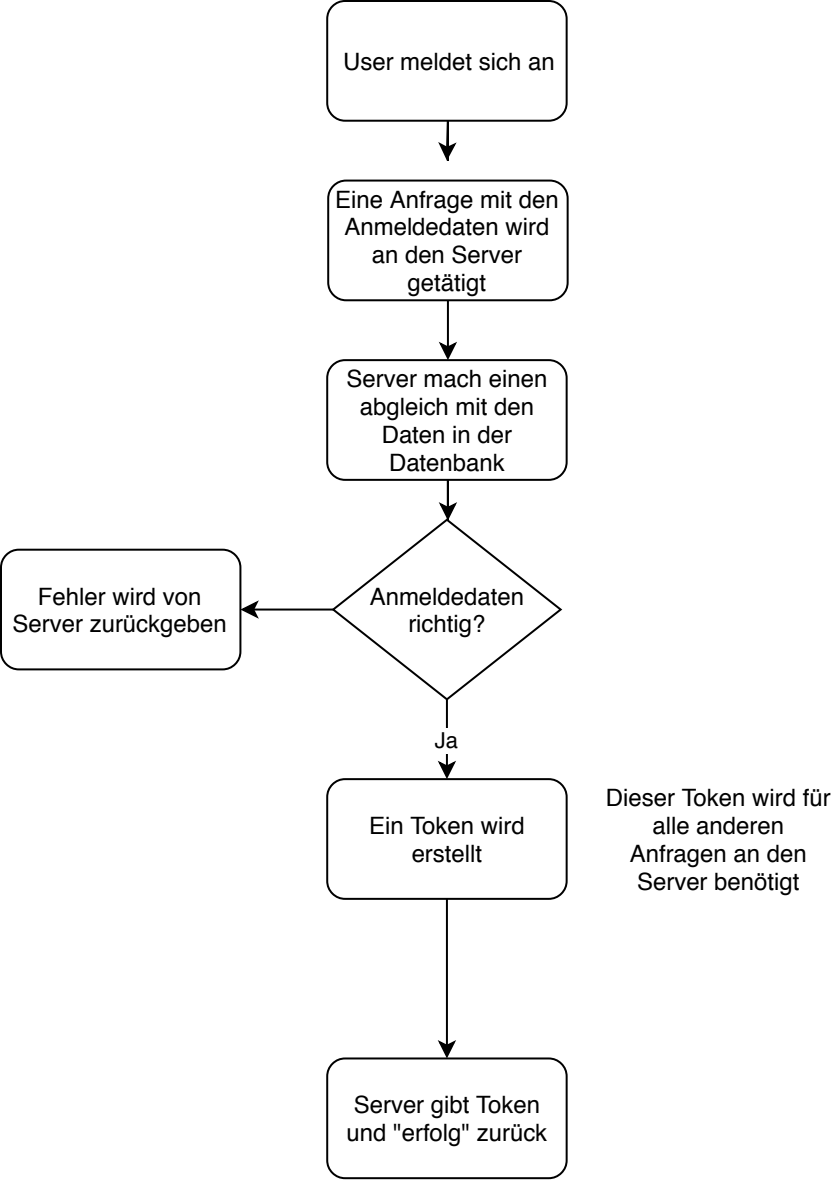
- election_id: …
- first_vote: …
- second_vote: …
- valid: …

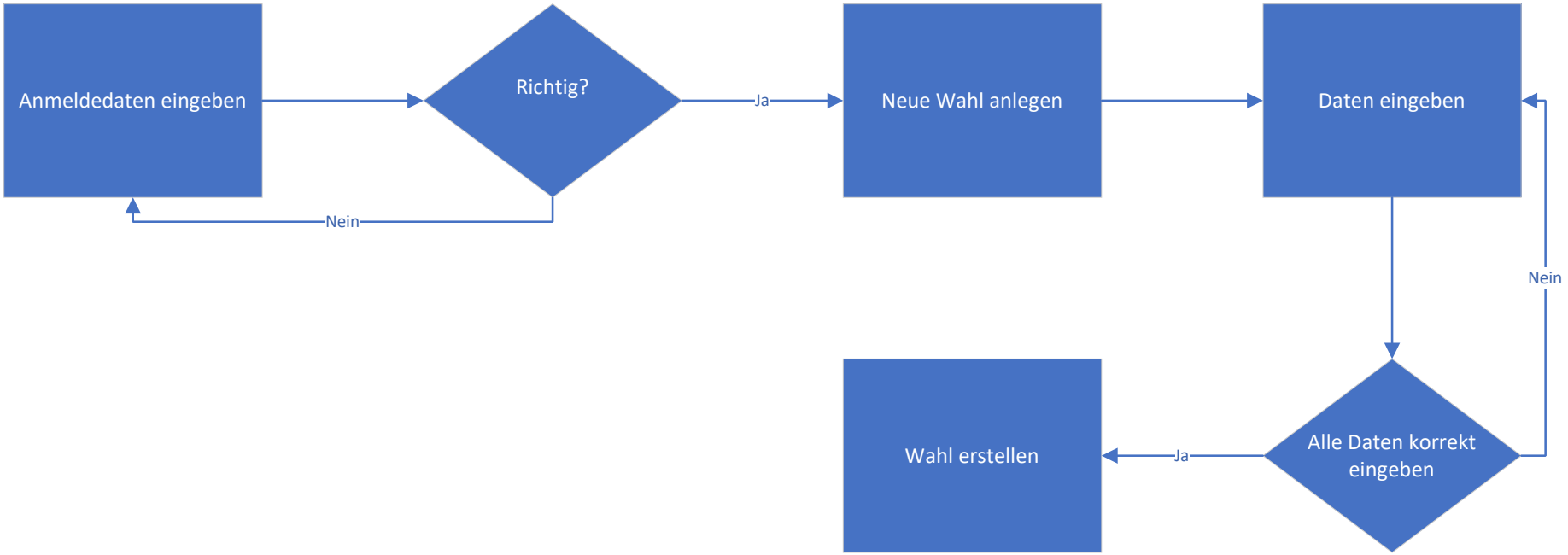
Es besteht die Möglichkeit, dass die jeweilige ID als Parameter, anstatt als URI mitgeschickt werden muss

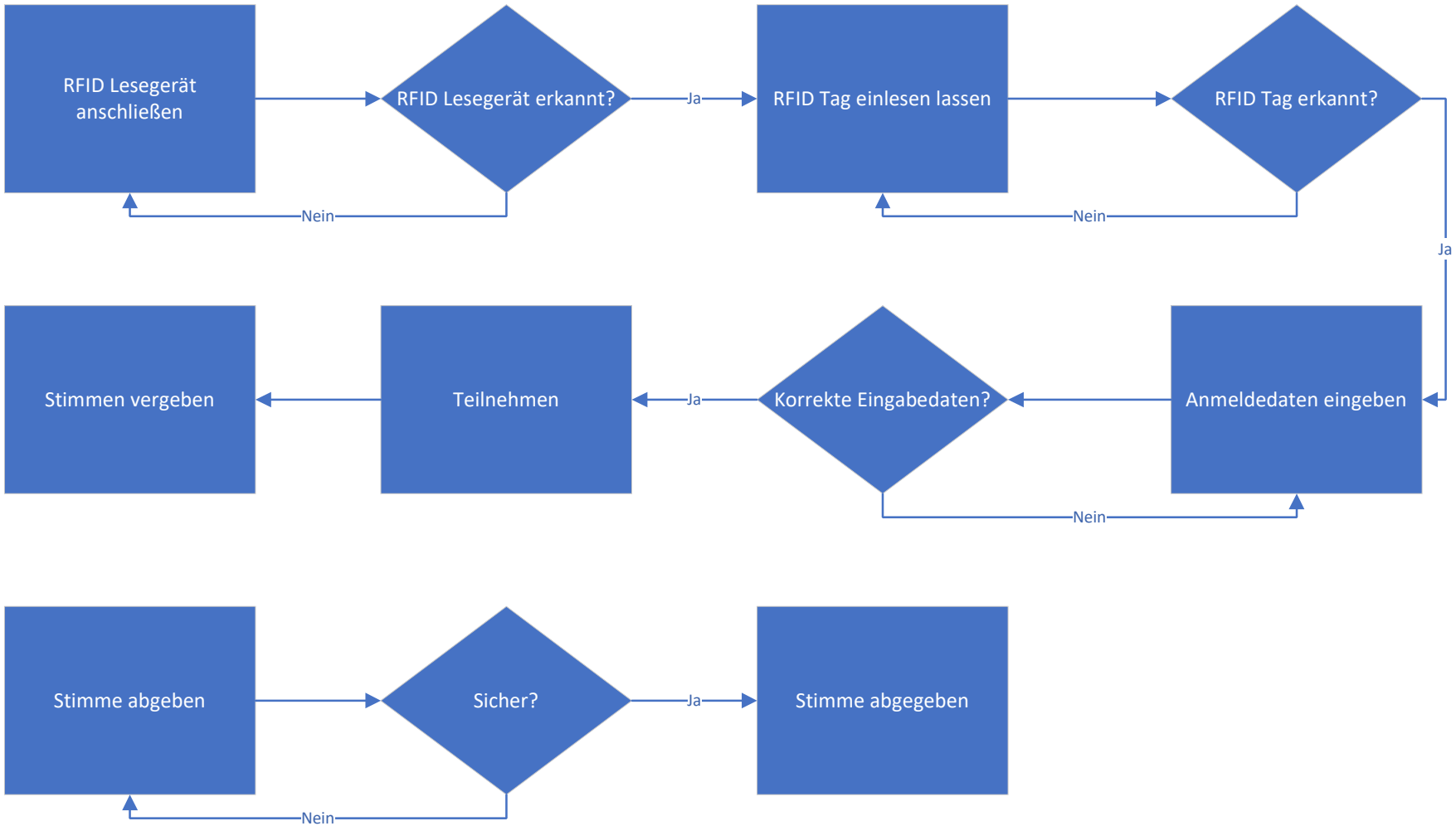
4. Ablaufplan

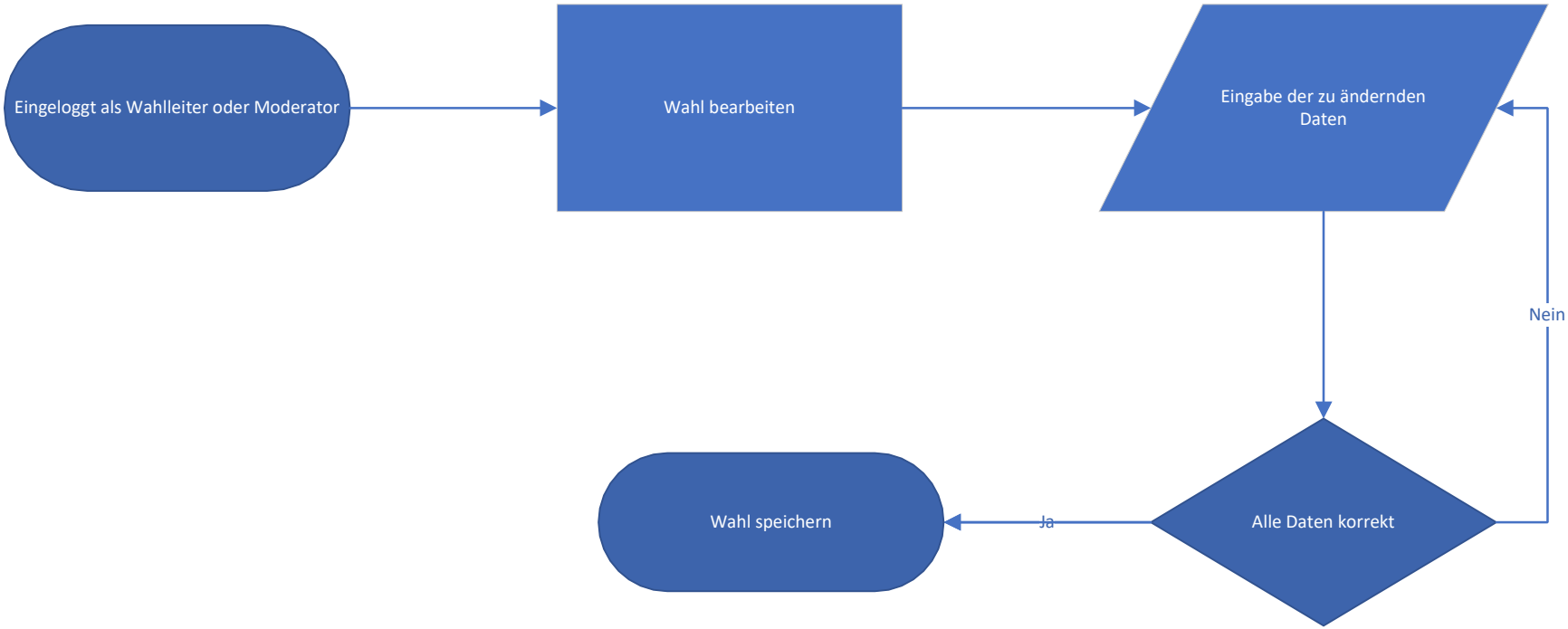
Zeitraum	Aktivität	Vorgehen
1 Jahr vor dem Wahltag bis 2 Wochen vor dem Wahltag, 17.59 Uhr	Wahlen erstellen	- Als Moderator oder Wahlleiter mit Login Daten anmelden. - „Neue Wahl erstellen“ anklicken. - Art der Wahl auswählen, Wahlkreis eingeben, Start- und Endzeit festlegen und Listen importieren. - „Erstellen“ anklicken. - Als Moderator wird die Wahl gespeichert und muss von einem Wahlleiter freigegeben werden. - Als Wahlleiter ist eine Wahl erstellt worden.
1 Jahr vor dem Wahltag bis 2 Wochen vor dem Wahltag, 17.59 Uhr	Wahlen bearbeiten/ freigeben	- Als Moderator oder Wahlleiter mit Login Daten anmelden. - Wahl aus vorhandener Liste auswählen und bearbeiten klicken - Daten bearbeiten, speichern oder ggf. als Wahlleiter freigeben klicken - Wahl wird mit Bearbeitungen gespeichert. Ggf. wird die Wahl freigegeben. In diesem Fall können die Wähler die Wahl nun sehen und wählen, wenn 1. sie sich bereits für das Online-Wahlsystem registriert haben. 2. sie im die Wahl betreffenden Wahlkreis gemeldet sind. 3. der Startzeitpunkt der Wahl bereits erreicht ist.
2 Wochen vor dem Wahltag, 18 Uhr bis Wahltag 18.00 Uhr	wählen	- Als Wähler mit Login-Daten anmelden und RFID-Tag einscannen. - Eine Wahl auswählen an der man teilnehmen will. Man kann nur wählen, wenn man sich für das Onlinewahlsystem registriert hat, im jeweiligen Wahlkreis gemeldet ist und der Zeitraum noch nicht überschritten wurde. - Seine Stimme abgeben und doppelt bestätigen indem man wieder den RFID-Tag einscannt. - Stimme wurde abgegeben.
Ab Wahltag, 18.00 Uhr	Wahlen auswerten	- Als Wahlleiter mit Login Daten anmelden. - Beendete Wahl auswählen und auswerten klicken. - Daten können nun exportiert werden.

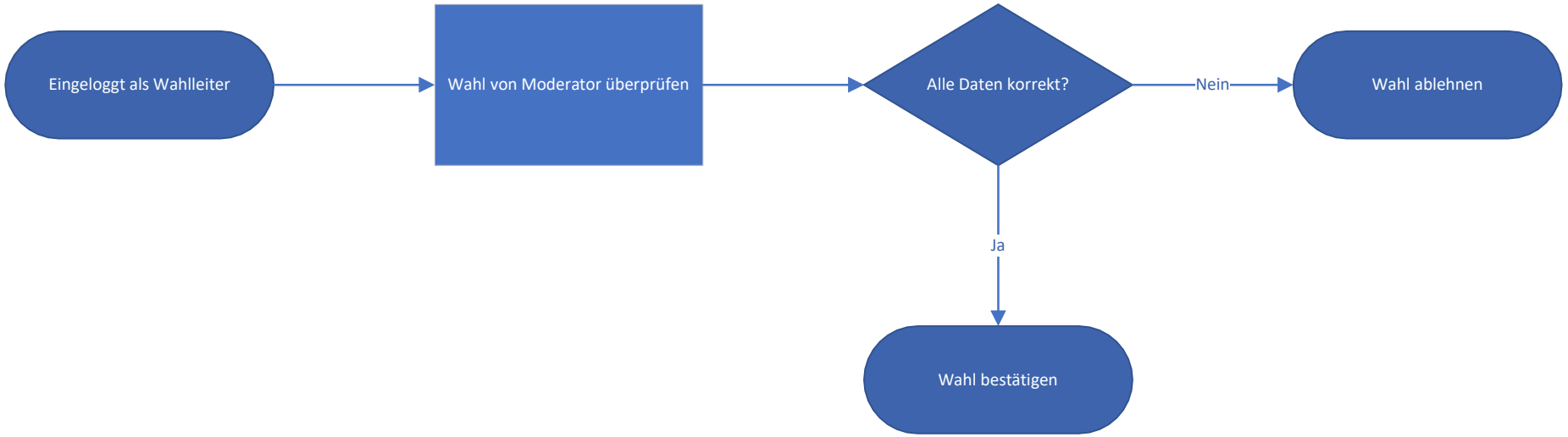
5. Ablaufdiagramme

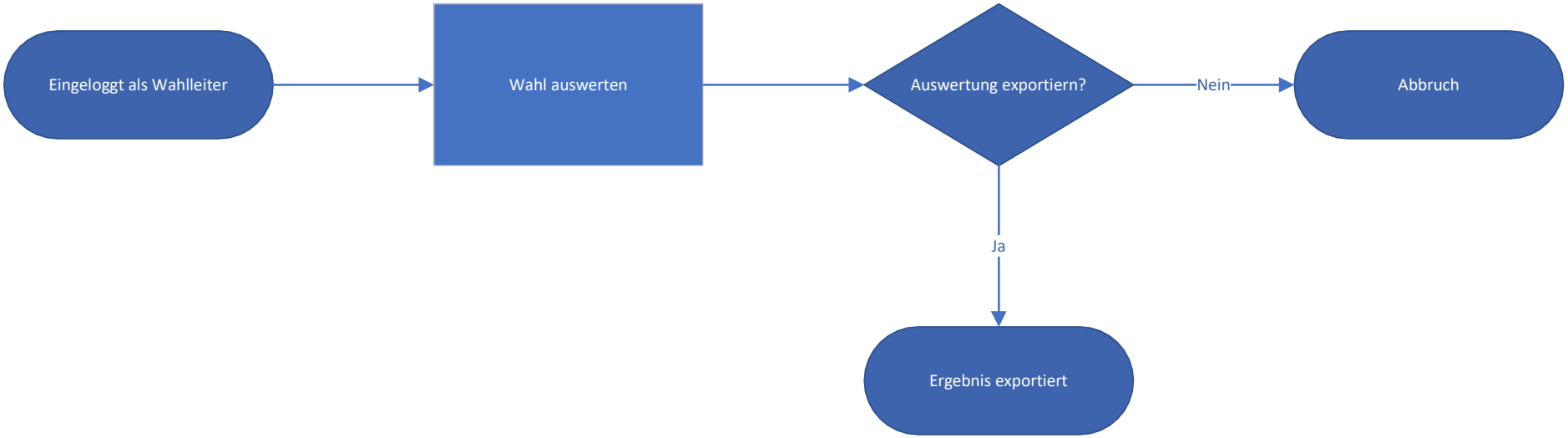

User meldet sich an
Eine Anfrage mit den Anmeldedaten wird an den Server getätigt
Server mach einen abgleich mit den Daten in der Datenbank
Anmeldedaten richtig?
Fehler wird von Server zurückgeben
Ja
Ein Token wird erstellt
Dieser Token wird für alle anderen Anfragen an den Server benötigt
Server gibt Token und "erfolg" zurück


Anmeldedaten eingeben
Richtig?
Ja
Nein
Neue Wahl anlegen
Daten eingeben
Nein
Alle Daten korrekt eingeben
Ja
Wahl erstellen


RFID Lesegerät anschließen
RFID Lesegerät erkannt?
Ja
Nein
RFID Tag einlesen lassen
RFID Tag erkannt?
Nein
Ja
Anmeldedaten eingeben
Korrekte Eingabedaten?
Ja
Nein
Teilnehmen
Stimmen vergeben
Stimme abgeben
Sicher?
Ja
Nein
Stimme abgegeben


Eingeloggt als Wahlleiter oder Moderator
Wahl bearbeiten
Eingabe der zu ändernden Daten
Alle Daten korrekt
Nein
Ja
Wahl speichern


Eingeloggt als Wahlleiter
Wahl von Moderator überprüfen
Alle Daten korrekt?
Nein
Wahl ablehnen
Ja
Wahl bestätigen


Eingeloggt als Wahlleiter
Wahl auswerten
Auswertung exportiern?
Nein
Abbruch
Ja
Ergebnis exportiert