JOINT TRANSPORTATION
RESEARCH PROGRAM
INDIANA DEPARTMENT OF TRANSPORTATION
AND PURDUE UNIVERSITY

Enhanced Traffic Pattern Knowledge Abstraction and Presentation from 511 Database

Jiannan Ding, Liya Koshy, Tanay Maheshwari, Abin Mathew, Jue Zhou, Stanley Yung-Ping Chien, Tianyi Li, Chengcheng Tao, and Yaobin Chen

SPR-4937 • Report Number: FHWA/IN/JTRP-2025/43 • DOI: 10.5703/1288284318608

RECOMMENDED CITATION

Ding, J., Koshy, L., Maheshwari, T., Mathew, A., Zhou, J., Chien, S. Y.-P., Li, T., Tao, C., & Chen, Y. (2025). *Enhanced traffic pattern knowledge abstraction and presentation from 511 database* (Joint Transportation Research Program Publication No. FHWA/IN/JTRP-2025/43). West Lafayette, IN: Purdue University. https://doi.org/10.5703/1288284318608

AUTHORS

Jiannan Ding
School of Construction Management Technology
Purdue University

Liya Koshy
Elmore Family School of Electrical and Computer Engineering
Purdue University

Tanay Maheshwari
School of Applied and Creative Computing
Purdue University

Abin Mathew
Elmore Family School of Electrical and Computer Engineering
Purdue University

Jue Zhou
Elmore Family School of Electrical and Computer Engineering
Purdue University

Stanley Yung-Ping Chien
Elmore Family School of Electrical and Computer Engineering
Purdue University

Tianyi Li
School of Applied and Creative Computing
Purdue University

Chengcheng Tao, PhD
School of Construction Management Technology
Purdue University

Yaobin Chen
Elmore Family School of Electrical and Computer Engineering
Purdue University
(765) 495-7816
chen62@purdue.edu
Corresponding Author

ACKNOWLEDGMENTS

This work was supported by the Joint Transportation Research Program (JTRP), administered by the Indiana Department of Transportation (INDOT) and Purdue University. The authors would like to thank all members of the INDOT Study Advisory Committee (SAC), including Shuo Li, Project Advisor; Jack Gallagher, Business Owner; Dan McCoy; Max Black; Nathan Shellhamer; Colton Amstutz; and Matthew Paradise for their guidance, advice, and technical support throughout the project period. Special thanks go to Dr. Darcy Bullock, Director of JTRP, and his staff at Purdue University for their administrative support and service.

JOINT TRANSPORTATION RESEARCH PROGRAM

The Joint Transportation Research Program serves as a vehicle for INDOT collaboration with higher education institutions and industry in Indiana to facilitate innovation that results in continuous improvement in the planning, design, construction, operation, management and economic efficiency of the Indiana transportation infrastructure. Learn more at engineering.purdue.edu/JTRP.

Published reports of the Joint Transportation Research Program are available at docs.lib.purdue.edu/jtrp/.

NOTICE

TECHNICAL REPORT DOCUMENTATION PAGE

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
FHWA/IN/JTRP-2025/43		

4. Title and Subtitle	5. Report Date
Enhanced traffic pattern knowledge abstraction and presentation from 511 database	December 31, 2025
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Jiannan Ding (https://orcid.org/0000-0002-1804-7339) Liya Koshy (https://orcid.org/0009-0008-5238-2043) Tanay Maheshwari Abin Mathew (https://orcid.org/0009-0004-1456-7363) Jue Zhou Stanley Yung-Ping Chien (https://orcid.org/0000-0002-1492-9959) Tianyi Li (https://orcid.org/0000-0002-1145-2526) Chengcheng Tao, PhD (https://orcid.org/0000-0003-2708-0912) Yaobin Chen (https://orcid.org/0000-0002-9875-4083)	FHWA/IN/JTRP-2025/43

9. Performing Organization Name and Address	10. Work Unit No.
Joint Transportation Research Program Hall for Discovery and Learning Research (DLR), Suite 204 207 S. Martin Jischke Drive West Lafayette, IN 47907	**11. Contract or Grant No.** SPR-4937

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
Indiana Department of Transportation (SPR) State Office Building 100 North Senate Avenue Indianapolis, IN 46204	Final Report
	14. Sponsoring Agency Code

15. Supplementary Notes

Conducted in cooperation with the U.S. Department of Transportation, Federal Highway Administration.

16. Abstract

The explosive growth in the volume, variety, and velocity of transportation data has placed considerable strain on traditional data infrastructures employed by state departments of transportation (state DOTs). Existing traffic information systems are constrained by fragmented data streams, limited historical archiving, and inadequate interoperability, thereby hindering their utility for comprehensive analysis, causal inference, and predictive modeling. This study addresses these challenges by presenting the design and initial implementation of a traffic data integration platform that unifies multi-source datasets and establishes a foundation for more applications. The platform focuses on data integration and access, incorporating systematic data ingestion pipelines, schema harmonization, quality management, and visualization capabilities. Three consecutive steps are performed: 1) identification of data sources, 2) platform data design and implementation, and 3) user interface design. Data source identification classifies source categories and formalizes collection and quality control procedures. Platform data design and implementation demonstrates database structures, ingestion pipelines, and interface protocols. Finally, user interface design supports dynamic user queries and customizable data display formats. It delivers a scalable and interoperable architecture, laying the groundwork for data-driven transportation systems and intelligent planning.

17. Key Words	18. Distribution Statement
traffic databases, data visualization, data ingestion, user interface design	No restrictions. This document is available through the National Technical Information Service, Springfield, VA 22161.

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	20	

EXECUTIVE SUMMARY

Introduction

511 is the nationwide traveler information service designated by the Federal Communications Commission to provide real-time traffic and travel updates. It integrates data from intelligent transportation systems, such as traffic sensors, cameras, dynamic message signs, and weather stations, into a unified platform for public access. Building on this national framework, the interactive map in the Indiana 511 platform provides information about road conditions, closures, and width/weight restrictions to help motorists and truck drivers drive safely and efficiently. The existing 511 data presentation platform, while providing valuable real-time traffic information, faces growing challenges in handling the explosive increase in data volume, variety, and velocity. Its current architecture is limited by fragmented data streams, insufficient historical archiving, and restricted interoperability which reduces its effectiveness for comprehensive analysis, causal inference, and predictive modeling. This project addresses these challenges by presenting the design and initial implementation of a traffic data integration platform that unifies multisource datasets and establishes a foundation for more applications. The platform focuses on data integration and access, incorporating systematic data ingestion pipelines, schema harmonization, quality management, and visualization capabilities. Three consecutive steps are performed: (1) identification of data sources, (2) platform data design and implementation, and (3) user interface design. Data source identification classifies source categories and formalizes collection and quality control procedures. Platform data design and implementation demonstrates database structures, ingestion pipelines, and interface protocols. Finally, user interface design supports dynamic user queries and customizable data display formats. This project demonstrated a working prototype of an advanced traffic data integration and visualization system. Built around Indiana's 511 database and enriched with complementary datasets, the platform enables transportation agencies to observe, interpret, and act upon spatiotemporal traffic patterns with unprecedented flexibility.

Findings

1. Multisource integration was achieved across traffic and other data feeds, covering incidents, speeds, travel times, rest area availability, truck parking, road weather conditions, and social events.
2. The metadata-driven "What-Where-When" schema proved effective for harmonizing heterogeneous datasets, reducing redundancy, and supporting rapid onboarding of new data sources.
3. The unified interface protocol enabled flexible querying across datasets, supporting spatial and temporal joins for visualization and demonstration.
4. The user interface design successfully guided users through a systematic request-building process, with real-time geographic and temporal scoping, contextual overlays (weather, events), and multiformat visualization options.

Implementation

The system architecture is composed of distinct frontend and backend components that were developed in parallel. The backend includes the continuous ingestion of 511 data and the execution of information retrieval queries. The front end provides a graphical user interface (GUI) that enables end users to formulate and submit these dynamic queries. A data-agnostic communication protocol ensures modular integration between the frontend and backend layers. Finally, the implementation was validated through rigorous testing of both components against predefined functional requirements.

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

In recent years, the transportation sector has been undergoing a fundamental transformation driven by the convergence of advanced data analytics, artificial intelligence (AI), and emerging cyber-physical systems (Bamakan & Banaeian Far, 2025; Spy Pond Partners, LLC et al., 2024; Wang et al., 2024). Within this context, national initiatives such as the U.S. Department of Transportation (USDOT) Smart City Program have emphasized the critical role of data platforms as the digital backbone of future transportation management. These platforms are expected to break down institutional data silos, integrate real-time and historical data, and support both operational management and long-term strategic planning through intelligent algorithms, predictive modeling, and systemwide optimization. Multiple United States smart city pilots are converging on building integrated, modular, API-driven data platforms that unify diverse urban data streams under common governance frameworks.

Efficient and comprehensive traffic management is essential for ensuring smooth transportation operations, enhancing safety, and improving travel reliability (Almukhalfi et al., 2024). In terms of the transportation field, both in operational and research domains, traffic decision making has become deeply reliant on large-scale, heterogeneous data sources, ranging from sensor feeds and weather stations to user-reported events and historical travel patterns. As of 2023, two-thirds of the state departments of transportation (DOTs) in the United States had established some form of agency-level data governance or were actively exploring setting up data governance in their agencies (Spy Pond Partners et al., 2024).

The ideal future traffic data system will serve as the smart brain. Through the data platform, which acts as the foundation, it will integrate resources, promptly process real-time data, scientifically analyze historical data, and effectively combine potential data sources. It enables a transition from isolated, linear traffic management to comprehensive systemwide optimization. While making informed decisions, the system will also predict future trends, contributing to overall optimization.

However, many transportation departments across the United States rely on state-specific systems and other independent platforms to provide traffic information. While these platforms individually offer valuable insights, their disparate nature restricts comprehensive analysis and cohesive data interpretation, hindering real-time integration, adaptive reasoning, and scientific insight. Furthermore, existing traffic information platforms, including the 511 systems operated by individual state DOTs, suffer from silos in data architecture, less unified data retrieval, inconsistent formats, and limited support for causal or predictive analytics. Furthermore, despite the abundance of data, existing systems often struggle with fragmentation, incompatible data formats, inconsistent standards, limited real-time interoperability, and inadequate causal analysis capabilities. These limitations significantly hinder the realization of integrated, adaptive, and predictive traffic operations, especially in the context of growing smart city infrastructures.

To address the above limitations, a detailed survey of existing traffic information websites and platforms was conducted to identify critical shortcomings and gaps, including:

1) **Fragmented data availability:** While real-time information is available, the absence of historical data storage and organized archiving significantly restricts long-term traffic pattern analysis, predictive modeling, and strategic planning capabilities;
2) **Limited actionability of current data:** The platform primarily offers data visualization without incorporating decision-support tools such as predictive analytics, congestion forecasting, or proactive rerouting capabilities;
3) **Incomplete core traffic metrics:** Essential operational variables (e.g., daily traffic volumes, peak-hour congestion, seasonal variations) are either unavailable or not systematically integrated, limiting the ability to fully assess network performance;
4) **Poor integration with external data sources:** There is limited interoperability with auxiliary datasets such as weather conditions, special events, or multimodal transportation systems, preventing dynamic systemwide coordination during disruptive events;
5) **Inadequate handling of irregular and extreme scenarios:** The system lacks mechanisms to account for nonrecurring disruptions such as large public events, emergency evacuations, or severe weather events, which often create atypical traffic patterns;
6) **Backend processing constraints:** The backend system does not provide or only provides basic data processing functions, without advanced analytical tools necessary for in-depth operational diagnostics, scenario analysis, or infrastructure optimization.

To address the gaps identified above, a next-generation traffic data platform shall serve as an integrated data hub and the intelligent brain of traffic management systems. This platform is not merely a data repository; it is a scalable, interoperable framework capable of ingesting real-time data streams, aligning them with historical records, and uncovering actionable patterns through advanced analytical algorithms. The platform is intended to unify disparate datasets, support customizable data presentation, and facilitate scenario-specific reasoning and predictive modeling.

The traffic data demonstration platform will be part of the smart, extensible, and scientifically grounded infrastructure that can be adopted by transportation agencies nationwide. By transitioning from fragmented, linear workflows to comprehensive, system-level optimization, the platform empowers both decision makers and researchers to anticipate future trends and make informed, data-driven interventions for safer and more efficient operations. By adopting a generic architecture, this versatile, scalable, and robust traffic data integration platform aims to facilitate easy adoption and customization by state DOTs nationwide, thereby fostering improved collaboration, data sharing, and enhanced traffic management strategies.

To achieve the envisioned next-generation traffic data platform, a three-step plan is required. As shown in Figure 1.1, the first step is the construction with a primary focus on unified data integration and visualization. This step involves developing a robust infrastructure for ingesting real-time and historical data, managing quality, and enabling seamless access through standardized interfaces and user-friendly displays. The second step will focus on advanced data analysis, including pattern detection,

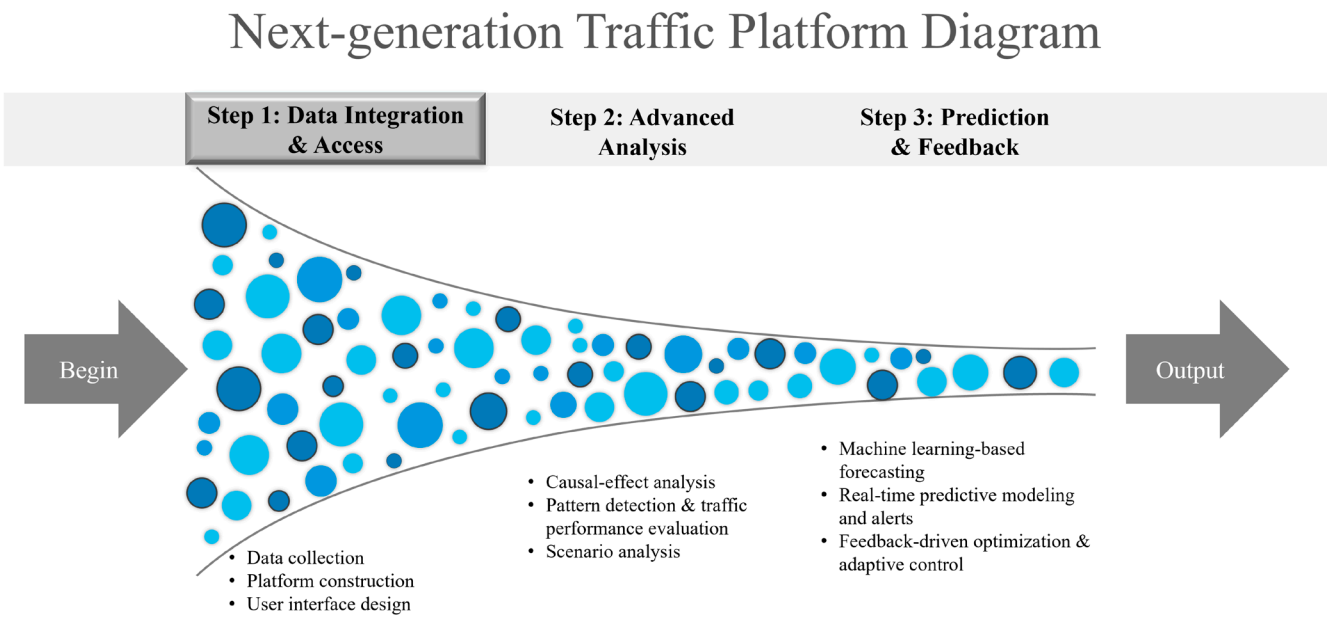

Figure 1.1 Three-Step Plan of the Envisioned Traffic Data Platform.

causal-effect analysis, and operational insights derived from multisource fusion. The third step will advance toward predictive modeling and feedback mechanisms, enabling proactive traffic management and long-term planning.

This project aligns with the long-term goal and primarily focuses on the first step of designing and demonstrating the data integration and access framework, forming the foundation for subsequent analysis and predictive capabilities. Specifically, it is organized into three key sections. Data source identification in Section 3 offers detailed description of data source category, collection procedures, and data fusion strategies, ensuring the availability of consistent and high-quality datasets. Platform design and implementation in Section 4 introduces the platform architecture, data ingestion pipeline, database design, frontend and backend interface protocols. Section 5 focuses on user interface design, presenting the systematic process for user request input and the configuration of flexible display formats for data visualization. Section 6 concludes the study and presents recommendations for future work.

2. PLATFORM REQUIREMENTS

As mentioned above, the development of the proposed next-generation traffic data platform can be divided into three steps. This project focuses on the first step of building a domain-oriented, data source–independent framework for traffic management, which enables the integration of future steps for traffic data analysis and prediction. This step encompasses data source identification, data collection and integration, user data requests, and flexible result presentation. Specifically, the core features of the traffic data platform in this study include:

- **Multisource data integration**: This platform should be able to consolidate diverse data streams into a singular, user-centric interface. The proposed system shall be designed to enable users to conveniently request and retrieve comprehensive datasets tailored to their specific analytical needs, ranging from transportation officials to academic researchers. It incorporates: (1) Seamless ingestion of real-time, historical, and external datasets; (2) Support for heterogeneous data sources, including traffic incidents, speed and volume data, weather and environmental data, event schedules, and third-party data; and (3) API-driven architecture to enable continuous data synchronization across agencies and platforms.
- **Historical data archiving and management**: To establish long-term data continuity to support planning, trend analysis, and predictive modeling, it incorporates structured storage that allows flexible querying based on spatial, temporal, and event-specific attributes.
- **Event-driven and disruption scenario analysis**: Traditional traffic platforms struggle to manage irregular events. This system is to evaluate alternative response strategies and improve resilience during disruptions by supporting modules to dynamically retrieve and visualize special events (e.g., festivals, parades, sporting events, and conventions) and emergency events (e.g., evacuations, severe weather, accidents). This will facilitate both event-driven and the scenario analysis both for historical and forecast purposes.
- **User-centric query and access control**: Efficient data utilization requires flexible query interfaces and controlled access protocols. The system allows users to retrieve specific spatial, temporal, and topical data subsets while enforcing multilevel permissions aligned with organizational roles. To empower users with targeted, on-demand data retrieval while ensuring data security, the query offers: (1) Flexible query interfaces allowing users to select spatial extent (e.g., administration-level and DOT-district-level, road-level) and temporal windows (e.g., real-time, historical, spanning multiple years, months, days and hours) based on the structure of "Event (what, where, when), consequence (traffic condition)"; (2) Data types and metrics of interest. Multi-level access controls are offered to serve different user groups (e.g., operators, planners, researchers).

- **Customizable data visualization and reporting:** To support diverse user needs through flexible data exploration and communication tools, this function enables users to request, visualize, and analyze traffic-related information in a flexible and intelligent manner. Using this function, the dimensions of the retrieved data shall be utilized in the result cards, visualized on a map view and chart view, and the user can interact with the data by specifying the definition of the axis of charts based the returned data. It shall also support replaying the data over time, as well as selecting subsets of the data to visualize.
- **Scalable and interoperable architecture:** The platform shall be traffic domain–specific but data source–independent, allowing all state DOTs and local agencies to inject their state-specific data into this platform and adapt to their specific operational environments, thereby promoting interoperability and future scalability. The architecture shall have: (1) a modular system design allowing deployment across different jurisdictions and transportation agencies; (2) an adaptable data schema supporting state DOTs' diverse operational environments and data sources.

3. IDENTIFICATION OF DATA SOURCES

Modern traffic management is increasingly dependent on large-scale, heterogeneous data generated from diverse sources, including traffic sensors, vehicle-to-everything (V2X) communications, weather monitoring systems, infrastructure IoT devices (networked sensors and field devices installed along the transportation network that generate real-time data, e.g., sensors for weather measuring; radar; or cameras reporting speed, volume, occupancy), user-reported events, mobility applications, and any potential data related to traffic. Effective traffic management relies heavily on the availability, diversity, and completeness of data inputs. To facilitate real-time responsiveness and long-term strategic analysis, the proposed system systematically integrates multisource traffic-related data spanning real-time operational feeds, historical archives, external social events, and weather conditions. This study uses Indiana data as an example. Data from other states can be used as the developed platform is data independent.

3.1 Data Source Category

3.1.1 Real-Time Traffic Data

The real-time traffic data system (Trafficwise/511) focuses on immediate traffic conditions, providing up-to-the-minute updates. It contains four parts, providing surveillance data images, thematic data groups updated in JavaScript Object Notation (JSON) format, Weigh-in-Motion (WIM) data, classification, and volume records through daily updates in CSV format. Additionally, JSON is a lightweight data-interchange format easy for humans to read and write and also easy for machines to parse and generate. The thematic data groups incorporate a variety of structured JSON feeds, updated every 5 min, from Indiana's Trafficwise 511 system. Each feed captures a distinct aspect of transportation conditions, enabling integration across operational, environmental, and behavioral domains:

The *Cars-event-feed.json* records contain active traffic events, including crashes, work zones, debris, and lane closures. Each entry is georeferenced with start and end mile markers; assigned a categorical event type and priority level; and includes consequence information for infrastructure, such as the effect on the ramp, shoulder, or lanes surrounding the area. These attributes enable spatial clustering, temporal trend analysis, and the detection of high-impact incidents.

The *cctv.json* feed provides metadata on roadside traffic cameras, including camera identifiers, road locations, and static snapshot URLs. These data enhance situational awareness by supporting visual verification of on-site traffic conditions in parallel with sensor-based data streams.

The *dmt_tt.json* and *tts.json* files report dynamic travel time messages shown on roadside message boards. These feeds contain active messages, corresponding corridors, and travel estimates between key interchanges. They are essential for tracking real-time congestion and providing user-oriented performance metrics.

The *rest_area.json* and *tpims.json* files capture information on rest area availability and truck parking. These data support freight logistics planning by including the number of available spaces, site status (e.g., open or closed), and route-level metadata, allowing the system to assess corridor-level parking demand.

Environmental data are sourced from *rwis.json*, which aggregates readings from roadside weather sensors. Attributes include air and pavement temperatures, humidity, wind speed, road surface status (e.g., dry, wet, icy), and black ice warnings. These are critical for winter maintenance analytics and safety forecasting.

Each JSON feed follows either GeoJSON or nested JSON structures with consistent spatial and temporal tagging. Together, these sources form the real-time data backbone of the platform, enabling real-time data collection, historical archiving, and fusion with external datasets for predictive analytics and event-response modeling.

3.1.2 Historical Traffic Data

Historical datasets are sourced from the Indiana Department of Transportation (INDOT) databases and include Annual Average Daily Traffic (AADT) for detailed volume and traffic pattern data, available annually; Vehicle Miles Traveled (VMT) for long-term county-level traffic data providing historical context; Traffic Count Database System (TCDS) with detailed records including speed, vehicle classification, and WIM data, updated annually; Regional Integrated Transportation Information System (RITIS) for big data aggregation and dissemination platform for solving challenging and complex transportation problems.

Among them, this platform integrates speed data from RITIS, a nationally recognized data clearinghouse that aggregates real-time traffic performance metrics from agencies and private providers. The RITIS speed dataset delivers granular, segment-level speed records across interstates and arterial corridors, updated at intervals ranging from 5 min to an hour to a month. Each observation includes time-stamped average speeds, reference link IDs, free-flow speed baselines, and travel time, facilitating bottleneck

identification and performance trend analysis. The data are formatted in tabular structures and aligned with roadway geometries via a standardized linear referencing system (TMC codes). By integrating RITIS data with real-time traffic data feeds, this platform enhances spatial-temporal continuity and enables comparative assessments across regions and data sources.

3.1.3 External Event and Weather Data

To enhance predictive analytics and response capabilities, the platform incorporates data from external APIs, including the National Weather Service and Visual Crossing APIs, offering both real-time weather updates and historical data covering various weather conditions (e.g., temperature, precipitation, wind speed, and visibility). It also incorporates social event calendars from third-party event APIs (e.g., Ticketmaster, Eventbrite) to obtain information on major public events and festivals that significantly impact traffic flow (e.g., conventions, concerts, or sporting events with a forecasted attendance of over a certain number).

3.1.4 Third-Party Data

To supplement operational data and improve situational awareness, we integrate real-time user-reported incident updates, social media feeds (e.g., informal reports and traffic condition updates from platforms such as Castlerock, IoT devices [local or federal stations], and third-party sensors [e.g., Skyline]).

3.1.5 National-Level Supporting Data

To provide context, benchmarking, and validation, the platform references national databases, for example, Federal Highway Administration (FHWA), Highway Performance Monitoring System (HPMS), Bureau of Transportation Statistics (BTS), National Transportation Atlas Database (NTAD), and Open Government Data Portal (data.gov). These national-level resources are used for cross-validation and enhancing data robustness.

3.1.6 Geospatial Data

In support of spatial analytics and visual interpretation, the platform incorporates comprehensive geospatial data layers representing administrative boundaries, road networks, and sensor infrastructure. Core datasets include:

- *Road geometry shapefiles* from INDOT, covering interstates, U.S. highways, and state routes with attributes such as route ID, direction, and mile markers. These are supplemented by AADT route-based layers.
- *Administrative boundary files*: Boundary files include counties, cities, INDOT districts, and subdistricts, enabling region-specific filtering and aggregation.
- *Additional geospatial inputs* include the points of interest (e.g., Historic rites, venues, parks and recreation centers), locations of weather stations, CCTV cameras, rest areas, and dynamic message signs, each geocoded to support mapping and spatial queries.

All layers are standardized to a common coordinate system (WGS 84) and integrated using spatial joins, buffer operations, and topology validation routines. These spatial datasets form the foundation for corridor-level pattern recognition, hotspot detection, and interactive map-based interfaces within the platform.

3.2 Data Collection Procedures

Real-time data are continuously acquired via automated API calls at specified intervals of hour or minutes, and streams are integrated using standardized data schemas to facilitate interoperability and efficient data fusion. Historical datasets are regularly updated through batch downloads and stored in a structured database designed for fast retrieval.

3.3 Data Fusion and Quality Management

Data fusion is critical for integrating heterogeneous sources such as weather APIs, traffic-related inputs. Potential data fusion methods involve spatial-temporal alignment (aligns data based on geographic and temporal references, such as geospatial joins, time window matching), schema-level fusion (unifies heterogeneous attribute schemas across sources, such as field mapping and attribute standardization), rule-based logic integration (IF-THEN logic, condition-based triggers), machine learning–based fusion and ontology-based semantic fusion (uses semantic relationships to link and reason across data sources). In this system, spatial-temporal alignment and schema-level fusion are used as the main methods of data fusion. Other methods can serve as references and potential approaches for future purposes, including the understanding, organization, and analysis of the data. Data quality is most often defined as fitness for purpose, coming directly from quality management in the manufacture of physical products (Sebastian-Coleman, 2022). Based on the goal and function of the platform, the primary goal of data quality management is to ensure the representativeness and independence of various data sources while maintaining compatibility of different sources through a unified schema and highlighting the comprehension of the current system.

In the proposed platform, data sources are filtered during data ingestion to keep the most informative ones. Further, the platform implements redundancy management and standardization to ensure high-quality data integration. Redundancy management is conducted through the automated detection of duplicate entries through spatial-temporal indexing. These quality management practices ensure consistent, reliable, and analytics-ready datasets. Adopting standardized metadata schemas to unify disparate data sources reduces the number of data processing steps, prioritizing the richness and compatibility of different data sources.

4. PLATFORM DESIGN AND IMPLEMENTATION

4.1 Platform Design

The key requirement of the system is to integrate real-time and historical data streams from multiple heterogeneous data sources into a common data model and to build an archive that supports flexible data query and future advanced analytics and predictive modeling. Additionally, architecture is required to support

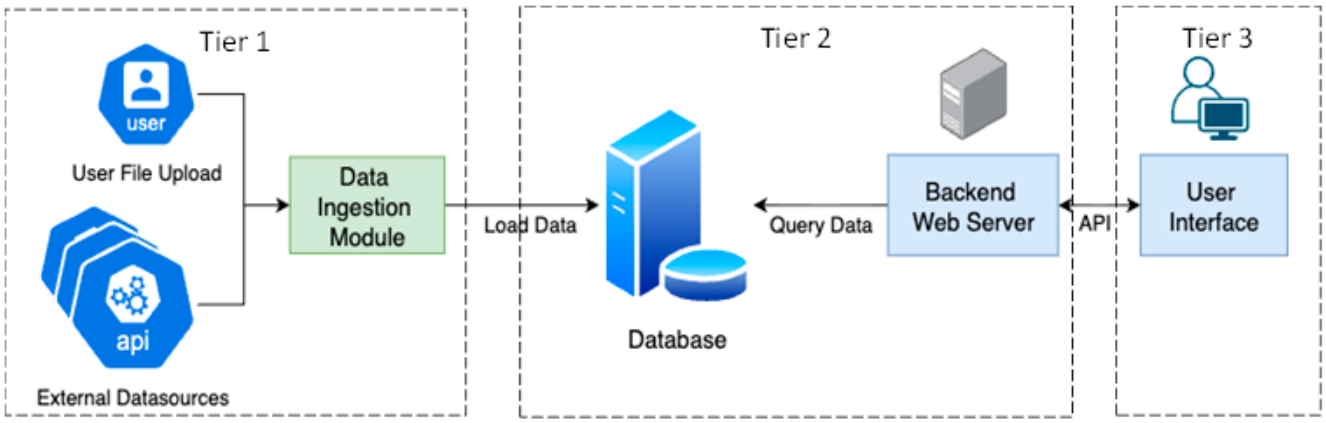

Figure 4.1 System Data Management Architecture.

scalability in terms of the number of incoming data sources and data storage. Aside from data-related requirements, a key objective is to reduce the complexity in adding external data sources to the system by optimizing the data ingestion process to be a configuration-driven process to prevent a model that requires full software development of a lifecycle for each new data source integration. To fulfill these requirements, a three-tier architecture is designed for traffic data ingestion, processing, and visualization. Figure 4.1 illustrates the system data management architecture.

Tier 1 is data gathering and ingestion, which collects data from various sources in different formats and converts them into a uniform format. In addition, it resolves data overlapping, inconsistency, and redundancy issues. The second tier is the data organization and interface to the user interface. The data should contain all necessary information to make the user interface data independent. Tier 3 is the user interface, which is data-independent, providing the user with the flexibility to ask any questions, translate the user's request into database queries, and retrieve the answer from the database.

4.1.1 Data Ingestion Module

The data ingestion module was designed in direct response to the core system requirements of integrating diverse data types and simplifying the addition of new sources. The complete data lifecycle, from initial acquisition to final storage, is managed by this layer to produce a clean, reliable, and analysis-ready data archive.

4.1.1.1 Data Ingestion Pipeline. The easy integration of new data sources with a template process and minimal software development is a key system requirement. To meet this, a flexible, configuration-driven ingestion pipeline was designed. The hard-coded logic for specific data sources is avoided. Instead, a new source can be added by an administrator through the creation of a configuration file that defines its schema and access method. This decision was crucial for achieving a scalable, low-maintenance system. The design of the pipeline was further driven by the need to handle both continuous and static data, which resulted in two ingestion methods: Automated Polling supports the integration of real-time data streams from external APIs, and Manual Uploads incorporate historical or custom datasets provided by users in standard file formats.

4.1.1.2 Data Cleaning and Processing. The requirement for building a reliable data archive for advanced analytics cannot be fulfilled if raw data is used directly due to data incompleteness, overlap, and repetition from data sources. Therefore, a mandatory formatting and cleaning process must be included. Raw, often inconsistent data is transformed into a standardized and uniform format through this step, which involves handling tasks such as removing duplicates, filling in missing values, and structuring the data for analysis.

4.1.2 Database Design

To support the heterogeneous nature of data from the traffic domain-specific applications, the system's persistence layer is designed around a highly flexible, metadata-driven model. This approach enables the platform to ingest, store, and query a wide range of data sources without requiring modifications to the core database schema or application logic when new data sources are added.

4.1.2.1 Database Design Principle. The database design is founded on the principle of categorizing all data attributes into three fundamental dimensions for events and traffic conditions: What, Where, and When.

- **What (Information):** This dimension describes the intrinsic characteristics of an event or observation that may contribute to the traffic conditions. It includes attributes such as event type (e.g., 'CRASH,' 'CONSTRUCTION'), traffic speed, weather conditions (e.g., precipitation rate), the message displayed on a dynamic sign, or social events (e.g., sports, conferences).
- **Where (Location):** This dimension provides the spatial context. It defines the geographic location of an event, which can be represented as a point (e.g., a CCTV camera), a line (e.g., a road segment with a traffic jam), or a polygon (e.g., a county affected by a weather event).
- **When (Time):** This dimension provides the temporal context. It captures the timing of an event, including start times, end times, update timestamps, or a specific point in time for an observation.

By abstracting data into these three categories, the system can convert diverse data sources into a unified framework for querying across disparate datasets. For example, a user can request "all traffic crashes (What) that occurred on Interstate 465 (Where) during the last 24 hours (When)."

A metadata-driven architecture is employed for the database to ensure that adding a new data source is more akin to a data-entry task (i.e., populating the metadata tables) rather than a software development effort, thereby significantly reducing the time and cost associated with system expansion.

4.1.2.2 Traffic Causal Effect Analysis. A fundamental functional requirement of the system is to support the analysis of causal effects on traffic flow and the various factors that affect it, both dependent and independent. Some external events are weather events or social events. To enable the identification of causal effects, the database schema supports the logical join of multiple data sources or data fusion. The merged data will be used for visualization of the user interface. Currently, this operation is carried out by utilizing the "Where" (Spatial) and "When" (Temporal) dimensions.

4.1.3 Backend–Frontend Interface Protocol Design

A key system requirement is the ability to query across a growing number of diverse data sources without continuous backend development. A traditional approach of creating a unique API endpoint for every type of query would be unscalable and brittle. To avoid this, a unified interface protocol was designed based on a single, structured API request format.

This design was chosen because it decouples the frontend's query capabilities from the backend's implementation. The frontend gathers information from the meta-table in the database to determine what information is available, allowing it to formulate complex queries for a single data source or multiple sources simultaneously by constructing a request that conforms to this standard protocol. The backend, in turn, is designed to be a generic interpreter of these structured requests. It parses the request and dynamically generates the appropriate database query based on the database metadata, rather than relying on predefined, query-specific logic. This decision directly supports the goal of rapid data source integration, as the backend server requires no adaptation when a new source is added; it learns how to query the new source from the updated metadata. Furthermore, the protocol supports multisource queries through spatial and temporal join conditions, allowing for complex correlational analysis.

A sample instance of the protocol is depicted in Figure 4.2.

4.1.4 Frontend (Client-Side)

The frontend of the 511 Map Visualization application serves as the main interface through which users explore and analyze transportation data. Designed for usability, it simplifies complex data interactions so that even users without technical or database expertise can perform meaningful analysis. The goal is to make transportation analytics intuitive. Users define what they want to see, and the system handles the rest.


```
{
  "table_name": "traffic_events",
  "selected_columns": [
    "id",
    "event_title",
    "priority_level",
    "date_start",
    "date_end"
  ],
  "filters": {
    "expressions": [
      {
        "expressions": [
          "event_status = 'active'",
          "priority_level > 2"
        ],
        "logic": "AND"
      },
      {
        "expressions": [
          "date_start > 2025-01-02",
          "date_start < 2024-12-31"
        ],
        "logic": "OR"
      }
    ],
    "logic": "AND"
  },
  "sortby": {
    " priority_level ": "DESC",
    " date_start ": "ASC"
  }
}
```


Figure 4.2 Backend–Frontend Interface Structure.

At its core, the frontend provides a visual environment for selecting datasets, applying data request criteria, defining geographic or time-based constraints, and viewing results through interactive charts and maps. All of this is done without exposing users to the complexity of the underlying query or data structure.

When a user selects a dataset, applies temporal filters, or defines a geographic area, the frontend translates those selections into a structured JSON query, identical to the one in Figure 4.1.

The front end constructs a well-defined object that includes:

- Tables and selected columns
- Filter expressions (based on user-defined conditions)
- Spatial joins
- Temporal joins for filtering by date and time

Once the query is ready, it is sent to the backend through a POST request using the fetch API. After receiving the response from the API, the front end:

- Validates the response and handles any errors
- Parses the returned data for use in maps, charts, or other user interface (UI) components
- Processes special cases depending on the dataset (e.g., parsing location data or extracting event timestamps)

This layered approach keeps the frontend flexible and modularized. By offloading query logic to a centralized service, UI components remain lightweight and easier to maintain, with no direct involvement in backend request construction. This makes the code easier to read, test, and extend. From a user perspective,

this design results in a smooth, responsive experience. Filters and selections update in real-time, and charts and maps reflect the user's choices without delay.

In short, the frontend acts as both a translator and a guide: it takes user input, builds meaningful queries, and presents the results in a way that can be conveniently displayed for comprehension and further analysis.

The scalability of the interoperable architecture would be evaluated on the ability to ingest additional sources of traffic domain data using basic configuration. In the initial phase of the project, six sources of traffic data (see Section 3.1 Data Source Category) were integrated into the system using only JSON based configuration which included data preprocessing and organization. Along with the initial sources of the data, a new data source (Castlerock Traffic Events) was also integrated into the system using minimal configuration. This further highlights the ability of the system to adapt and scale to multiple sources of data.

4.2 Data Organization Implementation

4.2.1 Data Ingestion Implementation

The data ingestion pipeline is implemented using a custom-built Data Loader. This toolkit is designed to accept multiple data sources and perform the necessary extraction and loading process into the database. This process is driven by data source–specific configuration files written in JSON format. These configuration files are created once by a developer for each new data source and contain all the necessary information for the ingestion process, such as the API address, authentication credentials, and the mapping of data attributes to retrieve and store.

The implementation follows a four-step process:

1. **Acquisition**: The tool is initialized with a configuration file to acquire raw data from a source. It either makes an API call to an external online service or reads a user-provided file. The system currently supports CSV, XML, and JSON as input formats.
2. **Format Standardization**: Regardless of the source format, the raw data is transformed into a standardized internal JSON structure. This step ensures that all data, no matter its origin, is handled consistently in subsequent stages.
3. **Cleaning and Transformation**: The standardized JSON data is then cleaned. This involves filtering out records with critical missing values and unifying the formats for all spatial and temporal fields to ensure consistency across the database. The program also separates related fields into the appropriate structures for loading into different database tables.
4. **Storage**: Finally, cleaned, structured, and transformed data is loaded into the database, populating the tables defined by the system's metadata-driven schema.

4.2.2 Database Implementation

The database is implemented using a relational database with extensions to handle geospatial data. The schema is designed to be self-describing using metadata tables.

- **Metadata Core**: The schema includes a metadata core, which is a central registry that defines the structure and properties of every data source integrated into the system. This approach was chosen to create a self-describing schema, which is essential for the system's flexibility. It allows new data sources to be added without modifying the core application code, as the system can learn about new data structures by simply reading the metadata. The core consists of four tables: one main table stores high-level information for each source, such as its name and type, while three corresponding tables dynamically define the schema for each source by mapping its attributes to the "What, Where, and When" design philosophy.
- Data Storage Tables:
  - Dedicated tables store event and event description data.
  - A centralized location table stores all geospatial information of events to reduce redundancy and improve query performance.

4.2.3 Backend Implementation

The backend server constitutes the system's core, handling all business logic and API services. It is implemented using the Python-based framework. The backend web server provides the API through which the user interface queries the databases. The request structure used for the data retrieval queries is detailed in Section 4.1.3.

4.2.4 Interface Protocol Implementation

The interaction between the client and server is managed through a well-defined data flow, initiated by the structured API request. The flow proceeds from request formulation on the client to processing, query generation, execution, and response serialization on the backend, culminating in a UI update on the client. Requesting information from multiple data sources requires retrieving data from more than one database table. The backend fetches data from each table and combines it. When querying from multiple tables, it is essential to specify a logical join condition. These joins can be of two types: Spatial Join (linking data with location criteria) and Temporal Join (combining data for a specific time segment). The API request structure, formatted as a JSON object, allows a user to define these queries by specifying tables, columns, filters, and the explicit spatial or temporal join conditions. The corresponding JSON response holds the results of this request, with data organized by its source table. This response is sent to the UI from the backend API server.

4.2.5 Operational Deployment

The framework can be easily and quickly deployed to a virtual machine or a cloud environment, as all software is packaged by leveraging Docker-based deployment, which avoids all software version dependencies and environment incompatibility issues. This approach ensures consistency, scalability, and simplified management across applications hosted on a common infrastructure.

The deployment requirements for the system were sourced from INDOT to enable quick turnaround time for the process. The primary software requirements revolve around using an Ubuntu 24.04–based environment and specific versions for the database (Postgres 17). The hardware requirements include using 1 TB storage with 32 GB of memory on the server.

The primary limitations to large-scale deployment are hardware constraints on database storage and web servers. These challenges can potentially be addressed through a cloud-based approach that incorporates auto-scaling based on the resource usage of various system components. Furthermore, enabling component-level scaling can mitigate bottlenecks, particularly on the database server, as most system interactions involve querying large volumes of data. Additionally, when a new data source is integrated into the system, a limited time window is anticipated for the configuration of the data source to be activated by the system.

5. USER INTERFACE DESIGN

The design of the unified analytics platform's webpage is driven directly by the diverse needs of its intended users, from strategic planners to real-time operators. The core design philosophy is to guide the user through a logical and systematic process, first defining a precise data request and then formatting the resulting information in the most effective way for their specific analytical task. This approach ensures that the platform is both powerful for expert users and intuitive for those performing more routine checks, transforming a complex data ecosystem into an accessible and actionable tool.

5.1 A Systematic Process for User Request Input

To move beyond the limitations of the current fragmented toolset, the platform's interface will guide users through a structured workflow to construct their data queries (Box 2 in Figure 5.1). This systematic process is designed to be flexible, allowing users to start with a broad question and progressively narrow their focus, or to target a very specific event from the outset.

5.1.1 Selection of Primary Analytical Lens

The user's journey begins by selecting their primary data focus, which corresponds to the core questions they seek to answer. As shown in Figure 5.2, the main options, derived from user needs, include:

- Performance (Effect) Data: Centered on speed, travel time, and reliability metrics. This is the starting point for both real-time monitoring and long-term strategic analysis.
- Event (Cause) Data: Focused on discrete occurrences, including unplanned incidents (crashes, debris), planned construction and maintenance activities, and special events (concerts, games).

5.1.2 Geographic Scoping

Once the primary lens is chosen, the user then defines Spatio-Temporal Filtering based on intuitive "where" and "when" of their query using a combination of interactive map tools and form-based controls, as shown in Figure 5.3. This directly addresses the need for both broad and granular analysis. Users can define their area of interest by (1) Selecting a route and mile marker range; (2) Choosing a predefined corridor, county, or district; (3) Drawing a custom polygon or bounding box directly on the map interface. (4) Selecting a combination of

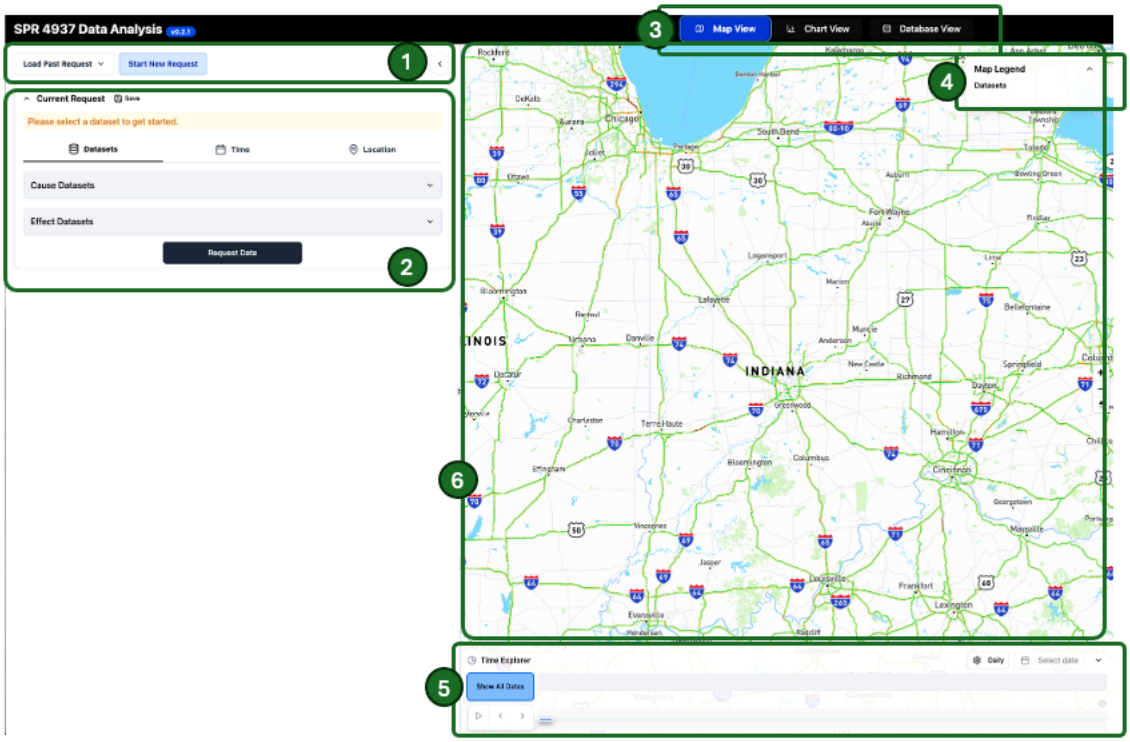

Figure 5.1 Interface Overview.

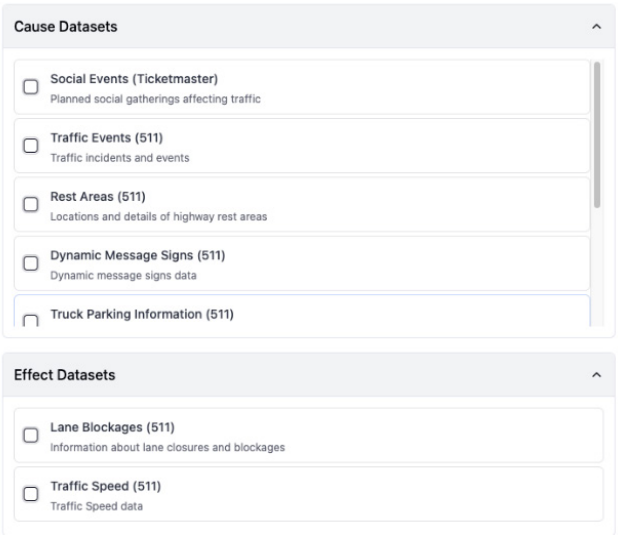

Figure 5.2 User Interface for Selecting Cause and Effect Datasets.

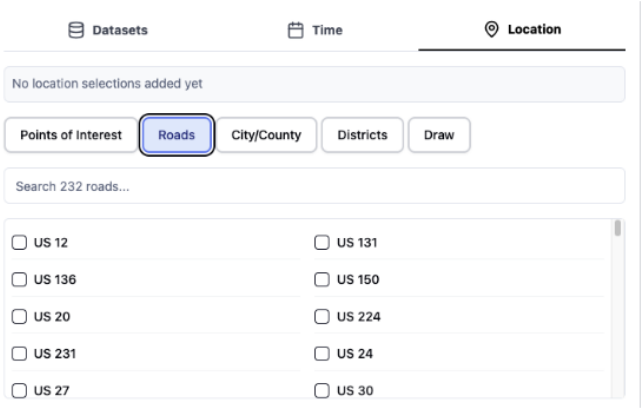

Figure 5.3 User Interface for Selecting Locations.

road selections with political or operational boundaries (e.g., counties or districts).

In addition, the map view will visualize the selected locations in real-time to support users when constructing geographical criteria for defining data requests. For example, in Figure 5.4, the user has selected Highway I-65 (highlighted in red) and District Greenfield (highlighted in blue). When multiple locations are selected and there are overlaps in between, a pop-up window will be displayed for the user to specify if they would like to keep both locations or only find data in the intersections.

5.1.3 Temporal Scoping

The platform's "time machine" functionality is central to the user experience, in addition to the where and what information. As shown in Figure 5.5, users can define their time frame by (1) Selecting a specific date and time range; (2) Using a "time-slider" to intuitively scrub back and forth through the historical record; (3) Filtering for recurring time periods, such as "all Friday afternoons in July," to analyze repeating patterns.

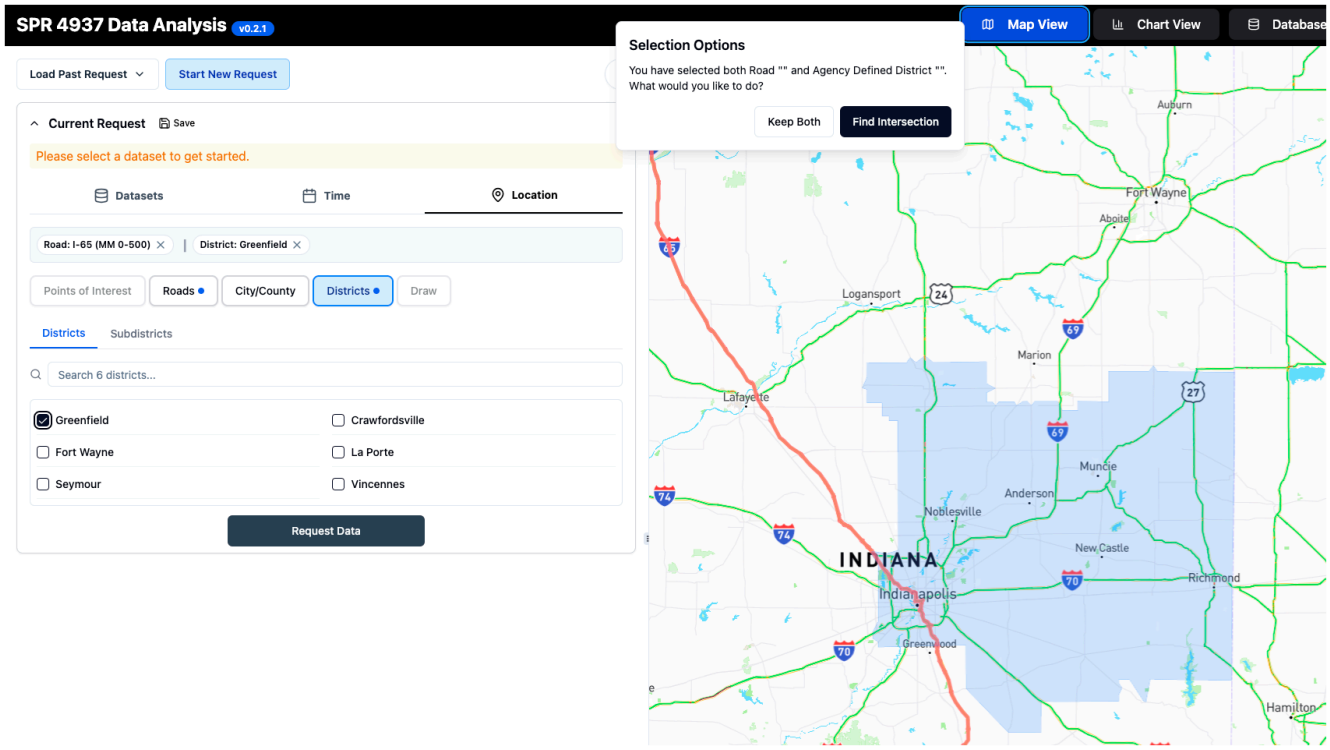

Figure 5.4 Interactive Location Visualization to Support Data Request Criteria Construction.

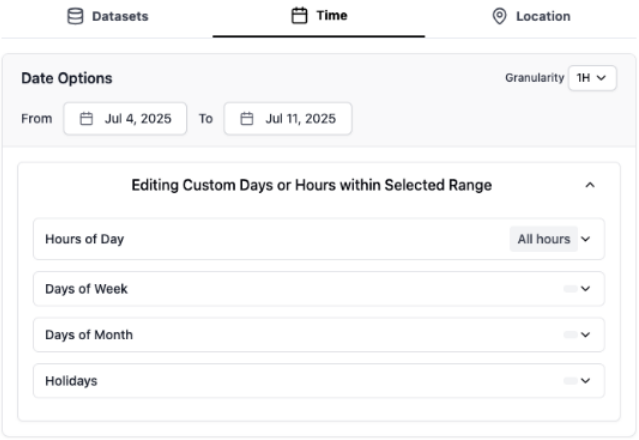

Figure 5.5 User Interface for Selecting Date Range and Temporal Filters.

5.1.4 User Query Input Support Features

5.1.4.1 Layering of Contextual Data. A key innovation of the platform is the ability to seamlessly overlay contextual data to help users "connect the dots" and diagnose the root cause of traffic phenomena. After defining their primary query, users can select from a palette of contextual layers to add to their analysis, as shown in Figure 5.6:

- Weather: Overlaying historical radar data or records of precipitation and temperature.
- Special Events: Displaying the location and time of major events from an integrated calendar.
- Network State: Showing the status of Dynamic Message Signs (DMS), real-time truck parking availability, or the configuration of lane closures at the selected time.

5.1.4.2 Query Management. The query management system in the 511 Map Visualization application enables users to save their analytical views and load previously saved queries, enhancing collaboration and efficiency. This system is implemented through two primary buttons: "Load Past Request" and "Start New Request" (Box 1 in Figure 5.1).

The "Load Past Request" button functions as a dropdown menu that allows users to access previously saved analytical configurations. When clicked, it toggles the visibility of a dropdown containing a searchable list of saved requests. Users can filter through their saved queries using the search box, making it easy to locate specific configurations. When a saved request is selected, the system creates a new filter with the saved request's name and populates it with all previously configured settings, including selected datasets, attribute filters, location selections, and timeframe parameters. This functionality ensures that users can quickly return to previous analytical views without having to manually reconfigure all settings.

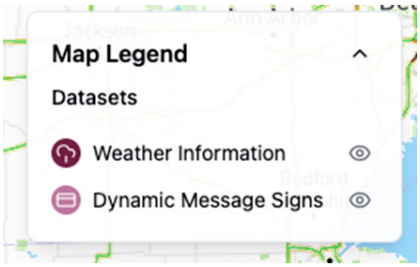

Figure 5.6 Map Legend for Toggling Dataset Visibility.

The "Start New Request" button allows users to begin a fresh analysis. When clicked, it creates a new filter with default settings, resets all selections, and marks the request as started. This provides users with a clean slate for configuring new data queries without being influenced by previous selections.

Behind the scenes, the system uses a comprehensive data structure to store all aspects of a query, including:

- Location selections (roads, cities, districts, points of interest)
- Timeframe parameters (date ranges, weekdays, hours, holidays)
- Selected datasets and their specific attribute filters
- UI state information for accurate reconstruction of the interface

When saving a request, the application captures the complete UI state and sends it to the server via an API endpoint. The saved request includes a name, description, creation timestamp, and the full configuration data needed to recreate the exact view. This data is stored in JSON format, allowing for efficient retrieval and reconstruction of complex analytical views.

5.1.5 Generate Requests for Retrieving Data and Receiving Results

After the user has constructed their request, the graphical user interface (GUI) translates the request into the JSON file according to the protocol specified in Section 4.2.4 and sends this file to the backend web server. The returned results are in the format specified by the protocol in Section 4.2.4.

The communication protocol consists of the following key steps:

1. **Query Construction:** The application first builds a comprehensive query request object using the buildQueryRequest function. This function transforms the user's selections into a structured JSON payload defined in Section 4 'Interface Protocol Implementation' containing:
   a. Tables with selected datasets and specific columns to retrieve
   b. Dataset-specific attribute filters with precise expressions (column, operator, value)
   c. Spatial join conditions derived from location selections (roads, cities, districts)
   d. Temporal join conditions based on timeframe selections (date ranges, weekdays, hours)
2. **Request Formatting:** The constructed query is formatted as a QueryRequest object with a standardized structure that the backend API expects. This ensures compatibility and proper interpretation of the query parameters.
3. **API Communication:** The formatted request is sent to the backend server using the executeQuery function, which:
   a. Makes a POST request to the API proxy endpoint (/api/proxy)
   b. Sets the appropriate Content-Type header to 'application/json'
   c. Converts the query object to a JSON string using JSON.stringify
   d. Sends the request body with all query parameters

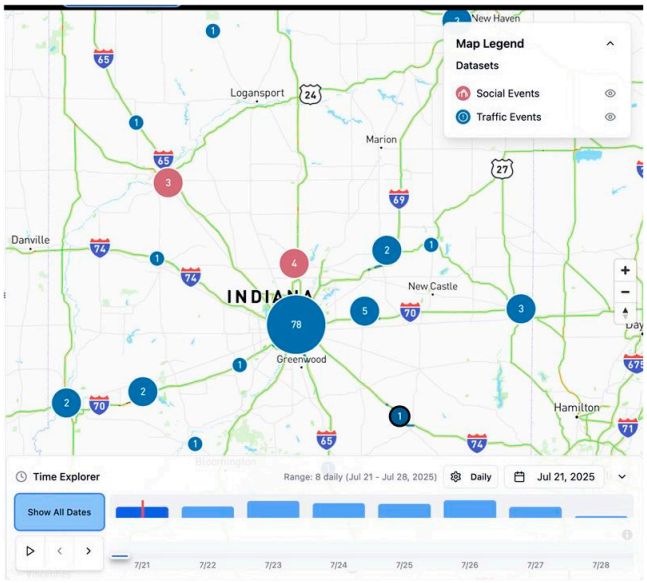

Figure 5.7 Interactive Map and Timeline View for Exploring Spatiotemporal Traffic Data.

Figure 5.9 Database View.

4. **Response Processing:** Upon receiving the response from the API, the application:
   a. Validates the response status to ensure successful communication
   b. Parses the JSON response data
   c. Processes and formats the results for visualization on the webapp
   d. Updates the application state with the retrieved data

5.2 A Systematic Process for Defining the Display Format

With the backend providing search results for user requests, the platform offers a flexible and systematic way to define how information is visualized and formatted. This allows the output to be tailored for different audiences and purposes, from an internal diagnostic session to a formal project justification report.

Choice of Primary Visualization: Users can select the most appropriate format for their analysis from several distinct visualization modes, as shown in Box 3 Figure 5.1:

- **Map View:** The default, interactive display showing data geospatially. This view is ideal for situational awareness and understanding the physical location and data counts (Figure 5.7).
- **Timeline View:** This time-space diagram is the primary tool for granular diagnosis of queue formation and duration (Figure 5.7).
- **Chart View:** Charts and graphs for the selected query. This view is suited for developing in-depth analysis and generating executive summaries (Figure 5.8).
- **Database View:** A tabular representation of raw or filtered query results. This view is ideal for data auditing, verification, and exporting records for external analysis (Figure 5.9).

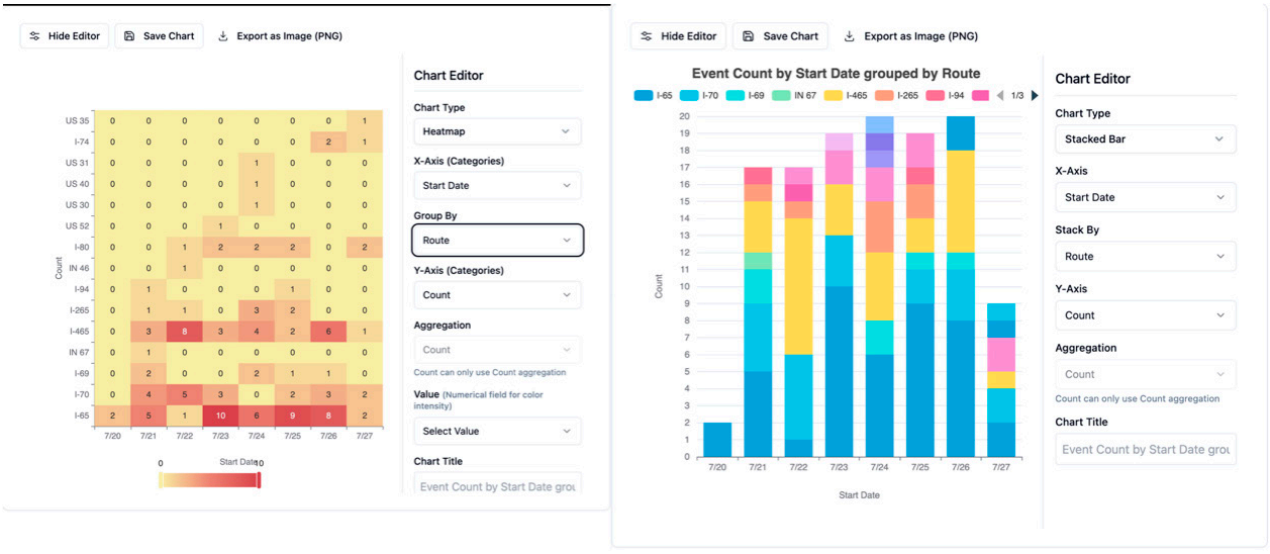

Figure 5.8 Example Visualizations Supported by the Chart View.

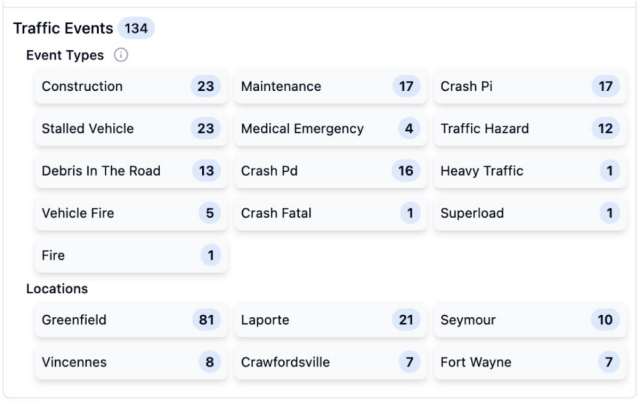

Figure 5.10 Results Summary Card.

Customization of Visualization Parameters: Within each view, users can fine-tune the display. For instance, in the Map View, they can toggle the visibility of different layers. In the Timeline View, they can adjust the time range and granularity for speeds to match specific thresholds relevant to their analysis, as shown in Figure 5.7.

Marker Tooltips: Clicking on any data point reveals a contextual tooltip, displaying key attributes such as speed, event type, or timestamp. This enables quick, in-place insights without disrupting the exploration workflow.

Highlighted Selection Feedback: When a location is selected via the filter interface (e.g., road, district, or city), it is visually highlighted on the map. This provides immediate confirmation and spatial orientation for the selected context.

Results Summary Card Filtering: The results summary card provides a real-time, structured overview of the dataset returned by the current query (Figure 5.10). It consistently displays a breakdown by location, along with additional groupings based on the most relevant attributes of the selected dataset (e.g., event type for traffic events, severity for incidents, etc.). This summary adapts dynamically to reflect all applied filters, giving users clear, immediate feedback on what is being analyzed.

Data Export and Reporting: A critical step in the user workflow is the creation of a final deliverable. As shown in Figure 5.8 and Figure 5.9, the platform systematizes this process by offering robust export options:

- Export Visualizations: Any map, chart, or heat map can be exported as a high-resolution image (e.g., PNG) for direct inclusion in reports and presentations.
- Export Data Tables: Users can download the underlying data for their query in a tabular format (e.g., CSV), allowing for further analysis in external tools like Excel. This is crucial for users who need to create custom tables and perform specialized calculations.

By structuring both the input and output processes, webpage design guides users from a complex question to a clear answer.

5.3 Performance Optimization and Rendering Strategy

To ensure a responsive and scalable user experience, the platform integrates multiple performance-oriented strategies across rendering and data handling workflows.

5.3.1 Rendering Efficiency

- **Map Layer Management:** Marker rendering on the map uses a layer-based approach with dynamic clustering. Unused layers are efficiently removed to reduce memory consumption. To reduce visual clutter when analyzing large datasets, the platform automatically clusters spatial markers based on zoom level. This helps preserve readability, especially in high-density areas, while still allowing users to access individual data when zoomed in.
- **Front End React Optimizations:** The platform leverages React hooks such as useMemo and useCallback to minimize unnecessary re-renders. Effects are dependency-aware, and UI components are modularized for rendering only when relevant input changes.

5.3.2 Network Optimization

- **Request Handling:** API calls are optimized through request batching, reduced column selection, and built-in retry logic for improved robustness in case of transient failures.
- **Asset Loading:** Static resources like GeoJSON files are lazy-loaded and cached, reducing initial load times and improving subsequent navigations.

5.4 Adaptive Architecture

The webapp is designed with a fully dynamic, data-agnostic architecture that eliminates the need for hardcoded logic when new datasets are introduced. On startup, the app uses a backend metadata discovery system to fetch all available tables and their attributes via the **fetchDataSourcesMetadata** service. This metadata powers dynamic UI components like the Dataset Selector (what), which automatically reflects any new datasets added to the backend—meaning there are no manual updates required.

The **buildQueryRequest** function builds queries dynamically based on the metadata and user selections, not fixed table names or columns. All filters, visualizations, and analytics adapt in real-time to any dataset structure. This abstraction layer enables instant integration of new data sources, making the platform scalable, extensible, and future-proof—an edge over rigid, hardcoded systems.

6. CONCLUSION AND FUTURE WORK

This study presented the design of a traffic data integration and visualization platform based on detailed survey of existing traffic information websites and platforms. The main work in this study focuses on the traffic data integration, ingestion, and visualization with three key sections. The developed platform supports multisource data integration, data-independent data organization structure and user interface, taking Indiana data as a demonstration example. The developed user interface supports two primary views of traffic pattern view and correlation analysis. It can display multidimensional traffic patterns based on time frames, geographical scopes, and specific locations, while also showcasing the flexibility to facilitate relationship analysis coupled with environmental factors, such as weather conditions and local events. The advantages of the tool include convenience of comprehensive data merge, versatility as a research tool, scalability and extensibility across states. A web-based prototype system was implemented with

an easy-to-use web interface. The usability and practicality of the developed prototype has been beta-tested by Study Advisory Committee members and other INDOT traffic engineers.

Future studies will focus on advanced analysis, prediction and feedback. By construction of features and built-in support for machine learning-based forecasting models, real-time anomaly detection, congestion prediction, and routing optimization, cause-effect analysis between traffic and external factors (e.g., weather, social events), this platform will offer more capability to support both operational decision-making and research-grade model development.

REFERENCES

Almukhalfi, H., Noor. A., & Noor. T. H. (2024). Traffic management approaches using machine learning and deep learning techniques: A survey. *Engineering Applications of Artificial Intelligence*, *133, Part B*, 108147. https://doi.org/10.1016/j.engappai.2024.108147

Bamakan, S. M. H., & Banaeian Far, S. (2025). Distributed and trustworthy digital twin platform based on blockchain and Web3 technologies. *Cyber Security and Applications*, *3*, 100064. https://doi.org/10.1016/j.csa.2024.100064

Sebastian-Coleman. L. (2022). The process challenge: Managing for quality. In L. Sebastian-Coleman (Au.), *Meeting the challenges of data quality management* (pp. 93–117). Academic Press. https://doi.org/10.1016/B978-0-12-821737-5.00005-5

Spy Pond Partners, LLC, Vanasse Hangen Brustlin, Inc., & Anything Awesome, LLC. (2024). *Implementing data governance at transportation agencies* (Volume 1: Implementation guide) (NCHRP Web-Only Document 419). The National Academies Press. https://doi.org/10.17226/28837

Wang, Y., Yin, S., Nasri, M., Liu, C., Zhang, S., Shankar, V., Venkataraman, N., Shrestha, R., Chandler, B., Brown., L., Davis, N., Ghandour, H., Mannering, F., Gates, J., & Cai, J. (2024). *Leveraging artificial intelligence and big data to enhance safety analysis: A guide* (NCHRP Research Report 1152). The National Academies Press. https://doi.org/10.17226/29098

About the Joint Transportation Research Program (JTRP)

On March 11, 1937, the Indiana Legislature passed an act which authorized the Indiana State Highway Commission to cooperate with and assist Purdue University in developing the best methods of improving and maintaining the highways of the state and the respective counties thereof. That collaborative effort was called the Joint Highway Research Project (JHRP). In 1997 the collaborative venture was renamed as the Joint Transportation Research Program (JTRP) to reflect the state and national efforts to integrate the management and operation of various transportation modes.

The first studies of JHRP were concerned with Test Road No. 1—evaluation of the weathering characteristics of stabilized materials. After World War II, the JHRP program grew substantially and was regularly producing technical reports. Over 1,600 technical reports are now available, published as part of the JHRP and subsequently JTRP collaborative venture between Purdue University and what is now the Indiana Department of Transportation.

Free online access to all reports is provided through a unique collaboration between JTRP and Purdue Libraries. These are available at docs.lib.purdue.edu/jtrp/.

Further information about JTRP and its current research program is available at engineering.purdue.edu/JTRP.

About This Report

An open access version of this publication is available online. See the URL in the citation below.

Ding, J., Koshy, L., Maheshwari, T., Mathew, A., Zhou, J., Chien, S. Y.-P., Li, T., Tao, C., & Chen, Y. (2025). *Enhanced traffic pattern knowledge abstraction and presentation from 511 database* (Joint Transportation Research Program Publication No. FHWA/IN/JTRP-2025/43). West Lafayette, IN: Purdue University. https://doi.org/10.5703/1288284318608