Projektgruppe INFRASense

Projektzeitraum: 1. Oktober 2022 bis 30. September 2023

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Die Qualität von Radwegen ist von grundelgendem Intresse aller Radfahrer. Um diese zu verbessern, muss erst festgestellt werden, wo dies notwendig ist. In diesem Sinne ist es Ziel der Projektgruppe, Radwege zu vergleichend zu bewerten. Dies geschieht in Form einer Webseite. Dazu Erkennen wir Bilder auf Schäden, sammeln Daten aus weiteren Datenquellen und stellen die gewonnenen Ergebnisse auf unserer Webseite dar.

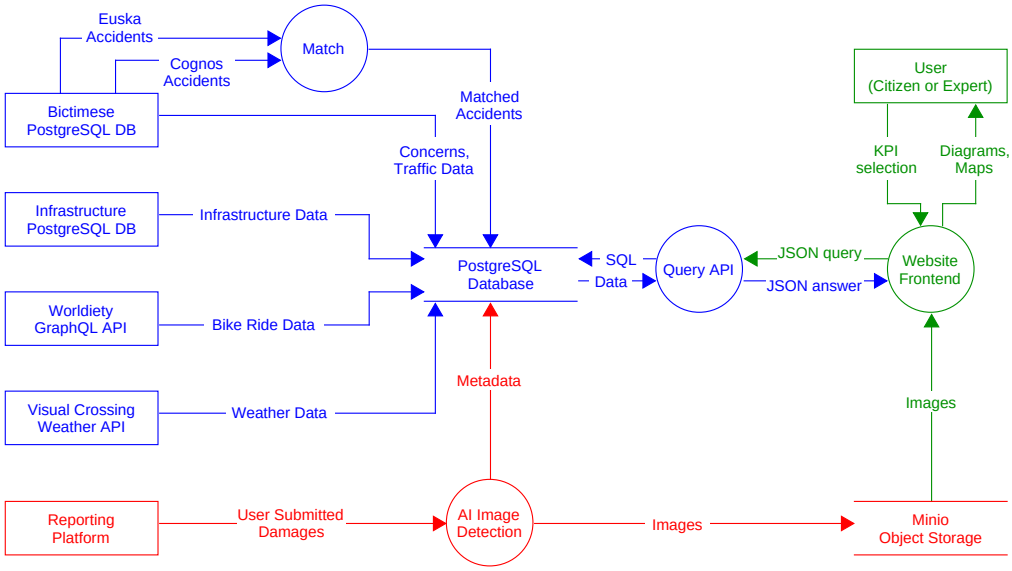

Abbildung 1: Datenflussdiagramm

Um die Effizienz unserer Arbeit zu erhöhen, haben wir uns in drei Gruppen aufgeteilt. In Abbildung 1 ist ein Datenflussdiagramm nach DeMarco dargestellt, in dem die Aufgabenbereiche der verschiedenen Gruppen ihre eigenen Farben haben. In Blau ist der Arbeitsbereich der Data-Gruppe, die Daten von den Datenquellen abfragt, aggregiert, in unserer Datenbank speichert und über eine API bereitstellt. Die Arbeitsgruppe KI (rot) erstellte eine Meldeplattform und ein KI-Modell, um Schäden in dort eingereichten Bildern zu klassifizieren. Die Metadaten werden in der Datenbank abgelegt, die Bilder im Object Storage Minio. Die Arbeitsgruppe Dashboard (grün) hat die Webseite erstellt. Dazu gehören die Entwicklung der verschiedenen Ansichten und Diagramme, sowie der Auswahlmöglichkeiten für die Nutzer.

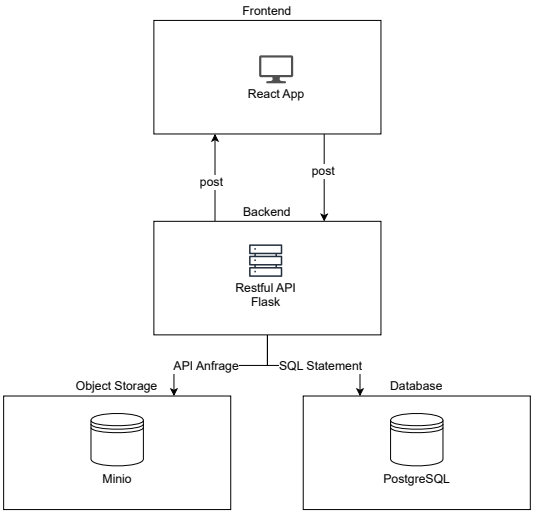

Abbildung 2: Architekturdiagramm der KI-Gruppe

2 KI Gruppe

Die Entwicklung unseres Anwendungsfalls wurde von einem klaren Ziel geleitet: Die Schaffung einer sinnvollen Anwendung, die einen spürbaren Mehrwert für die Fahrradmobilität bietet. Wir haben uns gefragt, wie wir Bilddaten effektiv nutzen können, insbesondere in Bezug auf die Erfassung und Verarbeitung von Informationen über Radwegschäden. Schließlich sind Schäden auf Radwegen nicht nur ein Ärgernis, sondern sie beeinflussen auch die Qualität unserer Fahrradinfrastruktur erheblich. Oftmals fehlt den Städten und Kommunen jedoch ein flächendeckender Überblick über die Lage der Schäden und welche davon dringend repariert werden müssen. Genau diesem Problem möchten wir entgegenwirken.

In diesem Kontext spielt unsere selbst entwickelte Künstliche Intelligenz eine entscheidende Rolle. Die von den Bürgern gemeldeten Schäden sollen durch unser KI-Modell analysiert und klassifiziert werden. Diese automatisierte Verarbeitung der Meldungen gibt unserer KI einen sinnvollen und praktischen Einsatzzweck. Die Bürgerbeteiligung löst gleichzeitig das Problem der Datengrundlage. Zusätzlich profitieren wir von einem kontinuierlichen Strom von Bildern, die zur Verbesserung unseres KI-Modells beitragen. Abbildung 2 zeigt die Architektur des entwickelten Systems. Es basiert auf einer 3-Schichten-Architektur, um das KI-Modell als eigenen Layer ins Backend zu legen. Zusätzlich werden die gemeldeten Bilder abgespeichert in einem Object Storage und einer Datenbank.

Das KI-Modell, welches mithilfe einer Flask-App mit dem Frontend und der Datenbank kommuniziert, wurde über mehrere Iterationen entwickelt. Als Vorgehensmodell wurde CRISP-DM verwendet. Insgesamt wurden sechs Iterationen durchlaufen, in denen jeweils verschiedene Ansätze verfolgt wurden, um die Ergebnisse des erstellten Modells zu verbessern. Dabei ist es wichtig zu betonen, dass der Datensatz von Null aus erstellt und selbst gelabelt wurde anhand eines eigens erstellten Ebenenkonzept. Im Laufe des Projektes hat man sich auf die Klassen Schlagloch und Riss fokussiert, da andere Klassen wie Absenkung und Erhebung nicht geeignet sind, um durch Computer Vision klassifiziert zu werden. Als Ansatz wurde zunächst Object Detection mit YOLO (You Only Look Once) betrachtet und dies im weiteren Verlauf noch erweitert um eine zusätzliche Klassifikation mit efficienet. Die Metriken der Durchläufe finden sich in Tabelle 1 und Tabelle 2.

Tabelle 1: Metriken der YOLO-Modelle

Model	**Precision**	**Recall**	**mAP50**
Iteration 4D	0.75357	0.59375	0.65509
Iteration	0.96457	0.86105	0.91608
Iteration 5D	0.96128	0.84733	0.91604
Iteration	0.67665	0.42463	0.49554
Iteration 5D	0.60201	0.49097	0.51116

Tabelle 2: Metriken der efficienet-Modelle

Model	**Precision**	**Recall**
Iteration 4	0.96938776	0.87962963
Iteration 5	0.950735321	0.99126467

Die Ergebnisse lassen darauf schließen, dass vor allem das Modell aus Iteration 5 verwendet werden kann. Die Ergebnisse der Klassifikation waren stets gut. Zusätzlich zu den erkannten Klassen wurden die Koordinaten der Bounding Boxen sowie weitere Attribute in verschiedenen Tabellen abgelegt. Eine physische Kopie des Bildes wird auf dem Object Storage Minio abgelegt. Ein Beispielbild, auf dem das Modell angewendet wird, ist in Abbildung zu sehen.

Das Frontend wurde entwickelt, um dem entwickelten KI-Modell einen Kontext zu geben. Dabei ist wichtig zu betonen, dass das Frontend eher als Prototyp anzusehen ist, der die nötigen Funktionalitäten für ein Meldeportal liefert. Es soll ein mögliches Meldeportal zeigen und kann bei Bedarf ausgetauscht werden, sodass man das KI-Modell durch die unterschiedlichen Endpunkte auch in weitere, bestehenden Lösun-

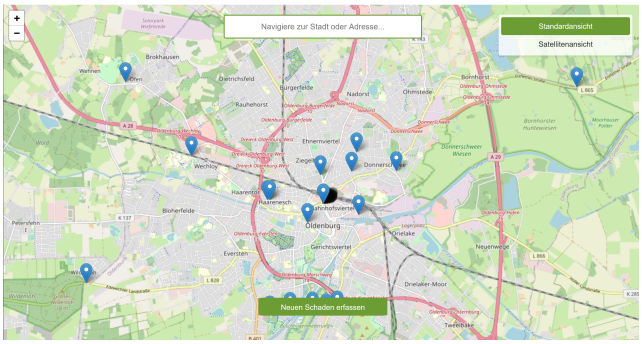

Abbildung 3: Meldeplattform

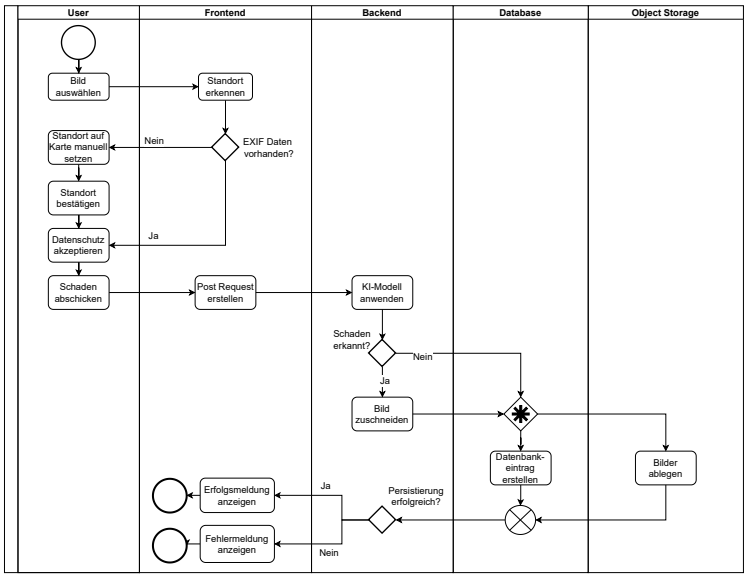

Abbildung 4: Aktivitätsdiagramm zum Usecase „Schaden melden“

gen implementieren und nutzen kann. Es wurden die vier Haupt-Usecases Schaden melden, Schaden klassifizieren, Schaden anzeigen und Standort bestimmen umgesetzt. Ein Beispielbild der Meldeplattform findet sich in Abbildung Abbildung 3. Das Ablaufdiagramm zum Melden eines Schadens findet man in Abbildung Abbildung 4.

3 Data

Die Arbeitsgruppe Data beschäftigte sich mit Anbindung der Datenquellen, Verarbeitung der erhaltenen Daten und bereitstellung der Daten für die Webseite. In

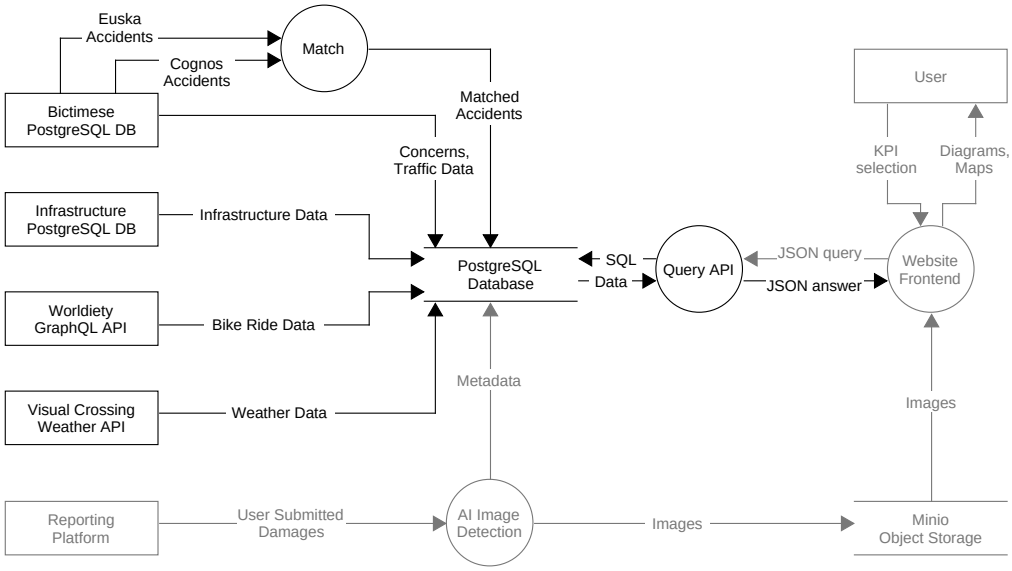

Abbildung 5: Datenflussdiagramm mit Fokus auf Data-Gruppe

Abbildung 5 ist ein Datenflussdiagramm nach DeMarco dargestellt, in dem die Arbeitsanteile der anderen Arbeitsgruppen ausgegraut sind. Einen Überblick über die verschiedenen Datenquellen liefert Unterabschnitt 3.2. Der Aufbau der Datenbank ist in Unterabschnitt 3.1 beschrieben. Die Architektur der API wird in Unterabschnitt 3.3 dargestellt.

3.1 Data Warehouse

Ein wesentlicher Bestandteil der Arbeitsgruppe Data ist der Aufbau eines Data Warehouses zur Speicherung der bereitgestellten Daten. In diesem Kapitel wird daher unser Data Warehouse Konzept vorgestellt.

In unserem Data Warehouse liegen Daten in einem hybriden Data-Warehouse-Konzept vor, was bedeutet, dass unterschiedliche Daten je nach Anforderungen normalisiert sowie denormalisiert und aggregiert vorliegen. Dabei enthält das zentrale Enterprise Data Warehouse (EDW) nach dem Inmon-Prinzip die vollständigen, normalisierten Daten. Dieses EDW dient als Rohdatenbasis für alle Data Marts. Die Data Marts werden nach dem Kimball-Modell erstellt und bieten den Benuztern den schnellen Zugriff auf spezifische Daten in einem denormalisierten Format. Dies erleichtert den Benutzern den Zugriff auf relevante Informationen.

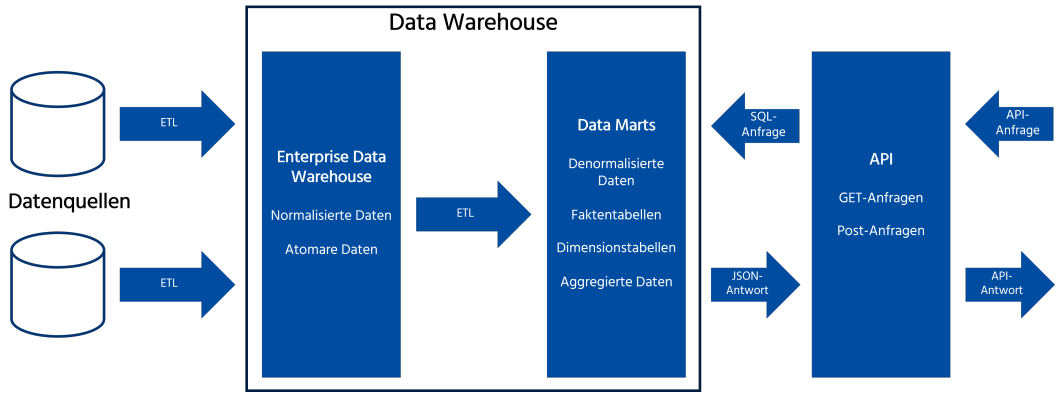

Abbildung 6: Data Warehouse Architektur in Anlehnung an Kimball[KR13]

Unter den normalisierten Daten sind bspw. die Infrastrukturdaten, Fahrtdaten, Verkehrsdaten, Bilddaten und Wetterdaten zu finden. Die denormalisierten und aggregierten Daten umfassen die aggregierten Fahrtdaten aus dem BIQEmonitor, aggregierte Unfalldaten, Meldedaten sowie aggregierte Spitzenwerte der unterschiedlichen Verkehrsdaten.

3.2 Daten

Uns wurde eine Reihe von verschiedenen Datenquellen von den unterschiedlichen Projektpartnern und Stakeholdern bereitgestellt. Diese haben wir in unserer Datenbank zusammengetragen und für die Abfragen unserer API aggregiert.

Verkehrszählдaten enthalten primär die Anzahl von Fahrzeugen, die pro Viertelstunde an verschiedenen Messstationen gemessen wurden. Einige Messtationen haben detailliertere Informationen für bestimmte Fahrzeugtypen oder Richtungen, oder zusätzlicher Informationen zu Auslastungen und Geschwinddigkeit.

Bürgermeldungen enthalten Eventbezogene Meldungen von Bürgern. Die Meldungen enthalten Zeit der Erstellung und Bearbeitung, Koordinaten und Beschreibung, einschließlich Kategorien für Meldungen von Bürgern über Vorfälle im Verkehrsbereich.

Unfalldaten liegen uns aus zwei unterschiedlichen Systemen, EUSKA und COGNOS, vor. Allerdings gibt es für jedes der Systeme Unfälle, die im anderen nicht

vorliegen. Dabei zählen wir zwei Unfälle als zusammengehörend, wenn sich höchstens ein Feld unterscheided oder eine Person in der Unfallschwere verschoben ist.

Infrastruktur-Daten lagen ehemals in einer weiteren Datenbank, die später in unsere Hauptdatenbank überführt wurde. Die enthaltenen Daten sind Busnetz- und Straßendaten für die Städte Oldenburg und Osnabrück.

Fahrradfahrtdaten wurden uns Vom Projektpartner Worldiety bereitgestellt. Diese sind auch über die BIQEmonitor-Website erreichbar und wurden im Rahmen des Projektes InfraSense von Bürgern erhoben. Uns wurden sowohl die Rohdaten der Fahrten, als auch lokal aggregierte Daten bereitgestellt.

Wetterdaten kauft die Abteilung VLBA wir von VisualCrossing. Dies ermöglicht es, diese anderen Datenpunkte nach ihrem begleitenden Wetter zu filtern oder zu gruppieren.

Bilder mit erkannten Straßenschäden legt die KI-Gruppe in unserer Datenbank ab. Die Bilder werden im Object-Storage Minio abgelegt, Metadaten in unserer Datenbank in den Tabellen image und annotations.

3.3 Schnittstelle

Wir stellen mit Python und FastAPI eine Schnittstelle bereit. Einige unserer Endpunkte sind über GET-reqests erreichbar, die meisten jedoch über POST-requests. Als Inhalt enthalten diese eine Anfrage im JSON-Format, wodurch nicht für jeden KPI ein eigener Endpunkt notwendig ist und die Anfragen flexibler gestellt werden können. Darüber hinaus ist es uns bei diesen Anfragen möglich, SQL-Injection zu verhindern und die Datenbank zu abstrahieren.

4 Dashboard

Wir haben die Entwicklung der Website in zwei Phasen entwickelt: Zuerst haben wir uns auf die Entwicklung der Online-Präsenz konzentriert, um Interessenten umfassende Informationen zu bieten und den Kontakt zu erleichtern. In der zweiten Phase haben wir das interaktive Dashboard entwickelt, welches den Kern unseres Projekts darstellt und zahlreiche komplexe Funktionen und Visualisierungen enthält.

4.1 Überblick über die Entwicklung

Überblick Technologien Ein visueller Überblick über unsere Technologien wird in Abbildung 7 in einem Schaubild präsentiert.

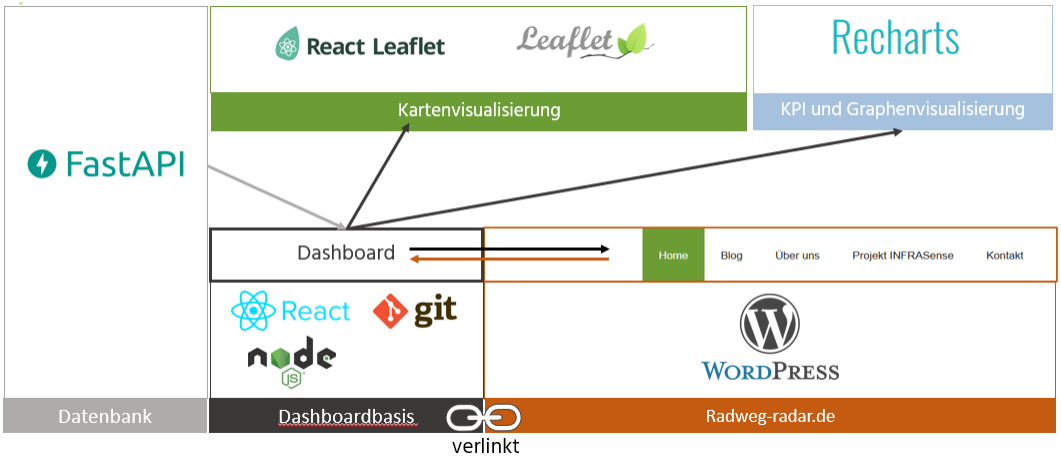

Abbildung 7: Überblick über die verwendeten Technologien [Bil23, RL23, Wik23b, Wik23a, Wik23c, See23, Fas23]

Trennung von Website und Dashbaord Ursprünglich wurde die Website in React entwickelt, was sich als zeitintensiv erwies, da für Blogbeiträge, auch nach Ende der Projekgruppe, einzelne React-Komponenten erstellt werden mussten. Dies hätte die Wartung erschwert. Daher haben wir uns entschieden, die Online-Präsenz auf WordPress umzustellen, jedoch das Dashboard weiter in React zu entwickeln. Ein weiterer Vorteil dieser Trennung besteht darin, dass nach Abschluss der Projektgruppe die Website weiterhin gewartet oder abgeschaltet werden kann. Gleichzeitig kann das Dashboard von einem anderen Team weiterentwickelt werden, was zu einer erhöhten Flexibilität und Nachhaltigkeit führt.

Hosting und Domains Die Website ist über die Domain `https://radweg-radar.de/` erreichbar. Für das Hosting des Dashboards und die dahinter liegende React-Applikation wurde eine virtuelle Maschine an der Uni Oldenburg eingerichtet. Auf dieser läuft die React-Applikation innerhalb eines Apache Webservers. Über `https://dashboard.radweg-radar.de/` ist das Dashboard weltweit erreichbar.

Kennzahlen für das Dashboard Für die Darstellung auf dem Dashboard ergeben sich 30 Kennzahlen. Die Oberkategorien lauten Fahrrad-Verkehrsaufkommen, Bürgermeldungen, Unfalldaten und Katendaten. Darüber hinaus wurden die Katendaten mit weiteren Informationen, z. B. Buslinien und Haltestellen, ergänzt.

4.2 Umsetzung des Dashboards

Das Dashboard ist in Boxen unterteilt, die als separate Komponenten integriert sind. Dieses Layout ist vollständig responsiv und passt sich der Bildschirmgröße an, was es auch für die mobile Nutzung optimiert. Ein möglicher Auschnitt des Dashboards ist in Abbildung 8 zu sehen.

Diagramme Als Diagramm-Framework wurde Recharts aufgrund seiner Modernität und Anpassbarkeit ausgewählt. Es bietet extreme Flexibilität durch Code-Anpassungen. Um ein Diagramm zu erstellen, müssen Benutzer verschiedene Informationen festlegen, einschließlich Stadt, Kennzahl, Daten-Gruppierung (z. B. nach Jahr, Monat, Tag, usw.), und den gewünschten Zeitraum. Dafür haben wir eine benutzerfreundliche Kennzahl-Auswahlmaske entwickelt, die den Auswahlprozess schrittweise führt. Bei sehr detailreichen Diagrammen, oder Diagrammen, die viele Daten enthalten, ist es hilfreich, sich diese in einer vergrößerten Form anzeigen zu lassen.

Kartendaten Über Filter kann die angezeigte Menge an Daten eingeschränkt werden. Es sind unter anderem Straßeneinfärbungen, Heatmaps, Kreuzungseinfärbungen, Unfalldaten, Meldungen, Buslinien und Haltestellen verfügbar. Alle Daten lassen sich sehr detailliert filtern, z. B. Unfälle nach Schadenshöhe, oder ob Alkohol beim Verursacher nachgewiesen werden konnte. Mit Hilfe der Kartenansicht können sehr komplexe Karten erstellt werden. So kann z. B. das Fahrradverkehrsaufkommen als Heatmap, die Fahrradzählstellen, Unfalldaten mit Kreismarker je nach Schwere des Unfalls und Bürgermeldungen durch verschiedene Icons angezeigt werden. Dann kann analysiert werden, ob ein Zusammenhang zwischen diesen Themen gefunden werden kann. Dabei hilft auch die Funktion, dass beim Klick auf diese angezeigten Objekte weitere Informationen als Pop-Up erscheinen.

Straßen-Score Ein Straßenscore fasst verschiedene Kennzahlen zusamen. Es gibt die Möglichkeit, zwei Straßen mit Hilfe eines Netz-Diagramms nach Kriterien und

dem Score zu vergleichen, während eine Tabellenvariante die Top und Flop fünf jeder Kategorie ausgibt.

4.3 Erweiterungsmöglichkeiten und Ausblick

Das Dashboard bietet Raum für zusätzliche Funktionen. Dies könnte die Erweiterung von Nutzerprofilen für personalisierte Ansichten, sowie die Integration von Exportmöglichkeiten für Diagramme, Tabellen und Bilder umfassen. Unsere flexible Architektur erlaubt auch die Integration verschiedener Visualisierungsoptionen. Aktuell verwenden wir Recharts-Diagramme, aber es besteht die Möglichkeit, andere Diagramm-Frameworks einzubinden. Neue Kennzahlen können problemlos ins System integriert werden, indem sie auf dem bestehenden Code aufbauen.

Abbildung 8: Das Box-Layout des Dashboards mit Diagrammansicht, Kartenansicht und Straßenscore

Literatur

[Bil23] Bilginc. React logo. `https://bilginc.com/editorFiles/3147aa77.png`, 2023. zuletzt aufgerufen am 25.09.2023.

[Fas23] FastAPI. Fastapi logo. `https://fastapi.tiangolo.com/img/logo-margin/logo-teal.png`, 2023. zuletzt aufgerufen am 25.09.2023.

[KR13] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. Wiley, Indianapolis, IN, 3 edition, 2013.

[RL23] React-Leaflet. React-leaflet logo. `https://react-leaflet.js.org/img/logo-title.svg`, 2023. zuletzt aufgerufen am 25.09.2023.

[See23] Seeklogo. Wordpress logo. `https://seeklogo.com/images/W/wordpress-logo-FC322694E8-seeklogo.com.png`, 2023. zuletzt aufgerufen am 25.09.2023.

[Wik23a] Wikipedia. Git logo. `https://de.m.wikipedia.org/wiki/Datei:Git-logo.svg`, 2023. zuletzt aufgerufen am 25.09.2023.

[Wik23b] Wikipedia. Leaflet logo. `https://de.m.wikipedia.org/wiki/Datei:Leaflet_logo.svg`, 2023. zuletzt aufgerufen am 25.09.2023.

[Wik23c] Wikipedia. Node.js logo. `https://de.wikipedia.org/wiki/Node.js#/media/Datei:Node.js_logo.svg`, 2023. zuletzt aufgerufen am 25.09.2023.