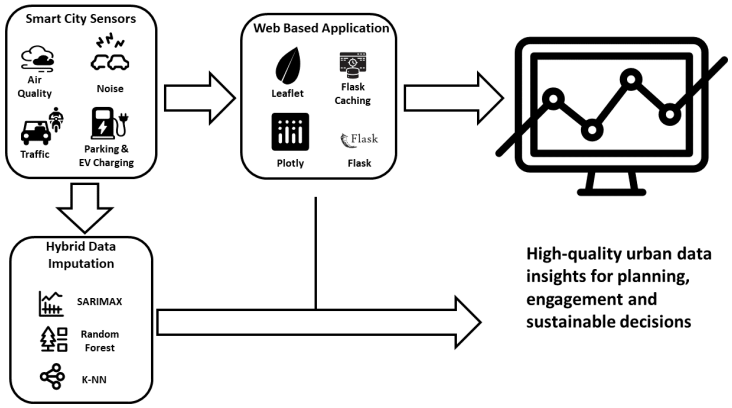
INTERNATIONAL HELLENIC UNIVERSITY

INTERNATIONAL HELLENIC UNIVERSITY
DEPARTMENT OF INFORMATION AND ELECTRONIC ENGINEERING

BACHELOR THESIS

Highlighting the Importance of Modern Sensors in Smart Cities: A Web-Based Application for Data Analysis and Visualization



Student:
Panagiotis Karampakakis
Student ID: 164673

Supervisor:
Konstantinos A. Tsintotas
Assistant Professor

23 May 2025

Highlighting the Importance of Modern Sensors in Smart Cities:
A Web-Based Application for Data Analysis and Visualization
Code of Dissertation: 23101
Student name: Karampakakis Panagiotis
Supervisor name: Dr. Konstantinos A. Tsintotas
Date of undertaking: 13-03-2025
Date of completion: 31-05-2025

We hereby affirm the authorship of this paper and the acknowledgement and credit of any assistance. We received in its composition. We have, furthermore, noted the various sources from which we extracted data, ideas, visual or written material, in paraphrase or exact quotation. Moreover, we affirm the exclusive composition of this paper by me, for it is a dissertation in the Department of Information and Electronic Engineering of the International Hellenic University.

The approval of this dissertation by the Department of Information and Electronic Engineering of the International Hellenic University does not necessarily entail the adoption of the author's views on behalf of the Department.

Dedication

To my family, whose unwavering support, understanding, and encouragement throughout the years of my academic journey have been fundamental to my personal and intellectual growth. Their presence has been a constant source of strength during every challenge and achievement. I also wish to sincerely thank Dr. Tsintotas Konstantinos, whose insightful guidance and careful review during the final stages of this thesis played an important role in shaping its outcome. His contribution reflects the academic standards and dedication that have guided me throughout my studies.

Prolog

The concept of the "smart city" now stands at the centre of current urban studies. Global growth of urban populations and the rapid spread of digital technology are moving municipalities away from their traditional forms and toward integrated digital ecosystems. Demographic projections indicate that by 2050, more than two-thirds of the world's inhabitants will reside in cities. This shift places unprecedented demands on infrastructure, environmental quality, and public administration. As a result, the smart city has moved from a theoretical construct to an operational requirement for coping with the composite challenges of modern urban life.

Smart cities are distinguished by their close coupling of sensor networks, data analytics, and automated control within everyday municipal functions. These layers monitor and optimise air quality, traffic flow, energy use, water distribution, and public safety. Their effectiveness depends on continuous data produced by ubiquitous sensing and converting that information into actionable knowledge. Reliability of hardware, rigour of analytical routines, and transparency in decision processes all contribute to the value of these systems.

The present thesis addresses these issues by creating a modular web-based platform that analyses and displays heterogeneous sensor streams collected in urban settings. Development drew upon data supplied by the Basel "*Smarte Strasse*" initiative and concentrates on five domains: air pollution, acoustic noise, vehicular activity, parking utilisation, and patterns of electric vehicle charging. Considering these domains together yields a comprehensive perspective on urban dynamics.

The backend is implemented in Python with Flask and extensively uses the `pandas` library for efficient data ingestion and preprocessing. Sensor readings, supplied as static comma-separated files, are first aligned in time and harmonised semantically. Missing observations are resolved with a composite imputation engine that combines Seasonal AutoRegressive Integrated Moving Average with eXogenous variables (SARIMAX) for temporal dependencies, k-Nearest Neighbour (k-NN) techniques for spatial similarity, and Random Forest regressors for non linear interactions across variables. Each method contributes a distinct advantage, and their combination maintains data integrity in a balanced fashion.

On the client side, Plotly.js and Leaflet.js deliver interactive visualisations. Users inspect temporal trends, spatial patterns, and correlations through an interface that adapts to desktop and mobile devices. The visual layer supports time series exploration, spatial clustering, and cross-variable mapping, serving domain experts and engaged citizens.

Ethical and legal considerations are treated with equal seriousness. Noise sensors were configured at

limited resolution to protect personal privacy in compliance with the General Data Protection Regulation. All transformations, analyses, and figures are reproducible and transparent, reflecting the principle that acceptance of innovative city systems depends on public trust as much as on technical performance.

Beyond its technical contribution, this work has implications for urban planning, environmental regulation, and civic technology. As municipal management becomes increasingly data-oriented, the demand for open, reliable, and transparent tools will intensify. The platform presented here demonstrates how openly available technologies can convert raw readings into insights of practical relevance. It is intended for software engineers and data scientists, planners, environmental professionals, and citizens who wish to participate in evidence-based governance.

In conclusion, the thesis narrows the gap between conceptual models of smart urbanism and operational solutions grounded in measured data. It underscores that the future of cities relies not only on sophisticated devices but also on ethical, inclusive, and adaptable system design. While the research marks the end of an academic project, it also lays the groundwork for subsequent exploration in continuous sensor integration, predictive analytics, and participatory governance. In this way, the prospect of more responsive and sustainable cities becomes a tangible objective rather than a distant aspiration.

Abstract

As urban environments grow increasingly complex, the demand for real-time, data-driven tools to monitor and manage city infrastructure has never been more critical. This thesis presents the design and implementation of a modular, web-based platform for visualizing and analyzing heterogeneous urban sensor data streams, developed using real-world datasets from the Basel *Smarte Strasse* initiative. The system consolidates domains such as air quality, traffic mobility, environmental noise, parking usage, and electric vehicle charging into a unified and responsive interface. At the core of the platform is a hybrid imputation engine that combines SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors), *k*-Nearest Neighbors (k-NN), and Random Forest regression to address the pervasive issue of missing sensor values. This blended approach ensures temporal continuity and cross-feature consistency, producing robust data suitable for longitudinal exploration. The backend is implemented in Python using Flask and exposes JSON Application Programming Interfaces (APIs) for real-time communication, while the frontend leverages Plotly.js and Leaflet.js for dynamic visualization of spatial and temporal patterns. Evaluation results confirm the system's reliability, with low Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) metrics across pollutant and mobility datasets. Performance profiling shows the API is optimized for speed and memory efficiency through caching and data downsampling. The system's usability is further enhanced by a responsive design that accommodates desktop and mobile users, supporting inclusive and transparent smart city engagement. Beyond its technical contributions, this work reflects on data ethics, sensor calibration variability, and the necessity of privacy-aware design. The application serves as a proof-of-concept for smart city monitoring and a foundation for future extensions, including real-time alerting, predictive modeling, and integration with external systems such as weather feeds and public health APIs. In doing so, this thesis contributes toward building resilient, responsive, and sustainable urban infrastructures.

Περίληψη

Καθώς τα σύγχρονα αστικά περιβάλλοντα καθίστανται ολοένα και πιο περίπλοκα, η ανάγκη για εργαλεία βασισμένα σε δεδομένα που προέρχονται σε πραγματικό χρόνο για την παρακολούθηση και διαχείριση των υποδομών των πόλεων είναι πιο επιτακτική από ποτέ. Η παρούσα πτυχιακή εργασία παρουσιάζει τον σχεδιασμό και την υλοποίηση μιας αρθρωτής, διαδικτυακής πλατφόρμας για την οπτικοποίηση και ανάλυση ετερογενών ροών δεδομένων αισθητήρων πόλης, χρησιμοποιώντας πραγματικά δεδομένα από το ερευνητικό πρόγραμμα "*Smarte Strasse*" της Βασιλείας. Το σύστημα μας ενοποιεί πολλαπλά πεδία όπως ποιότητα αέρα, κυκλοφορία, ακουστικό θόρυβο, χρήση χώρων στάθμευσης και φόρτιση ηλεκτρικών οχημάτων σε ένα ενιαίο και δυναμικό περιβάλλον διεπαφής. Στην καρδιά της εφαρμογής βρίσκεται μια υβριδική μηχανή αναπλήρωσης ελλειπόντων τιμών, που συνδυάζει τις μεθόδους: εποχιακού αυτοπαλίνδρομου ολοκληρωμένου κινητού μέσου με εξωγενείς μεταβλητές (SARIMAX), *k*-Πλησιέστεροι Γείτονες (*k*-NN) και μοντέλο παλινδρόμησης με χρήση δέντρων απόφασης σε τυχαία δάση (Random Forest Regressor), για την ενίσχυση της αξιοπιστίας των δεδομένων σε χωροχρονικό επίπεδο. Το backend της εφαρμογής υλοποιείται με Flask (Python), παρέχοντας JSON API για επικοινωνία σε πραγματικό χρόνο, ενώ το frontend βασίζεται στα Plotly.js και Leaflet.js για την διαδραστική αναπαράσταση των δεδομένων σε μορφή χαρτών, χρονικών διαγραμμάτων και πινάκων συσχέτισης. Η αξιολόγηση της εφαρμογής καταδεικνύει υψηλή ακρίβεια, με χαμηλές τιμές Μέσου Απόλυτου Σφάλματος (MAE) και Ρίζα Μέσου Τετρα-γωνικού Σφάλματος (RMSE) στις κατηγορίες ρύπων και κυκλοφοριακών δεδομένων. Η βελτι- στοποίηση μέσω caching και downsampling εξασφαλίζει άμεση απόκριση και περιορισμένη χρήση μνήμης ακόμη και σε απαιτητικά σενάρια χρήσης. Πέρα από την τεχνική του διάσταση, η εργασία προσεγγίζει ζητήματα ηθικής, ακρίβειας αισθητήρων και προστασίας προσωπικών δεδομένων. Η εφαρμογή αποτελεί ένα αποδοτικό εργαλείο παρακολούθησης της αστικής δρα- στηριότητας και ταυτόχρονα μια αφετηρία για μελλοντικές επεκτάσεις, όπως μοντέλα πρόβλε- ψης ρύπανσης, δυναμική διαχείριση κυκλοφορίας, και διασύνδεση με API μεταφορών ή υγείας. Συνολικά, η πτυχιακή συμβάλλει στην ανάπτυξη διαφανών, βιώσιμων και έξυπνων πόλεων του μέλλοντος.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Historical Evolution of Smart Cities

Since the term “smart city” was first introduced in the early 1990s, its definition has expanded significantly. Initially, the concept emerged at the intersection of urban planning and information technology [1], as scholars and practitioners explored how digital tools could improve municipal services [2]. Early initiatives focused primarily on automating infrastructure and digitising public administration, particularly in technologically advanced nations such as the United States and Japan [3], [4]. These efforts were driven by the urgency posed by accelerating urbanisation, which brought logistical, demographic, and environmental challenges.

Rapid urban growth intensified the practical necessity of these smart-city projects. In 1950, only about thirty percent of humanity resided in urban areas, yet by 2014 this proportion had risen to fifty-four percent [5]. A separate global survey confirmed this upward trajectory [6]. United Nations projections indicate that by 2050, over sixty-six percent of the global population will live in urban regions [7], with the fastest growth expected in Asia and Africa [8]. Cities such as Lagos, Dhaka, and Mumbai are expanding at an unprecedented pace [9], driven by both natural population growth [10] and rural-to-urban migration seeking employment opportunities [11]. This rapid urbanisation challenges housing infrastructure, pollution management, and traffic control [12], straining sustainable resource usage [13].

China exemplifies this dramatic urban transformation, with urbanisation levels soaring from forty point five three percent to fifty-three point seven three percent within just one decade [6]. This surge has given rise to “megacities,” defined as urban areas with at least ten million residents [14]. Governments responded by investing heavily in infrastructure, water systems, education, public transportation, and healthcare [6]. Nonetheless, infrastructure alone cannot guarantee sustainability [15], and rapidly growing cities continue to face issues such as air pollution, resource depletion, and crime [16]. Similar social and environmental challenges have been documented across various regions globally [17].

In response to these multifaceted challenges, integrating information and communication technology (ICT) with urban management gave rise to the smart-city paradigm. Advances in the Internet of Things (IoT), cloud computing, big data analytics, and mobile technology enabled real-time monitoring of transport networks, energy grids, and other urban systems [18]. Cloud computing significantly lowered barriers to large-scale data analysis [19], while mobile networks expanded public access to smart-city applications [20]. Barns highlights how dense sensor grids translate raw data into actionable policy insights [21], and Meijer demonstrates the potential for data-guided governance to optimize resource

allocation and reduce waste [22].
Several strategic programmes facilitated the transition of smart-city concepts from laboratories to widespread adoption. IBM's "Smarter Planet" campaign brought global attention to digital city management [1]. Subsequently, the European Union implemented the Digital Agenda to standardize data across member states [23]. Countries developed their national roadmaps, such as Japan's i-Japan Strategy [6] and Singapore's Intelligent Nation initiative [4]. Emerging economies followed suit, with China undertaking over two hundred pilot smart-city projects and India initiating one hundred urban transformations through its Smart Cities Mission [6]. Common elements across these initiatives include economy, governance, mobility, environment, people, and overall quality of life [24].
Recent research expands the scope to include social and ethical dimensions. Greenfield cautions against the uncritical adoption of technology, highlighting risks to civil liberties [25]. Sadowski further argues that poorly governed data systems may exacerbate existing inequalities [26]. Goodspeed advocates participatory planning processes to ensure equitable distribution of smart-city benefits [27]. This broadened perspective continues shaping digital technology deployment to enhance urban life inclusively.

1.2 The Need for Smart Cities and Sensor Networks

Urbanisation is fundamentally reshaping global demographics, with recent estimates indicating that more than fifty-eight percent of the world's population now resides in metropolitan areas [7]. Projections suggest this figure will reach seventy percent by 2050 [28]. Such growth intensifies pressure on public infrastructure, transportation, housing, environmental monitoring, and energy management, as comprehensively analysed by Vlachokostas and colleagues [29]. The increasing complexity and interconnectedness of urban systems render traditional static planning methods inadequate, a limitation underscored by Batty [14] and Giffinger [24].
Smart-city initiatives seek to address these challenges through informed decision-making and adaptive service delivery [15]. Harrison's foundational work emphasizes how sensor networks support these goals [1]. Typical sensor deployments include air-quality monitors, traffic cameras, sound-level meters [30], electric-vehicle charging logs [31], smart water meters, intelligent lighting systems, and parking detectors [18]. Collectively, these devices create a digital nervous system that facilitates timely interventions across transportation, public health, energy management, and emergency response [32]. Barnaghi highlights how semantic tools convert raw sensor data into actionable intelligence [33].
Despite these advancements, deploying sensor networks exposes technical and social challenges. Power disruptions, communication failures, and environmental interference can result in data gaps or corrupted measurements [34]. Calibration drift further complicates data integrity over time [35]. Consequently, robust data preprocessing pipelines use imputation techniques [36], statistical smoothing, and outlier detection methods [37].
Ethical considerations, such as privacy, add another layer of complexity. Urban sensors frequently collect location data, behavioural patterns, or visual imagery. Managing this sensitive information necessitates rigorous practices like data minimisation, anonymisation, and transparency [23]. Tonsager emphasizes privacy risks associated with unchecked sensor use [38], while Barnes details foundational information-policy frameworks [39]. Without proper governance, deployments risk amplifying surveillance and inequality, warnings echoed by Eubanks [40] and Greenfield [25].
Visualization dashboards transform raw sensor data into intuitive insights for policymakers and resi-

dents. Interactive maps effectively highlight congestion and pollution hotspots, enabling quick identification of problematic urban areas [41]. Dashboards also facilitate transparency and encourage active community participation in urban governance by openly sharing relevant data [42]. Additionally, they support civic engagement and inclusive decision-making by making complex datasets accessible and understandable, thus empowering citizens to participate meaningfully in urban planning and policy formation [21], [22]. Ultimately, responsibly managed sensor networks represent a significant step towards creating resilient, sustainable, and equitable urban ecosystems.

1.3 Main Contributions

In alignment with the increasing complexity of urban systems and the critical need for interpretable, ethical, and actionable data pipelines, this thesis presents a modular, web-based data analytics and visualization platform tailored for smart city sensor networks. The system is designed to support both real-time and historical sensor data workflows, encompassing environmental, mobility, and infrastructural domains.

At the core of this platform is a robust and scalable imputation pipeline capable of addressing the inherent challenges of missing data in sensor networks. Missingness in urban data arises from sensor outages, communication delays, environmental interference, and hardware degradation. The hybrid imputation approach integrates Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors (SARIMAX), k-Nearest Neighbors (k-NN), and Random Forest Regression to ensure temporal and contextual coherence across time-series and spatially distributed data sources.

The visualization layer of the platform enables intuitive, map-based interaction, empowering users to explore spatial trends, temporal anomalies, and multivariate correlations. This is realized through the seamless integration of a Python Flask backend with a JavaScript frontend powered by Plotly (v1.58.5) and Leaflet (v1.7.1). These technologies allow dynamic heatmaps, statistical rollups, and real-time overlays to be rendered efficiently and responsively.

Furthermore, the system architecture emphasizes ethical urban data governance. Privacy-aware mechanisms have been embedded throughout the data lifecycle, ranging from data anonymization to GDPR-aligned usage guidelines. This ensures not only technical functionality but also regulatory compliance and societal trust. In recognition of the open science paradigm, the system is made publicly available as open-source software, encouraging transparency, replication, and reuse. The modular architecture also facilitates extensibility to other smart city domains such as energy, waste, and security.

The main contributions of this work are as follows:

- A fully integrated, web-based platform for urban data analysis and interactive visualization that fuses Python Flask for backend processing with JavaScript-based frontends using Plotly and Leaflet for spatial and analytical representation.
- A robust data imputation module combining SARIMAX for seasonality modeling, k-NN for spatial neighborhood inference, and Random Forest regression for feature interaction learning and addressing multi-modal urban sensor datasets.
- A map-centric visualization suite supporting exploratory data analysis across space and time, incorporating statistical summaries, anomaly detection interfaces, and privacy-respecting overlays tailored for non-expert users and policymakers.

- A modular and privacy-aware system architecture built for reproducibility, extensibility, and sustainable deployment, aligned with open-source principles. The source code is publicly available [1] to foster further academic and civic innovation.

This bachelor's thesis has been peer-reviewed and published in the open-access journal *MDPI Future Internet*, further validating its scientific relevance and contribution to the field of smart city informatics. The full article is accessible at: A Web-Based Application for Smart City Data Analysis and Visualization.

1.4 Structure of the Thesis

The thesis is organized into six chapters, each building logically on the previous to deliver a comprehensive exploration of smart city sensor analytics and visualization. This Chapter introduced the motivations behind smart cities, emphasizing urbanization challenges, the critical role of sensor networks, and the need for responsive urban management informed by real-time data analytics.
Chapter 2 reviews existing research and technological developments in urban data infrastructure, sensor types, analytic methodologies, and visualization techniques. It contextualizes this study within the broader academic and technological landscape, highlighting gaps this work aims to address.
Chapter 3 describes the methodological framework of the research. It details the system architecture, covering dataset acquisition, preprocessing pipelines, data imputation strategies, and the modular design philosophy guiding backend and frontend integration.
Chapter 4 outlines the practical implementation of the platform. It presents specific dataset management practices, system evaluation protocols (including endpoint runtime performance and imputation accuracy measures using RMSE and MAE), interactive visualization features, and domain-specific use cases derived from sensor data from Basel's urban environment.
Chapter 5 summarizes the key contributions of the thesis and identifies avenues for future research. Proposed future directions include enhancing the predictive analytics capabilities with advanced artificial intelligence methods, scalability improvements, and refinements in user interface design to support broader stakeholder engagement.
Chapter 6 provides additional technical details, including code snippets, detailed algorithmic frameworks, and dataset descriptions.

[1] `https://github.com/PanagiotisKara/Web-Based-Data-Analysis-and-Visualization`

Chapter 2

Background and Related Work

2.1 Sensors in Smart Cities: Types and Applications

This chapter is providing a comprehensive review of the key technologies, methods, and ethical considerations underlying sensor-driven smart city infrastructures, starting by exploring the various types of sensors used in urban environments, outlining their roles in monitoring air quality, traffic, noise, and infrastructure health. Next, there is a review of application domains such as environmental monitoring, transportation analytics, and energy management. Finally, mentioning emerging trends in data fusion, edge computing, and visualization techniques, situating this thesis within the broader context of sensor-enabled urban governance. The effectiveness of smart city operations critically depends on the breadth, resolution, and coordination of heterogeneous sensor deployments. These sensors form the bedrock of urban intelligence by translating physical-world dynamics—such as pollution, mobility, infrastructure usage, and noise—into structured digital signals amenable to computational analysis and intervention [18], [32]. Unlike traditional city management systems that rely on sparse manual data entry, smart cities employ real-time sensor feeds to enable both reactive and anticipatory governance [2]. Each sensor category offers a distinct but complementary perspective on urban life. For instance, air quality sensors—such as electrochemical and optical particle counters—measure pollutants like nitrogen dioxide (NO_2), ozone (O_3), and particulate matter ($PM_{2.5}$) to assess respiratory risks, enforce emissions regulations, and support public health advisories [43], [44]. These data are increasingly correlated with meteorological variables (e.g., wind speed, humidity) to improve pollution modeling and exposure risk estimation [45]. Traffic sensors, including loop detectors, radar, and computer vision-based counters, provide fine-grained temporal data on vehicle flow, type, and speed. This information facilitates traffic signal optimization, congestion mitigation, and even accident prediction via predictive analytics [46], [47]. Integrated with noise sensors, these systems allow cities to identify not only movement patterns but also acoustic pollution hotspots, which are increasingly linked to chronic stress and cardiovascular disease risks [48]. Smart parking sensors—based on magnetic, ultrasonic, or camera technologies—enable dynamic pricing models and occupancy forecasting, alleviating parking search traffic and optimizing land use [49], [50]. Additionally, electric vehicle (EV) charging stations are increasingly instrumented with sensors that track session duration, energy delivered, and user patterns, supporting load balancing strategies for urban grids and fostering equitable access to sustainable transport infrastructure [51], [52].

Table 2.1 provides a structured overview of common IoT sensor types used in smart cities, detailing their output variables, typical use cases, and relevant challenges (e.g., calibration, spatial cover-

age, energy consumption). As cities grow denser and more complex, these sensor systems serve not only as monitoring tools but as operational feedback mechanisms—enabling responsive systems that adapt automatically to changes in the urban environment [15], [18], [30]. Modern deployments are increasingly characterized by sensor-data fusion, where inputs from diverse sensor types are combined to yield context-aware decision-making frameworks [33], [53]. For example, coupling traffic density with pollution measurements enables cities to identify pollution sources more accurately or trigger green wave traffic signaling during peak NO_2 hours. Moreover, techniques like multivariate time-series modeling and graph neural networks are now being applied to extract latent patterns across distributed sensor nodes [54]. The shift toward edge computing—processing data closer to where it is generated—has reduced latency and bandwidth dependency, allowing for near-real-time anomaly detection and control [55], [56]. Concurrently, advances in low-power wide-area networks (LPWAN) such as LoRaWAN, NB-IoT, and Sigfox have improved sensor deployment scalability by lowering energy requirements and expanding geographic coverage, even in resource-constrained urban settings [57], [58]. Ultimately, the future of smart city sensing lies not just in collecting more data but in designing architectures that are transparent, privacy-preserving, equitable, and adaptable. The orchestration of multimodal sensors into unified platforms—such as the one demonstrated in this thesis—represents a necessary evolution toward more resilient, inclusive, and intelligent cities [14], [59], [60].

Table 2.1: Common sensor types, data outputs, and applications in smart cities.

Sensor Type	**Data Output**	**Primary Applications**
Air Quality Sensors	NO_2, O_3, CO, $PM_{2.5}$, PM_{10} concentrations ($\mu g/m^3$)	Pollution monitoring, emissions tracking, health risk alerts, environmental policy [31]
Traffic and Mobility Sensors	Vehicle counts, speed, occupancy, pedestrian flow	Traffic optimization, congestion management, safety enforcement, flow analysis [47]
Noise Sensors	Sound level (dB), frequency spectrum	Noise mapping, zoning compliance, quality-of-life assessment, urban planning [30]
Parking and EV Sensors	Availability status, occupancy, energy transferred (kWh)	Optimizing parking space utilization and monitoring EV charging infrastructure [46]
Weather Sensors	Temperature, humidity, wind, rainfall, UV index	Microclimate modeling, disaster preparedness, energy forecasting, agriculture support [43]
Structural and Utility Sensors	Vibration, tilt, pressure, current/voltage, flow rates	Infrastructure health monitoring, predictive maintenance, utility management [32]

Initial smart-city research focused primarily on digital infrastructure and electronic governance, leveraging developments in information and communication technologies [61]. Over the past decade, the emphasis has shifted significantly due to the proliferation of affordable, networked sensors deployed at an urban scale [62]. These wireless devices provide real-time, high-frequency data essential for adaptive control systems [18], [63], with end-to-end analytics converting raw data streams into ac-

tionable insights for urban agencies [64]. A significant application area is environmental monitoring, where distributed sensor networks tracking particulate matter and trace gases enable early-warning systems and public health interventions [31]. However, challenges persist, including calibration drift over time [65] and spatial variability between sensor locations [66]. Hybrid methodologies combining satellite and ground measurements are increasingly employed to enhance coverage and accuracy [67], extending recently to continental-scale studies integrating machine learning for real-time estimations [68]. Transportation analytics leverage sensors such as loop detectors, Bluetooth beacons, and mobile-phone traces to infer traffic congestion patterns [47]. These data sources feed travel-time estimation algorithms, optimizing signal timing and route guidance [46]. Additionally, vehicle-to-infrastructure communication now offers near-instantaneous telemetry, augmenting roadside sensors and improving safety features [69]. Urban soundscape monitoring utilizes both static microphones and crowdsourced mobile data to map noise levels. These sensor grids aid zoning policy decisions and correlate sound data with quality-of-life metrics [30], [70]. Advanced edge-computing deployments allow local processing of audio, reducing latency and preserving user privacy [71], [72]. Integrating sensor data with resident feedback further strengthens community engagement and trust in urban planning [73]. Parking management relies on diverse technologies such as ultrasonic probes, magnetic sensors, LiDAR, and video analytics to determine occupancy status [49]. Real-time availability is published through cloud dashboards, enhancing user experience and reducing traffic congestion caused by parking searches [74], [75]. Empirical research highlights that searching for parking spaces significantly contributes to inner-city traffic, emphasizing the importance of efficient parking solutions [76]. Electric vehicle (EV) charging infrastructure faces distinct data challenges, with load-balancing algorithms essential for preventing grid overload [52]. Predictive models allocate capacity efficiently during peak periods [51]. Blockchain technologies are also explored for secure, peer-to-peer energy transactions, although still experimental [51]. Weather sensing in urban environments uses local sensors for detailed short-range forecasts, employing edge computing to ensure timely severe-weather alerts [43], [77]. Sensor fusion methodologies further refine models to account for complex urban topographies [78], with machine learning enhancing forecast accuracy [79]. Structural health monitoring employs embedded sensors like accelerometers, fiber-optic gauges, and tiltmeters to detect early damage in infrastructure such as bridges and high-rise buildings [80]. Vibration and tilt measurements complement each other, providing comprehensive structural health assessments [81], [82]. Interoperability remains crucial, with frameworks like SOSA and CityGML ensuring consistency across heterogeneous sensor systems [83], [84]. Microservices architectures further enhance scalability and fault tolerance in large IoT deployments [85], [86]. Ethical and legal considerations, including GDPR compliance, algorithmic bias, and privacy-preserving techniques, have become integral components of smart-city design [23], [38], [40]. Explainable AI further ensures transparency and accountability in decision-making [87]. Visualization research complements these advancements, offering dashboards and immersive environments for intuitive data interaction and public engagement [21], [41], [42], [88]. Collectively, this literature establishes the technical, methodological, and ethical foundations informing this thesis's unified sensor-driven analytics platform, promoting transparent and effective urban governance.

Chapter 3

Methodology

3.1 System Architecture and Design

This chapter details the methodological framework behind the proposed web-based analytics platform, emphasizing its modular architecture, comprehensive multi-stage data pipeline, and stringent privacy protocols. The platform is specifically designed to handle diverse urban data streams spanning environmental, mobility, and infrastructure domains, thereby ensuring broad applicability.

The system employs a full-stack modular architecture characterized by loosely coupled components, facilitating ease of maintenance, scalability, and extensibility. According to Gubbi, modular architectures simplify horizontal scaling by isolating system components [89], while Perera notes the advantages of context-aware designs in adapting to new data sources dynamically [90]. Figure 3.1 illustrates the detailed data flow: beginning with a broker that aggregates heterogeneous sensor inputs, including air quality (NO_2, $PM_{2.5}$, CO)[31], acoustic data (decibel levels, frequency spectra)[30], and mobility metrics (vehicle counts, pedestrian flow, parking occupancy, EV charging activities) [46].

Given the inherent challenges of urban sensor data such as gaps, temporal drift, and random noise [34], the platform incorporates a robust three-stage imputation engine. Initially, seasonal autoregressive models with exogenous inputs identify and address periodic and short-term data gaps [65]. Subsequently, a k-nearest neighbors algorithm utilizes spatial similarities to improve data completeness [36]. Lastly, random forest regression tackles nonlinear residual patterns, leveraging auxiliary variables for enhanced accuracy [37].

Post-processing, cleaned and imputed data are served via a Flask backend, providing standardized JSON endpoints to the frontend. Plotly generates interactive charts [91], while Leaflet supports sophisticated geospatial visualizations [92]. This integration allows users to dynamically explore temporal patterns, spatial correlations, and anomalies with minimal latency.

The entire data pipeline follows privacy-by-design principles aligned with GDPR. Fine-grained data such as individual trajectories are either aggregated or anonymized, ensuring personal information protection [23]. Additional security layers, including token-based authentication and encrypted data transfer, safeguard against unauthorized access [38].

While the current emphasis lies on exploratory visualization, the architecture readily supports predictive analytics. Li highlights the effectiveness of machine learning methods for short-term forecasting given high-quality data inputs [53], while Papastefanopoulos suggests transformer-based networks as suitable for complex multivariate data interactions [54]. Open-source availability ensures adaptability across diverse urban analytics applications.

3.2 Data Preprocessing

Effective preprocessing is crucial to maintaining data consistency, reliability, and analytical robustness, particularly given the heterogeneity and noise inherent in urban sensor datasets [93].
The preprocessing stage begins by parsing raw data files, accommodating multiple encoding formats and delimiters for versatility [46]. Data headers are standardized to lowercase ASCII, and timestamps are uniformly converted into datetime objects to facilitate subsequent alignment and resampling operations [94]. Records with invalid or malformed timestamps are discarded to preserve temporal accuracy.
Next, numeric fields undergo rigorous validation checks against known physical limits, for example, values of carbon monoxide exceeding realistic environmental thresholds trigger manual review [95]. Mandatory schema validations ensure completeness of critical fields such as geolocation coordinates and unique sensor identifiers [67].
Addressing missing data, arising typically from network disruptions, power outages, or sensor degradation [96], involves a layered imputation methodology. Initially, simple statistical imputation techniques address small data gaps. Decision-tree models handle structured missingness, while ensemble methods, such as random forests, resolve complex, multivariate gaps [97].
Application's pipeline employs:

- **SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors)**: Used for capturing long-term trends and cyclical behavior in time-series data [98].
- **k-Nearest Neighbors (k-NN)**: Exploits similarity across spatially or temporally adjacent observations [36].
- **Random Forest and MissForest**: Applies nonlinear multivariate regression to infer missing values using correlated features [99], [100].

This ensemble approach preserves both temporal coherence and cross-variable dependencies. During imputation, null values are temporarily replaced by placeholders compatible with web serialization (e.g., 'null' in JSON), facilitating seamless integration with visualization frameworks. Furthermore, extreme values generated during modeling are clipped to remain within the empirical data distribution, avoiding unrealistic outputs [54].
Once cleaned, additional feature engineering is performed: temporal flags (e.g., weekend, rush hour), lagged variable creation (e.g., 1-hour delay), and rolling window statistics (e.g., moving averages) [101]. These derived attributes enhance the model's ability to detect seasonal trends and behavioral patterns, aligning with smart city objectives like anomaly detection or policy evaluation.
For computational efficiency, processed datasets are cached using server-side memory stores and updated incrementally based on event-driven triggers [102]. Before transmission to the frontend, datasets are converted to standardized formats (e.g., JSON or CSV) using type-safe serializers that guarantee compatibility with visualization libraries and APIs. This ensures robustness in rendering, responsiveness in user interaction, and modularity in system expansion.

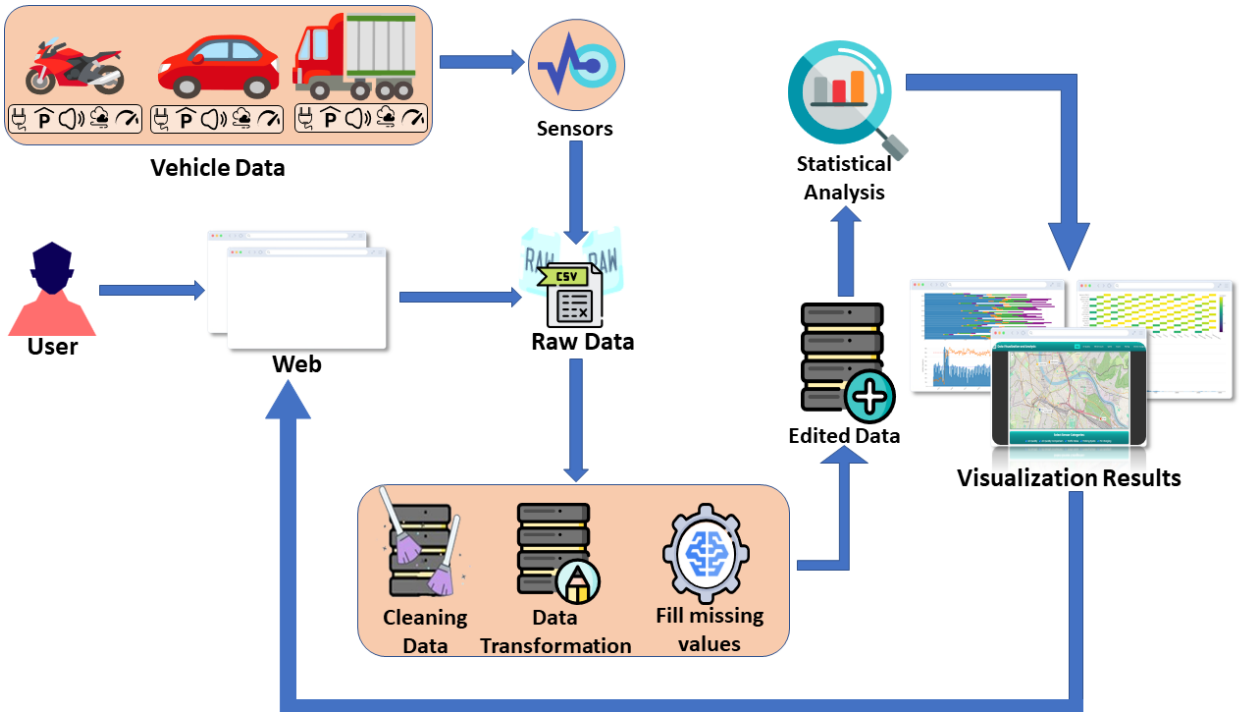


Figure 3.1: Raw data from various urban sensors, including air quality monitors, traffic counters, speed detectors, noise recorders, and EV charging stations, which are automatically retrieved when a user accesses the dashboard. These inputs undergo preprocessing, which includes cleaning, formatting, and hybrid imputation. The final harmonized dataset is then used for statistical and correlational analyses that uncover spatial patterns, temporal dynamics, and cross-variable relationships. The outcomes are rendered through interactive charts and maps for intuitive, real-time exploration of urban data.

3.3 Data Analysis

Post-preprocessing, the validated sensor data undergo comprehensive statistical analysis, revealing temporal trends, spatial anomalies, and intricate inter-variable dynamics essential for informed urban planning and management [53], [54]. The analytical process starts with temporal aggregation into hourly, daily, or weekly intervals, generating descriptive statistics such as minimum, maximum, mean, median, interquartile range (IQR), and standard deviation [93]. These metrics characterize baseline data variability, crucial for anomaly detection and alert mechanisms [34], [95].

Advanced smoothing techniques, including moving averages and exponential smoothing, mitigate short-term fluctuations, elucidating longer-term patterns, especially beneficial for volatile data such as noise levels or traffic flows [103]. Downsampling strategies optimize visualization performance without sacrificing critical data resolution [91].

Cross-variable correlation analyses further uncover meaningful relationships, such as the connection between traffic volume and noise pollution or between temperature fluctuations and EV charging demands [24], [32]. Analytical results are cached, ensuring rapid frontend rendering and interactive exploration.

3.4 Data Visualization

The visualization layer serves as the user interface for communicating complex urban datasets through intuitive and interactive graphics. It is primarily built using Leaflet [104] for interactive map rendering and Plotly.js [105] for dynamic, browser-based charting. These technologies were selected for their cross-platform compatibility, rich feature sets, and responsive performance in handling high-frequency sensor data [88], [91].

The visual components support multiple formats including time-series line and scatter plots for observing temporal trends, with smoothed line charts effectively highlighting moving averages, seasonal effects, and long-term cycles in data streams such as air quality or traffic flow [41], [101]. Stacked and grouped bar charts represent proportional distributions across categorical variables (e.g., vehicle types, sensor zones), while box plots reveal variance and outliers within sensor distributions that are important for identifying anomalous behavior or spatial inequities in service delivery [21], [54].

Histograms are used to display frequency distributions, helping to detect skewness, modality, and irregular behaviors in sensor outputs. Correlation heatmaps depict inter-variable relationships, providing insight into multi-sensor dynamics such as co-occurrence of air pollution and traffic congestion, or correlations between weather patterns and infrastructure usage [24], [53].

Every plot is embedded with responsive interactivity through hover tooltips, drag-and-zoom capabilities, and clickable filters. The chart views also include a suite of user tools (Figure 3.2) for capturing snapshots, toggling axes, and selecting data subsets using box or lasso selectors. These features enable both novice and expert users to perform exploratory data analysis with minimal friction.

Figure 3.2: The interactive Plotly toolbar offers users various controls for refining their chart views and interactions [105]. From left to right, these icons enable features such as screenshot capture, zooming, panning, selecting data points (using a box or lasso), autoscaling, resetting the chart view, and toggling specific chart modes.

All visualizations are dynamically updated using asynchronous JavaScript modules, allowing charts and maps to refresh automatically as new data becomes available or filters are applied [102]. Data is transformed into browser-native formats (e.g., JSON-compliant date objects) to maintain rendering accuracy and time zone consistency across client environments.

The layout of the visualization interface enables multi-view comparisons by arranging charts side-by-side or stacking them vertically, facilitating contextual exploration. This unified view supports real-time cross-referencing between temporal dynamics, spatial overlays, and statistical distributions, yielding a holistic understanding of urban system behavior [21], [88]. Through this integration of analytical depth and visual accessibility, the system empowers stakeholders to make data-informed decisions in urban governance and planning.

Mathematical Framework of Imputation Algorithms

1. Random Forest Regression. Random Forest is an ensemble learning method that constructs a multitude of decision trees during training and outputs the average prediction of the individual trees to reduce variance and improve generalization [106]. For a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, the random forest estimator computes:

$$\hat{y}_t = \frac{1}{T} \sum_{i=1}^{T} h_i(\mathbf{x}_t) \tag{3.1}$$

where T is the total number of trees and h_i represents the prediction from the i-th tree. Random Forest handles non-linearity, high-dimensionality, and inter-feature interaction, making it suitable for imputation when data show irregular patterns or multimodal trends.

2. k-Nearest Neighbors (k-NN). k-NN is a non-parametric algorithm that estimates missing values by identifying k closest complete samples based on a chosen distance metric (typically Euclidean). Given a sample $\mathbf{x}_i$ with missing value x_{ij}, its imputation is given by:

$$\hat{x}_{ij} = \frac{1}{k} \sum_{r \in \mathcal{N}_k(i)} x_{rj} \tag{3.2}$$

where $\mathcal{N}_k(i)$ is the set of k nearest neighbors to instance i with observed x_{rj} values. This approach is effective when spatial or temporal proximity implies data similarity, as is often the case with environmental sensor networks [107].

3. SARIMAX (Seasonal AutoRegressive Integrated Moving Average with Exogenous Variables). SARIMAX extends ARIMA models by incorporating both seasonality and external regressors. The general model for an observation y_t is:

$$\Phi_P(B^s)\phi_p(B)(1-B)^d(1-B^s)^D y_t = \Theta_Q(B^s)\theta_q(B)\varepsilon_t + \beta^\top \mathbf{X}_t \tag{3.3}$$

where:

- $\phi_p(B)$ and $\theta_q(B)$ are the non-seasonal AR and MA polynomials,
- $\Phi_P(B^s)$ and $\Theta_Q(B^s)$ are the seasonal AR and MA polynomials,
- d and D are the orders of differencing,
- $\mathbf{X}_t$ is a vector of exogenous variables, and
- β represents the coefficients of the external regressors.

It is well-suited for pollution data which exhibit both cyclical patterns (e.g., daily peaks) and dependencies on exogenous factors like traffic or meteorology [108]. This hybrid ensemble of statistical and machine learning models combines the interpretability of SARIMAX with the flexibility of non-parametric learning methods, providing robust handling of various data behaviors and enhancing the resilience of smart city sensor analytics.

Chapter 4

Experimental Protocol

This chapter elaborates on the technical implementation and empirical outcomes of the proposed smart city data analysis and visualization platform, detailing its core architectural layers, underlying design principles, and resulting insights from real-world sensor data. The implementation consists of three interconnected components: (1) dataset acquisition and structured storage, (2) backend infrastructure for data processing, imputation, and API orchestration, and (3) a frontend layer for interactive visualizations and user interactions. Each component is engineered with modularity and scalability, allowing independent development, testing, and updates. Performance optimization strategies, such as efficient data caching, rendering, and endpoint runtime evaluations (including execution time and memory usage benchmarks), ensure smooth operation in resource-constrained, browser-based environments. The accuracy and reliability of data imputation methods are evaluated using robust statistical metrics, including RMSE and MAE, to ensure data quality and analytical integrity. Systematic dataset versioning, standardized schemas, and modular code organization further support reproducibility, aligning with best practices in reproducible data science and civic informatics [93], [102]. Collectively, these design choices facilitate a scalable, robust, and ethical smart city infrastructure capable of adapting to evolving urban sensing technologies, stakeholder requirements, and analytical objectives.

The outcomes from the implementation highlight the effectiveness of the interactive visualization system, showcase practical use cases derived from actual sensor datasets, and provide interpretative insights into urban phenomena. The discussion is structured into three key parts: visualization, responsiveness, and interactivity, domain-specific use cases, and analytical implications and limitations.

4.1 Dataset Management and Description

The datasets utilized in this study originate from publicly accessible repositories managed by the city of Basel, Switzerland[1], accessed on 25 August 2023. These datasets include detailed time-series records for various urban sensing modalities such as air quality (NO2, PM2.5), traffic volume and speed, environmental noise, parking occupancy, and electric vehicle (EV) charging activities. Data are provided in standardized CSV format along with comprehensive metadata.

Datasets are stored locally to ensure analytical consistency and to mitigate runtime disruptions. Several factors justify this approach: firstly, portions of Basel's sensor infrastructure, particularly for

[1] `https://data.bs.ch/explore/?refine.tags=smarte+strasse`

parking and noise monitoring, are intermittently offline or decommissioned, affecting reliable real-time ingestion [31]. Secondly, local storage mitigates common network issues, such as latency, API throttling, schema drift, and fluctuating availability of open data portals [89], [109].
Local storage facilitates rigorous preprocessing, modeling, and experimentation, supporting reproducible empirical machine learning and urban informatics research. Datasets are maintained with meticulous version control in a sandboxed environment, enhancing reliability for pipeline testing, parameter tuning, and comparative analyses without external interference or dependency changes [93], [110].
This offline data management strategy enables detailed schema validation, integrity checks, and historical backtesting, which are essential for robust evaluations of predictive algorithms. [98], [99]. Additionally, it aligns with the increasing adoption of synthetic and simulation-based environments, ensuring deterministic testing and reproducibility [53].
From a governance standpoint, isolating datasets from cloud infrastructure and external third-party APIs reduces risks related to data leakage, enhancing compliance with privacy regulations like GDPR [23], [38]. Thus, the platform adheres strictly to ethical urban sensing principles, ensuring secure, reliable data handling.
In summary, the dataset management strategy emphasizes data integrity, reproducibility, and security, providing a robust foundation for urban decision-making and academic research.

Figure 4.1: Overview of the sensors used in the Basel Smart Road project [111]: from left to right, an air quality sensor, a noise sensor, a traffic monitoring camera, and an electric vehicle charging station.
Before delving into the specific sensor modalities utilized in this application, it is essential to clarify that the current system version integrates only a curated subset of available urban sensing technologies. This decision was shaped by both the availability of high-quality, well-annotated datasets and the immediate urban challenges addressed by the project. Specifically, the application leverages sensor streams related to air quality, traffic and vehicle activity, acoustic pollution, parking utilization, and electric vehicle charging infrastructure. These categories were selected for their established relevance in urban health monitoring, transport optimization, and sustainability planning [32], [89], [95].
Air quality sensors, including NO_2, O_3, and $PM_{2.5}$, are central to environmental health research due to their strong associations with respiratory and cardiovascular outcomes, as documented in numerous longitudinal cohort studies and World Health Organization guidelines [67], [95], [112]. Traffic-related emissions and meteorological conditions significantly influence pollutant dispersion, which makes high-resolution spatiotemporal monitoring essential [31], [53]. These sensors are widely used

in urban risk assessment, regulatory compliance, and forecasting systems designed to preempt exposure peaks [66], [68].

Mobility data, derived from vehicle counters and speed detectors, offers insight into congestion levels, modal share, and street-level performance. These data streams are critical for dynamic traffic management, infrastructure planning, and emissions modeling [21], [46], [103]. Temporal patterns in vehicle activity, such as peak-hour surges or weekend shifts, can be correlated with pollution metrics and urban accessibility measures.

Environmental noise is a key urban health and livability indicator, with chronic exposure to elevated sound levels increasingly linked to adverse outcomes such as stress, insomnia, cognitive impairment, and cardiovascular disease [48], [95]. In this system, sound sensors are deployed to monitor both broadband sound pressure levels (measured in dB) and spectral frequency distributions, allowing for fine-grained characterization of the urban acoustic landscape. Unlike traditional noise monitoring stations that only report aggregate dB values, the sensors here support time-resolved, source-specific detection. The captured raw audio signals are processed using a Sound Event Detection pipeline, transforming continuous waveforms into temporally segmented and labeled sound events. As shown in Figure 4.2, the system can differentiate between overlapping acoustic sources such as speech, car passing by, and bird singing, by applying machine learning models trained on annotated urban sound datasets [113], [114]. These models often rely on spectrogram-based feature extraction (e.g., MFCCs or log-Mel features) and sequence modeling using RNNs, CNNs, or attention-based architectures [115], [116]. This capability supports semantic acoustic monitoring, enabling the system to separate traffic-related noise from other sources such as construction, voices, or natural sounds. It enhances urban zoning analysis, allowing decision-makers to identify noise hotspots, enforce zoning compliance, and evaluate the effectiveness of noise-reduction measures (e.g., traffic rerouting, sound barriers) [30], [73], [117]. Moreover, by recording frequency-specific information, the system can distinguish between low-frequency rumble from trucks and high-frequency disturbances such as alarms or sirens, essential for responsive and context-aware city planning [118].

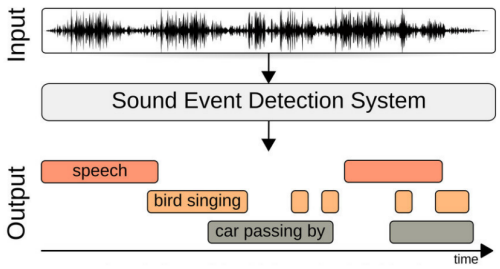


Figure 4.2: Illustration of the Sound Event Detection process used by urban acoustic sensors. The raw audio waveform (top) serves as the input, representing ambient sounds captured in real-world city environments such as streets or public areas. This waveform is processed by a machine learning-based system, which extracts acoustic features and identifies temporal segments corresponding to specific sound categories. The output (bottom) displays labeled time intervals for events like *speech*, *bird singing*, and *car passing by*. Multiple overlapping events can be detected simultaneously, providing a fine-grained understanding of the soundscape. This process enables urban systems to distinguish traffic-related noise from background sounds, supporting applications in environmental monitoring, noise pollution assessment, and smart mobility analysis [111].

Parking and EV infrastructure sensors offer a complementary perspective on how urban space is utilized and how the transition toward electrified mobility is progressing. This study's monitored area includes white-line and yellow-line parking spots, each governed by distinct usage policies based on Basel-Stadt's municipal parking regulations. White-line zones typically denote public parking spaces available without restriction, although time limits or payment rules may apply. In contrast, yellow-line spots are designated for restricted use, allowing legal parking only during specific working hours, often limited to commercial deliveries, employees, or short-term loading [119]. This dual-spot monitoring strategy, which is illustrated in Figure 4.3 (Figure 4.3), enables a differentiated analysis of spatial demand and temporal occupancy patterns, revealing not only usage intensity but also regulatory compliance and turnover behavior. These patterns are essential for optimizing parking allocation, minimizing misuse, and informing time-based pricing models [49], [74]. Additionally, electric vehicle (EV) charging sensors complement this data by recording charging durations and delivered kilowatt-hours, offering insight into infrastructure utilization, equity in spatial access, and long-term energy planning [51], [52].

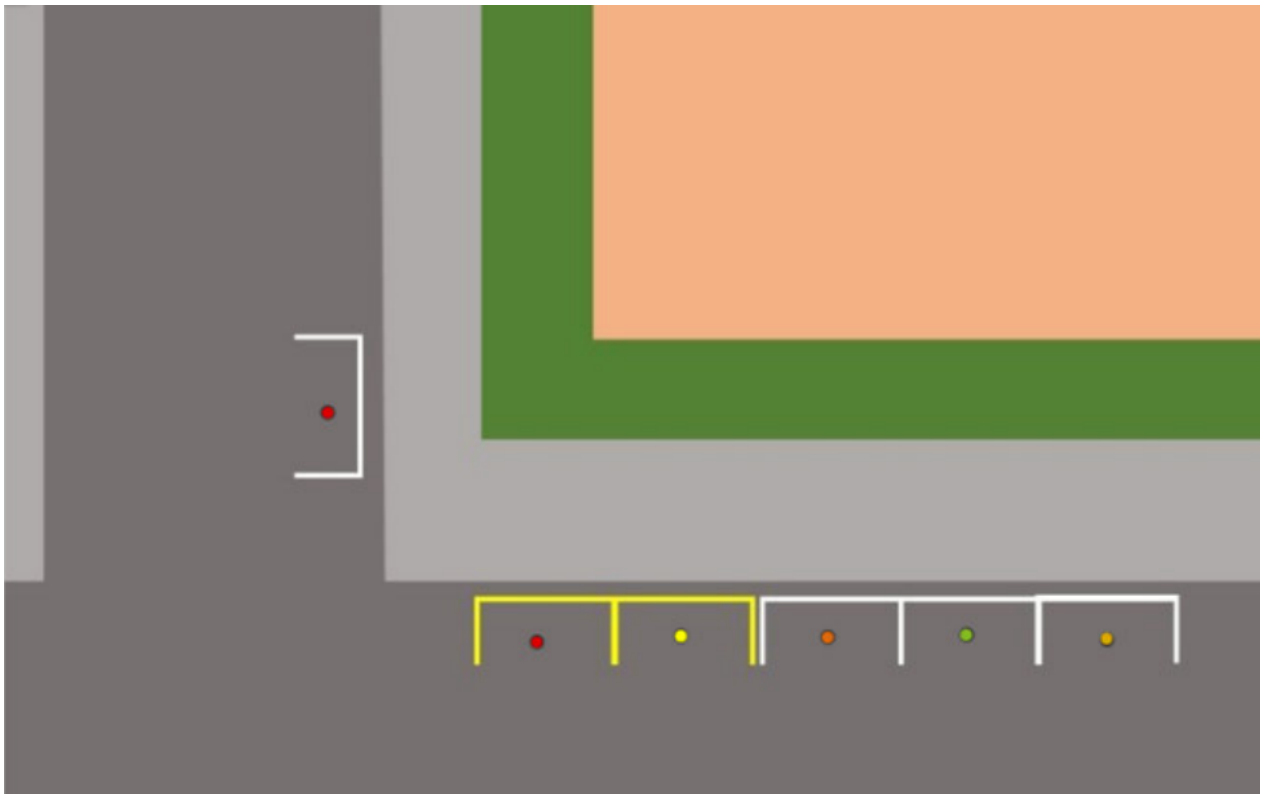


Figure 4.3: Visual layout of the monitored parking area. These specific spots are covered by the fixed-position camera used for detecting occupancy [111].

These carefully selected sensor categories not only align with international research priorities in smart cities but also support scalable, interpretable, and ethically governed urban analytics [23], [38], [54]. The current implementation balances operational feasibility with analytical power and establishes a modular baseline for future extensions into domains such as air humidity, waste tracking, or urban greenery monitoring.

4.2 Data Layer

The data layer of the proposed platform constitutes the backend engine that ingests, processes, transforms, and serves heterogeneous urban sensor datasets in a modular and scalable manner. Developed using the Python programming, the system is built atop the Flask microframework due to its lightweight design, extensibility, and robust community support [120], [121]. Flask orchestrates static content rendering and dynamic application programming interface (API) endpoints, providing an integrated stack that supports data-driven urban informatics.

At its architectural core, the backend separates responsibilities between data logic, visualization routing, and statistical computation. This separation of concerns aligns with best practices in full-stack web development [122], allowing scalable maintenance and clear extensibility pathways, such as onboarding new sensor types or linking live data feeds from IoT gateways [32], [89]. For instance, sensor-specific API routes are defined modularly using Flask Blueprints, allowing for decoupled development and testing.

The data layer is optimized for handling preprocessing tasks such as data normalization, timestamp alignment, schema validation, and missing value imputation. Imputation is performed using a hybrid strategy combining statistical forecasting (SARIMAX from the Statsmodels library), local neighborhood learning (via k-NN from Scikit-learn), and ensemble regression (Random Forest-based imputation with MissForest) [36], [98], [99]. This approach ensures temporal coherence and multivariate consistency across sensor modalities, a common challenge in real-world urban data streams [54].

Once imputed, the data is subjected to aggregation procedures, including computation of descriptive statistics (min, max, mean, median, standard deviation, quartiles) and time-binning. These operations are performed using Pandas and NumPy [93], [123], which support high-performance array and table computations. Results are serialized into JSON-compatible structures and returned to the frontend for rendering. To support exploratory analytics, a dedicated endpoint computes correlation matrices across all numeric features. These matrices allow visualization of interdependencies between sensor categories (e.g., traffic volume vs. NO_2 levels or noise vs. speed) and support cross-domain policy analysis [24], [53]. Caching is a core component of the backend design. Using Flask-Caching and Redis integration, the system avoids redundant execution of expensive functions such as imputation, correlation, or cross-temporal aggregation. This reduces latency and enables fast response times for frontend visualizations, even when handling multi-megabyte CSVs or complex computation chains [102]. Finally, a custom logging and profiling utility tracks the execution time of every request route. These logs aid in performance diagnostics and enable system administrators to optimize route execution, load balancing, and compute resource allocation over time. Overall, this backend architecture combines modern data science tooling, scalable API design, and computational rigor to deliver a resilient and extensible data infrastructure for smart city analytics.

4.3 Frontend Interface and UX Design

The frontend interface of the smart city analytics platform is meticulously designed to facilitate seamless human-data interaction, enabling users to explore complex urban phenomena with clarity, flexibility, and responsiveness [124]–[126]. Built with a user-centric philosophy, the interface supports both exploratory and operational tasks by integrating intuitive interaction paradigms and adaptive design principles grounded in human-computer interaction (HCI) theory [127], [128]. Usability heuristics

such as consistency, visibility of system status, error prevention, and user control are consistently applied throughout the platform's interface design [125], [129].

The design framework prioritizes responsive design and modular componentization, enabling interface elements such as dynamic charts, statistical dashboards, and real-time maps to adapt fluidly to a diverse range of screen resolutions and device types [130], [131]. Leveraging mobile-first principles and scalable vector-based rendering technologies, the application maintains pixel-precision responsiveness across desktops, tablets, and smartphones [132]. High-performance visualization libraries such as Plotly.js and Leaflet.js are used to render temporal and geospatial data interactively, supporting exploratory analysis through user-driven interactions like zoom, drag, lasso selection, and hover tooltips [91], [104], [105].

The layout architecture features a persistent navigation bar that provides seamless access to domain-specific dashboards, each dedicated to a particular sensor modality such as NO_2, $PM_{2.5}$, traffic flow, or parking utilization [21]. This top-level consistency reduces cognitive load, while the modular dashboard layout accommodates progressive disclosure of information, from high-level statistics to low-level data tables and heatmaps [88]. Sensor-specific maps are populated with color-coded, clickable markers whose styling reflects real-time or historical data states, enhancing geospatial legibility [133].

Visual cognition principles drive the platform's use of layout spacing, typographic hierarchy, and color palette design. Colors are semantically encoded (e.g., gradient scales for pollutant concentration) and verified against color blindness simulators to maximize accessibility [134], [135]. Contrasts are managed using WCAG 2.1 AA standards to ensure legibility in varied lighting conditions [136]. Hierarchical layering, opacity scaling, and tool-assisted annotations promote visual clarity when multiple variables are overlaid. AJAX-based asynchronous data requests allow for live interaction with backend services without reloading the page, maintaining application state while streaming new data or user-selected filters [137], [138]. This is particularly important for visualizing time-evolving data and enabling linked-view updates across dashboard widgets [139]. A global CSS-based design system ensures branding consistency and enforces style uniformity across UI components such as dropdowns, toggles, and sliders. All form elements adhere to ARIA roles and keyboard navigability standards to support users with disabilities [136], [140]. Dynamic layout breakpoints adapt to various viewport sizes to ensure interface readability and touch control support for mobile contexts. Overall, this frontend implementation operationalizes universal design principles to support equitable, insightful, and real-time interaction with smart city sensor data [141]. Through rigorous design conventions, responsive graphics pipelines, and accessibility compliance, the interface serves as a comprehensive tool for policy analysts, urban researchers, and civic stakeholders alike.

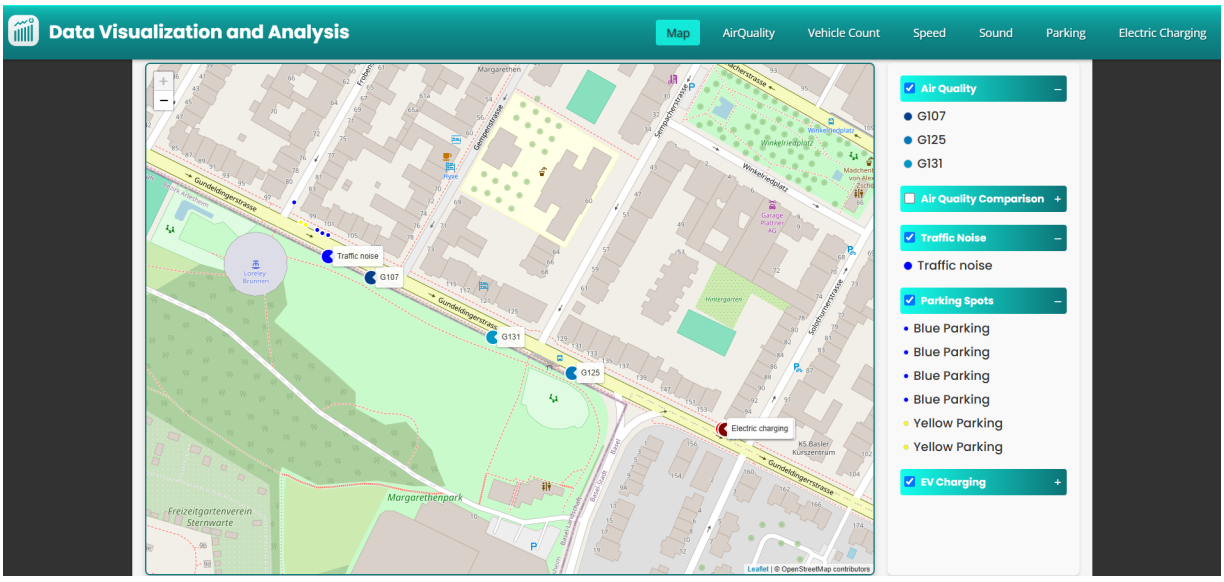


Figure 4.4: Interactive map view of the application visualizing multiple urban sensor categories. Color-coded markers with permanent labels identify air quality stations (G107, G125, G131), comparative air quality reference points, traffic noise sensors, electric vehicle charging stations, and individual parking spots, which are denoted by small blue and yellow dots. Users can toggle sensor categories using checkboxes, which dynamically adjust the map view to focus on selected features. Integrating zoom-level control, real-time filtering, and semantic labeling enables fast and intuitive exploration of key urban infrastructure data.

4.4 Interactive Visualization Features

One of the platform's most prominent features is its real-time, interactive visualizations that empower users to engage directly with urban data through responsive and modular dashboards [91], [104]. The interactive components are embedded across multiple views, including air quality, noise pollution, traffic density, and parking utilization, and are optimized to support user-specific filtering, comparative analysis, and spatial awareness [21], [88]. The visualization system supports toggling of data layers, allowing users to select or deselect sensor stations or variable categories within line charts, bar plots, and heatmaps. This interactivity reduces cognitive overload by decluttering the display and enabling precision-driven insights [127]. For instance, in the air quality dashboard (Figures 4.5 and 4.7), toggling $PM_{2.5}$ and NO_2 readings by station allows localized comparisons across neighborhoods. Responsive charts are further enhanced by embedded median lines and percentile shading, allowing the user to quickly identify statistical outliers or anomalous trends [134]. Visual encodings are informed by perceptual research, ensuring that hue, size, and spacing provide intuitive data cues without requiring textual labels [135]. Dashboards also allow dynamic resizing and minimization of components. These features provide flexible layouts that support a variety of user preferences, facilitating both macro- and micro-level explorations [130]. Time-series visualizations (e.g., Figure 4.8) dynamically align sensor readings across temporal windows, offering context-aware comparisons.
A key analytic component is the system's correlation heatmap (Figure 4.9), which visually encodes Pearson correlation coefficients across pollutant and traffic variables. This matrix reveals positive

and negative relationships, enabling hypothesis generation for multivariate modeling or policy simulation [139]. Asynchronous data loading through AJAX prevents full-page reloads when navigating between tabs or adjusting filters, resulting in a smoother and uninterrupted exploratory experience [137], [138]. This is particularly valuable in longitudinal analyses, where users may toggle between daily and weekly aggregation levels without incurring performance lags. Stylistically, charts maintain WCAG-compliant contrast and font scalability to ensure readability across devices and screen resolutions [136]. Colorblind-safe palettes and hover-tooltips enhance accessibility and facilitate interpretation of complex datasets, especially when multiple metrics co-exist on a single chart [140], [141]. Together, these visual tools form a cohesive interactive environment in which users can explore, filter, and interpret rich urban datasets in a seamless and personalized manner. Whether the goal is anomaly detection, trend forecasting, or infrastructure planning, the system's visual analytics framework empowers both lay users and domain experts to derive actionable insights from raw sensor streams [53], [54].

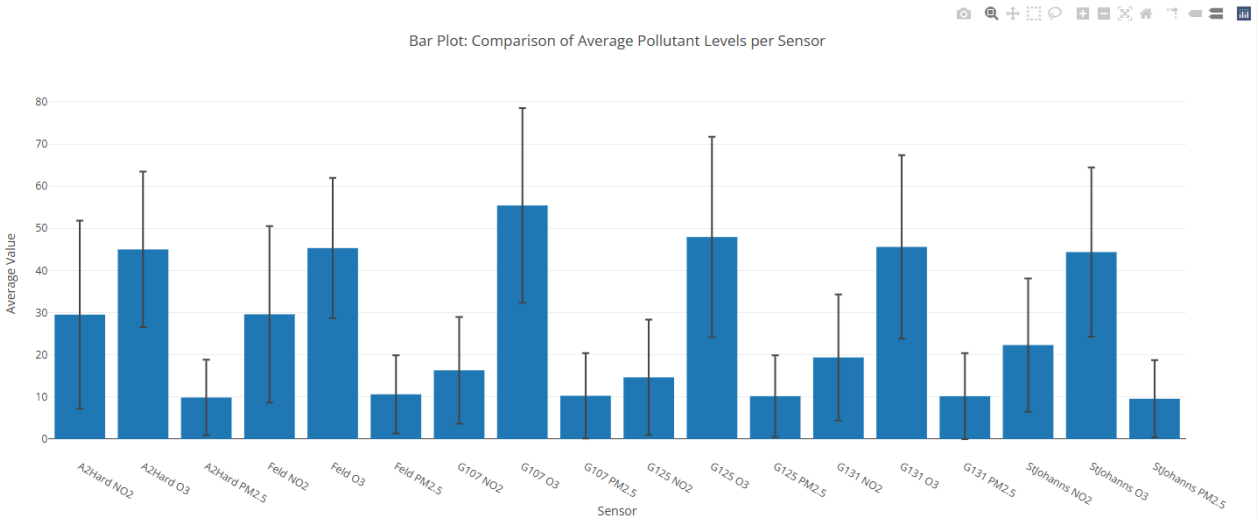


Figure 4.5: Bar plot illustrating average pollutant levels (NO_2, O_3, $PM_{2.5}$) captured across various sensor locations. Error bars denote standard deviation, reflecting measurement variability. Comparative assessment supports spatial analysis of air quality across the urban network.

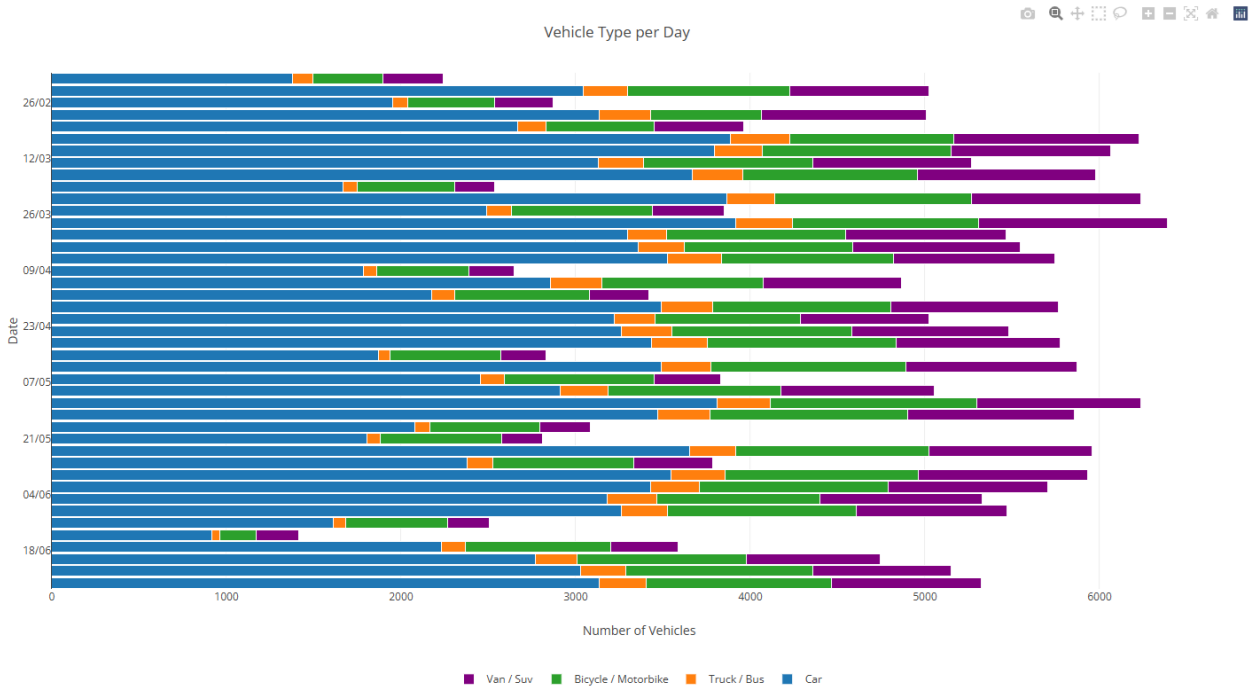


Figure 4.6: This horizontal stacked bar chart illustrates the composition of different vehicle categories recorded on selected days. Each bar represents the total number of vehicles detected per day, categorized into four types: cars (blue), trucks or buses (orange), bicycles or motorbikes (green), and vans or SUVs (purple). The figure highlights the predominance of standard cars in daily traffic volumes, while also capturing the recurring contribution of heavy vehicles, two-wheelers, and light commercial vehicles. This breakdown facilitates analysis of modal share trends, enabling transportation planners to assess usage diversity and evaluate infrastructure demand for each vehicle class.

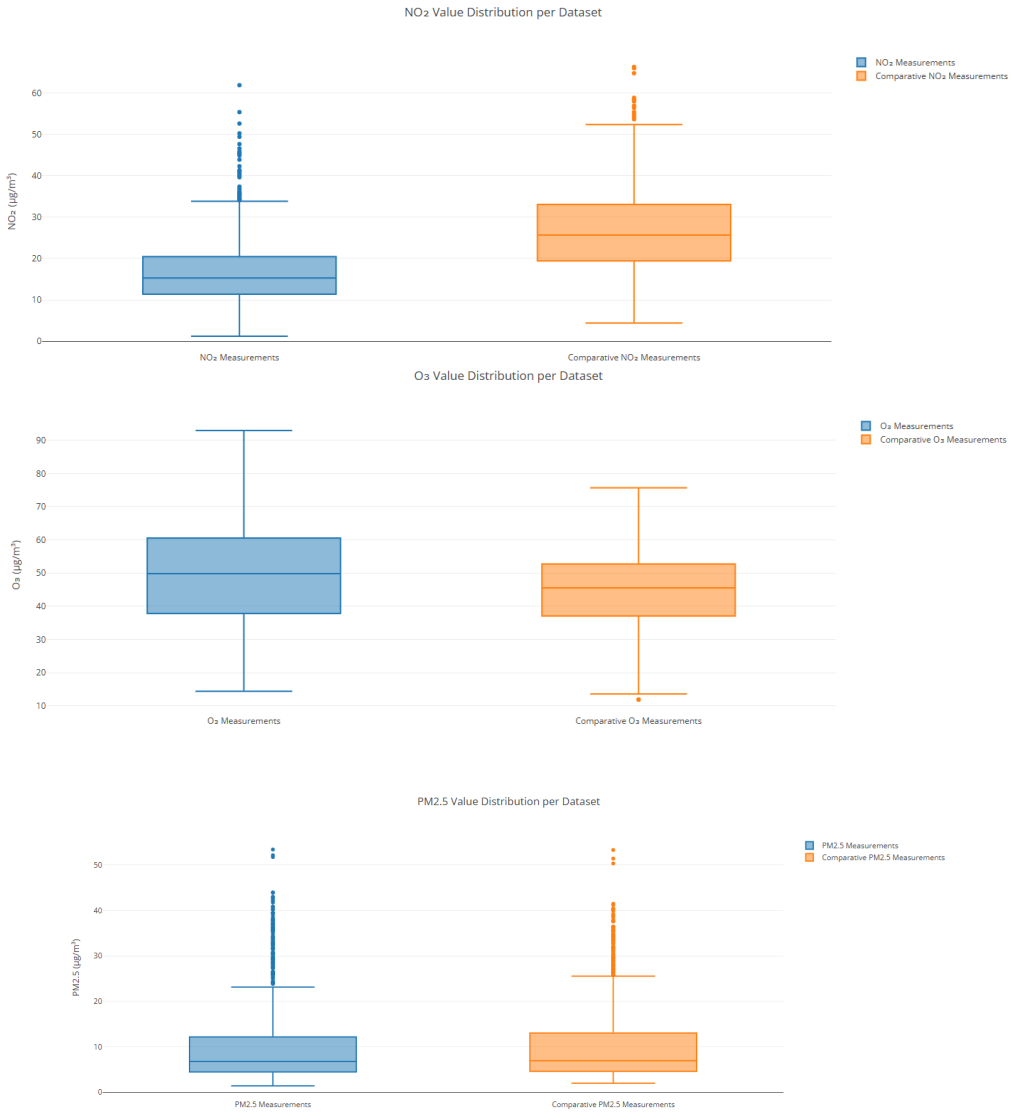


Figure 4.7: Comparative distribution of air quality pollutants (O_3, $PM_{2.5}$, and NO_2) between sensor datasets. Boxplots compare the value distributions of three major air pollutants: ozone (O_3), particulate matter ($PM_{2.5}$), and nitrogen dioxide (NO_2) across two datasets: one collected from the primary urban sensor network (blue) and the other from comparative reference stations (orange). The O_3 measurements show a wider range and higher upper whisker in the primary dataset, indicating sporadically elevated ozone concentrations. $PM_{2.5}$ distributions are relatively similar in median values, but the comparative dataset exhibits greater variability and more outliers. For NO_2, the reference data show a significantly higher median and broader distribution, suggesting denser traffic exposure or local emission sources.

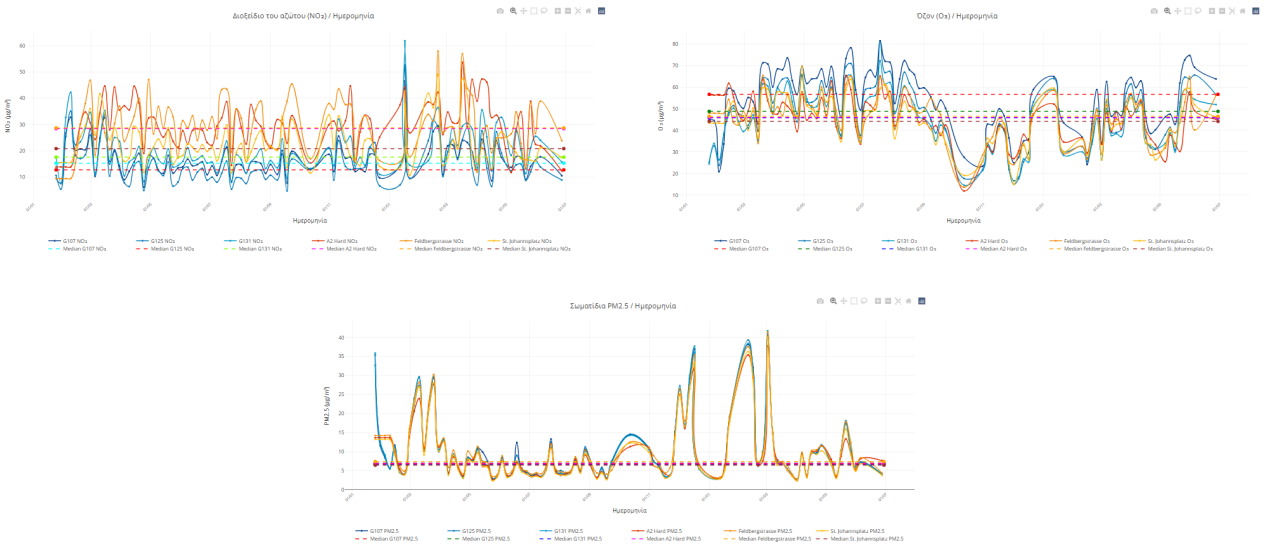


Figure 4.8: Time-series visualizations from three air quality sensors (e.g., G107, G125, G131) used in the Basel Smart Road project. Each chart displays pollutant levels (e.g., NO_2, O_3, $PM_{2.5}$). The X-axis represents timestamps, and the Y-axis shows $\mu g/m^3$ concentrations. Median reference lines highlight central tendencies, helping to identify trends, spikes, and deviations across different monitoring locations.

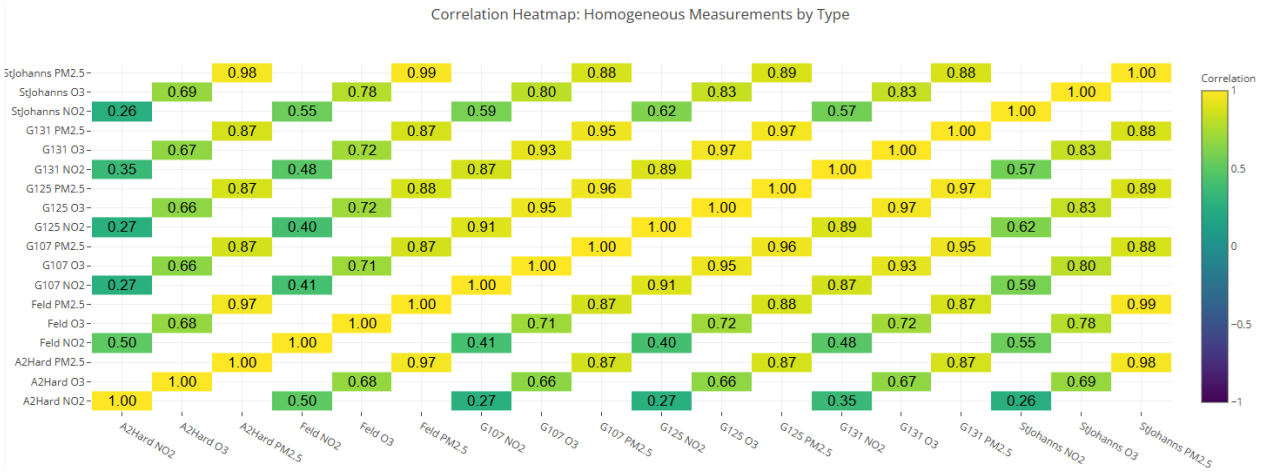


Figure 4.9: Correlation heatmap visualizing the degree of correlation between different sensors and pollutants(O_3, NO_2, $PM_{2.5}$). The color scale on the side reflects correlation values ranging from -1 to +1, with cooler tones (e.g., purple) indicating weaker or negative correlations and warmer tones (yellow) signifying strong positive correlations. Each cell represents the relationship between two specific measurements of the same type (e.g., "G107 NO_2", and "StJohanns NO_2"). Strong intra-pollutant clustering often reflects shared atmospheric behavior or familiar emission sources. Weak or negative correlations indicate local environmental differences or independent dynamics of pollutants. Overall, the heatmap provides a quick visual summary of which pollutant measurements tend to rise and fall together, highlighting meaningful relationships in urban air quality patterns.

4.5 Use Cases and Data-Driven Insights

The real-world applicability of the platform is demonstrated through several use cases involving environmental monitoring, urban planning, and infrastructure optimization. These examples highlight how city administrators, researchers, and citizens can derive practical value from data visualizations and analytics workflows.

In air quality monitoring, daily aggregates of NO_2, O_3, and $PM_{2.5}$ levels enable granular tracking of pollution distribution across neighborhoods. High-pollution periods, such as rush hours or industrial peaks, are visually distinguishable, offering timely insight into exposure risks [67], [95]. Through sensor-to-sensor comparisons, anomalies can be flagged when a specific station deviates from baseline trends, potentially indicating faulty sensors, localized pollution events, or sensor drift [65], [66].

Traffic sensor data, such as vehicle count and speed measurements, that support congestion analysis and mobility planning. By observing these variables alongside noise pollution levels (as seen in Figure 4.12), decision-makers can locate mobility bottlenecks and noise-intensive zones [30], [103]. These insights aid in adjusting traffic light timing, plan road upgrades, or restrict access during sensitive hours.

Parking utilization and EV charging datasets provide rich feedback on temporal usage trends (Figure 4.13). Monitoring vehicle turnover and charge session durations helps optimize parking policy and charging infrastructure, particularly in high-demand districts [49], [51]. These insights support predictive maintenance and load balancing across the EV grid [52].

Furthermore, the public-facing nature of the dashboards democratizes access to urban metrics, encouraging community-driven participation in environmental stewardship [73]. By visualizing key metrics like air pollution and EV usage through intuitive graphs and maps, the platform fosters civic literacy and behavioral change. Citizens can view real-time air quality trends or parking space availability, reinforcing data transparency and facilitating behavioral nudges [21], [38].

In combination, these domain-specific applications showcase the power of contextual, real-time sensor integration in creating actionable insights for urban governance. They underscore the platform's ability to support both high-level planning and granular, neighborhood-scale interventions.

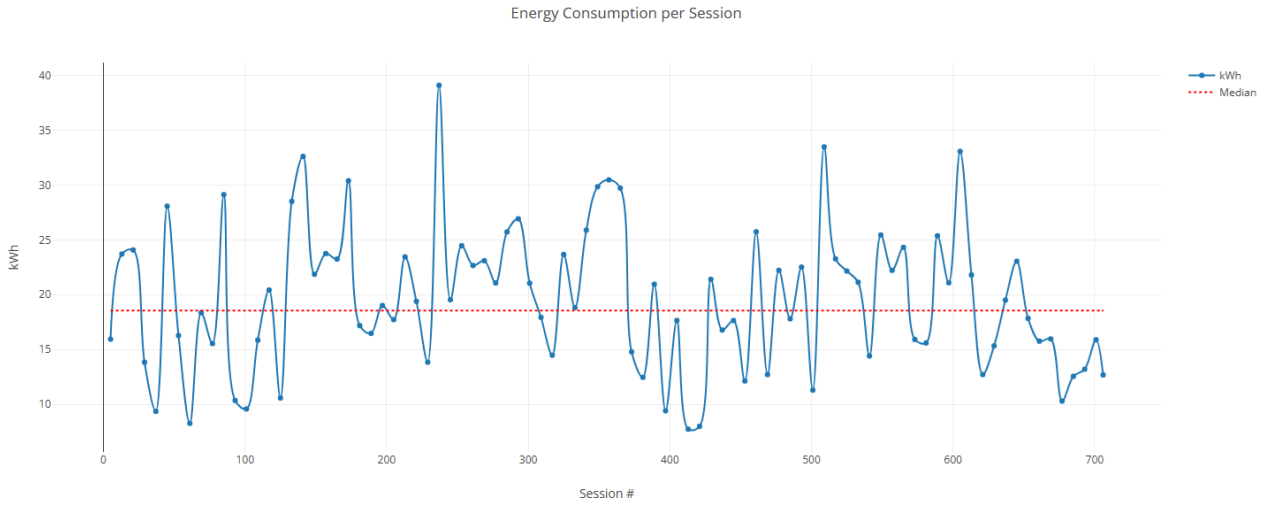


Figure 4.10: Time-series representation of energy consumption per electric vehicle charging session. Each point corresponds to the unique sessions' energy usage in kWh, with the red dashed line denoting the median value. Peaks and fluctuations illustrate diverse charging behaviors among users.

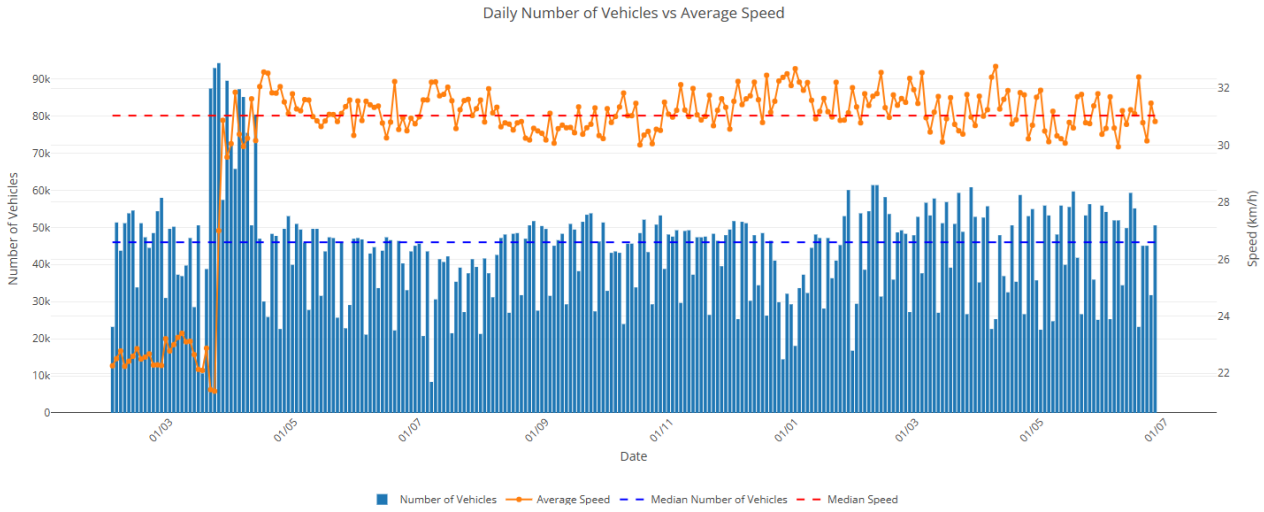


Figure 4.11: Daily number of vehicles vs. average speed. This plot displays the daily vehicle count (blue bars, left axis) in conjunction with the average vehicle speed (orange line, right axis). Dashed horizontal lines indicate median values. Although vehicle volume remains relatively stable throughout the period, fluctuations in speed are observed, particularly during peak periods of congestion or reduced flow. The visualization aids in understanding traffic dynamics High vehicle counts correspond to lower average speeds, indicating potential congestion or the effects of traffic regulation.

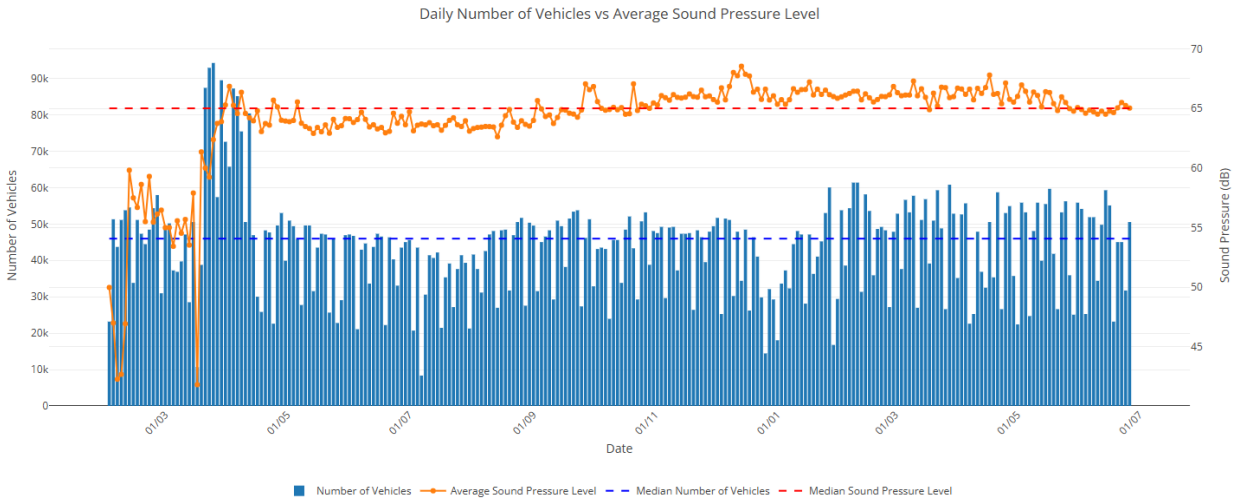


Figure 4.12: Daily number of vehicles vs. average sound pressure level. This plot illustrates the relationship between the daily number of vehicles (blue bars, left axis) and the average sound pressure level (orange line, right axis) recorded at a monitoring location. Horizontal dashed lines represent the median values of each respective dataset. A strong alignment between peaks in vehicle count and elevated noise levels suggests a correlation between traffic intensity and urban sound pollution. The dual-axis chart effectively highlights how increased traffic volumes tend to coincide with higher average decibel levels over time.

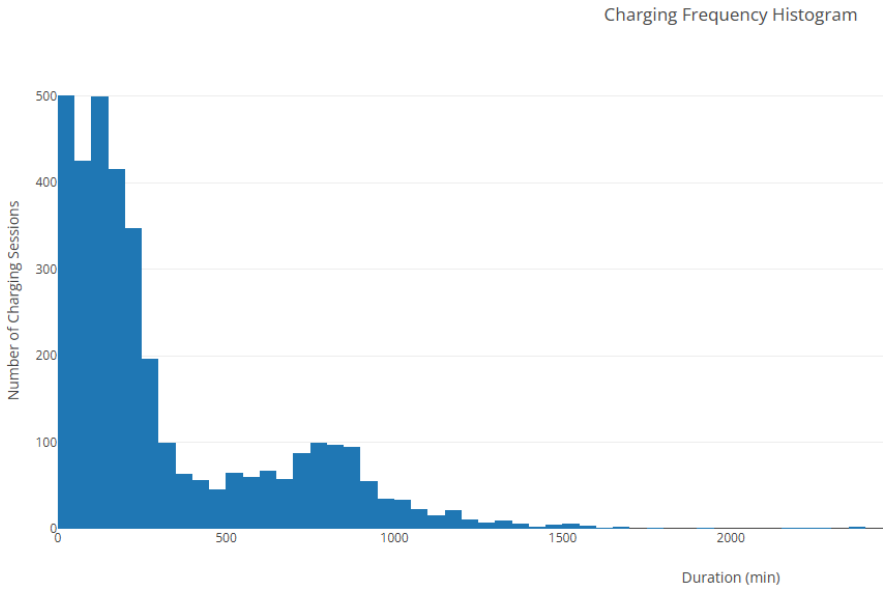


Figure 4.13: Histogram showing the distribution of electric vehicle charging session durations. The data indicate that most sessions are relatively short, with a high frequency of charges under 200 min. Longer sessions occur less frequently, exhibiting a heavy right-tail characteristic of usage skew.

4.6 Evaluation

The smart city platform was evaluated from two complementary perspectives: the accuracy of the hybrid imputation framework and the system's runtime performance. To evaluate the accuracy of imputation, two commonly used metrics in time-series and environmental modeling were employed: Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), both of which are foundational tools in statistical learning and data quality assessment [142], [143].
RMSE is sensitive to large errors due to its squaring component, thereby penalizing models heavily when they produce imputed values far from the actual measurement. It is calculated as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{t=1}^{N}(Y_t - \hat{Y}_t)^2} \tag{4.1}$$

where Y_t is the true observed value, $\hat{Y}_t$ the imputed value, and N the number of observations. RMSE provides a quadratic scoring rule that is particularly effective in reflecting the model's sensitivity to large deviations; thus, it is often used when large errors are especially undesirable in application contexts [108], [144].
MAE, on the other hand, measures the average magnitude of errors in a set of predictions, without considering their direction, using the absolute value of the differences. It is defined as:

$$\text{MAE} = \frac{1}{N}\sum_{t=1}^{N}|Y_t - \hat{Y}_t| \tag{4.2}$$

MAE is less sensitive to outliers than RMSE and provides a more interpretable measure in cases where all errors are treated equally [142], [145]. Its simplicity makes it especially useful in evaluating the average quality of imputation without the influence of extreme deviations.
In environmental sensor contexts such as air quality, both metrics are commonly applied in tandem, providing complementary perspectives on estimation accuracy. For example, low MAE but high RMSE may indicate that while most values are predicted reasonably well, a few high-magnitude errors are distorting the RMSE. Thus, both are essential to fully understand model performance and identify underlying noise or calibration discrepancies in the data [54], [146].
Furthermore, RMSE aligns with the Euclidean norm (L2 loss) in optimization settings, while MAE corresponds to the Manhattan norm (L1 loss), linking them to broader frameworks in regression modeling and regularization techniques [147]. RMSE is differentiable and therefore better suited for gradient-based optimization, whereas MAE provides robustness in the presence of non-Gaussian noise, which is particularly useful when dealing with urban environmental datasets prone to sensor drift, signal dropout, or abrupt changes [102].
These metrics assess the quality of imputed data generated through the platform's hybrid approach, combining SARIMAX, k-Nearest Neighbors, and Random Forest regressors. As demonstrated in Table 4.1, results showed strong imputation performance across most sensors, especially for $PM_{2.5}$, and provided a reliable foundation for downstream visualization and policy decision-making tasks.
Table 4.1 provides a detailed summary of the MAE and RMSE scores across pollutants and sensor locations. The consistency of low-error values in $PM_{2.5}$, with RMSE below 3 $\mu g/m^3$ in all tested sensors, highlights the reliability of hybrid imputation in low-noise particulate datasets. NO_2 exhibits higher error variance due to short-term volatility and proximity to traffic-saturated zones.

Table 4.1: Imputation accuracy per sensor and pollutant (air quality).

Sensor–Pollutant	**MAE**	**RMSE**
A2Hard NO_2	5.374	8.049
Feld NO_2	4.794	6.746
G125 O_3	4.922	6.083
G107 O_3	4.699	5.865
StJohanns O_3	4.515	5.707
A2Hard O_3	4.301	5.653
G131 O_3	4.534	5.598
Feld O_3	3.807	4.945
StJohanns NO_2	3.403	4.832
G131 NO_2	3.178	4.689
G125 NO_2	2.652	3.926
G107 NO_2	2.508	3.783
G131 $PM_{2.5}$	1.827	2.895
G125 $PM_{2.5}$	1.781	2.580
Feld $PM_{2.5}$	1.757	2.428
G107 $PM_{2.5}$	1.729	2.372
A2Hard $PM_{2.5}$	1.704	2.327
StJohanns $PM_{2.5}$	1.667	2.302

To evaluate system responsiveness and resource usage, execution time and memory consumption of key API endpoints under realistic workloads was measured. These measurements reflect the platform's ability to efficiently deliver insights without imposing significant computational overhead.

Table 4.2 summarizes execution benchmarks for the most critical backend services, such as correlation matrix generation, audio spectrum processing, and parking data aggregation. Execution times varied from 0.01 seconds for static data endpoints to 8.67 seconds for complex audio feature extraction. Correspondingly, memory usage ranged between 2.4 MB and over 400 MB, depending on the nature and complexity of the data processed.

These findings demonstrate that the platform's caching strategies (via Flask-Caching) and data downsampling procedures effectively maintain responsiveness even for compute-intensive tasks. The endpoints were optimized to avoid repeated computations through memoization and to load only the minimal necessary dataset slices.

Despite the lack of formal usability trials, internal testing across multiple devices (desktop, tablet, and mobile) verified consistent rendering speeds, stable navigation, and cross-browser compatibility. These results support the design goal of building an accessible, scalable, and multi-stakeholder-friendly platform suitable for urban deployment scenarios.

Table 4.2: Performance evaluation of selected API endpoints.

Endpoint	Execution Time (s)	Memory Start (MB)	Memory Increase (MB)
`get_spso_data`	0.01	165.12	–
`get_echarging_data`	0.13	183.73	62.21
`get_corr_matrix`	0.46	165.10	84.09
`get_road_data`	0.66	165.04	135.11
`get_vehicle_data`	2.69	165.11	419.95
`get_soundHz_data`	8.67	180.53	419.28
`get_parking_data`	0.51	388.28	2.43

4.7 Discussion

As the proliferation of smart city infrastructures accelerates, the reliance on pervasive sensing and data-driven decision-making must be matched by a rigorous evaluation of their limitations and broader implications. While the system presented in this thesis demonstrates significant potential in aggregating and visualizing multisource urban data, it also reveals several challenges intrinsic to real-world deployments. One of the most pressing issues is sensor data quality. Low-cost sensors are widely adopted for scalability, but they often lack reference-grade equipment's long-term precision and durability. Environmental variables such as humidity, pollution accumulation, and temperature shifts can significantly impact sensor accuracy. Our deployment revealed deviations in speed measurement and object classification rates, echoing concerns raised in broader studies that suggest routine calibration and self-diagnostic routines are essential to any urban sensing strategy [65]. Another recurring concern is missing or intermittent data. In sensor-based environments, data gaps emerge frequently due to battery drain, hardware malfunctions, or communication losses. While imputation methods such as SARIMAX, k-Nearest Neighbors, and ensemble learning models help mitigate this, they rely on assumptions about temporal continuity and spatial correlation. When missingness coincides with anomalous events, such as power outages or natural disasters, standard imputation techniques may be inadequate, affecting immediate analytics and any predictive modeling or forecasting layered atop the dataset. Spatial equity in sensor deployment presents another layer of complexity. High-density areas tend to receive greater sensor coverage due to economic or political prioritization, while marginalized or low-income regions remain under-monitored. This uneven distribution risks reinforcing existing urban disparities by omitting key data from policy decisions. Integrating equity-aware sensor placement algorithms or applying spatial interpolation techniques could help alleviate this imbalance [59]. Ethical and legal dimensions are equally critical. Even when data is anonymized, movement patterns, occupancy, or infrastructure use can reveal sensitive behavioral trends. While our platform avoids invasive features such as facial recognition or license plate tracking, future expansions—particularly those involving image capture, audio classification, or real-time surveillance—must be evaluated through the lens of data minimization, consent, and transparency. Regulations such as the GDPR offer necessary guardrails, but technical solutions like differential privacy, federated learning, or edge-based filtering can add further safeguards. Visual privacy deserves special attention. Cameras and visual sensors, while rich in contextual information, raise concerns about the inadvertent capture of personally identifiable data. Our system currently avoids this by abstracting visuals and suppressing high-resolution features in maps and time-series plots. Nevertheless, as machine learning techniques for behavior prediction become more common, ensuring that visual data is treated with the same scrutiny as textual

or numerical data becomes vital [88]. Furthermore, interpretability and inclusivity must be prioritized in dashboard design. Data visualizations must not only be accurate but also understandable to a broad audience, including policymakers, residents, and advocacy groups. Interactive interfaces, tooltips, and semantic layers can improve usability, but they should be complemented by contextual explanations and decision support mechanisms. Finally, citizen participation remains a cornerstone of ethical smart city development. While sensor platforms often operate in the background, fostering active feedback loops with communities—through open APIs, transparency reports, or participatory sensing campaigns—can transform passive infrastructures into democratic instruments of urban co-creation. In conclusion, innovative city platforms, like the one developed in this thesis, reflect the immense promise of integrated data for urban optimization. However, they also foreground a spectrum of socio-technical challenges. Navigating these requires an interdisciplinary approach—combining engineering rigor with ethical foresight, participatory governance, and long-term urban resilience planning. In the next development phase, these values must be embedded not just in code and architecture but in institutional practices and public trust frameworks.

Figure 4.14: Sample output from the privacy-aware low-resolution camera used in the project. Despite limited fidelity, parking occupancy detection remained effective while protecting individual identities. The entire field of view, except for the parking spots, is blacked out to ensure no surrounding personal or identifiable information is visible [111].

Chapter 5

Conclusions

This thesis introduced a modular, web-based platform for real-time analysis and visualization of smart city sensor networks, utilizing heterogeneous datasets collected from the Basel Smarte Strasse project. By integrating air quality data, noise levels, vehicular metrics, parking availability, and EV charging logs, the system provides an accessible yet comprehensive window into the operational and environmental dimensions of urban life.

The platform's architecture supports flexible data ingestion, hybrid imputation for missing values and an efficient caching backend. The frontend offers an interactive interface combining geospatial maps, time-series plots, and correlation heatmaps that facilitate intuitive exploration of complex patterns. The outcome empowers urban planners, environmental analysts, and civil stakeholders with data-driven insights that guide infrastructure optimization, mobility planning, and environmental health interventions.

Among the primary contributions of this work is the successful demonstration of a unified data pipeline capable of ingesting and synchronizing multi-source sensor data streams with high temporal and spatial fidelity. This pipeline ensures reliability and responsiveness, making it suitable for both exploratory use and policy-critical applications in smart urban environments. Another important achievement is the integration of a hybrid imputation strategy that leverages statistical models and machine learning techniques to mitigate the impact of missing or noisy data. This hybrid approach preserves not only the temporal dynamics of each variable but also the correlations across different sensor types, thus ensuring analytical robustness.

Moreover, the project showcases advanced visualization methods that go beyond static plotting by offering dynamic, user-driven exploration of sensor data through interactive charts, geospatial overlays, and temporal dashboards. These features serve both novice users and expert analysts, transforming raw sensor logs into actionable insights through visual storytelling and spatial reasoning. Finally, the system adheres to ethical and regulatory best practices by embedding privacy-conscious design into its monitoring components. This includes selective data abstraction, anonymization mechanisms, and low-resolution imaging for parking surveillance, all aligning with GDPR and urban data governance principles.

5.1 Proposed Extensions and Future Work

To evolve the current proof-of-concept into a production-grade smart city monitoring platform, several technological, analytical, and governance-driven enhancements are proposed. A critical extension is the support for real-time streaming and dynamic alerting. Although full-scale stream processing tools like Apache Kafka or Apache Flink offer comprehensive functionality, an intermediate solution could involve implementing inotify-based file watchers to detect new data arrivals. These triggers can initiate background ingestion jobs and push real-time updates to the frontend using WebSockets, enabling threshold-based alerts (e.g., NO_2 levels exceeding 50 $\mu g/m^3$) for environmental monitoring and rapid response.

To ensure performance scalability as sensor networks grow, transitioning from flat-file CSV ingestion to time-series databases like InfluxDB or TimescaleDB is also necessary. These databases offer optimized support for temporal indexing, continuous queries, retention policies, and seamless scaling, which are vital for operational continuity and long-term storage of urban telemetry [148].

Another major opportunity lies in the integration of advanced predictive modeling. Once higher-frequency sensor data becomes available, deep learning techniques such as long short-term memory (LSTM) networks or transformer architectures could be deployed via scheduled Celery tasks or orchestrated using Kubernetes CronJobs. These models would enable short-term forecasting of pollutants, noise disturbances, and traffic anomalies, thus supporting proactive planning and crisis mitigation.

Furthermore, the backend infrastructure could be extended by incorporating modular RESTful APIs using frameworks such as FastAPI. These endpoints would facilitate public and third-party access, support OpenAPI documentation, and expose endpoints for real-time sensor event handling and predictive output delivery. This would establish a more interoperable platform for both public transparency and commercial applications.

User accessibility and inclusivity should also remain a design priority. Accessibility improvements could involve automated testing through tools like axe-core, addition of alternative text to graphical content, and compliance with Web Content Accessibility Guidelines (WCAG) 2.1. Features like keyboard navigation and screen reader compatibility will ensure that the system is usable by individuals with disabilities or varying levels of digital literacy [149].

Lastly, a roadmap for long-term sustainability includes the establishment of role-based access control, data transparency logs, and governance frameworks that enable stakeholder deliberation. Enabling collaborative dashboards where city officials, citizens, and researchers co-interpret data will democratize insights and promote participatory urban innovation.

These enhancements collectively aim to transform the current platform into a scalable, intelligent, and civic-focused infrastructure for the next generation of smart cities.

Chapter 6

Appendices

Hybrid Imputation with SARIMAX, k-NN and Random Forest

Listing 6.1: Python function for imputing missing time-series values using SARIMAX.

```
def arima_impute(series):
    s_obs = series.dropna()
    if len(s_obs) < 10:
        return series.interpolate(method="time", limit_direction="both"
            )
    try:
        model = SARIMAX(s_obs, order=(1,1,1),
                        enforce_stationarity=False,
                        enforce_invertibility=False)
        res = model.fit(disp=False)
        pred = res.get_prediction(start=series.index[0], end=series.
            index[-1])
        series_imputed = series.copy()
        series_imputed[series_imputed.isna()] = pred.predicted_mean[
            series_imputed.isna()]
        return series_imputed
    except:
        return series.interpolate(method="time", limit_direction="both"
            )
```

This Python function demonstrates a hybrid imputation method for handling missing values in time-series datasets using the SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors) model. The procedure begins by extracting the observed (non-missing) entries of the input series using `dropna()`. If fewer than 10 observations are available, it defaults to a fallback approach using time-based linear interpolation, as the SARIMAX model requires a minimum amount of historical data to be fitted effectively. If sufficient observations exist, the function initializes a SARIMAX model with an ARIMA(1,1,1) structure—meaning one autoregressive term, one differencing operation to ensure stationarity, and one moving average component. It sets `enforce_stationarity` and `enforce_invertibility` to `False` to allow greater model flexibility and avoid estimation errors for borderline cases. After fitting the model to the observed

data, it retrieves in-sample predictions spanning the full index range of the original time series. It then creates a copy of the input series and replaces all missing values with the predicted values computed by the imputation model. This replacement is limited only to missing positions to ensure the integrity of originally observed data. The entire modeling step is enclosed in a `try-except` block to handle potential failures gracefully. If any error occurs during model fitting or prediction (such as convergence issues or ill-conditioned inputs), the function falls back to a simpler time-based interpolation as a last resort. This dual-mode strategy ensures robust handling of edge cases while favoring statistical forecasting when feasible.

Listing 6.2: Blending SARIMAX and k-NN imputation with clipping to sensor bounds.

```
knn_imputer = KNNImputer(n_neighbors=5)
knn_arr = knn_imputer.fit_transform(df_all[measure_cols].values)

for col in measure_cols:
    mask = df_all[col].isna()
    df_all[col][mask] = (
        0.7 * arima_imputed[col][mask]
      + 0.3 * knn_arr[:, measure_cols.index(col)][mask]
    )
    df_all[col] = df_all[col].clip(*original_bounds[col])
```

This code snippet presents a hybrid blending strategy that combines statistical forecasting (SARIMAX) and distance-based machine learning (k-Nearest Neighbors, or k-NN) to impute missing values in multivariate sensor time-series data. The process begins by initializing a `KNNImputer` from `scikit-learn`, specifying `n_neighbors=5` to base imputation on the five most similar samples across all columns listed in `measure_cols`. The imputer is applied to the raw values of `df_all[measure_cols]`, generating a NumPy array `knn_arr` where missing values have been filled by averaging neighboring samples within the feature space. For each measurement column, a boolean `mask` is computed to isolate the positions where data was originally missing. At these positions, a weighted average is computed: 70% of the imputed value comes from the earlier SARIMAX forecast (stored in `arima_imputed[col]`), while 30% comes from the corresponding entry in `knn_arr`. This linear combination is designed to blend temporal forecasting with spatially aware statistical similarity, producing robust and contextually accurate estimates. Finally, the line `df_all[col] = df_all[col].clip(*original_bounds[col])` ensures that all imputed values remain within known valid sensor bounds (e.g., NO_2 concentrations or decibel levels), avoiding unrealistic or out-of-distribution values. This final step enforces physical plausibility and guards against extrapolation artifacts, particularly relevant when imputed data feeds into downstream visualization or decision-making systems.

Listing 6.3: Combining SARIMAX and Random Forest predictions using a weighted hybrid approach.

```
model = MultiOutputRegressor(RandomForestRegressor(n_estimators=100))
model.fit(X_train, y_train)

# Predict missing values
rf_pred = model.predict(X_missing)
arima_pred = sarimax_fit(...).get_prediction(...).predicted_mean

# Hybrid blending: 70% ARIMA, 30% Random Forest
combined = 0.7 * arima_pred + 0.3 * rf_pred
```

This code demonstrates a multi-model ensemble technique for imputing missing data points by combining predictions from a Random Forest regressor with those from a SARIMAX time-series model. The process begins with the initialization and training of a `MultiOutputRegressor` wrapper around `RandomForestRegressor` from `scikit-learn`, enabling the model to predict multiple target variables simultaneously (e.g., pollutant types across sensors). Here, the Random Forest ensemble uses 100 estimators (decision trees), which helps reduce variance and improve generalization. The trained model is applied to the missing data subset `X_missing`, yielding `rf_pred`, a prediction array that estimates missing values based on the structural correlations and feature distributions present in the training data `X_train`. Parallelly, SARIMAX is fitted to each univariate series (not shown in detail in this code but typically handled per variable), and `get_prediction()` extracts the corresponding forecasted values, stored as `arima_pred`. These represent the temporal expectations inferred from the time-series dynamics. The key step is the hybrid fusion, where a linear blend is applied: 70% of the final imputation comes from the SARIMAX forecast, while the remaining 30% is sourced from the Random Forest estimate. This weighted approach is particularly beneficial in scenarios where time-series trends are significant but may be complemented by cross-feature associations learned by the Random Forest. By fusing both model outputs, the strategy mitigates the limitations of any single method. Temporal consistency from SARIMAX is preserved, while Random Forest adds robustness against structural non-linearity or seasonal deviations. The final result `combined` represents a reliable imputation strategy for filling sensor data gaps in urban environments, balancing statistical rigor with machine learning generalization.

6.1 Flask API Endpoint

Listing 6.4: Flask route for computing and returning a correlation matrix from numeric sensor fields.

```
@app.route("/get_corr_matrix")
@cache.cached()
def get_corr_matrix():
    df = pd.read_csv(..., parse_dates=["Zeitstempel"])
    cols = [c for c in df.columns if c not in non_numeric]
    corr = df[cols].corr().fillna(0)
    return jsonify(corr.to_dict())
```

This Flask route defines an API endpoint, `/get_corr_matrix`, that dynamically computes a correlation matrix from numerical sensor inputs. It begins by reading a dataset using `pandas.read_csv()` with explicit parsing of the `Zeitstempel` (timestamp) column into datetime format. The code then

filters out non-numeric columns using a predefined list called `non_numeric`, isolating the numerical fields necessary for correlation analysis. Using `df[cols].corr()`, a Pearson correlation matrix is calculated between all pairs of numeric variables. This matrix quantifies the linear dependencies between variables, such as pollutant levels, noise measurements, or vehicle counts, helping users identify interrelationships within the urban sensing data. Missing values in the resulting matrix (due to insufficient overlapping data) are filled with zeros via `fillna(0)` to ensure complete JSON serializability. The result is converted into a dictionary and returned to the client through Flask's `jsonify()`, which formats the output in JSON structure suitable for frontend visualizations (e.g., correlation heatmaps). The route is decorated with `@cache.cached()`, a caching mechanism from Flask-Caching that ensures repeated calls do not redundantly recompute the matrix, thereby improving API responsiveness and performance, particularly when working with large datasets or multiple concurrent users.

6.2 Evaluation Algorithms

Listing 6.5: Computing the Mean Absolute Error (MAE) between predicted and true values.

```
import numpy as np

def mean_absolute_error(y_true, y_pred):
    """
    Compute MAE between true and predicted values.
    """
    y_true = np.array(y_true)
    y_pred = np.array(y_pred)
    return np.mean(np.abs(y_true - y_pred))

# Example usage:
# y_true = [3.0, 5.0, 2.5, 7.0]
# y_pred = [2.5, 5.5, 2.0, 8.0]
# mae = mean_absolute_error(y_true, y_pred)
# print(f"MAE = {mae:.4f}")
```

This Python function calculates the Mean Absolute Error (MAE), which is one of the most commonly used metrics for evaluating the accuracy of regression models. MAE measures the average magnitude of errors in a set of predictions, without considering their direction. It is calculated as the mean of the absolute differences between the true values y_i and the predicted values $\hat{y}_i$, formally expressed as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{6.1}$$

where n is the number of observations, y_i denotes the ground truth, and $\hat{y}_i$ represents the model's prediction.

The implementation uses NumPy arrays to ensure compatibility and performance when working with vectorized data structures. The function is robust and simple, offering interpretability due to its unit consistency with the data being predicted. Unlike squared-error metrics such as RMSE, MAE does

not penalize large errors more heavily than small ones, making it suitable when outliers are not disproportionately important.
This function is particularly useful in time-series forecasting, environmental data imputation, and sensor analytics, where it provides a reliable scalar measure of average predictive performance. The example usage illustrates how to apply it to a small dataset and output the result with four decimal places for readability.

Listing 6.6: Computing the Root Mean Squared Error (RMSE) between predicted and true values.

```
import numpy as np

def root_mean_squared_error(y_true, y_pred):
    """
    Compute RMSE between true and predicted values.
    """
    y_true = np.array(y_true)
    y_pred = np.array(y_pred)
    mse = np.mean((y_true - y_pred)**2)
    return np.sqrt(mse)

# Example usage:
# y_true = [3.0, 5.0, 2.5, 7.0]
# y_pred = [2.5, 5.5, 2.0, 8.0]
# rmse = root_mean_squared_error(y_true, y_pred)
# print(f"RMSE = {rmse:.4f}")
```

The Root Mean Squared Error (RMSE) is a standard way to quantify the differences between predicted values and observed values for a continuous variable. It is defined as the square root of the average of the squared differences between prediction and actual observation. This metric is especially sensitive to large errors due to the squaring step, making it more suitable for cases where large deviations should be penalized more strongly.
The formula for RMSE is given by:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{6.2}$$

where y_i is the true value, $\hat{y}_i$ is the predicted value, and n is the total number of observations.
In the Python function above, both input vectors `y_true` and `y_pred` are first cast into NumPy arrays. The Mean Squared Error (MSE) is computed using the squared differences, followed by taking the square root to yield the RMSE. This two-step process preserves the original units of the data, enhancing the interpretability of error magnitude.
RMSE is widely used in regression evaluation, environmental modeling, and forecasting contexts, particularly where a few large errors can disproportionately affect outcomes. The example usage demonstrates its application to a small dataset and prints the final error rounded to four decimal places for clarity.

6.3 JavaScript Snippets for Frontend Integration

Listing 6.7: Plotly.js configuration object for rendering a sensor correlation heatmap.

```
var traceHeatmap = {
  z: zData,
  x: sensorNames,
  y: sensorNames,
  type: 'heatmap',
  colorscale: 'Viridis',
  text: zData,
  texttemplate: '%{text:.2f}',
  textfont: { size: 10, color: 'white' }
};
```

This JavaScript code snippet defines a Plotly.js trace configuration for rendering a heatmap that visualizes pairwise correlations between different sensor readings. The variable `traceHeatmap` serves as the core configuration passed to Plotly's rendering engine. The `z` field contains the actual numerical correlation matrix `zData`, representing the strength of linear relationships between sensor variables. Both `x` and `y` axes are labeled using `sensorNames`, ensuring the heatmap presents a symmetric matrix with labeled rows and columns for interpretability. The `type` is set to `'heatmap'`, instructing Plotly to render a two-dimensional color-mapped matrix. The `colorscale` parameter specifies the color gradient used to represent different correlation strengths; `'Viridis'` is a perceptually uniform colormap suitable for scientific visualizations. The `text` field overlays raw correlation values directly onto the heatmap cells, while `texttemplate: '%text:.2f'` ensures these values are formatted to two decimal places. The `textfont` parameter customizes the overlay font size and color (white), ensuring readability against the colored background. Together, these options enhance the usability of the heatmap, enabling users to interpret both the magnitude and direction of relationships at a glance.

Listing 6.8: Function to dynamically add checkboxes for toggling Leaflet.js layer groups.

```
function addCheckbox(label, layerGroup) {
  let cb = document.createElement('input');
  cb.type = 'checkbox'; cb.checked = true;
  cb.onchange = () => {
    map[cb.checked ? 'addLayer' : 'removeLayer'](layerGroup);
  };
  legend.appendChild(cb);
  legend.appendChild(document.createTextNode(label));
}

addCheckbox('Air quality sensors', airLayer);
```

This JavaScript function `addCheckbox` enables dynamic addition of UI checkboxes to control the visibility of specific Leaflet.js layer groups on a web map. The function takes two arguments: a text label and a Leaflet layer group (e.g., for air quality sensors). It creates an HTML `input` element

of type checkbox using `document.createElement`, then sets it to be initially checked. The `onchange` handler is a conditional toggle that either adds or removes the associated layer group from the map using Leaflet's `map.addLayer()` or `map.removeLayer()` methods depending on the checkbox state. This provides seamless interactivity for users to control what layers they see. The checkbox and its text label are then appended to a predefined `legend` DOM element, which is typically styled to appear on the map interface. This functionality supports modular and scalable UI design, making it easier to integrate additional sensor layers (e.g., noise, traffic, parking) in future expansions.

Bibliography

[1] C. Harrison and I. Donnelly, "Foundations for smarter cities," *IBM Journal of Research and Development*, vol. 54, no. 4, pp. 1–16, 2010.

[2] A. Caragliu, C. Del Bo, and P. Nijkamp, "Smart cities in europe," *Journal of urban technology*, vol. 18, no. 2, pp. 65–82, 2011.

[3] A. Cocchia, "Smart and digital city: A systematic literature review," pp. 13–43, 2014.

[4] J. Lee, R. Phaal, and S. Lee, "Towards an effective framework for building smart cities: Lessons from seoul and san francisco," *Technological Forecasting and Social Change*, vol. 89, pp. 80–99, 2014.

[5] UN-Habitat, "World cities report 2016: Urbanization and development – emerging futures," 2016. [Online]. Available: `https://unhabitat.org/world-cities-report`.

[6] J. Wang, Z. Xiong, Y. Wang, D. Zhao, Y. Jiang, and S. Liu, "A literature survey on smart cities," *Science China Information Sciences*, vol. 58, no. 10, pp. 1–18, 2015.

[7] K. C. Seto *et al.*, "Human settlements, infrastructure and spatial planning," *Climate Change 2014: Mitigation of Climate Change*, 2014.

[8] U. N. D. of Economic and P. D. Social Affairs, "World urbanization prospects: The 2018 revision," 2018. [Online]. Available: `https://population.un.org/wup/`.

[9] UN-Habitat, *World cities report 2020: The value of sustainable urbanization*, `https://unhabitat.org/wcr/`, Accessed: 2025-05-19, 2020.

[10] M. R. Montgomery, "The urban transformation of the developing world," *Science*, vol. 319, no. 5864, pp. 761–764, 2008.

[11] M. Chen, W. Liu, and D. Lu, "Urbanization and sustainability: Comparative pathways in china and india," *Sustainability*, vol. 12, no. 5, p. 1885, 2020.

[12] S. Angel *et al.*, "Making room for a planet of cities," 2011.

[13] C. Moreno, Z. Allam, D. Chabaud, C. Gall, and F. Pratlong, "Urban sustainability and resilience: From theory to practice," *Sustainability*, vol. 12, no. 1, p. 20, 2020.

[14] M. Batty, K. W. Axhausen, F. Giannotti, *et al.*, "Smart cities of the future," *The European Physical Journal Special Topics*, vol. 214, pp. 481–518, 2012.

[15] V. Albino, U. Berardi, and R. M. Dangelico, "Smart cities: Definitions, dimensions, performance, and initiatives," *Journal of Urban Technology*, vol. 22, no. 1, pp. 3–21, 2015.

[16] Z. Huang, M.-P. Kwan, and et al., "Urban environments and covid-19 in three eastern states of the united states," *Environment and Planning B: Urban Analytics and City Science*, vol. 47, no. 4, pp. 491–510, 2020.

[17] D. Boyd and K. Crawford, "Critical questions for big data," *Information, Communication & Society*, vol. 15, no. 5, pp. 662–679, 2012.

[18] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of things for smart cities," *IEEE Internet of Things journal*, vol. 1, no. 1, pp. 22–32, 2014.

[19] I. A. T. Hashem *et al.*, "The rise of "big data" on cloud computing: Review and open research issues," *Information Systems*, vol. 47, pp. 98–115, 2015.

[20] Y. Xu, Z. Zhang, H. Liang, and A. Y. Zomaya, "An iot-oriented data storage framework with hierarchical reliability and cost efficiency," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1684–1692, 2018.

[21] S. Barns, "Smart cities and urban data platforms: Designing for a new civic infrastructure," *City, Culture and Society*, vol. 12, pp. 5–12, 2018.

[22] A. Meijer and M. P. R. Bolívar, "Co-creating smart cities: Governance, citizen engagement and digital technology," *Information Polity*, vol. 21, no. 1, pp. 1–14, 2016.

[23] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide*, vol. 1, pp. 10–555, 2017.

[24] R. Giffinger and et al., "Smart cities—ranking of european medium-sized cities," *Vienna University of Technology*, 2007.

[25] A. Greenfield, "Against the smart city," *Do projects*, 2013.

[26] J. Sadowski, "Too smart? city science and the rise of smart urbanism," *MIT Press*, 2020.

[27] R. Goodspeed, "Smart cities: Moving beyond urban cybernetics to tackle wicked problems," *Cambridge Journal of Regions, Economy and Society*, vol. 8, no. 1, pp. 79–92, 2015.

[28] U. N. D. of Economic and S. Affairs, "World urbanization prospects: The 2022 revision," *Population Division*, 2022.

[29] C. Vlachokostas *et al.*, "A digital twin paradigm for smart cities: The case of water management," *Sustainable Cities and Society*, vol. 64, p. 102 507, 2021.

[30] L. M. Aiello, R. Schifanella, D. Quercia, F. Aletta, G. Newman, and P. Atkinson, "Chatty maps: Constructing sound maps of urban areas from social media data," *Royal Society Open Science*, vol. 3, no. 3, p. 150 690, 2016.

[31] R. Gangwar and et al., "State-of-the-art sensor technologies in smart cities: A comprehensive review," *Journal of Urban Technology*, vol. 30, no. 2, pp. 1–21, 2023.

[32] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[33] P. Barnaghi, W. Wang, C. Henson, and K. Taylor, "Semantics for the internet of things: Early progress and back to the future," *International Journal on Semantic Web and Information Systems*, vol. 8, no. 1, pp. 1–21, 2012.

[34] N. U. Okafor and D. T. Delaney, "Missing data imputation on iot sensor networks: Implications for on-site sensor calibration," *IEEE Sensors journal*, vol. 21, no. 20, pp. 22 833–22 845, 2021.

[35] R. Toledo *et al.*, "Drift-aware modeling for environmental sensor calibration," *Environmental Modelling & Software*, vol. 160, p. 105 551, 2023.

[36] G. E. Batista and M. C. Monard, "A study of the k-nearest neighbour as an imputation method," *His*, vol. 87, no. 251-260, p. 48, 2002.

[37] J. M. Jerez, I. Molina, *et al.*, "Missing data imputation using statistical and machine learning methods in a real breast cancer problem," *Artificial intelligence in medicine*, vol. 50, no. 2, pp. 105–115, 2010.

[38] L. Tonsager and J. Ponder, "Privacy frameworks for smart cities," *JL & Mobility*, p. 1, 2022.

[39] C. Barnes, "Information privacy: Official secrecy and civil liberties in the uk," *Government and Opposition*, vol. 41, no. 3, pp. 431–456, 2006.

[40] V. Eubanks, "Automating inequality: How high-tech tools profile, police, and punish the poor," *St. Martin's Press*, 2018.

[41] P. Fricker, R. Krueger, and X. Torres, "Dashboard design for urban analytics: Lessons from real-time systems," *Urban Informatics Journal*, vol. 3, no. 2, pp. 150–167, 2019.

[42] Kharakhash, "Data visualization: Transforming complex data into actionable insights," *Automation of technological and business processes*, vol. 15, no. 2, pp. 4–12, 2023.

[43] M. Mahmud, M. Ali, *et al.*, "Air quality monitoring using iot and big data: A review," *MDPI Electronics*, vol. 9, no. 9, p. 1453, 2020.

[44] Y. Zheng, S. Consolvo, T. Choudhury, B. Harrison, and B. Lambert, "Ubifit garden: A ubiquitous and persuasive technology to promote physical activity," *ACM Transactions on Computer-Human Interaction (TOCHI)*, vol. 19, no. 1, 2013. DOI: 10.1145/2147783.2147787.

[45] H. Shahraiyni and S. Sodoudi, "A review on statistical modeling and machine learning approaches for short-term air quality forecasting," *Atmospheric Pollution Research*, vol. 7, no. 5, pp. 768–777, 2016. DOI: 10.1016/j.apr.2016.03.006.

[46] Y. Zheng and et al., "Urban computing: Concepts, methodologies, and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 5, no. 3, p. 38, 2014.

[47] E. Dmitrieva, A. Pathani, G. Pushkarna, P. Acharya, M. Rana, and P Surekha, "Real-time traffic management in smart cities: Insights from the traffic management simulation and impact analysis," in *BIO Web of Conferences*, EDP Sciences, vol. 86, 2024, p. 01 098.

[48] M. Basner, W. Babisch, A. Davis, *et al.*, "Auditory and non-auditory effects of noise on health," *The Lancet*, vol. 383, no. 9925, pp. 1325–1332, 2014.

[49] M. Li, W. Liu, *et al.*, "A smart parking system based on iot and cloud computing," *Sensors*, vol. 19, no. 5, p. 1023, 2019.

[50] J. Gante, G. Falcao, and R. Dinis, "Urban mobility analytics using crowdsourced data: A survey," *IEEE Access*, vol. 9, pp. 143 337–143 359, 2021. DOI: 10.1109/ACCESS.2021.3119745.

[51] M. Masetti *et al.*, "Charging infrastructure optimization for electric vehicles using predictive analytics," *IEEE Transactions on Smart Grid*, vol. 13, no. 1, pp. 618–630, 2022.

[52] F. Lopes *et al.*, "Load forecasting for electric vehicle charging stations in smart cities," *Renewable and Sustainable Energy Reviews*, vol. 158, p. 112 117, 2022.

[53] B. Li, M. C. Kisacikoglu, C. Liu, N. Singh, and M. Erol-Kantarci, "Big data analytics for electric vehicle integration in green smart cities," *IEEE Communications Magazine*, vol. 55, no. 11, pp. 19–25, 2017.

[54] V. Papastefanopoulos, P. Linardatos, T. Panagiotakopoulos, and S. Kotsiantis, "Multivariate time-series forecasting: A review of deep learning methods in internet of things applications to smart cities," *Smart Cities*, vol. 6, no. 5, pp. 2519–2552, 2023.

[55] A. Yassine, S. Singh, and E. Aggoune, "Edge computing for smart cities: A survey," *IEEE Access*, vol. 7, pp. 152 168–152 192, 2019.

[56] D. Carneiro, A. Amaral, M. Carvalho, and L. Barreto, "An anthropocentric and enhanced predictive approach to smart city management," *Smart Cities*, vol. 4, no. 4, pp. 1366–1390, 2021.

[57] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: The rising stars in the iot and smart city scenarios," *IEEE Wireless Communications*, vol. 23, no. 5, pp. 60–67, 2016. DOI: 10.1109/MWC.2016.7721743.

[58] A. Kotagi and P. Hiremath, "Low power wide area network technologies for smart cities: A survey," *Internet of Things*, vol. 18, p. 100 510, 2022. DOI: 10.1016/j.iot.2022.100510.

[59] V. Pereira, H. Ferreira, N. Silva, and R. Pinto, "Privacy-preserving smart cities: A review of enabling technologies and policies," *Sensors*, vol. 21, no. 7, p. 2308, 2021.

[60] E. Ismagilova, L. Hughes, N. P. Rana, and Y. K. Dwivedi, "Security, privacy and risks within smart cities: Literature review and development of a smart city interaction framework," *Information Systems Frontiers*, pp. 1–22, 2022.

[61] D. Washburn, U. Sindhu, S. Balaouras, R. A. Dines, N. Hayes, and L. E. Nelson, "Helping cios understand "smart city" initiatives," *Growth*, vol. 17, no. 2, pp. 1–17, 2009.

[62] T. Singh, A. Solanki, S. K. Sharma, A. Nayyar, and A. Paul, "A decade review on smart cities: Paradigms, challenges and opportunities," *IEEE Access*, vol. 10, pp. 68 319–68 364, 2022.

[63] T.-h. Kim, C. Ramos, and S. Mohammed, *Smart city and iot*, 2017.

[64] S. Aslam and H. S. Ullah, "A comprehensive review of smart cities components, applications, and technologies based on internet of things," *arXiv preprint arXiv:2002.01716*, 2020.

[65] D. Hasenfratz *et al.*, "Calibration and quality control of sensor data for urban air quality monitoring," *IEEE Sensors Journal*, vol. 15, no. 1, pp. 224–234, 2015.

[66] L. Spinelle and et al., "Review of calibration strategies for low-cost sensors for air pollution," *Sensors*, vol. 17, no. 3, p. 561, 2017.

[67] N. Castell and et al., "Can commercial low-cost sensor platforms contribute to air quality monitoring and exposure estimates?" *Environment International*, vol. 99, pp. 293–302, 2017.

[68] P. Yu and et al., "Monitoring air pollution using hybrid satellite-ground data fusion," *Remote Sensing of Environment*, vol. 259, p. 112 418, 2021.

[69] T. Abbas *et al.*, "Review of vehicle-to-infrastructure (v2i) communication in connected vehicle systems," *IEEE Access*, vol. 9, pp. 103 547–103 567, 2021.

[70] F. Aletta, J. Kang, and . Axelsson, "Soundscape descriptors and a conceptual framework for developing predictive soundscape models," *Landscape and Urban Planning*, vol. 149, pp. 65–74, 2016.

[71] E. Miranda *et al.*, "Urban sound classification using edge-ai and deep learning in smart cities," *Sensors*, vol. 22, no. 4, p. 1538, 2022.

[72] Y. Yang, M. Zhou, and C. Li, "Ai-driven urban sound event detection for smart city noise monitoring," *Applied Acoustics*, vol. 202, p. 109 122, 2023.

[73] N. Maisonneuve, M. Stevens, M. Niessen, and L. Steels, "Participatory noise mapping works! an evaluation of participatory sensing as an alternative to standard techniques for environmental monitoring," *Ecological Informatics*, vol. 6, no. 1, pp. 27–38, 2011.

[74] S. Amar *et al.*, "Smart parking solutions using computer vision and iot: A review," *IEEE Access*, vol. 10, pp. 45 678–45 695, 2022.

[75] D. Johnson and P. Sharma, "Iot-based smart parking management system," *International Journal of Engineering Research and Technology*, vol. 13, no. 6, pp. 1432–1440, 2020.

[76] D. Shoup, *The High Cost of Free Parking*. Planners Press, American Planning Association, 2005.

[77] M. S. Islam *et al.*, "An iot-based weather monitoring system using low-cost sensors and edge analytics," *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, pp. 4053–4067, 2020.

[78] Q. Du, M. Wang, and T. Liu, "Urban microclimate modeling using edge-enabled sensor fusion," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3224–3235, 2021.

[79] L. Chang, J. Zhang, and Y. Wang, "Iot-based real-time urban weather data collection and forecasting with edge computing," *Sensors*, vol. 23, no. 1, p. 199, 2023.

[80] S. De and B Bhattacharya, "Structural health monitoring using wireless sensor networks: A review," *Shock and Vibration*, vol. 10, no. 3, pp. 197–216, 2003.

[81] R. Morales *et al.*, "Structural monitoring using fiber optic sensors: A review," *Measurement*, vol. 90, pp. 117–134, 2016.

[82] D. D. Mascarenas, J. P. Lynch, C. R. Farrar, and S. W. Doebling, "Development of structural health monitoring system using wireless smart sensors," *IEEE Sensors Journal*, vol. 7, no. 5, pp. 743–751, 2007.

[83] K. Janowicz, A. Haller, S. J. Cox, D. Le Phuoc, and K. Taylor, "Sosa: A lightweight ontology for sensors, observations, samples, and actuators," *Journal of Web Semantics*, vol. 56, pp. 1–10, 2019.

[84] T. H. Kolbe, "Representing and exchanging 3d city models with citygml," in *3D Geo-Information Sciences*, Springer, 2009, pp. 15–31.

[85] N. S. Pargoo, M. Ghasemi, S. Xia, *et al.*, "The streetscape application services stack (sass): Towards a distributed sensing architecture for urban applications," *arXiv preprint arXiv:2411.19714*, 2024. DOI: 10.48550/arXiv.2411.19714. [Online]. Available: https://arxiv.org/abs/2411.19714.

[86] L. Sánchez, L. Muñoz, J. A. Galache, *et al.*, "Iot-based scalable data collection system for smart cities," *Future Generation Computer Systems*, vol. 76, pp. 221–232, 2017.

[87] V. N. Gudivada, D. Rao, and V. V. Raghavan, "Explainable ai for the semantic web: A survey," *arXiv preprint arXiv:1802.01798*, 2018.

[88] M. Ali, F. Shah, N. Ahmad, and H. u. Rahman, "Urbanviz: Interactive urban data visualization platform for policy support," *Cities*, vol. 117, p. 103 296, 2021.

[89] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[90] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the internet of things: A survey," *Computer Communications*, vol. 83, pp. 1–22, 2016.

[91] M. Bostock, V. Ogievetsky, and J. Heer, "D3: Data-driven documents," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2301–2309, 2011.

[92] V. Agafonkin and contributors, *Leaflet: An open-source javascript library for interactive maps*, https://leafletjs.com, Accessed: 2024-04-10.

[93] W. McKinney, "Data structures for statistical computing in python," *Proceedings of the 9th Python in Science Conference*, vol. 445, pp. 51–56, 2010.

[94] J. Muckell, A. Gandhe, J.-S. Hwang, and C. T. Lawson, "A critical analysis of the use of timestamps in gps data," in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, 2011, pp. 430–433.

[95] L. Morawska *et al.*, "Applications of low-cost sensing technologies for air quality monitoring and exposure assessment," *Environment International*, vol. 116, pp. 286–299, 2018.

[96] A. Alghamdi *et al.*, "Urban sensor networks: A systematic review of challenges and solutions," *IEEE Access*, vol. 9, pp. 90 642–90 660, 2021.

[97] S. Adhikari and Q. Ye, "A comprehensive survey of missing data handling techniques in the internet of things," *Journal of Big Data*, vol. 9, no. 1, pp. 1–36, 2022.

[98] F. R. Alharbi and D. Csala, "A seasonal autoregressive integrated moving average with exogenous factors (sarimax) forecasting model-based time series approach," *Inventions*, vol. 7, no. 4, p. 94, 2022.

[99] D. J. Stekhoven and P. Bühlmann, "Missforest—non-parametric missing value imputation for mixed-type data," *Bioinformatics*, vol. 28, no. 1, pp. 112–118, Oct. 2011, ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btr597. eprint: https://academic.oup.com/bioinformatics/article-pdf/28/1/112/50568519/bioinformatics_28_1_112.pdf. [Online]. Available: https://doi.org/10.1093/bioinformatics/btr597.

[100] A. Aleryani, A. Rehman, and A. Ali, "Multiple imputation of multivariate time series data: A novel approach based on ensemble learning," *Applied Soft Computing*, vol. 91, p. 106 222, 2020.

[101] G. Bontempi, S. B. Taieb, and Y.-A. Le Borgne, "Machine learning strategies for time series forecasting," *European Business Intelligence Summer School*, vol. 3, pp. 62–77, 2012.

[102] M. M. Rahman *et al.*, "A survey on serverless computing and function-as-a-service (faas): Foundations and future directions," *Journal of Systems and Software*, vol. 170, p. 110 708, 2020.

[103] J. Zhang, X. Lin, and D. Zeng, "Data-driven intelligent transportation systems: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1624–1639, 2011.

[104] P. Crickard III, *Leaflet. js essentials*. Packt Publishing Ltd, 2014.

[105] Plotly Technologies Inc., *Plotly.js - a javascript graphing library*, `https://plotly.com/javascript/`, Accessed April 2025, 2025.

[106] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[107] O. e. a. Troyanskaya, "Missing value estimation methods for dna microarrays," *Bioinformatics*, vol. 17, no. 6, pp. 520–525, 2001.

[108] R. J. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed. OTexts, 2021. [Online]. Available: `https://otexts.com/fpp3/`.

[109] L. Atzori, A. Iera, and G. Morabito, "The evolution of the internet of things towards the future internet of things," *MDPI Future Internet*, vol. 9, no. 5, p. 41, 2017.

[110] A. Beaulieu, N. Schuurman, and B. Williams-Jones, "Smart cities and the governance of urban data: Policy, practice and ethics," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 379, no. 2199, p. 20 200 462, 2021.

[111] Smart Strasse Basel Project Team, *Smartestrasse smartsuisse 2023 presentation*, Accessed March 2025, 2023.

[112] R. Maia *et al.*, "Low-cost sensors for urban air quality: A review of technologies and performance," *Environmental Monitoring and Assessment*, vol. 192, no. 7, pp. 1–22, 2020.

[113] T. Mesaros, T. Heittola, and T. Virtanen, "Tut database for acoustic scene classification and sound event detection," in *24th European Signal Processing Conference (EUSIPCO)*, IEEE, 2016, pp. 1128–1132.

[114] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," *Proceedings of the 22nd ACM International Conference on Multimedia*, pp. 1041–1044, 2014.

[115] H. Bilen, T. Freeman, J. Mayer, *et al.*, "Weakly supervised event detection in audio using convolutional neural networks," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2016, pp. 295–299.

[116] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[117] P. Monserrate, L. Remaggi, S. Ceriani, J. Lopes, F. Ribeiro, and J. Davis, "Urban sound monitoring using low-cost sensors: Challenges and limitations," *Sustainable Cities and Society*, vol. 66, p. 102 656, 2021.

[118] T. Gloaguen, G. Lafay, and S. Rossignol, "Environmental sound classification for smart cities: A review," *Applied Sciences*, vol. 12, no. 1, p. 354, 2022.

[119] Stadt Basel, *Parkieren in basel-stadt: Informationen zur parkraumbewirtschaftung*, `https://www.parking.bs.ch/parkregeln/`, Accessed: 2025-05-19, 2021.

[120] A. Ronacher, *Flask (a python microframework)*, `https://flask.palletsprojects.com/`, Accessed: 2024-04-12, 2010.

[121] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*. O'Reilly Media, Inc., 2018.

[122] M. Fowler, *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2002.

[123] C. R. Harris, K. J. Millman, S. J. van der Walt, and et al., "Array programming with numpy," *Nature*, vol. 585, pp. 357–362, 2020.

[124] J. J. Garrett, *The Elements of User Experience: User-Centered Design for the Web and Beyond*. New Riders, 2010.

[125] J. Nielsen, *Usability Engineering*. Morgan Kaufmann, 1994.

[126] S. Krug, *Don't Make Me Think, Revisited: A Common Sense Approach to Web Usability*. New Riders, 2014.

[127] B. Shneiderman, C. Plaisant, and M. Cohen, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, 2010.

[128] A. Dix, J. Finlay, G. Abowd, and R. Beale, *Human-Computer Interaction*. Pearson Education, 2004.

[129] D. Benyon, *Designing Interactive Systems: A Comprehensive Guide to HCI, UX and Interaction Design*. Pearson Education, 2014.

[130] B. F. Reed, *Responsive Web Design with HTML5 and CSS3*. Packt Publishing Ltd, 2016.

[131] E. Marcotte, *Responsive Web Design*. A Book Apart, 2010.

[132] L. Wroblewski, *Mobile First*. A Book Apart, 2011.

[133] J. H. Jo, B. Jo, J. H. Kim, and I. Choi, "Implementation of iot-based air quality monitoring system for investigating particulate matter (pm10) in subway tunnels," *International journal of environmental research and public health*, vol. 17, no. 15, p. 5429, 2020.

[134] E. R. Tufte, *The Visual Display of Quantitative Information*. Graphics Press, 2001.

[135] C. G. Healey and et al., "Visualizing vector fields using line integral convolution and dye advection," *IEEE Transactions on Visualization and Computer Graphics*, vol. 2, no. 2, pp. 131–149, 1996.

[136] W. W. W. Consortium, *Web content accessibility guidelines (wcag) 2.1*, `https://www.w3.org/TR/WCAG21/`, Accessed: 2024-05-18, 2018.

[137] D. Flanagan, *JavaScript: The Definitive Guide*. O'Reilly Media, 2006.

[138] *Using xmlhttprequest*, `https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest`, Accessed: 2024-05-18.

[139] D. A. Keim, "Information visualization and visual data mining," in *Proceedings IEEE Symposium on Information Visualization*, IEEE, 2002, pp. 1–6.

[140] J. Lazar, J. H. Feng, and H. Hochheiser, *Research Methods in Human-Computer Interaction*. Morgan Kaufmann, 2015.

[141] C. Stephanidis, *Universal Access in Human-Computer Interaction. Users Diversity*. Springer, 2011.

[142] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005.

[143] T. L. Chai and R. R. Draxler, "Root mean square error (rmse) or mean absolute error (mae)?–arguments against avoiding rmse in the literature," *Geoscientific Model Development*, vol. 7, no. 3, pp. 1247–1250, 2014.

[144] E. Gholipour, A. Abedini, and Z. Ghaemi, "Machine learning-based air quality prediction models: A systematic review," *Environmental Monitoring and Assessment*, vol. 193, no. 10, pp. 1–24, 2021.

[145] H. Zhang, Y. Zhu, Y. Zhang, D. Wang, J. Wu, and X. Xie, "A comprehensive review of missing data handling in time series," *Neurocomputing*, vol. 491, pp. 244–257, 2022.

[146] F. Tang, H. Ishwaran, and V. R. Rao, "A comprehensive survey on missing data in machine learning," *Journal of Big Data*, vol. 7, no. 1, pp. 1–37, 2020.

[147] C. M. Bishop, *Pattern recognition and machine learning*. New York: Springer, 2006.

[148] R. e. a. Taft, "Timescaledb: Sql made scalable for time-series data," in *Proceedings of the International Conference on Management of Data (SIGMOD)*, 2018, pp. 1041–1053.

[149] Y. Wang and T. Nguyen, "Accessibility for all: Design guidelines for inclusive web dashboards," *ACM Transactions on Accessible Computing*, vol. 14, no. 3, pp. 1–28, 2021.