

Ausarbeitung

Software Architektur Labor

- Thema -
Wahlergebnisse Web-App

Studenten: Florian Kloss, Tom Göckel, Johannes Karcher

Universität: Hochschule Karlsruhe (HKA)

Studiengang: Informatik (M.Sc.)

Semester: Sommersemester 2025

Dozent: Prof. Dr. Carsten Sinz

Inhaltsverzeichnis

Einleitung	3
Motivation	3
Zielsetzung.....	3
Grundlagen	4
Problemstellung.....	4
Technologieauswahl.....	4
Use Cases	5
Muss- /Kann-Kriterien.....	7
Umsetzung	9
Benutzerhandbuch	9
Entwicklerdokumentation.....	14
Neue Komponente hinzufügen	15
Neue Schnittstelle hinzufügen	16
Fazit.....	19
Literaturverzeichnis.....	20

Einleitung

Motivation

Wählen sind ein zentrales Element jeder Demokratie. Sie ermöglichen es den Bürgerinnen und Bürgern, aktiv an politischen Entscheidungsprozessen teilzunehmen und die politische Richtung ihres Landes mitzubestimmen. Gerade in einer zunehmend digitalisierten Gesellschaft ist der Anspruch gestiegen, politische Informationen schnell, zuverlässig und übersichtlich abrufen zu können. Dazu zählen insbesondere Wahlergebnisse, die sowohl unmittelbar nach der Wahl als auch in historischer Perspektive von großem Interesse für die Öffentlichkeit sind.

Obwohl viele offizielle Stellen Wahlergebnisse bereitstellen, ist die Art der Darstellung häufig unübersichtlich, fragmentiert oder technisch wenig zugänglich. Gerade für weniger technisch affine Personen oder für Medienvertreter, die schnelle und visuell aufbereitete Informationen benötigen, stellt dies eine Hürde dar. Hinzu kommt, dass der Vergleich aktueller und vergangener Wahlergebnisse oft nur mit hohem manuellem Aufwand möglich ist. Eine benutzerfreundliche und strukturierte Plattform zur Darstellung dieser Daten würde nicht nur die politische Transparenz fördern, sondern auch das politische Interesse in der Bevölkerung stärken.

Zudem stellt die Verarbeitung und Darstellung von Wahlergebnissen interessante Herausforderungen an die Softwarearchitektur. Es gilt, große Datenmengen effizient zu verarbeiten, eine hohe Verfügbarkeit und Skalierbarkeit zu gewährleisten und gleichzeitig eine intuitive Benutzeroberfläche bereitzustellen.

Zielsetzung

Ziel des Projekts ist die Konzeption und Umsetzung einer Software (Web-App) zur übersichtlichen Darstellung von Wahlergebnissen in Deutschland. Dabei sollen sowohl aktuelle Wahlergebnisse möglichst zeitnah nach der Wahl präsentiert als auch vergangene Ergebnisse archiviert und vergleichbar aufbereitet werden. Im Zentrum steht die Entwicklung einer modularen und wartbaren Systemarchitektur, die den Anforderungen an Performance, Skalierbarkeit und Usability gerecht wird.

Die geplante Lösung richtet sich primär an zwei Zielgruppen: Zum einen an Bürgerinnen und Bürger, die sich über das politische Geschehen informieren möchten, und zum anderen an Medienvertreter, die Wahlergebnisse für Berichterstattung, Analysen und Vergleiche benötigen. Durch eine intuitive Benutzeroberfläche und eine performante Backend-Architektur soll sichergestellt werden, dass beide Gruppen effizient auf die gewünschten Informationen zugreifen können.

Im weiteren Verlauf dieser Ausarbeitung werden die Anforderungen an das System detailliert analysiert, verschiedene Architekturentwürfe evaluiert und ein geeigneter Architekturansatz ausgewählt und begründet. Zudem werden Aspekte wie Datenintegration, Benutzerinteraktion sowie mögliche Erweiterbarkeit und Wartbarkeit der Lösung betrachtet.

Grundlagen

Problemstellung

Ziel der geplanten Applikation ist es, aktuelle und vergangene Wahlergebnisse in Deutschland auf eine übersichtliche, benutzerfreundliche und interaktive Weise darzustellen. Die Daten sollen dabei direkt von offiziellen Quellen – insbesondere der Webseite des Bundeswahlleiters – bezogen und in geeigneter Form aufbereitet werden. Die Anwendung richtet sich an ein breites Publikum, insbesondere an Bürgerinnen und Bürger sowie an Medienvertreter, die zuverlässige, aktuelle und klar visualisierte Informationen benötigen.

Die Umsetzung dieses Vorhabens ist mit mehreren Herausforderungen verbunden:

- **Datenqualität und Aktualität:** Die Gewährleistung, dass die von offiziellen Stellen bezogenen Wahldaten stets korrekt, vollständig und aktuell sind, ist essenziell. Insbesondere bei Live-Daten zur laufenden Auszählung müssen Aktualisierungen in kurzen Intervallen verarbeitet und ohne Verzögerung dargestellt werden.
- **Datenaufbereitung und -integration:** Die von der Bundeswahlseite bereitgestellten Rohdaten liegen häufig in komplexen, maschinenlesbaren Formaten (z. B. XML, CSV oder JSON) vor. Diese müssen automatisiert eingelesen, validiert, transformiert und in ein strukturiertes Format überführt werden, das sich für die Anzeige und Analyse innerhalb der Applikation eignet.
- **Benutzerfreundlichkeit und Visualisierung:** Eine der zentralen Anforderungen ist die Gestaltung einer intuitiven und barrierearmen Benutzeroberfläche, die auch umfangreiche und mehrdimensionale Wahldaten – etwa nach Parteien, Regionen oder Zeitverläufen – leicht verständlich und visuell ansprechend aufbereitet.
- **Skalierbarkeit und Performance:** Die Anwendung muss auch bei starkem Nutzeraufkommen, z. B. am Wahlabend, stabil und performant bleiben. Dies erfordert eine Architektur, die Lastspitzen abfedern kann und sich bei Bedarf flexibel skalieren lässt.

Zusätzlich gilt es, die Lösung so zu gestalten, dass sie auch zukünftige Erweiterungen – wie z. B. die Integration weiterer Datenquellen, interaktive Vergleiche zwischen Wahlen oder benutzerdefinierte Filterfunktionen – unterstützt. Diese Anforderungen erfordern eine durchdachte Softwarearchitektur, die Modularität, Wartbarkeit und Erweiterbarkeit gleichermaßen berücksichtigt.

Technologieauswahl

Für die Umsetzung der Applikation zur Darstellung deutscher Wahlergebnisse wurde eine moderne und bewährte Technologieauswahl getroffen, die sowohl eine hohe Entwicklungsproduktivität als auch eine gute Wart- und Erweiterbarkeit gewährleistet. Die Architektur der Anwendung folgt dabei dem klassischen Frontend-Backend-Modell, wobei das Frontend für die Benutzerinteraktion und das Backend für die Datenverarbeitung und -bereitstellung verantwortlich ist.

Frontend: React

Für die Entwicklung des Frontends wird das JavaScript-Framework React eingesetzt. React ist eines der weltweit am weitesten verbreiteten Frameworks für die Entwicklung von Webanwendungen und bietet zahlreiche Vorteile für dieses Projekt:

- Komponentenbasierter Ansatz: React basiert auf dem Prinzip wiederverwendbarer Komponenten, was eine strukturierte, modulare und wartbare Entwicklung ermöglicht. UI-Elemente wie Diagramme, Tabellen oder Filterfunktionen können in eigenständigen Komponenten implementiert und flexibel zusammengesetzt werden.
- Effizientes State Management: Die Verwaltung und Aktualisierung des Applikationszustands (State) ist in React durch interne Mechanismen und zusätzliche Bibliotheken wie Redux oder Zustand gut unterstützt. Dadurch lassen sich dynamische Daten wie Wahlergebnisse oder Nutzerinteraktionen effizient verarbeiten und in Echtzeit anzeigen.
- Breites Ökosystem und Community-Support: React verfügt über eine sehr aktive Community und ein umfangreiches Ökosystem. Zahlreiche frei verfügbare Bibliotheken – etwa für Datenvisualisierung (z. B. Recharts, D3.js oder Chart.js) – können direkt integriert werden, was die Entwicklung beschleunigt und qualitativ hochwertige Ergebnisse ermöglicht.

Backend: FastAPI

Für das Backend wurde FastAPI gewählt, ein modernes, leichtgewichtiges Web-Framework für Python, das speziell auf die Entwicklung von RESTful APIs ausgerichtet ist. Die Entscheidung für FastAPI basiert auf mehreren technischen und praktischen Überlegungen:

- Hohe Performance durch asynchrone Verarbeitung: FastAPI unterstützt asynchrone Programmierung nativ, was bei der Verarbeitung großer Datenmengen – wie sie beim Abruf und der Aktualisierung offizieller Wahldaten vorkommen – zu einer hohen Systemperformance und geringen Antwortzeiten führt.
- Nahtlose Integration in das Python-Ökosystem: Da viele Datenquellen und Analysewerkzeuge rund um Wahldaten in Python zur Verfügung stehen, erleichtert FastAPI die Anbindung und Weiterverarbeitung dieser Informationen. Bibliotheken wie pandas, requests oder lxml lassen sich problemlos integrieren.
- Automatisch generierte API-Dokumentation: FastAPI bietet integrierte Unterstützung für die automatische Erstellung einer OpenAPI-konformen API-Dokumentation (z. B. über Swagger UI). Dies verbessert nicht nur die Lesbarkeit und Wartbarkeit der Schnittstellen, sondern fördert auch die Zusammenarbeit im Entwicklungsteam und mit externen Stakeholdern.

Die Kombination aus React im Frontend und FastAPI im Backend stellt somit eine robuste, leistungsfähige und zukunftssichere Basis für die Umsetzung der geplanten Anwendung dar. Beide Technologien sind flexibel einsetzbar und können bei Bedarf um zusätzliche Funktionen und Dienste erweitert werden – etwa um Authentifizierung, Caching oder eine Datenbankanbindung.

Use Cases

Zur Unterstützung der definierten Zielgruppen – insbesondere Bürgerinnen und Bürger sowie Medienvertreter – deckt die Applikation eine Reihe zentraler Anwendungsfälle ab. Diese Use Cases beschreiben die grundlegenden Funktionen und Interaktionen zwischen Nutzer und System.

1. Datenabruf und -anzeige

Beschreibung: Der Nutzer kann aktuelle Wahlergebnisse aufrufen. Die Anwendung stellt sicher, dass die jeweils neuesten verfügbaren Daten angezeigt werden.

Ablauf:

- Der Benutzer navigiert zur Startseite der Anwendung.
- Die Applikation ruft automatisiert die aktuellen Wahldaten von der offiziellen Webseite des Bundeswahlleiters ab.
- Die Daten werden in strukturierter und verständlicher Form angezeigt (z. B. als Tabellen oder Diagramme).

2. Filterung und Suche

Beschreibung: Die Nutzer haben die Möglichkeit, Wahlergebnisse gezielt nach verschiedenen Kriterien einzuschränken.

Mögliche Filteroptionen:

- Wahltyp (z. B. Bundestagswahl, Landtagswahl)
- Bundesländer
- Zeiträume bzw. Wahljahre
- Parteien

Ziel: Nutzer können schnell relevante Daten finden, vergleichen oder analysieren.

3. Abruf und Vergleich historischer Daten

Beschreibung: Die Anwendung ermöglicht den Zugriff auf vergangene Wahlergebnisse.

Funktionalität:

- Abruf von Wahlergebnissen früherer Jahre oder Legislaturperioden.
- Vergleichsfunktionen, z. B. gegenüber aktuellen Werten oder zwischen verschiedenen Wahlen.

Ziel: Förderung politischer Transparenz und Unterstützung analytischer Betrachtungen durch Langzeitvergleiche.

4. Datenvisualisierung

Beschreibung: Die Anwendung stellt Wahlergebnisse visuell ansprechend und interaktiv dar.

Visualisierungstypen:

- Balken- oder Kreisdiagramme zur Stimmenverteilung
- Kartenvizualisierungen für regionale Ergebnisse (z. B. auf Ebene von Bundesländern oder Wahlkreisen)
- Dynamische Veränderungen bei Filterauswahl

Ziel: Komplexe Daten werden verständlich und ansprechend präsentiert, um die Benutzerfreundlichkeit zu erhöhen.

5. Responsive Design und plattformübergreifende Nutzung

Beschreibung: Die Applikation ist für verschiedene Endgeräte optimiert.

Gerätetypen:

- Desktop-PCs
- Tablets
- Smartphones

Ziel: Unabhängig vom verwendeten Gerät wird eine gleichbleibend hohe Benutzerfreundlichkeit und Funktionalität gewährleistet.

Muss- /Kann-Kriterien

Für die Konzeption und Realisierung der Applikation werden funktionale und nicht-funktionale Anforderungen in Muss-Kriterien (verbindlich umzusetzen) und Kann-Kriterien (wünschenswerte Erweiterungen) unterteilt. Diese Unterscheidung hilft dabei, den Entwicklungsfokus zu priorisieren und den Projektumfang realistisch zu planen.

Muss-Kriterien

Die folgenden Anforderungen gelten als essentiell für die Erfüllung der Projektziele:

- Aktualität: Die Anwendung muss in der Lage sein, aktuelle Wahlergebnisse zuverlässig und zeitnah darzustellen. Eine regelmäßige Aktualisierung der Datenquellen ist sicherzustellen.
- Datenintegrität: Die angezeigten Daten müssen korrekt, vollständig und konsistent mit den offiziellen Informationen sein. Fehlerhafte oder veraltete Informationen sind auszuschließen.
- Nutzerfreundlichkeit: Die Benutzeroberfläche soll intuitiv bedienbar sein, mit klarer Navigation, verständlicher Darstellung der Inhalte und responsivem Design für verschiedene Endgeräte.
- Performance: Die Anwendung muss kurze Ladezeiten und schnelle Reaktionszeiten bieten, auch bei hoher Nutzerlast, insbesondere während großer Wahlen mit erhöhtem Zugriff.
- Sicherheit: Der Umgang mit externen Datenquellen sowie die Verarbeitung innerhalb des Systems muss gegen Manipulationen, Datenverlust und unautorisierte Zugriffe abgesichert sein.

Kann-Kriterien

Die folgenden Funktionen stellen wünschenswerte Erweiterungen dar, die den Funktionsumfang und den Mehrwert der Anwendung erhöhen, aber nicht zwingend für den MVP (Minimum Viable Product) erforderlich sind:

- **Erweiterte Visualisierungen:** Ergänzung der Darstellung durch komplexere Visualisierungen, z. B. Heatmaps, interaktive Karten oder animierte Zeitreihen, um tiefere Einblicke zu ermöglichen.
- **Personalisierung:** Möglichkeit für registrierte Nutzer, bestimmte Wahlen, Regionen oder Parteien als Favoriten zu markieren oder ein personalisiertes Dashboard zu erstellen.
- **Exportfunktionen:** Bereitstellung von Exportmöglichkeiten für Wahldaten in verschiedenen Formaten (z. B. CSV, Excel, PDF), etwa zur Weiterverarbeitung durch Medien oder Forschung.

- **Benachrichtigungen:** Implementierung von Alert- oder Newsletter-Funktionen, die Nutzer über neue Wahlergebnisse oder relevante politische Ereignisse automatisch informieren.
- **Mehrsprachigkeit:** Unterstützung weiterer Sprachen (z. B. Englisch, Türkisch), um auch nicht-deutschsprachige Bürgerinnen und Bürger oder ein internationales Publikum anzusprechen.
- **KI-Prognosen:** Integration eines Machine-Learning-Modells (z. B. mit TensorFlow.js), das auf Basis historischer Wahldaten und aktueller Umfragen mögliche Trends oder Sitzverteilungen prognostiziert.

Umsetzung

Die Konzeption und Umsetzung der Wahlergebnisse Web-App erfolgte im Rahmen einer gemeinschaftlichen Teamarbeit. Die Idee zur Anwendung sowie die einzelnen funktionalen und technischen Entscheidungen wurden im Team diskutiert, geplant und iterativ weiterentwickelt. Eine klare, starre Aufgabenzuordnung war im Projektverlauf nicht gegeben. Vielmehr arbeiteten alle Teammitglieder flexibel an verschiedenen Teilen des Systems. Dieses kollaborative Vorgehen förderte einen kontinuierlichen Wissensaustausch und trug wesentlich zur Gesamtlösung bei.

Benutzerhandbuch

Nach dem Aufruf der Anwendung begrüßt die Startseite die Nutzerinnen und Nutzer mit dem Titel „**Wahlergebnisse in Deutschland**“ und einem kurzen Einführungstext. Die Anwendung bietet eine interaktive Visualisierung von Wahlergebnissen aus den deutschen Bundesländern.

The screenshot shows the homepage of the 'Wahlergebnisse in Deutschland' web application. At the top, there is a navigation bar with links: 'Wahlergebnisse' (highlighted in blue), 'Interaktive Karte', 'Vergleich', 'Vorhersage', 'Partei Analyse', 'Wahlbeteiligung & Demografie', 'Wahlumfragen', and 'News'. Below the navigation bar, the main title 'Wahlergebnisse in Deutschland' is displayed, followed by a subtitle 'Interaktive Visualisierung von Wahlergebnissen aus den deutschen Bundesländern'. There are two buttons: 'Karte anzeigen' (blue) and 'Mehr erfahren' (white). The page is divided into six sections, each with an icon and a brief description:

- Interaktive Karte**: Erkunden Sie die Wahlergebnisse auf einer interaktiven Karte.
- Vergleichsansicht**: Vergleichen Sie Ergebnisse zwischen Regionen und Jahren.
- Vorhersage-Tool**: Analysieren Sie mögliche Wahlausträge.
- Partei-Analyse**: Entwicklung und regionale Stärke einer Partei.
- Wahlbeteiligung & Demografie**: Zusammenhang zwischen Bevölkerung und Beteiligung.
- News**: Entwicklung der Wahlergebnisse über die Jahre.

Im oberen Bereich befindet sich eine Navigationsleiste mit den Hauptfunktionen der Webanwendung:

- **Interaktive Karte**
- **Vergleich**
- **Vorhersage**
- **Partei-Analyse**
- **Wahlbeteiligung & Demografie**
- **Wahlumfragen**
- **News**

Darunter sind die wichtigsten Themenbereiche als **Kacheln mit Symbolen** dargestellt. Diese führen direkt zu den jeweiligen Modulen der Anwendung. Durch einen Klick auf „**Karte anzeigen**“ gelangt man direkt zur interaktiven Kartenansicht mit aktuellen Wahlergebnissen.

Die Startseite ist übersichtlich gestaltet und ermöglicht einen schnellen Einstieg in die verschiedenen Analysemöglichkeiten. Jede Kachel ist mit einem kurzen Beschreibungstext versehen, der die jeweilige Funktion erläutert.

Die **interaktive Kartenansicht** ermöglicht eine visuelle Darstellung der Wahlergebnisse für jedes Bundesland Deutschlands. Nutzer:innen können Bundesland und Wahljahr auswählen, um sich die Ergebnisse sowie ergänzende Landesinformationen anzeigen zu lassen.



Funktionen im Überblick:

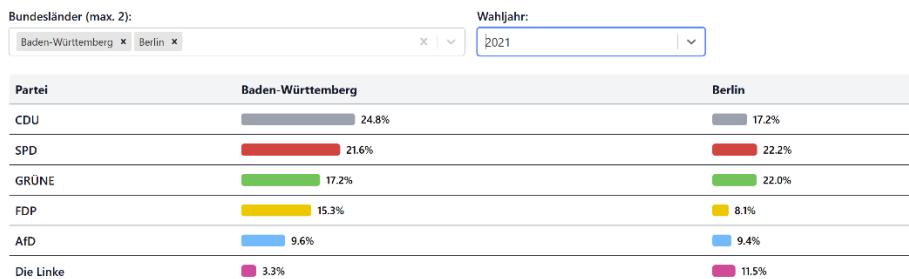
Bereich	Beschreibung
Bundesland-Auswahl	Dropdown zur Auswahl eines Bundeslandes.
Jahresauswahl	Dropdown zur Auswahl des Wahljahrs (z. B. 2017, 2021, 2025).
Karte	Zeigt Deutschland mit den Bundesländern; das gewählte Bundesland wird hervorgehoben.
Wahlergebnisse	Darstellung der Stimmanteile der wichtigsten Parteien (Balken mit Prozentwert).
Landesinformationen	Zeigt Basisdaten des gewählten Bundeslands an: Hauptstadt, Fläche, Gründungsjahr, Bevölkerung, wirtschaftliche Kennzahlen etc.

Benutzerhinweise:

- Die Karte lässt sich **zoomen und verschieben**, um genauere Betrachtung zu ermöglichen.
- Die **Datenaktualität** ist je nach Wahljahr unterschiedlich.

In der **Vergleichsansicht** lassen sich die Wahlergebnisse von **zwei Bundesländern gleichzeitig gegenüberstellen**. So erhält man einen schnellen Überblick über politische Unterschiede oder Entwicklungen in verschiedenen Regionen.

Vergleich der Wahlergebnisse



Funktionen im Überblick:

Bereich	Beschreibung
 Bundesland-Auswahl (max. 2)	Es können genau zwei Bundesländer ausgewählt werden, deren Wahlergebnisse miteinander verglichen werden sollen.
 Wahljahr-Auswahl	Auswahl des Wahljahres, für das die Vergleichsdaten angezeigt werden (z. B. 2017, 2021, 2025).
 Ergebnistabelle	Die wichtigsten Parteien werden zeilenweise dargestellt. Die Spalten zeigen die Stimmanteile der gewählten Bundesländer, jeweils farblich hervorgehoben mit Prozentwert.

Benutzerhinweise:

- Die farbigen Balken visualisieren den Stimmenanteil pro Partei.
- Die Auswahl von mehr als zwei Bundesländern ist **nicht möglich**, um die Übersichtlichkeit zu wahren.
- Fehlende oder nicht verfügbare Daten (z. B. bei zukünftigen Jahren) werden mit „**Keine Daten**“ dargestellt.

Die Komponente **Wahlprognose** bietet eine vereinfachte, modellbasierte Vorhersage der Wahlergebnisse für zukünftige Jahre. Sie basiert auf linearen Trends früherer Ergebnisse und dient als exploratives Werkzeug – nicht als exakte Voraussage.

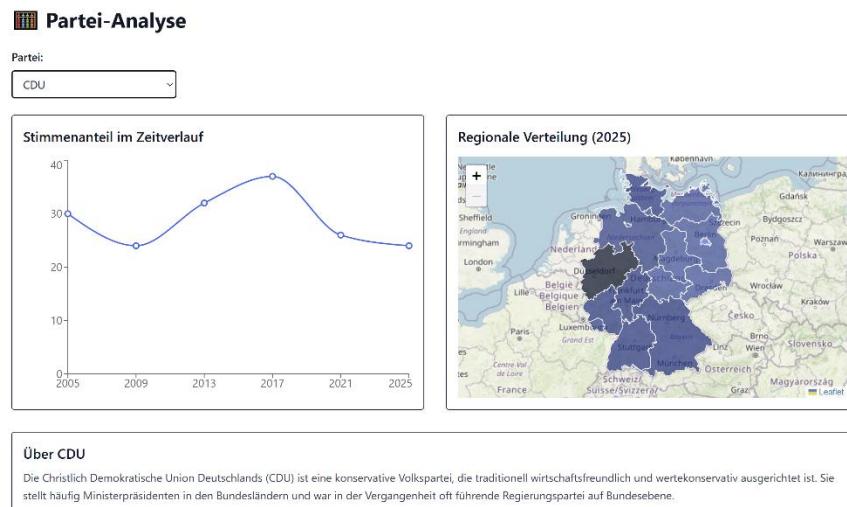
Wahlprognose (Beta)



Funktionen im Überblick:

Bereich	Beschreibung
 Bundesland-Auswahl	Auswahl eines Bundeslandes, für das eine Prognose berechnet werden soll.
 Zieljahr	Freie Eingabe oder Auswahl eines zukünftigen Wahljahres (z. B. 2029, 2030, 2035).
 Ergebnisanzeige	Grafische Darstellung des prognostizierten Stimmenanteils je Partei – Balken visualisieren die relativen Werte.

Die Komponente **Partei-Analyse** ermöglicht es den Nutzer:innen, die historische Entwicklung und regionale Verteilung einer ausgewählten Partei detailliert zu untersuchen.



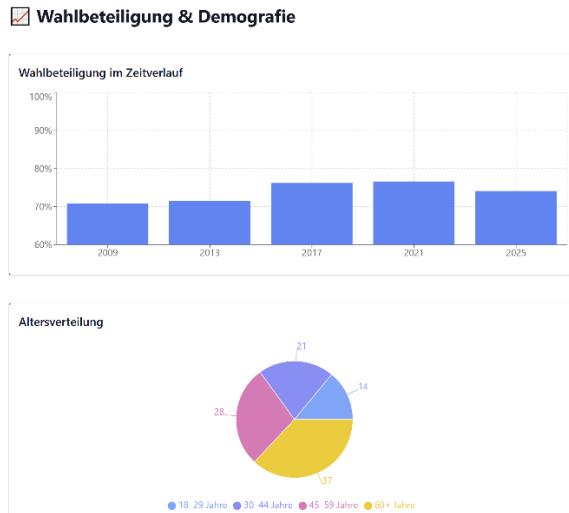
Funktionen im Überblick:

Bereich	Beschreibung
 Parteiauswahl	Auswahl einer Partei aus dem Dropdown-Menü zur Analyse.
 Stimmenanteil im Zeitverlauf	Liniendiagramm zeigt die Entwicklung der Stimmenanteile bei Bundestagswahlen seit 2005 bis zur Prognose 2025.
 Regionale Verteilung (2025)	Interaktive Deutschlandkarte visualisiert die regionalen Unterschiede im Wahlergebnis je Bundesland im Jahr 2025. Die Farbintensität gibt die Stärke der Partei an.
 Parteibeschreibung	Kurzporträt der gewählten Partei mit allgemeinen Informationen zu Ausrichtung und historischer Bedeutung.

Benutzerhinweise:

- Das Zeitdiagramm erlaubt es, **Trends und Entwicklungen** leicht zu erkennen – z. B. Wachstum, Einbrüche oder Stabilität über die Jahre.
- Der Textbereich informiert über die parteipolitische Ausrichtung und typische Wählerschaft.

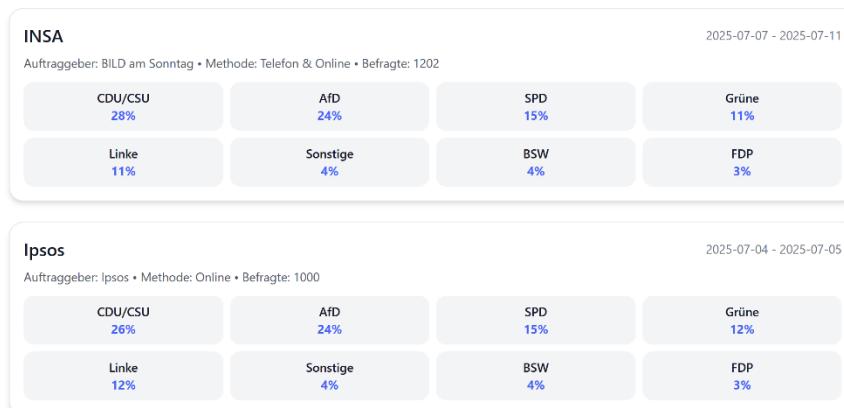
Die Komponente **Wahlbeteiligung & Demografie** zeigt anschaulich, wie sich die Wahlbeteiligung über die Jahre hinweg entwickelt hat und welche Altersgruppen die Wählerschaft dominieren.



Diese Ansicht bietet eine kompakte Übersicht über die Wahlbeteiligung im Zeitverlauf sowie demografische Merkmale wie Altersverteilung, Geschlechterverhältnis, Bildungsstand und berufliche Situation – zentrale Faktoren zur Analyse des Wählerprofils und gesellschaftlicher Trends.

Die Komponente **Wahlumfragen** bietet eine strukturierte Übersicht aktueller Wahlumfragen, sortiert nach Instituten. Neben Prozentwerten für die wichtigsten Parteien werden auch Befragungsmethoden, Stichprobengröße sowie der Erhebungszeitraum angegeben. Die kompakten Karten ermöglichen einen schnellen Vergleich unterschiedlicher Institute und Umfrageergebnisse.

Aktuelle Umfragen zur Bundestagswahl



Die Komponente **News** zeigt politische und gesellschaftlich relevante Nachrichten an, die häufig im Zusammenhang mit Wahlen, Parteien oder demokratischen Entwicklungen stehen. Die Beiträge sind mit Titel, Kurzbeschreibung, Bild und Veröffentlichungsdatum versehen. Die Komponente bietet einen schnellen Einblick in aktuelle Diskussionen und dient der politischen

Meinungsbildung auf Grundlage verlässlicher Quellen. Per Klick auf die Nachricht wird man direkt zur Quelle geleitet.



Mehr erfahren informiert Nutzer:innen über die Zielsetzung der Anwendung, ihre Funktionen sowie über potenzielle Anwendungsbereiche. Diese Seite sollte insbesondere neuen Nutzer:innen als Orientierung dienen, um die Navigation und das Ziel der Plattform schnell zu erfassen.

Mehr erfahren

Diese Webanwendung bietet eine moderne, interaktive Visualisierung von Wahlergebnissen in Deutschland. Sie richtet sich an Bürger:innen, Forschende, Medien und Bildungseinrichtungen, die sich für politische Entwicklungen auf Landes- und Bundesebene interessieren.

Einführung

Die Plattform stellt verschiedene Werkzeuge bereit: eine interaktive Karte zur Erkundung der Wahlergebnisse auf Landesebene, ein Vergleichsmodul zwischen Bundesländern und ein Prognosetool auf Basis vergangener Trends. Ergänzend werden demografische und sozioökonomische Informationen bereitgestellt, um die Ergebnisse besser einordnen zu können.

Funktionen

- **Interaktive Karte:** Visualisierung der Zweitstimmenverteilungen pro Bundesland und Jahr
- **Wahlvergleich:** Vergleich von Wahlergebnissen zwischen bis zu zwei Bundesländern – auch jahrgangsübergreifend
- **Demografische Analyse:** Kombination von Wahlergebnissen mit Bevölkerungsstruktur, Altersdurchschnitt, Arbeitslosenquote u.a.
- **Prognose (Beta):** Einfache Schätzung möglicher Wahlausgänge auf Basis linearer Trendmodelle

Zielgruppen

Die Anwendung richtet sich an alle, die sich für politische Entwicklungen interessieren – insbesondere:

- Politisch interessierte Bürger:innen
- Studierende und Forschende der Politikwissenschaft
- Lehrkräfte und Bildungseinrichtungen
- Journalist:innen und Redaktionen

Entwicklerdokumentation

Voraussetzungen

- Python: Version 3.9 oder höher
- Docker: Docker Desktop muss installiert und lauffähig sein
- Visual Studio Code: Empfohlen für die lokale Entwicklung

Projektstruktur

Die Applikation besteht aus einem Frontend (React) und einem Backend (FastAPI). Beide Komponenten werden über Docker Compose orchestriert.

Setup (Docker)

1. Docker Desktop installieren
Falls noch nicht geschehen, Docker Desktop installieren und starten.
2. Docker container bauen
Im Projektverzeichnis das Terminal in Visual Studio Code öffnen und folgenden Befehl ausführen:
docker compose build --no-cache
Hinweis: Falls Probleme mit node_modules auftreten, den Build-Cache mit --no-cache explizit leeren.
3. Docker container starten:
docker compose up
Dadurch werden alle benötigten Container (Frontend, Backend, ggf. Datenbank) erstellt und gestartet.

Startseite aufrufen

Sobald alle Container erfolgreich laufen, ist die Anwendung über die Startseite erreichbar:

http://localhost:3000

(Falls abweichend konfiguriert: Portnummer ggf. im docker-compose.yml prüfen)

Änderungen sichtbar machen

Bei Änderungen am Code (z. B. UI oder Backend-Logik), ist ein Neustart der Container notwendig:

docker compose restart

Neue Komponente hinzufügen

In diesem Abschnitt wird beschrieben, wie eine neue React-Komponente zur Darstellung spezifischer Wahlergebnisse (z. B. für einen Wahlbezirk oder Kandidaten) zur Webanwendung hinzugefügt wird.

1. Komponente erstellen

Erstelle eine neue Datei im Komponentenverzeichnis (z. B. `src/components`). Inhalt dieser Datei ist zum Beispiel die neue Komponente `Comparison.tsx`:

```
⚙️ Comparison.tsx 2 ✘
frontend > src > components > ⚙️ Comparison.tsx > 📁 Vergleich
1   import React, { useEffect, useState } from "react";
2   import Select from "react-select";
3   import { bundeslandInfos } from "../data/bundeslandInfos";
4   import { Bundesland } from "./Map"; // falls in Map.tsx exportiert
5   import { farben } from "./Map"; // falls Farben ausgelagert wurden
6   import { bundeslaender } from "./Map";
7
8
9   const parteien = ["CDU", "SPD", "GRÜNE", "FDP", "AfD", "Die Linke"];
10
11  const Vergleich = () => [
12    const [selectedStates, setSelectedStates] = useState<Bundesland[]>([]);
13    const [results, setResults] = useState<Record<string, Record<string, number> | null>>({});

14
15  useEffect(() => {
16    selectedStates.forEach((state) => {
17      if (results[state.value]) return;
18
19      fetch(`http://localhost:8000/election-results?state=${encodeURIComponent(state.value)}`)
20        .then((res) => res.json())
21        .then((data) => {
22          setResults((prev) => ({ ...prev, [state.value]: data }));
23        })
24        .catch(() => {
25          setResults((prev) => ({ ...prev, [state.value]: null }));
26        });
27    });
28  }, [selectedStates]);
29
30  return (
31    <div>
32      <h1>Vergleich</h1>
33      <Select
34        options={parteien}
35        onChange={(value) => setSelectedStates(value)}
36      />
37      <table>
38        <thead>
39          <tr>
40            <th>Partei</th>
41            <th>Ergebnis</th>
42          </tr>
43        </thead>
44        <tbody>
45          {Object.entries(results).map(([key, value]) => {
46            return (
47              <tr>
48                <td>{key}</td>
49                <td>{value}</td>
50              </tr>
51            );
52          })}
53        </tbody>
54      </table>
55    </div>
56  );
57}
```

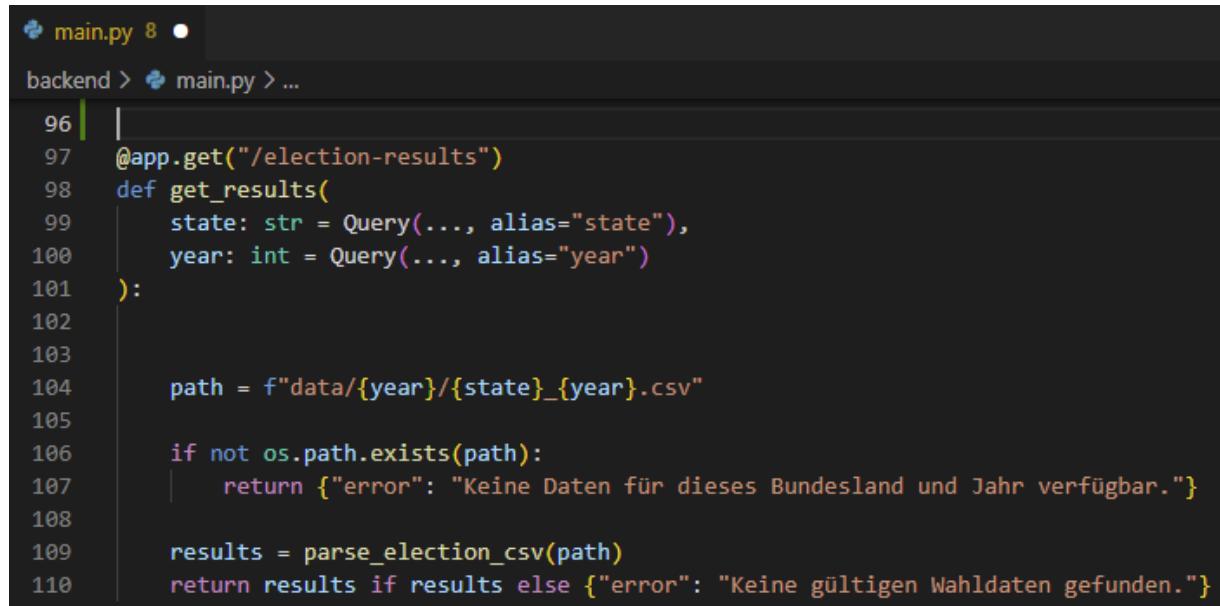
2. Komponente einbinden

Importiere und verwende die Komponente in einem übergeordneten Element, z. B. in *App.tsx*:

```
⚙️ App.tsx 1 ●
frontend > src > ⚙️ App.tsx > ...
1
2  import React from "react";
3  import { Routes, Route } from "react-router-dom";
4  import Home from "./components/Home";
5  import Navbar from "./components/Navbar";
6  import MapPage from "./components/Map";
7  import MehrErfahren from "./components/LearnMore";
8  import ComparisonPage from "./components/Comparison";
9  import PredictionPage from "./components/Prediction";
10 import PartyAnalysisPage from "./components/PartyAnalysis";
11 import SurveysPage from "./components/SurveysPage";
12 import NewsPage from "./components/NewsPage";
13 import ParticipationAndDemographicsPage from "./components/ParticipationAndDemographics";
```

Neue Schnittstelle hinzufügen

Für das hinzufügen einer neuen Schnittstelle wird zunächst ein API Endpoint von FastAPI benötigt. In dem Beispiel den */election-results* Endpoint (Datei *main.py*):



```

main.py 8
backend > main.py > ...
96 |
97     @app.get("/election-results")
98     def get_results(
99         state: str = Query(..., alias="state"),
100        year: int = Query(..., alias="year")
101    ):
102
103
104     path = f"data/{year}/{state}_{year}.csv"
105
106     if not os.path.exists(path):
107         return {"error": "Keine Daten für dieses Bundesland und Jahr verfügbar."}
108
109     results = parse_election_csv(path)
110     return results if results else {"error": "Keine gültigen Wahldaten gefunden."}

```

Dann kann im Frontend über den Endpunkt auf die Daten zugegriffen werden (Datei *Map.tsx*):



```

const MapPage = () => {
  const [selected, setSelected] = useState<Bundesland | null>(null);
  const [geoData, setGeoData] = useState<FeatureCollection | null>(null);
  const [electionResults, setElectionResults] = useState<Record<string, number> | { error: string } | null>(null);
  const [year, setYear] = useState<number>(2025);

  useEffect(() => {
    fetch("/data/bundeslaender.geo.json")
      .then((res) => res.json())
      .then((data) => setGeoData(data));
  }, []);

  useEffect(() => {
    if (!selected) return;
    fetch(`http://localhost:8000/election-results?state=${encodeURIComponent(selected.value)}&year=${year}`)
      .then((res) => res.json())
      .then((data) => setElectionResults(data))
      .catch(() => setElectionResults(null));
  }, [selected, year]);
}

```

In diesem Beispiel werden die Daten aus dem Backend gezogen - Funktion *fetch* ([Daten aus dem Backend]). Dann auf die erzeugte Komponente *ElectionResult* mit der Funktion *setElectionResult* ([Daten aus dem Backen]) weitergereicht.

Wahldaten

Die Daten zu den Bundestagswahlergebnissen stammen von der Webseite der Bundeswahlleiterin (https://bundeswahlleiterin.de/dam/jcr/ce2d2b6a-f211-4355-8eea-355c98cd4e47/btw_kerg.zip). Dort stehen CSV-Dateien für alle bisherigen Bundestagswahlen zur Verfügung. Seit der Wahl im Jahr 2017 wird zusätzlich ein neues Datenformat angeboten, das parallel zum bisherigen (alten) Format existiert. In diesem Projekt wird zur Vereinfachung ausschließlich das ältere Format verwendet. Jede Datei enthält für eine Bundestagswahl die Anzahl der Erst- und Zweitstimmen aller Parteien, die mindestens eine Stimme erhalten haben, auf Wahlkreisebene. Darüber hinaus sind Angaben zur Zahl der Wahlberechtigten, der tatsächlich Wählenden sowie der gültigen und ungültigen Stimmen enthalten.

Die Formatierung der Dateien ist jedoch nicht durchgehend einheitlich: So existieren beispielsweise für die Wahl 1949 keine Zweitstimmen; in den Jahren 1972 bis 1987 werden Parteinamen teilweise mit Punkten oder Leerzeichen abgekürzt (z. B. „F.D.P.“ statt „FDP“); und in der Datei zur Wahl 2021 fehlen die ersten drei Zeilen vollständig. Aufgrund dieser Inkonsistenzen und weil für das Projekt lediglich die Ergebnisse auf Bundeslandebene relevant sind, werden die Wahldaten mithilfe eines eigenen Python-Skripts extrahiert und in ein standardisiertes Format überführt.

Dabei wird für jede Wahl für jedes Bundesland eine eigene Datei erstellt. Diese enthält die Anzahl Erst- und Zweitstimmen für alle Parteien, sowie der Anteil an allen Stimmen in Prozent. Die folgende Abbildung zeigt die Datei für die Wahlergebnisse in Baden-Württemberg 2025:

Party	Erststimmen (tot)	Erststimmen (%)	Zweitstimmen (tot)	Zweitstimmen (%)
Sozialdemokratische Partei Deutschlands	1002327.0	15.8	898778.0	14.2
Christlich Demokratische Union Deutschlands	2280707.0	36.0	2006866.0	31.6
BÜNDNIS 90/DIE GRÜNEN	887608.0	14.0	865738.0	13.6
Freie Demokratische Partei	271794.0	4.3	357539.0	5.6
Alternative für Deutschland	1231221.0	19.4	1256430.0	19.8
Christlich-Soziale Union in Bayern e.V.				
Die Linke	360037.0	5.7	429484.0	6.8
FREIE WÄHLER	163869.0	2.6	86311.0	1.4
PARTEI MENSCH UMWELT TIERSCHUTZ	10548.0	0.2	55906.0	0.9
Basisdemokratische Partei Deutschland	17282.0	0.3	21845.0	0.3
Piratenpartei Deutschland				
Volt Deutschland	69168.0	1.1	49025.0	0.8
Ökologisch-Demokratische Partei - Die Naturschutzpartei	1798.0	0.0	12258.0	0.2
Südschleswiger Wählerverband				
Partei für Verjüngungsforschung				
Bündnis C - Christen für Deutschland	2021.0	0.0	11768.0	0.2
Bayernpartei				
Marxistisch-Leninistische Partei Deutschlands	3574.0	0.1	2379.0	0.0
Menschliche Welt - für das Wohl und Glücklichsein aller				
Partei des Fortschritts				
Bürgerrechtsbewegung Solidarität				
BÜNDNIS DEUTSCHLAND	4821.0	0.1	7411.0	0.1
Bündnis Sahra Wagenknecht - Vernunft und Gerechtigkeit			260219.0	4.1
MERA25 - Gemeinsam für Europäische Unabhängigkeit				
WerteUnion				
Übrige	5646.0	0.1		

Fazit

Mit der Wahlergebnisse-Web-App wurde eine moderne und funktionale Anwendung geschaffen, die wesentliche Anforderungen an eine datenbasierte Visualisierungsplattform erfüllt. Besonders hervorzuheben ist die hohe Abdeckung der gesetzten Muss-Kriterien: Die Anwendung reagiert performant, liefert korrekte und aktuelle Daten und stellt diese in einer übersichtlichen, responsiven Benutzeroberfläche dar. Auch zentrale Kann-Kriterien wie interaktive Karten, dynamische Diagramme und einfache Prognosemodelle wurden erfolgreich umgesetzt und werten die Nutzererfahrung deutlich auf.

Trotz des Verzichts auf Funktionen wie Personalisierung oder Exportoptionen zeigt die Architektur des Projekts, dass eine modulare Erweiterung jederzeit möglich wäre. Damit ist die Grundlage für ein nachhaltiges, ausbaufähiges System gelegt.

Insgesamt wurde ein solides und praxisnahes Ergebnis erreicht, das sowohl funktional als auch gestalterisch überzeugt. Die Arbeit an diesem Projekt hat nicht nur unsere technischen Fähigkeiten gestärkt, sondern auch verdeutlicht, wie wichtig strukturierte Planung, saubere Schnittstellen und eine klare Kommunikation innerhalb eines Softwareprojekts sind – Erfahrungen, die über das Projekt hinaus von großem Wert bleiben.

Literaturverzeichnis

<https://fastapi.tiangolo.com/>

<https://react.dev/>

<https://vite.dev/>

<https://tailwindcss.com/>

<https://www.bundeswahlleiterin.de/>

<https://dawum.de/>

<https://newsapi.org/>