Contents

[Analyse 3](#_Toc135920178)

[Aufgabenstellung 3](#_Toc135920179)

[Infos 3](#_Toc135920180)

[Anforderungen 3](#_Toc135920181)

[Ziele 3](#_Toc135920182)

[Must-have 3](#_Toc135920183)

[Nice-to-have 3](#_Toc135920184)

[Use-Cases 3](#_Toc135920185)

[Lösungsverfahren 3](#_Toc135920186)

[Design 4](#_Toc135920187)

[Mockup 4](#_Toc135920188)

[Testfälle 4](#_Toc135920189)

[Realisation 4](#_Toc135920190)

[Frontend 4](#_Toc135920191)

[Backend 4](#_Toc135920192)

[Testen 5](#_Toc135920193)

[Deployment 5](#_Toc135920194)

[Installation 5](#_Toc135920195)

[Anleitung 5](#_Toc135920196)

# Analyse

## Aufgabenstellung

Grossratswahlen Kanton Thurgau Parteistimmen nach Gemeinden visuell Darstellen

In dieser Dokumentation wird beschrieben, wie man mithilfe von JavaScript und der Chart.js-Bibliothek eine visuelle Darstellung der Thurgauer Grossratswahlen nach Parteien erstellen kann. Die Aufgabe besteht darin, interaktive Diagramme zu erstellen, welche die Verteilung und Entwicklung der Parteien darstellt

### Infos

* OGD-Thurgau
* Moodle

### Anforderungen

* Visuelle Darstellung mit chart.js
* Mind. 2 verschiedene Diagrammtypen
* Filter oder DB

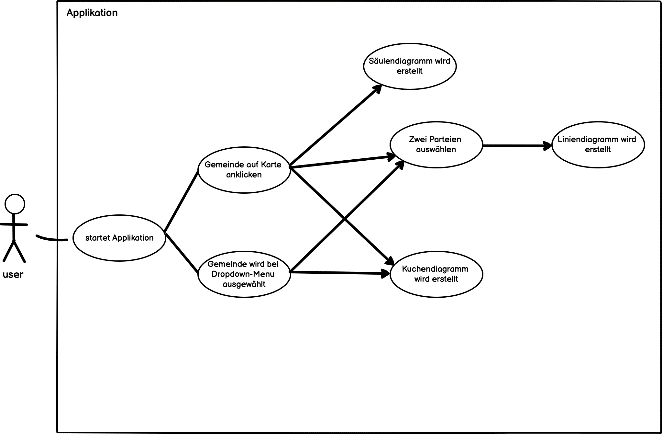
## Ziele

### Must-have

* Map mit tooltip
* 3 verschiedene Diagramme

### Nice-to-have

## Use-Cases



## Lösungsverfahren

* SVG inject mit styling
* Diagramme darstellen
* 4 Datenschnittstellen
* Daten abfragen und mit Diagrammen verknüpfen

## Zielgruppe

* Politikinteressierte Thurgauer welche die Parteiverteilung in Diagrammen sehen wollen
* An der Thurgauer Politik interessierte Leute welche keinen Zugriff auf aktuelle Daten haben

# Design

## Ein Bild, das Entwurf, Diagramm enthält. Automatisch generierte BeschreibungMockup

Diagramme werden untereinander dargestellt  
Auf der Karte kann per Hover-Tooltip die Anzahl Gesamtstimmen pro Gemeinde angesehen werden werden

Das Balkendiagramm stellt die Stimmen pro Partei im gewählten Jahr an

Das Kreisdiagramm zeigt die Stimmen pro Gemeinde und Jahr an

Im Liniendiagramm können 2 Parteien pro Jahr verglichen werden.

## Testfälle

Siehe [Testen](#_Testen)

# Realisation

## Frontend

* Map und Diagramme in HTML einfügen
* Map als SVG inject konfiguriern
* Tooltip zu Map hinzufügen
* Daten vom Backend mit Diagrammen und Tooltips verknüpfen

## Backend

* Struktur mit Model, Contoller, Router erstellen
* JSON-Logik pro Diagramm erstellen
* Routes konfigurieren

## Technologien

### Produkt

* ChartJS.org für Diagramme
* NodeJS mit Express für Backend
* Axios für API Tätigkeiten
* SVG Inject um SVG als img-Tag einzubinden
* Plain HTML, CSS und JS für Frontend
* NPM für Paketverwaltung

### Tools

* IntelliJ IDEA Ultimate für Programmierung
* MS Word für Dokumentation und Dokumente
* Balsamiq Wireframes für Mockups
* Git zur Versionsverwaltung
* GitLab als gemeinsamen Versionstorage

# Testen

|  |  |
| --- | --- |
| Testfall | passed |
| Man kann das Jahr auswählen und das Balkendiagramm wird angepasst |  |
| Man kann auf der Map über eine Gemeinde hovern und dann werden die Gesamtstimmen angezeigt |  |
| Man kann eine Gemeinde wählen (Dropdown) und dann wird das Diagramm aktualisiert |  |
| Man kann ein Jahr wählen (Dropdown) und dann wird das entsprechende Diagramm aktualisiert. (2x) |  |
|  |  |
| Man kann zwei Parteien auswählen (Dropdown) und das Liniendiagramm wird aktualisiert |  |

# Deployment

## Installation

1. GitLab Repository klonen
2. Im Backend und im Frontend «npm install» ausführen, um Packages zu installieren
3. «npm run start» ausführen, um die Applikation zu starten
4. Have fun

## Anleitung

Die Applikation sollte soweit selbsterklärend sein, Durch die Filter können die Diagramme verändert werden und die Angezeigten Daten den eigenen Vorlieben angepasst werden.

# Quellen

[w3schools  
stackoverflow](https://www.w3schools.com/)  
[Toolbox](https://moodle.kftg.ch/course/view.php?id=358&section=3)  
[Die ultimative Quelle für Alles](https://de.wikipedia.org/wiki/Verstand)