Open University of Catalonia (UOC) Master's Degree in Computer Science

MASTER'S THESIS

Area: Artificial Intelligence

Optimizing Bicing: Data Analytics for Operational Efficiency in Barcelona's Bike Sharing System

Development of a Web Application for the Analysis and Optimization of the Bicing Service

---

Author: Andrés Santos Sanz

Tutor: Ismael Benito Altamirano

Professor: Joan Núñez do Rio

Barcelona, January 23, 2025

Credits/Copyright

FINAL PROJECT RECORD

Title of the project:	Optimizing Bicing: Data Analytics for Operational Efficiency in Barcelona's Bike Sharing System
Author's name:	Andrés Santos Sanz
Collaborating teacher's name:	Ismael Benito Altamirano
PRA's name:	Joan Núñez do Rio
Delivery date (mm/yyyy):	01/2025
Degree or program:	Máster Universitario de Ingeniería Informática
Final Project area:	Artificial Intelligence
Language of the project:	English
Keywords	bike sharing, data analytics, web-based application

Dedication/Quote

I dedicate this work to my parents, without whom I would never have reached this point. To my parents-in-law, Pilar and Ángel, for helping me in every way they could during these years. And especially to Lucía, my life partner, for supporting me in all my endeavors and for continuing to do so.

Acknowledgements

I would like to thank my tutor, Ismael Benito Altamirano, for his support and guidance throughout this project. His help has been key to completing my work successfully.

I am also very grateful to the Universitat Oberta de Catalunya (UOC) and all the teachers and professionals who have shared their knowledge with me. Their support has helped me grow and learn during this program.

Abstract

Bicing, the bike sharing system in Barcelona, plays a crucial role in the promotion of sustainable urban mobility by significantly reducing carbon emissions and alleviating traffic congestion. As an integral component of the city's transport infrastructure, bicycling provides an accessible eco-friendly alternative to carbon-motorized transport, encouraging healthier and more sustainable lifestyles. However, despite these benefits, the system faces notable operational challenges, particularly regarding efficient management of resources and maintenance reliability. Current strategies have room for improvement, especially in terms of optimizing the availability and the condition of bicycles.

This thesis addresses these challenges by proposing a set of strategies to improve operational efficiency and user satisfaction. These include optimizing bicycle redistribution using advanced data analytics to predict demand patterns, enhancing predictive maintenance algorithms to reduce bicycle downtime. To implement these improvements, a web-based application has been developed. This application is designed to facilitate real-time monitoring and decision-making support. By streamlining operations, increasing bicycle availability, and delivering a more reliable and responsive service, this system ultimately aims to enhance the overall user experience and contribute further to the sustainability of urban mobility.

Keywords: bike sharing, sustainable urban mobility, Bicing Barcelona, predictive maintenance, data analytics, resource optimization, web-based application, real-time monitoring.

Resumen

Bicing, el sistema de bicicletas compartidas en Barcelona, desempeña un papel crucial en la promoción de la movilidad urbana sostenible al reducir significativamente las emisiones de carbono y aliviar la congestión del tráfico. Como un componente integral de la infraestructura de transporte de la ciudad, Bicing ofrece una alternativa accesible y ecológica al transporte motorizado, fomentando estilos de vida más saludables y sostenibles. Sin embargo, a pesar de estos beneficios, el sistema enfrenta importantes desafíos operacionales, particularmente en lo que respecta a la gestión eficiente de los recursos y la fiabilidad del mantenimiento. Las estrategias actuales presentan margen de mejora, especialmente en términos de optimizar la disponibilidad y el estado de las bicicletas.

Esta tesis aborda estos desafíos proponiendo un conjunto de estrategias orientadas a mejorar la eficiencia operativa y la satisfacción del usuario. Estas incluyen la optimización de la redistribución de bicicletas utilizando análisis avanzados de datos para predecir patrones de demanda, así como la mejora de los algoritmos de mantenimiento predictivo para reducir el tiempo de inactividad de las bicicletas. Para implementar estas mejoras, se ha desarrollado una aplicación web. Esta aplicación está diseñada para facilitar la monitorización en tiempo real y el soporte para la toma de decisiones. Al agilizar las operaciones, aumentar la disponibilidad de bicicletas y ofrecer un servicio más confiable y receptivo, este sistema tiene como objetivo mejorar la experiencia del usuario y contribuir a la sostenibilidad de la movilidad urbana.

Contents

1 Introduction

1.1 Context and motivation

Bike sharing has established itself as a sustainable mobility alternative in numerous cities around the world, contributing to the reduction of traffic congestion and carbon emissions, as well as to the promotion of a healthy lifestyle. Globally, the integration of these systems into the urban transport network has enabled citizens to choose an accessible, flexible and *eco-friendly* mode of transportation, complementing other means such as public transport and walking. This success has led to the existence of more than **2,000 bike sharing systems** operating in cities worldwide, with millions of daily users contributing to the **sustainability and efficiency of urban mobility** [1].

In the case of Barcelona, the shared bicycle system **Bicing** has experienced significant growth in recent years, as reflected in the constant increase in subscribers and the accumulated annual uses. According to **Bicing** statistics in 2024, the number of subscribers has reached **159,851 users**, with **1.7 million monthly uses** in September 2024 alone [2]. Despite these positive results, the management and maintenance of the system presents operational challenges that affect both the user experience and service efficiency.

The main issue revolves around optimizing the use and distribution of bicycles, as well as improving repair times and availability at stations. Currently, the system operates with **7,600 bicycles and 526 stations** (see Figure 1), with an average use of **7.73 rides per bicycle per day** and an average of **294 bicycles repaired daily** [2]. These figures highlight the need to implement improvements in system management to ensure availability and reduce bicycle downtime. Moreover, user perception of **Bicing**'s service quality still indicates certain areas for improvement. According to the latest evaluation conducted on *March 13, 2024*, system management received an average score of **6.35 out of 10** [4], indicating that there is room to optimize the user experience and the operational efficiency of the service.

Since 2007, the management and maintenance of the **Bicing** system, along with other shared mobility services in the Barcelona area such as **AMBici** and **Bicibox**, have been provided by the company *Movement* (*Movement Barcelona 2005 S.L*) [5]. This company is responsible for the maintenance of out-of-service bicycles, the repositioning of the bicycles, a task necessary due to the elevation differences present in the city of Barcelona, and the maintenance of the stations. According to their website, maintenance is performed in a *predictive* manner; however, areas for improvement have been identified.

To address these challenges, several measures could be implemented to increase **system efficiency** and **user satisfaction**. One proposed improvement is optimizing the redistribution of bicycles using **data analytics** to understand *demand patterns*, *peak usage times*, and *real-time*

Figure 1: Bike-sharing station of the Bicing system in Barcelona. The bicycles are lined up next to a control post, ready to be used by riders [3]

.

data, ensuring that bicycles are available where and when they are needed most. Furthermore, enhancing **maintenance frequency and effectiveness** through advanced *predictive algorithms* could help reduce bicycle downtime and improve overall reliability of service. **User feedback integration** is another key opportunity; Including feedback more dynamically into the operational model could help to resolve recurring issues faster and improve overall quality of service. **Expanding the fleet** and increasing the number of charging stations for **electric bicycles** could also alleviate current capacity limitations, accommodating the increasing demand. Another opportunity is to use data to identify stations with high demand relative to their size, allowing the proposal of new stations to better meet user needs.

1.2 Goals

1.2.1 General Objective

Develop a web-based system to help optimize the management of a bike sharing system in the city of Barcelona (**Bicing**).

1.2.2 Specific Objectives

- **Analyze station usage patterns** to identify areas requiring maintenance, ensuring greater availability of bikes, and improving overall service.

- Develop an intuitive web application that serves as a **front-end** to facilitate user interaction with the system in an efficient and user-friendly manner.

- Implement a **backend** API to manage data and provide key functionalities such as request processing and data analysis.

- Use notebooks to experiment with different models and analysis techniques, facilitating the integration of the results into the final system.

- **Optimize service efficiency** through continuous improvement of the analytical layer and the integration of predictive models for better demand management.

1.3 Sustainability, diversity, and ethical/social challenges

1.3.1 Sustainability

The shared bicycle monitoring and management system contributes to achieving **SDG 11 - Sustainable Cities and Communities** [6], by promoting the use of non-polluting transportation and reducing the dependence on private cars in the city of Barcelona. By encouraging bicycle use instead of motor vehicles, it supports the transition to a cleaner and more sustainable mobility model, helping to decrease **CO2 emissions** and thus contributing to **SDG 13 - Climate Action**. In addition, the project could improve the efficiency of bicycle distribution, minimize resource use and reduce the number of trips required for manual repositioning, contributing to **SDG 12 - Responsible Consumption and Production**.

1.3.2 Ethical behavior and social responsibility

The project also addresses ethical and social responsibility issues by facilitating equitable access to the transportation system, promoting shared mobility, and improving the availability of services in all areas of the city. This translates into a positive impact on **SDG 8 - Decent Work and Economic Growth**, by allowing more people, regardless of their location or resources, to benefit from an accessible and affordable transportation system.

1.3.3 Diversity, gender, and human rights

The project does not collect specific demographic data from users, making it difficult to assess its impact in terms of gender, diversity, or human rights. However, universal accessibility to the service is promoted, aligning with **SDG 10 - Reduced Inequalities**.

1.4 Approach and methodology

In this project, different strategies were evaluated to address the management of the bike-sharing system in Barcelona (**Bicing**), considering the most suitable technologies for task automation and planning.

- **Development of a new product from scratch**: Create a completely new system that includes the generation of optimized routes, the user interface, and the task assignment module. Although it would provide full control over all functionalities, this option would require extensive development and the creation of customized solutions for complex optimization problems, significantly increasing the time and resources needed.
- **Use of existing products**: Integrate pre-existing route management and planning tools, such as commercial software or optimization platforms. Although this strategy allows for faster implementation, it does not offer the flexibility needed to adapt to the specific characteristics of Barcelona's bike-sharing system and would limit the ability to customize decision-making logic for system management.
- **Hybrid strategy**: This option consists of developing a system that combines a specialized analysis module with existing tools, such as a third-party API, to optimize certain functionalities.

1.4.1 Chosen Strategy

The selected strategy is the **hybrid strategy**. This combination leverages the strengths of each technology, minimizing development from scratch and focusing on the customization of critical modules for the project.

Reasons for choosing this strategy:

- **Customization and flexibility**: The integration of third-party APIs allows for specific needs to be addressed, such as operator assistance and dynamic route planning.
- **Operational efficiency**: The hybrid strategy optimizes resource allocation, reducing response times and improving service quality based on actual traffic conditions and demand.

- **Scalability**: By combining specifically developed components with existing tools, the system is easier to scale and adapt to future increases in demand or changes in the operating environment.

- **Reduced costs and development time**: By integrating existing components and focusing on the development of specific modules, implementation times and associated costs are reduced.

- **Improved user experience**: The intuitive user interface and dynamic planning capability provide an enhanced user experience, both for operators and for the system's end-users.

1.5 Schedule

In this section, we provide an overview of the resources required to successfully carry out the project, alongside a comprehensive breakdown of the tasks to be completed.

The necessary resources include human and technical components. Human resources involve the author of the thesis and the thesis supervisor, each playing a critical role in guiding and executing different phases of the project. Technical resources include the following:

- **SaaS (Software as a Service)**: Various cloud-based tools were used, such as Overleaf [7] for the creation of the thesis and Google Sheets [8] for scheduling.

- **Programming Languages**: Python [9] was used for data analysis and back-end development, while Typescript [10] was used for front-end development.

- **Frameworks**: The project used FastAPI [11] for back-end development and Next.js [12] for front-end development.

- **PaaS (Platform as a Service)**: Supabase [13] was used for authentication and Vercel [14] was used for the deployment of the front-end.

- **Version Control System**: GitHub [15] was used for version control and collaboration.

- **IDE (Integrated Development Environment)**: Cursor [16] was used as the primary IDE for writing and debugging code.

- A dedicated server from the AIWell [17] research group was used to deploy the back-end API.

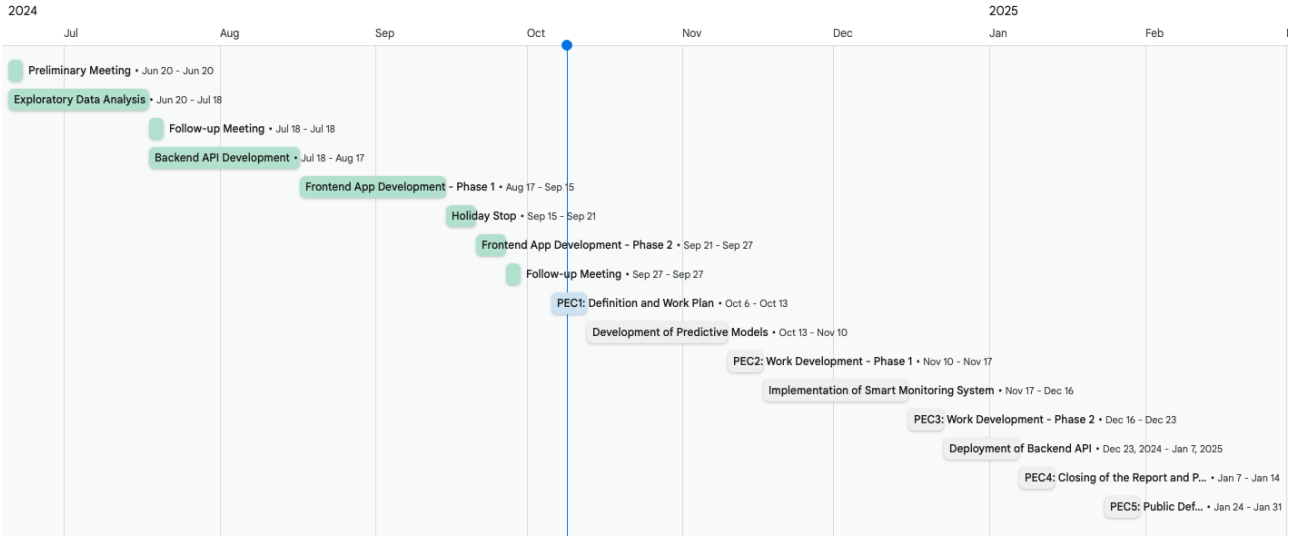

Figure 2: Project Gantt Diagram

The following schedule provides an overview of the project timeline, delineating key tasks and milestones critical to the successful completion of the project. It includes the various phases of development, implementation, and testing, highlighting dependencies and ensuring a structured progression from the initiation to the final defense. The **Gantt Diagram** (see Figure 2) visually represents the time frame for each task, offering a clear and concise perspective on the duration and overlap of the activities involved.

1.6 Summary of the outputs of the project

The main outputs of the project include a **user-friendly web application** that acts as the **frontend**, enabling efficient user interactions and making the overall system accessible and intuitive for users. This frontend ensures that both operators and end-users can navigate the system seamlessly, enhancing the overall **user experience**. Additionally, a **backend API** was developed to provide an **analytics layer,** manage data, and perform predictive analysis. This API is crucial for managing the flow of information between the front-end and the data storage systems, ensuring that the data is always up-to-date and actionable. **Predictive analysis** enables proactive maintenance and optimal resource allocation, significantly boosting **system reliability**. Notebooks were used extensively throughout the development phase to test and refine various models and analysis techniques. These notebooks allowed for iterative experimentation, providing a flexible environment to evaluate different approaches, document findings, and integrate successful models directly into the main system.

Table 1: Project Timeline

Task ID	Task Name	Start Date	End Date	Duration (days)	Status	Dependencies
1	Preliminary Meeting	20 Jun 2024	20 Jun 2024	1	Completed	None
2	Exploratory Data Analysis	20 Jun 2024	18 Jul 2024	29	Completed	1
3	Follow-up Meeting	18 Jul 2024	18 Jul 2024	1	Completed	2
4	Backend API Development	18 Jul 2024	17 Aug 2024	31	Completed	3
5	Frontend App Development - Phase 1	17 Aug 2024	15 Sep 2024	30	Completed	4
6	Holiday Stop	15 Sep 2024	21 Sep 2024	7	Completed	None
7	Frontend App Development - Phase 2	21 Sep 2024	27 Sep 2024	7	Completed	5
8	Follow-up Meeting	27 Sep 2024	27 Sep 2024	1	Completed	7
9	PEC1: Definition and Work Plan	6 Oct 2024	13 Oct 2024	8	Completed	8
10	Frontend App Development - Phase 3	13 Oct 2024	10 Nov 2024	29	Completed	9
11	PEC2: Work Development - Phase 1	10 Nov 2024	17 Nov 2024	8	Completed	10
12	Backend API Performance Improve	17 Nov 2024	16 Dec 2024	30	Completed	11
13	PEC3: Work Development - Phase 2	16 Dec 2024	23 Dec 2024	8	Completed	12
14	Deployment of Backend API	23 Dec 2024	7 Jan 2025	16	Completed	13
15	PEC4: Closing of the Report and Presentation	7 Jan 2025	14 Jan 2025	8	Completed	14
16	PEC5: Public Defense	24 Jan 2025	31 Jan 2025	8	In Progress	15

1.7 Brief description of the remaining chapters of the report

The second chapter reviews the **state-of-the-art**, summarizing the most relevant research in software for optimizing urban mobility services. This includes a detailed exploration of existing **bike-sharing systems**, discussing their features, limitations, and how recent technological advancements are driving their evolution. The chapter also provides an overview of technologies commonly used in these systems, including **IoT solutions**, **machine learning algorithms** for predictive maintenance, and integrated **data platforms**. Furthermore, it introduces the **General Bikeshare Feed Specification (GBFS)**, highlighting its importance in standardizing data exchange across various bike sharing networks, enabling better interoperability and service coordination.

The third chapter, **materials and methods**, describes the design and development process, elaborating on the technical infrastructure, the selection of appropriate development frameworks, and the integration of third-party services into the system. It also covers the methodologies used for data collection and analysis, the choice of **cloud infrastructure**, and the steps taken to ensure **scalability and maintainability**. The chapter provides a comprehensive look at the tools and technologies employed, such as the backend and frontend frameworks, **APIs**, **data management systems**, and **testing environments** used during development.

The **results chapter** presents findings from system implementation. Additionally, it details specific **challenges** encountered during implementation, such as technical hurdles or data integration issues. Solutions adopted to overcome these challenges are also discussed, with insights into future optimizations that could further enhance **system performance**.

2 State of Art

2.1 Introduction

The origin of the bicycle dates back to the late 18th and early 19th centuries, when the first prototypes like the *celerifere* and later the *draisienne* (see Figure 3) were created, the latter invented by Karl Drais in 1817. These early designs paved the way for the evolution of the modern bicycle as we know it today. During the 19th century, the bicycle underwent several significant improvements, such as the addition of pedals by Pierre Michaux in the 1860s, and the development of the high-wheeled design known as the *penny-farthing*. Eventually, the invention of the *safety bicycle*(see Figure 4) in the 1880s, featuring two wheels of similar size and a chain drive to transmit power to the rear wheel, marked the birth of the modern bicycle [18].

Figure 3: Draisienne invented by Karl Drais (around 1820) [18]

Figure 4: 1886 Swift Safety Bicycle [18]

The concept of bike sharing has its roots in the 1960s, when the first program known as "White Bikes Plan" (*Witteefietsenplan*) was implemented in Amsterdam, Netherlands. This system involved making public bicycles available for anyone to use, free of charge. Although the original project did not succeed due to a lack of infrastructure and issues with theft, it laid the foundation for future developments. Later, in the 1990s, cities like Copenhagen adopted more structured programs, and with the advent of digital technology and electronic locking systems, modern bike-sharing programs began to thrive. From the 2000s onwards, initiatives such as *Vélib* in Paris popularized the use of docked systems, and later dockless models expanded thanks to advances in mobile and GPS technologies [19].

2.2 Advanced Bike-Sharing Systems Around the World

Currently, the most advanced bike-sharing models can be observed in various cities around the world. **Citi Bike** [20], in New York, United States, is one of the most prominent systems, with over 24,000 bicycles. This system uses a combination of docking stations and user data to optimize bike availability and improve user experience. Additionally, it has successfully integrated its operations with other forms of public transport, facilitating smooth and efficient urban mobility.

In China, Hangzhou boasts one of the largest bike-sharing systems in the world, known as **Hangzhou Public Bicycle** [21]. This system has a fleet of more than 100,000 bicycles and is distinguished by its effective integration with the city's public transport. Efficient redistribution logistics ensure that bicycles are available according to demand in different areas, significantly contributing to urban mobility.

On the other hand, **Mobike** [22], also originating in China but with an international presence, was a pioneer in the dockless bike-sharing model. This modality offers greater flexibility for users, allowing them to pick up and drop off bicycles anywhere within designated areas. Mobike has integrated technologies such as GPS and the Internet of Things (IoT) into its infrastructure, enabling users to easily find and unlock bicycles using their mobile phones, while operators can manage the fleet more efficiently and in real-time.

Figure 5: Mobike bikes dockless in Huanggang, Hubei, China (CC BY-SA 4.0, Aiko Yamamoto)

2.3 Open Data

Open Data refers to data that is made freely available to the public for anyone to access, use, and share [23]. It is typically provided by governments, research institutions, and other organizations with the aim of promoting **transparency**, **innovation**, and **civic engagement**.

The concept of **Open Data** is fundamental to **smart cities** and modern governance. It typically includes datasets from governments, research institutions, corporations, and even crowdsourcing initiatives. The simplicity of **Open Data** lies in its universal principles of **openness**—anyone can access, use, and share the data without restrictions [24].

2.3.1 Challenges of Open Data

However, while current datasets are generally easy to share, **historical data** presents significant challenges. Collecting, cleaning, and storing historical data is **resource-intensive**, making it costlier and more complex to integrate into open platforms.

Additionally, the value of historical data can be difficult to justify, especially when compared to the immediate and clear utility of up-to-date datasets. The long-term benefits of maintaining historical data, like enriching trend analyses and informed policy-making, are not always easily **quantifiable**, making it harder to advocate for the necessary investment in **infrastructure** and **management** [25].

2.3.2 Open Data Portals

To facilitate smart city initiatives, many governments have established **Open Data portals**, which serve as central hubs for accessing various datasets. Some of the principal Open Data portals include **Barcelona's Open Data BCN** [26], as well as portals from cities like **London** [27], **New York** [28], and **Singapore** [29]. These portals help enhance decision-making, support research, and encourage civic engagement.

2.4 Open Data Standards in Bike-sharing Systems

Open data standards play a critical role in enabling interoperability, efficiency, and ease of integration in bike-sharing systems. This section delves into the various data standards used in bike-sharing systems, focusing specifically on the data structures that underpin these standards.

2.4.1 General Bikeshare Feed Specification (GBFS)

GBFS [30] is one of the most widely adopted standards for bike-sharing systems, developed by the North American Bikeshare Association (NABSA). This JSON-based specification is

designed to provide real-time information about bike-sharing systems, including station information, bike availability, and dock status.

GBFS uses a JSON array of objects, where each object represents a station with fields such as `station_id`, `lat`, `lon`, `num_bikes_available`, and `num_docks_available` (see Listing 1), which are part of station status. Additionally, a call can be made to retrieve more detailed information about the station, known as station information (see Listing 2).

Listing 1: JSON example of `station_status` in GBFS standard

```
{
  "data": {
    "stations": [
      {
        "station_id": "1234",
        "name": "Main Street Station",
        "lat": 40.7486,
        "lon": -73.9864,
        "num_bikes_available": 12,
        "num_docks_available": 5,
        "is_renting": true,
        "last_reported": 1633024800
      },
      {
        "station_id": "5678",
        "name": "Broadway Station",
        "lat": 40.7590,
        "lon": -73.9845,
        "num_bikes_available": 7,
        "num_docks_available": 8,
        "is_renting": true,
        "last_reported": 1633024800
      }
    ]
  }
}
```

Listing 2: JSON example of `station_information` in GBFS standard

```
{
  "data": {
    "stations": [
      {
        "station_id": "1234",
        "name": "Main Street Station",
        "lat": 40.7486,
        "lon": -73.9864,
        "address": "123 Main St, New York, NY",
        "capacity": 20
      },
      {
        "station_id": "5678",
        "name": "Broadway Station",
        "lat": 40.7590,
        "lon": -73.9845,
        "address": "456 Broadway, New York, NY",
        "capacity": 25
      }
    ]
  }
}
```

This structure is suitable for querying station-specific details and allows for straightforward traversal of station lists. JSON is easy to parse and has a hierarchical structure that simplifies data grouping. The format is flexible and human-readable, making it suitable for third-party developers and application integration. However, JSON's schema-less design can result in inconsistent data, especially when dealing with extensions like new vehicle types or dockless bikes. The verbosity of JSON also makes it challenging to handle very large datasets efficiently.

2.4.2 Mobility Data Specification (MDS)

MDS [31] was developed by the Open Mobility Foundation and was initially designed for managing e-scooter deployments but has since been expanded for bike-sharing. It provides a comprehensive framework that helps cities manage and regulate mobility services.

MDS relies on JSON objects to represent data about individual trips, including fields like `trip_id`, `vehicle_id`, `start_time`, `end_time`, and `route`. Listing 3 is an example of an MDS trip event data structure.

The structure is temporal, meaning that it needs to accommodate time-series data effectively. JSON provides a good balance of readability and flexibility, making it easier for mobility operators and city regulators to work with trip data. MDS also includes well-defined event types, facilitating standardization in regulatory oversight. However, as with GBFS, MDS inherits the limitations of JSON in handling massive datasets, particularly time-series data, without an efficient indexing mechanism. Complex operations involving route data and compliance reporting might face performance bottlenecks.

Listing 3: JSON example of the MDS standard

```
{
  "trip_id": "abc123",
  "vehicle_id": "bike789",
  "provider_name": "CityBikes",
  "start_time": "2021-09-30T08:00:00Z",
  "end_time": "2021-09-30T08:30:00Z",
  "route": [
    {
      "lat": 40.7486,
      "lon": -73.9864,
      "timestamp": "2021-09-30T08:05:00Z"
    },
    {
      "lat": 40.7496,
      "lon": -73.9850,
      "timestamp": "2021-09-30T08:15:00Z"
    }
  ],
  "status": "completed"
}
```

2.4.3 General Transit Feed Specification (GTFS)

GTFS [32] is a well-established standard for public transportation data, primarily focused on static schedules and routes. The bike-sharing systems integrate GTFS to align bike-sharing with public transit, offering seamless multi-modal transportation options.

GTFS uses CSV as the primary format to store static data, including `stops.txt`, `routes.txt`, and `trips.txt`. For a representation of `stops.txt` see Listing 4. Each CSV file represents a specific aspect of the transit system. CSV is highly efficient for tabular data, allowing for easy

ingestion by SQL databases. This makes it well-suited for joining data from different tables and efficiently managing relationships between stops, trips, and routes. However, CSV lacks native support for hierarchical relationships or temporal data, making it less versatile for representing dynamic real-time information. Data integrity checks must also be handled externally, as there is no schema enforcement.

Listing 4: CSV example of the `stops.txt` file of GTFS standard

```
stop_id,stop_name,stop_lat,stop_lon
1234,Main Street Station,40.7486,-73.9864
5678,Broadway Station,40.7590,-73.9845
9101,Central Park Station,40.7851,-73.9683
```

2.4.4 Service Interface for Real-Time Information (SIRI)

SIRI [33] is an XML-based specification designed for real-time transit information. While it originated in public transportation systems, it has found application in integrating bike-sharing data for real-time updates.

SIRI utilizes XML to represent dynamic hierarchical data, such as real-time vehicle locations (`VehicleMonitoring`) and stop arrival predictions (`StopMonitoring`), for an example of a `StopMonitoring` message see Listing 5.

XML is structured to handle complex nested relationships effectively. XML is capable of expressing highly nested data, which is useful for representing complex systems with multiple interdependencies. SIRI's schema provides strict validation, making it highly reliable for ensuring consistent data. However, XML is verbose and can become unwieldy for very large datasets. Parsing XML is also more computationally intensive compared to JSON, leading to potential inefficiencies when managing real-time bike-sharing updates.

Listing 5: XML example of `VehicleMonitoring` message of the SIRI standard

```
<VehicleMonitoringDelivery>
  <VehicleActivity>
    <MonitoredVehicleJourney>
      <LineRef>1</LineRef>
      <VehicleRef>Bus123</VehicleRef>
      <DirectionRef>Northbound</DirectionRef>
      <MonitoredCall>
        <StopPointName>Main Street Station</StopPointName>
        <ExpectedArrivalTime>2021-09-30T08:05:00Z</ExpectedArrivalTime
           >
      </MonitoredCall>
    </MonitoredVehicleJourney>
  </VehicleActivity>
</VehicleMonitoringDelivery>
```

2.4.5 Generic Technical Infrastructure

The technical infrastructure that supports data collection in bike-sharing systems is based on IoT and sensor networks embedded into the bikes and docking stations. The most advanced infrastructure typically relies on GPS modules for tracking bike positions, providing real-time location updates to fleet managers. Inertial sensors such as accelerometers and gyroscopes help detect unauthorized movement or potential accidents, while communication modules like 4G LTE, LoRaWAN, and NB-IoT are responsible for transmitting data to centralized cloud services. These cloud platforms, such as AWS, Azure, or Google Cloud, form the backend infrastructure for data aggregation and processing. Streaming platforms like Apache Kafka and Apache Flink are also crucial in managing real-time data streams, helping to monitor bike availability and optimize rebalancing of fleets [34].

2.5 Open Data Visualization in Bike-sharing

A global open data infrastructure provides the community with the opportunity to build applications and create innovative solutions. In the following paragraphs, I will cover tools that represent the cutting edge of functionality and user experience, highlighting how they make use of open data to improve urban mobility.

The first tool, **Bike Share Map** [35], is an interactive platform that provides real-time data about bike-sharing stations in every city. Developed by Oliver O'Brien, this tool is highly useful for both residents and tourists who wish to get around the city in a sustainable way. The

map shows the availability of bikes and docking stations, helping users plan their trips with ease. Its intuitive design and reliable data make it a convenient choice for anyone relying on bike-sharing services.

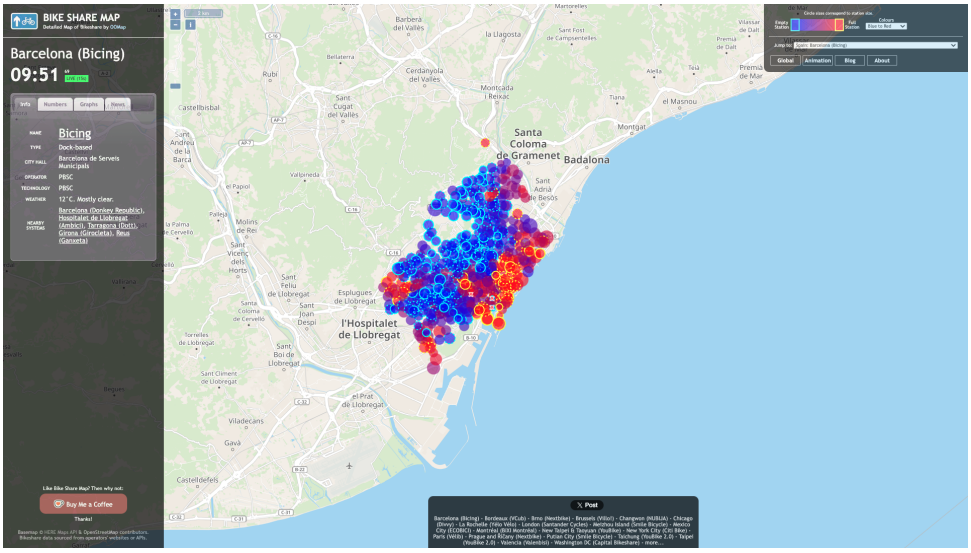

Figure 6: Bike Share Map [35]

Citymapper [36] is known for its comprehensive travel-planning features, It provides users with multiple transport options, including walking, biking, metro, buses, and even taxis. It integrates real-time data to give the most efficient routes available, making it ideal for daily commuters and newcomers alike. The app's user-friendly interface and the ability to compare different modes of transportation have established Citymapper as a go-to tool for navigating complex urban landscape.

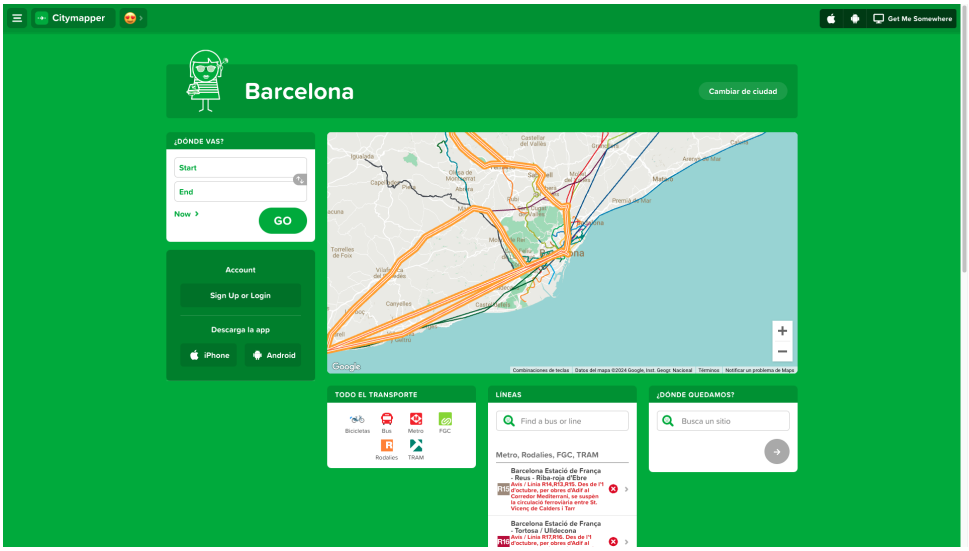

Figure 7: Citymapper [36]

The third tool, **Bike Sharing World Map** [37], offers a global perspective on bike-sharing systems, and Barcelona is one of the highlighted cities. Created by Paul DeMaio, this map serves as an overview of how bike-sharing networks operate worldwide, offering a bird's-eye view of different schemes. It is particularly valuable for urban planners, researchers, or individuals interested in comparative analysis of bike-sharing initiatives across different regions. The tool doesn't provide detailed station-level data like the Barcelona Bike Share Map but serves as an important resource for understanding the global context of shared mobility.

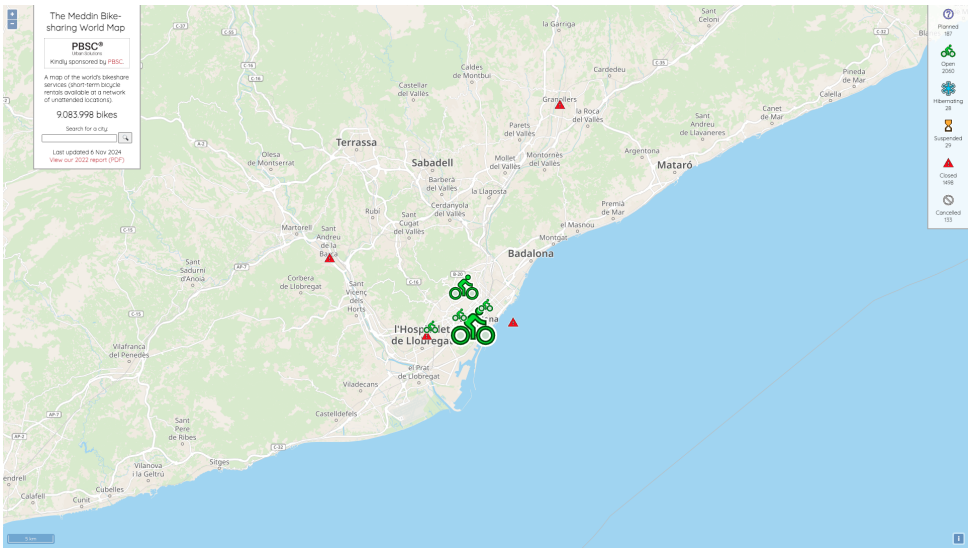

Figure 8: Bike Sharing World Map [37]

3 Methods and resources

3.1 Why Bicing?

The study of Barcelona's shared mobility system, particularly the **Bicing service** [5], is highly significant for the **Universitat Oberta de Catalunya (UOC)**, both academically and practically. Barcelona exemplifies a city that promotes open data access through its **Open Data system** [26], allowing developers, researchers, and citizens to access real-time urban metrics, including **public transport** and **sustainable mobility**.

Technically, Barcelona uses the **General Bikeshare Feed Specification (GBFS)** [30] to manage real-time data for Bicing. This globally adopted standard ensures interoperability among bike-sharing systems and serves as a model for integration with the city's Open Data platform. Through GBFS, data on **bike availability**, **station status**, and **operational conditions** are easily accessible, supporting real-time management and analysis.

A standout possibility of Bicing data is the access of a **minute-by-minute snapshot data storage system** since late 2021. Unlike most APIs providing only current data, these high-frequency data enables a detailed analysis of **user behavior**, **bike distribution**, and **public policy impacts** on urban mobility. Access to these detailed historical data allows for robust research into **usage patterns**, **system efficiency**, and **predictive models** for optimizing resource management.

3.2 Real-Time Data vs Historical Data Processing

The proposed Bicing system combines real-time and historical data, each with distinct but complementary objectives and uses. This section analyzes how both types of data are managed and integrated within the system.

3.2.1 Real-Time Data

Real-time data are provided directly by the Bicing API, following the GBFS standard. Every minute, a snapshot of bike availability and station status is obtained, which is displayed on the web application to help users make informed decisions about their journeys.

Real-time data offer several benefits. Firstly, it provides **instant information**, enabling users to know the exact availability of bikes and parking spots at any given moment. In addition, it contributes to **operational optimization**, helping operators make quick decisions about bike redistribution based on demand.

However, there are also challenges associated with real-time data. One challenge is the need for **minimal latency**; although the data is not processed, it must be presented with the

least possible delay to be useful for immediate decision making. Another limitation is that the real-time API does not provide **detailed information** on user trips, which restricts certain types of analysis.

3.2.2 Historical Data

Historical data analysis allows the study of trends and usage patterns. Data collected minute by minute are stored in a historical bucket, compressed in **Parquet** format to improve the efficiency of storage and analysis. This information is used to generate detailed reports and analyses using the **analytical API**, developed with **FastAPI**.

The analysis of historical data provides several benefits. It allows identification of **daily and seasonal trends** in bike demand, allowing a more effective analysis of usage patterns. Furthermore, it helps in **predicting demand**, which allows better anticipating future needs and optimizing the distribution of bikes based on historical data.

Historical data also present challenges. The accumulation of minute-by-minute data generates a **large volume** of information, requiring an efficient infrastructure to handle it. To address this, **compression using Parquet** was implemented to significantly reduce file size and improve performance.

3.2.3 Complementarity of Real-Time and Historical Data

Both types of data are essential for the system's operation. **Real-time data** provides a current view for users and facilitates immediate operational management, while **historical data** offers detailed insights for long-term planning. Real-time data informs the current situation of stations, whereas historical data helps analyze trends and continuously improve the system.

For example, while operators can observe **saturation stations** in real time to make immediate decisions, historical data helps them plan system expansion and **redistribute resources** more efficiently. This synergy between both data types ensures an optimal user experience and efficient operation of the Bicing system.

3.3 Project Structure

3.3.1 Internal Services - to Develop in This Project

3.3.2 Analytical API

The analytical API, developed using **FastAPI** [11] and **Python** [9], analyzes historical data stored in the snapshots bucket. Provides insights such as usage trends and peak hours. Hosting

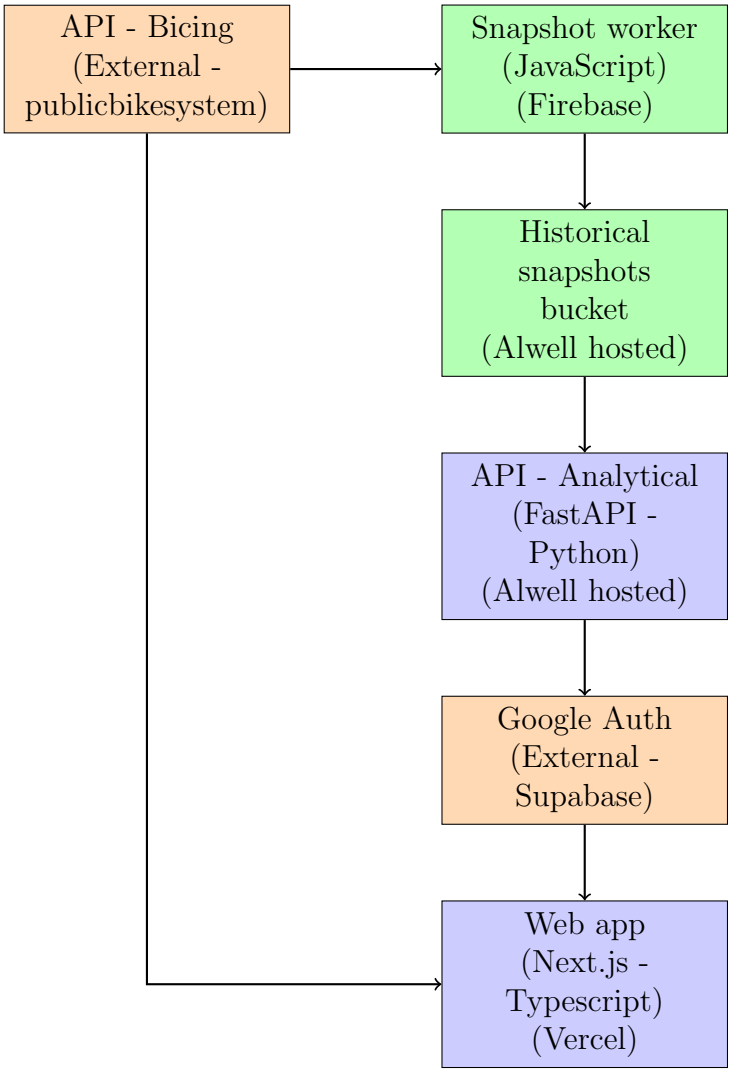

Figure 9: System architecture

this internally allows for greater control and lower latency when processing data. The API has two main functionalities:

1. Providing station statistics (number of bike check-ins/check-outs, station usage ranking, average time the station is empty, average time it is full, etc.) over a given period of time.

2. Providing an overview of the evolution of a specific metric over time (usage rate of a station or a group of stations during the last hour throughout the day).

3.3.3 Web application

The web application, created with **Next.js** [12] and **Typescript** [10], provides the user interface to view historical and real-time data. It integrates with both the analytical API and the real-

time data API to deliver an interactive experience for end-users.

In the application, there are mainly three views –pages– that logged-in users can access and only one (the home page) that nonlogged-in users can access:

- **Home**: Real-time visualization of bike sharing (bicing) data and detailed view of data for each station (see Figure 10).

- **Analytics**: Visualization of basic metrics for stations, neighborhoods, districts, or at the city level (see Figure 11).

- **Tasks**: A feature to assist managers in performing basic interventions in the system, such as supplying stations or removing disabled bikes from the system (see Figure 12).

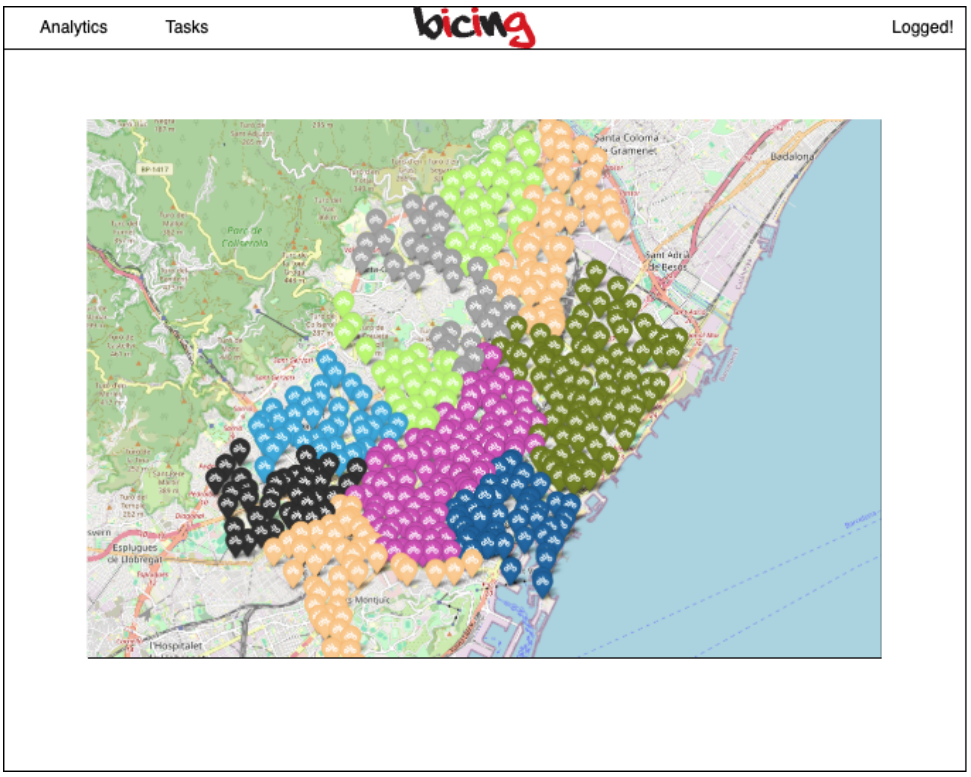

Figure 10: Mock-up - Home

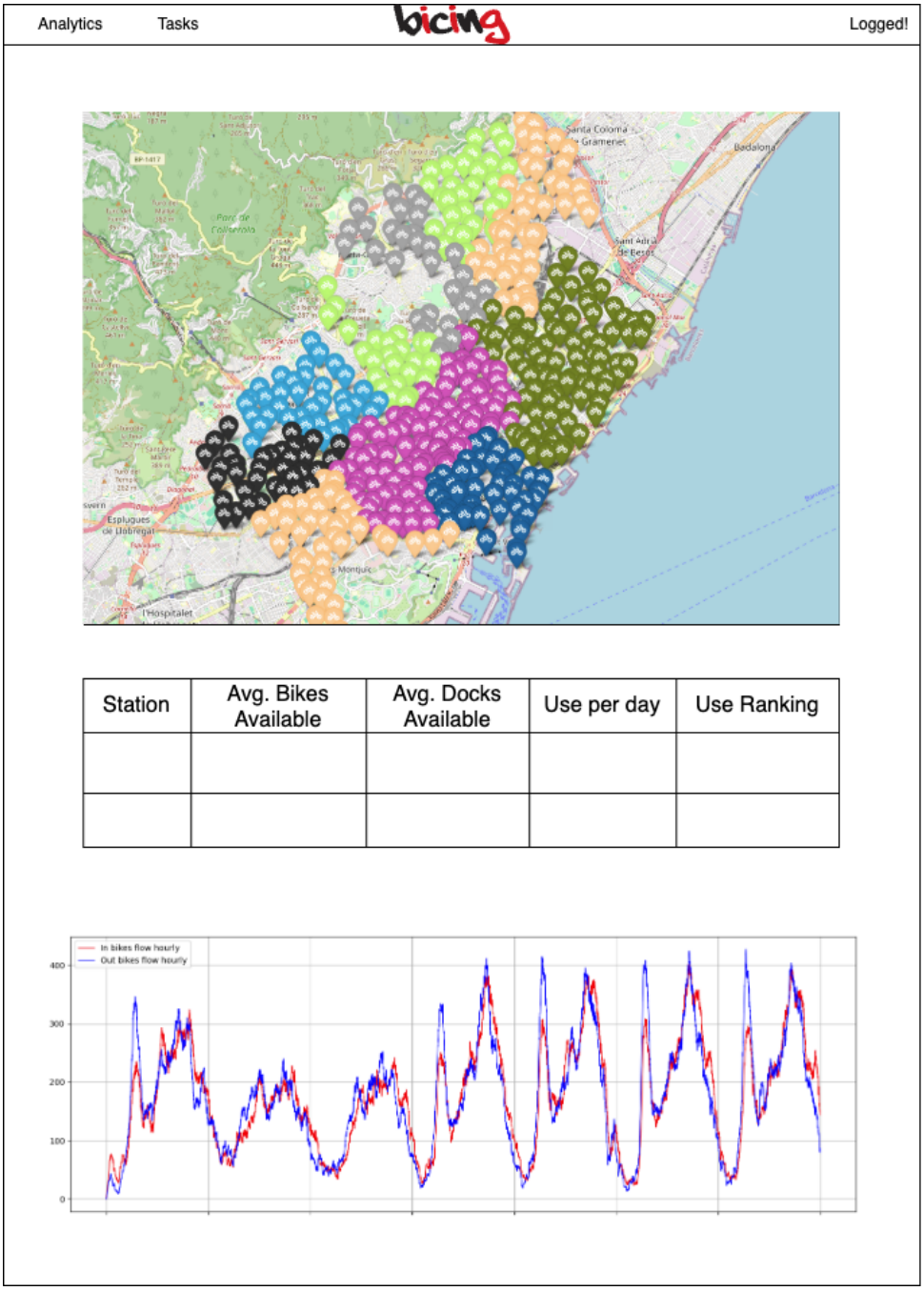

Figure 11: Mock-up - Analytics

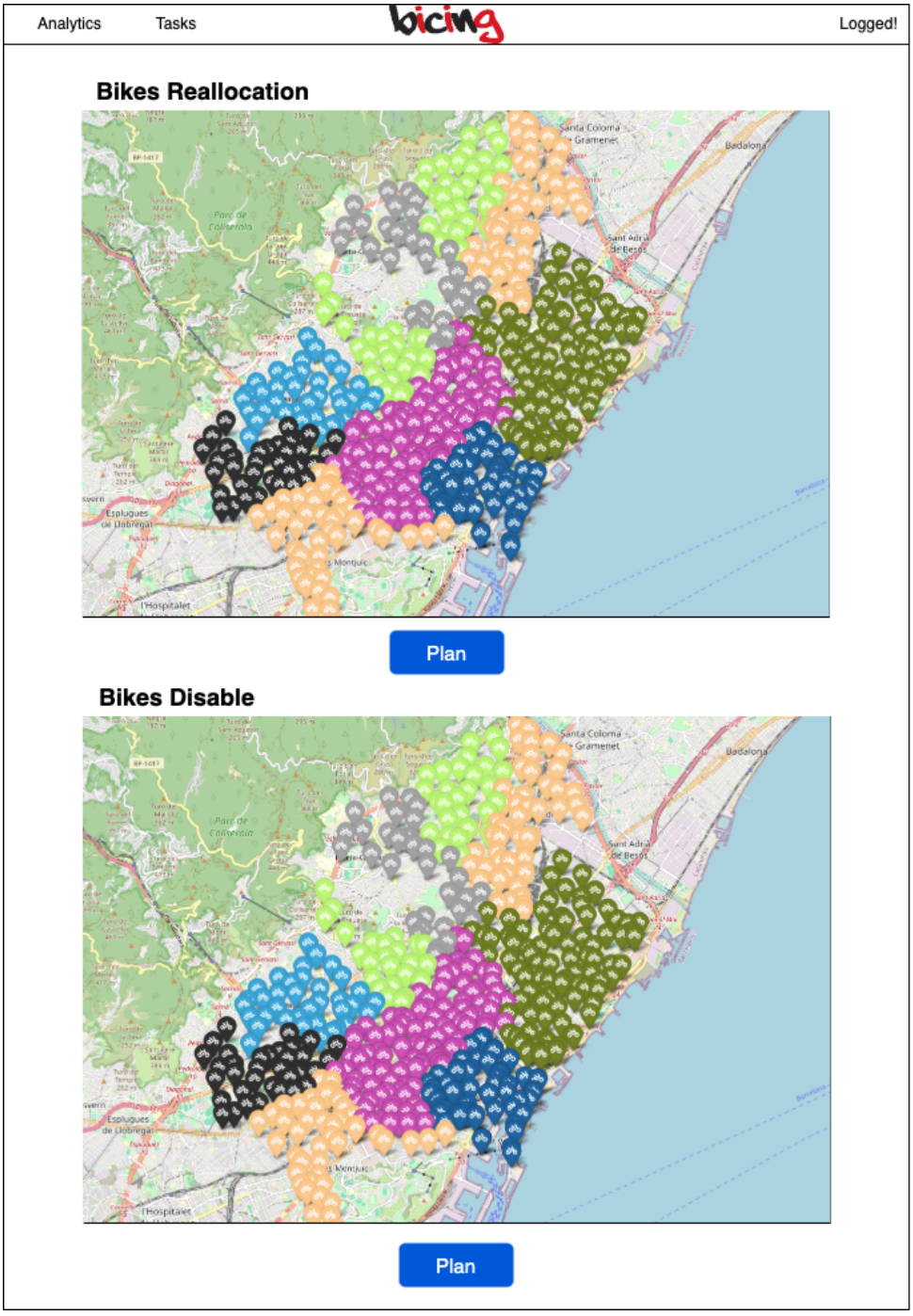

Figure 12: Mock-up - Tasks

3.3.4 Internal Services - Pre-existing

The snapshot worker collects data snapshots from the **Bicing API** [38], which provides data on bicycle availability. This worker stores snapshots in a historical bucket, allowing for comprehensive analysis over time to identify trends and usage patterns.

The historical snapshots bucket serves as internal storage for the data collected by the snapshot worker. It acts as a repository for future analytics and trend analysis.

3.3.5 External Services

The **Bicing API** is an external API used to access real-time bicycle availability data. The API only returns real-time data.

Google authentication, managed via **Supabase** [13], is used for secure user authentication through Google. It also serves to optimize the resources of the AIWell cluster, preventing unverified users from accessing the data or returning information from the analytics API.

The real-time data API, integrated with **Publicbikesystem** [38], provides live bicycle availability information to support timely user decisions.

4 Results

4.1 General Overview

In this updated system architecture, a **compressor** has been added between the **Snapshot Worker** and the **Historical Snapshots Bucket** stages. The compressor's role is to reduce the size of the data before it is stored in the bucket, improving storage efficiency and reducing potential costs. The data flow from the Snapshot Worker to the Compressor manually, where it undergoes compression, and then to the Bucket for long-term storage. This modification optimizes the process of managing historical data, ensuring that large datasets are stored in a more compact form. For a deeper understanding of the technical details behind this change and the data format transformation, please refer to section `Optimizing Data Storage: From JSON to Parquet`(see subsubsection 4.5.2).

4.2 Historical Data Capture

The process of data capture involves extracting relevant information from a specific source to create a comprehensive dataset. The main objective is to gather data that is **accurate, consistent, and ready for further processing** or analysis. In this approach, JavaScript functions are utilized and deployed as **Firebase Cloud Functions** [39] to connect to external APIs, extract data, and process it.

The data capture is initiated by making HTTP requests to the relevant APIs, specifically using the **GBFS standard** [30], and more concretely to the station-status endpoint. The received data is then cleaned and formatted to ensure consistency and to avoid discrepancies. The sampling latency is set to one minute, meaning that data is captured every minute, which ensures near real-time monitoring and updates. The following snippet shows the core of the data capture functions implemented in JavaScript using Firebase Cloud Functions.

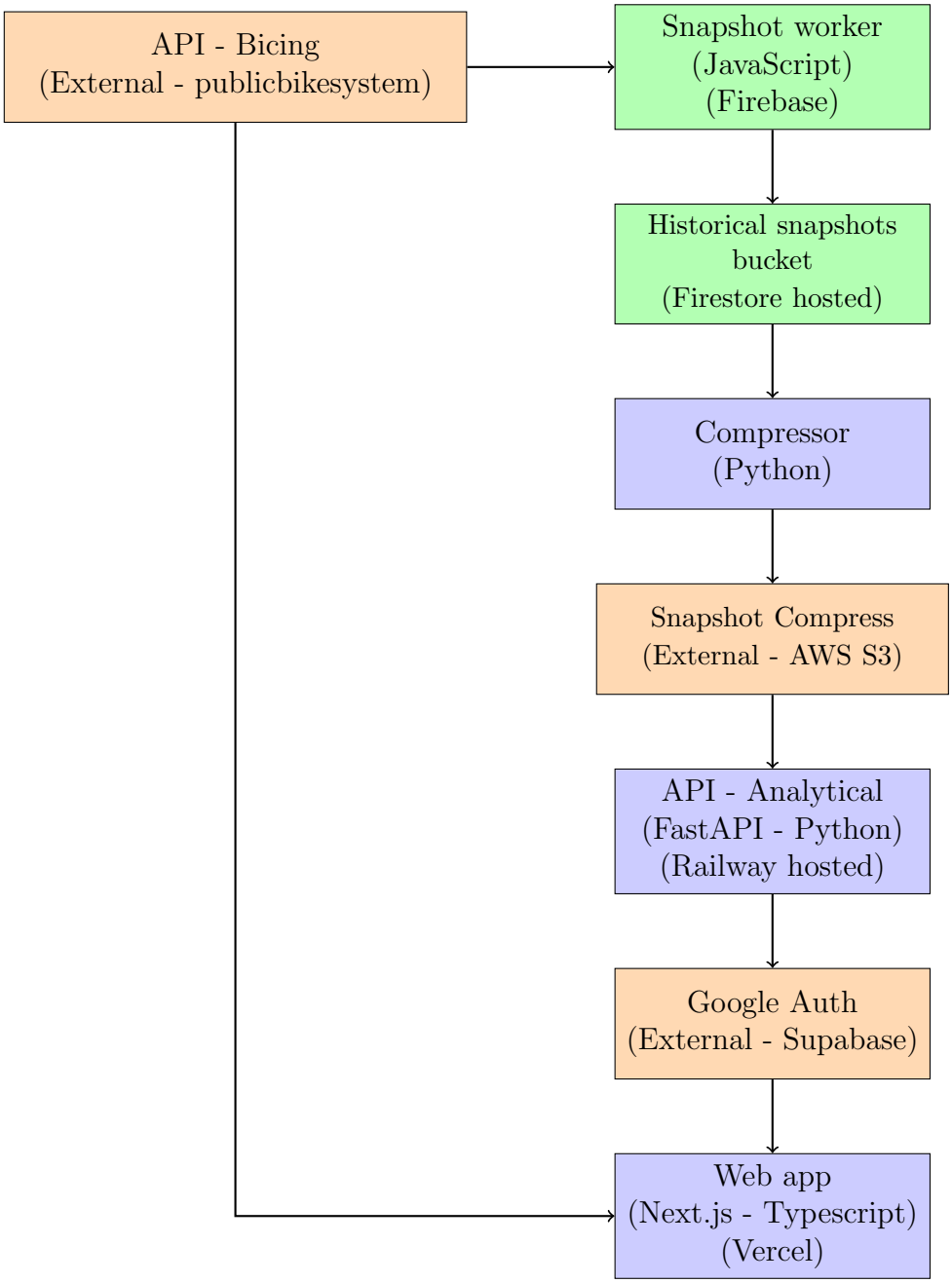

Figure 13: System architecture with data compression

Listing 6: Function to capture GBFS Bicing snapshots

```
const functions = require("firebase-functions");
const fetch = require('node-fetch');
const admin = require('firebase-admin');
admin.initializeApp();

exports.gbfsSnapshot = functions
    .pubsub.schedule('every minute')
    .onRun(async (context) => {
        let settings = { method: "Get" };
        let url = "https://barcelona-sp.publicbikesystem.net/customer
           /ube/gbfs/v1/en/station_status";
        let response_gbfs = await fetch(url, settings);
        let json_gbfs = await response_gbfs.json();

        let timestamp = json_gbfs["last_updated"];
        let date = new Date(timestamp * 1000);
        let year = date.getFullYear().toString();
        let month = (date.getMonth() + 1).toString().padStart(2, "0")
           ;
        let day = date.getDate().toString().padStart(2, "0");

        json_gbfs["ttl"] = -1;

        const writeResult = await admin.firestore()
            .collection(`snapshots_${year}-${month}-${day}`)
            .doc(timestamp.toString())
            .set(json_gbfs);

        functions.logger.info("Success writing snapshot.", {
           structuredData: true});

        return null;
    });
```

In the code snippet (see Listing 6), Firebase Cloud Functions and the "node-fetch" library are used to make HTTP GET requests to remote API endpoints. The responses are parsed, cleaned, and mapped into the desired structure before storing them in Firebase Cloud Store. The error-handling mechanism ensures that any issues during the data capture process are

logged, and appropriate responses are provided [40]. The sampling latency is configured to one minute, which allows frequent and timely updates of the data set.

One issue is that there are no origin and destination data in GBFS because it focuses on docks rather than individual bicycles. Although this structure is not ideal for tracking individual journeys, it is the only current available data structure. This design choice aligns with the data structure used in the Bicing project, which also emphasizes station-based information over individual bike tracking. As a result, GBFS data is suitable for applications that monitor dock availability and overall system status, but not for detailed analysis of specific bike journeys or user movement patterns.

4.3 Analytical API Overview

The **analytical API** [41] is a robust and scalable API designed to provide an advanced analytical layer for Barcelona's bike-sharing system, known as Bicing. This repository centralizes and facilitates access to critical data, which is essential for both system operators and end users, contributing significantly to the continuous optimization and improvement of urban mobility services.

4.3.1 Core Features

The API provides multiple endpoints to obtain data on current bike statuses, inflow and outflow, as well as historical information. These endpoints are easy to use and flexible enough to adapt to various query needs. It is designed to handle large volumes of data and multiple simultaneous requests, ensuring optimal performance even during peak demand, delivering a smooth experience for all users. By implementing industry-standard security practices, such as CORS management and proper exception handling, the API secures sensitive data and ensures system integrity. Interactive documentation is also included, accessible at `https://bcn-bicing-api.up.railway.app/docs`, which helps developers understand and use the API efficiently, promoting quick and effective adoption.

4.3.2 Main Endpoints

It includes several endpoints to manage and access data effectively. The `/stats` endpoint provides the current status of all Bicing stations, including available bikes and free docks. Additionally, the `/flow` endpoint shows inflow and outflow data for bikes in various areas, allowing operators to understand movement patterns throughout the city.

4.3.3 Technologies Used

The API is built using FastAPI, which is a modern, high-performance framework for building APIs with Python that allows for rapid deployment and ease of scalability. It utilizes Pandas and GeoPandas, which are powerful libraries for data analysis and manipulation, essential for processing the geospatial data of bikes and stations. Shapely and PyProj are used for geometric data processing and analysis, which are crucial for determining station locations and distribution. Additionally, Uvicorn is used as a high-speed ASGI (Asynchronous Server Gateway Interface) server that ensures optimal performance and rapid response delivery for API requests.

4.3.4 Deployment

The API deployment was carried out on Railway [42] servers, ensuring a streamlined build process and consistency between environments. The implemented API is optimized to handle multiple concurrent requests, ensuring reliability and scalability. Services such as the historical Bicing data bucket hosted on AWS S3 [43] via Supabase are seamlessly integrated, maintaining the modularity of the system.

4.4 Web App Overview

The **Barcelona Bicing Dashboard** [15] is a comprehensive, user-friendly visualization tool that combines real-time data with processed analytics via an API, designed to improve the user experience of the Bicing bike-sharing system in Barcelona. This dashboard provides real-time information on bike availability, station status, and usage patterns across the city.

4.4.1 Core Features

- **Interactive Map**: A geographic visualization of all Bicing stations, allowing users to easily navigate and identify available bikes or docks.
- **Real-Time Updates**: Instant information on the availability of bikes and docks, helping users plan their trips more effectively.
- **Advanced Filters**: Options to filter stations according to operational status, capacity, and other relevant criteria, providing a personalized search experience.
- **Data Visualizations**: Charts and tables that display usage metrics, trends, and historical analysis to support strategic decision making for both users and operators.

- **Responsive Design**: The app is optimized for a seamless experience on both desktop and mobile devices, ensuring accessibility and convenience for all users.

4.4.2 Main Routes

- **Home**: The home route (see Figure 14) provides an overview of the Bicing system, showing key metrics, recent updates, and a summary of bike availability across the city. It serves as the main entry point for users to quickly get an understanding of the current state of the system.

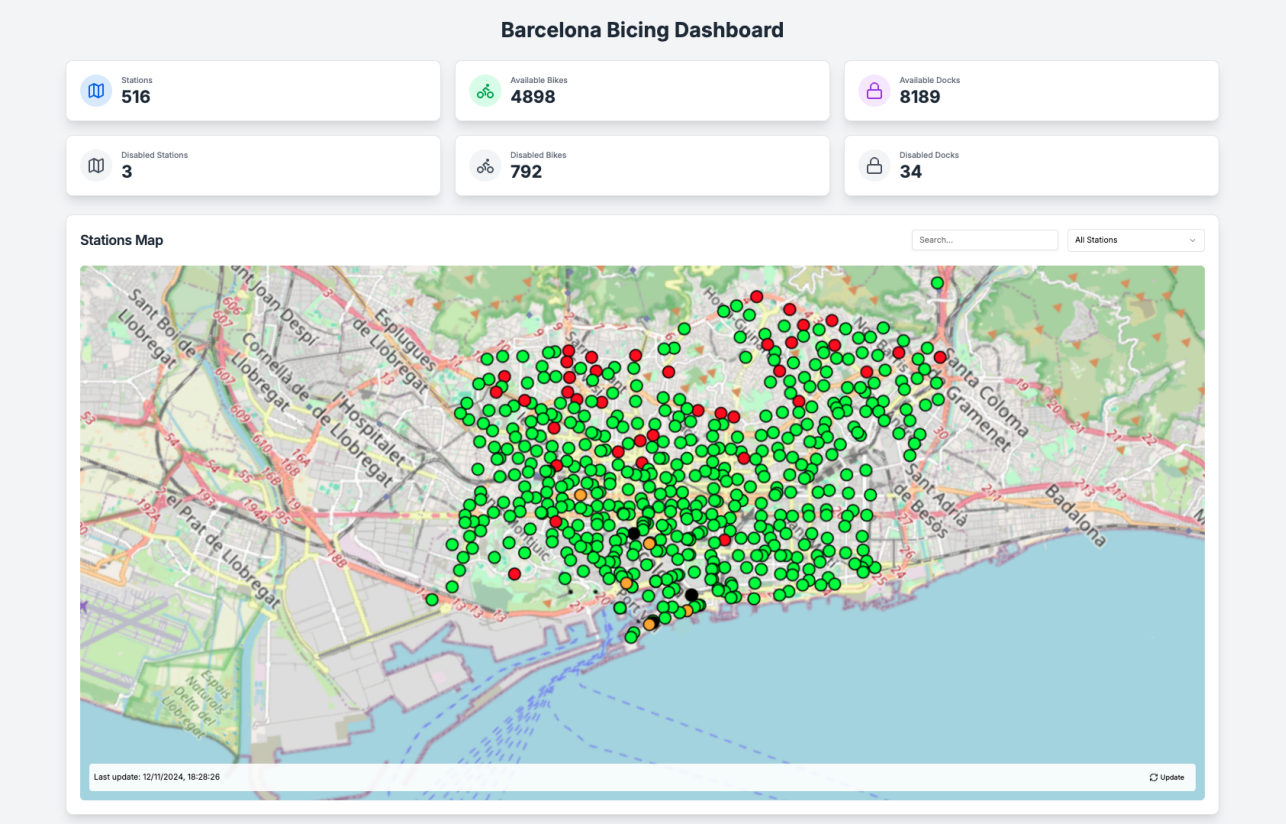

Figure 14: Home page

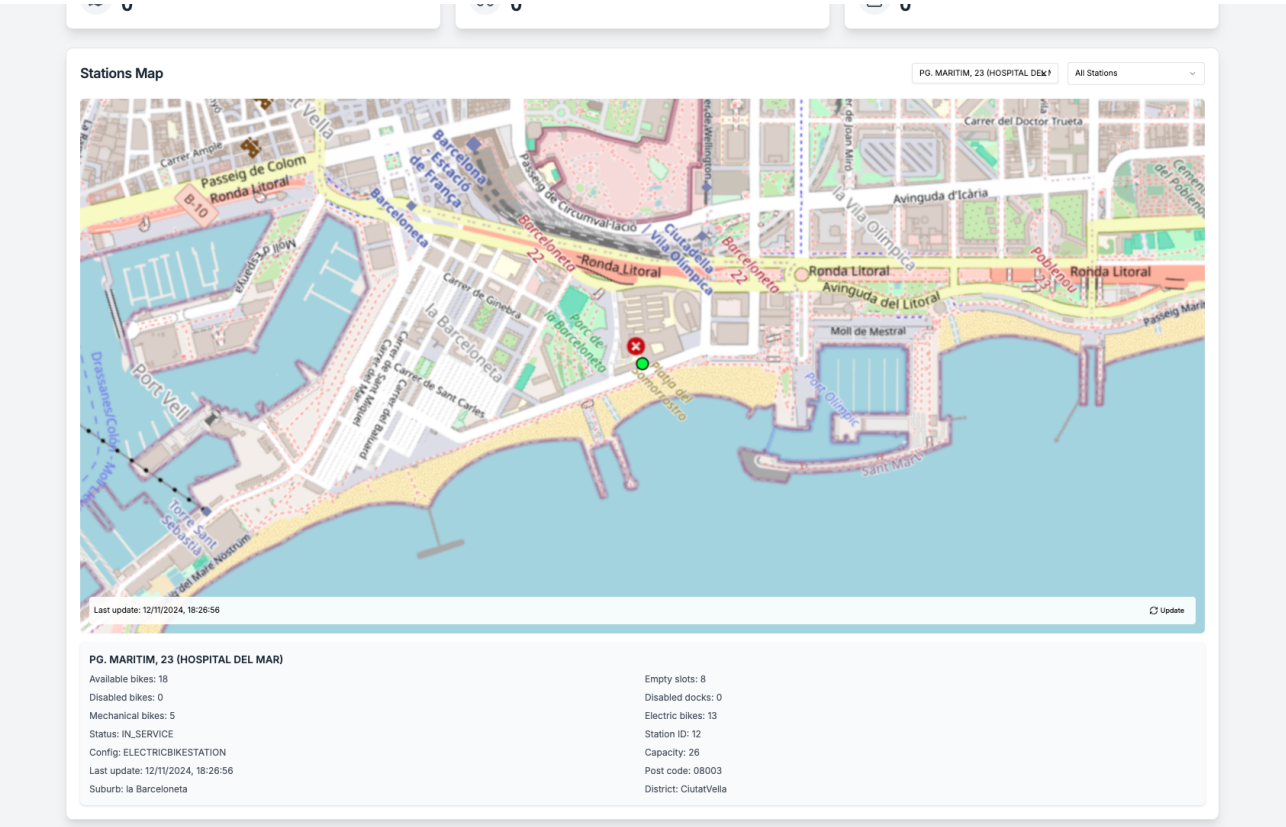

Figure 15: Home page - station detail

- **Analytics**: The analytics route offers detailed data visualizations and insights into usage patterns, trends, and station performance. This route is also connected to the analytics API, providing real-time data and deeper insights. This section is crucial for users interested in understanding the broader impact and efficiency of the Bicing system.

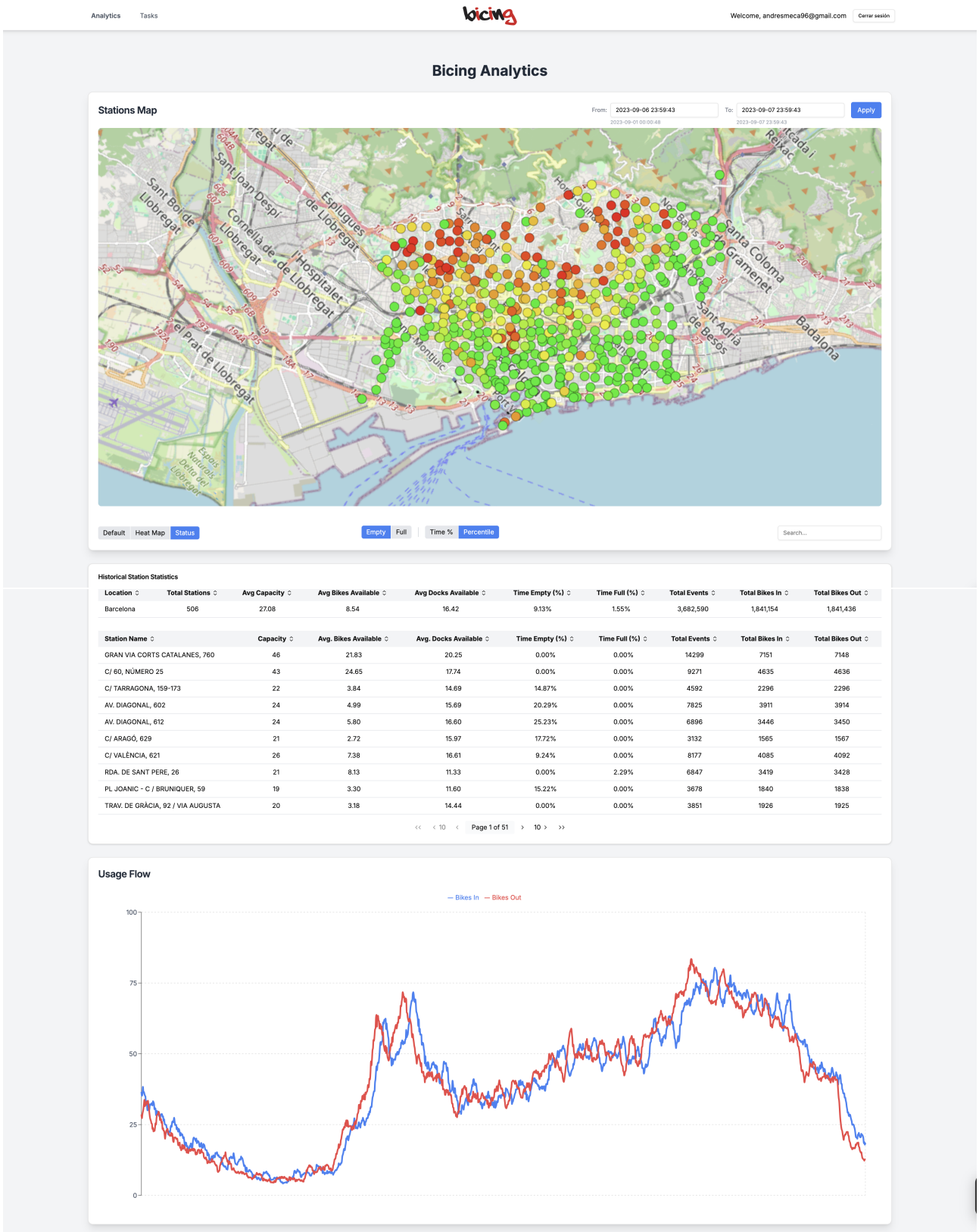

Figure 16: Analytics page

- **Tasks**: The tasks route(see Figure 17) is designed for operators and administrators to manage the redistribution of bikes, track maintenance activities, and monitor station statuses. It supports the operational side of Bicing by providing tools for efficient resource management.

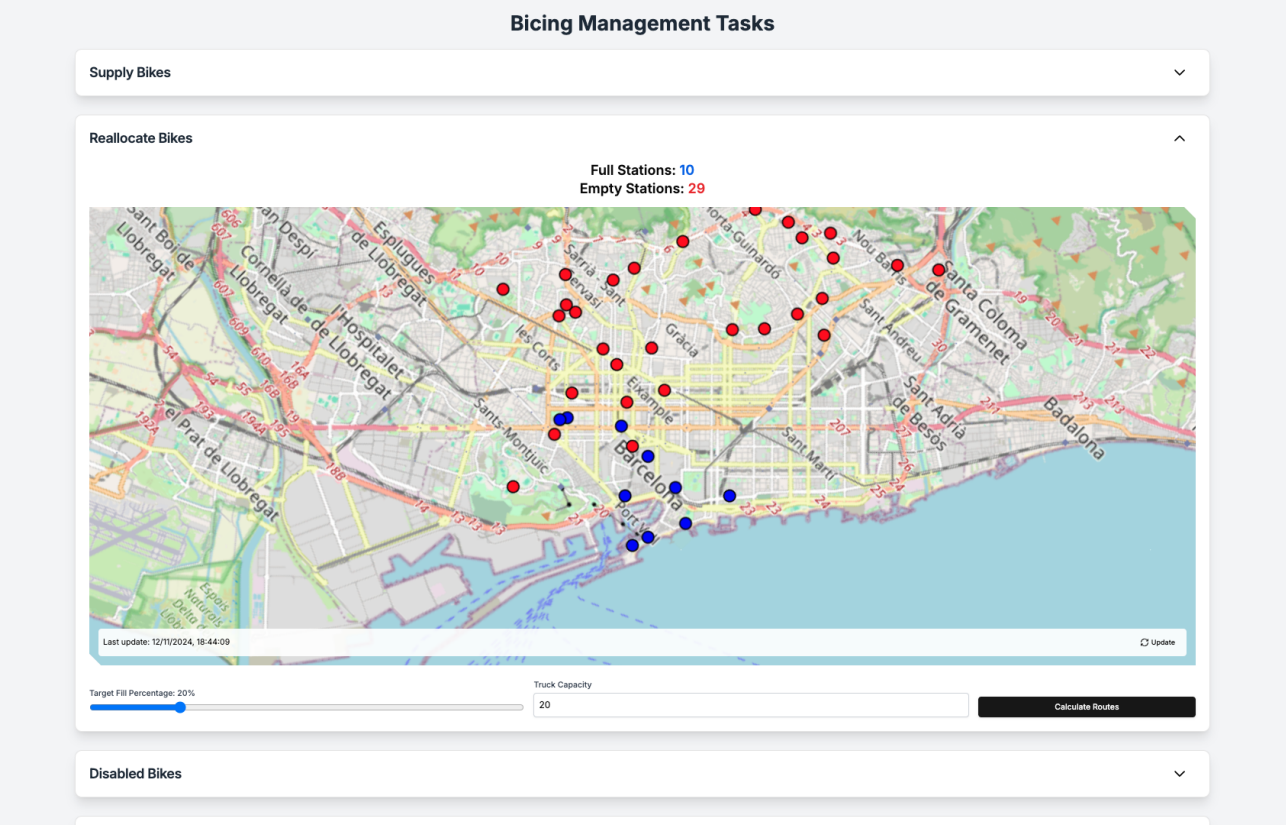

Figure 17: Tasks Page

4.4.3 Technologies Used

By integrating modern technologies such as **Next.js**, **React**, and **TypeScript**, the web application delivers a fast, scalable and user-friendly platform. Additional features, such as data visualizations through **Recharts** and mapping via **Leaflet**, provide a detailed and dynamic experience. The design is further enhanced by **Tailwind CSS**, ensuring a consistent and modern user interface. Google authentication is implemented via **Supabase** to provide a secure and seamless login experience.

4.4.4 Deployment

The deployment has been carried out on the **Vercel** [14] platform, using its robust hosting capabilities optimized for modern web applications. Vercel's seamless integration with **Next.js**

allows efficient server-side rendering and static site generation, ensuring that the dashboard remains **highly performant** and **scalable**.

Vercel provides **automatic deployment** upon merging changes to the main branch, offering continuous deployment that keeps the application up-to-date with the latest code. This ensures that the development process remains fluid, with **rapid iteration** and **immediate reflection** of updates in production.

Environment variables and secrets, such as **API keys** for Bicing data and **Supabase credentials**, are securely managed through Vercel's built-in environment management, ensuring secure handling of **sensitive information**.

4.5 Modifications During Development

4.5.1 From Citibik API to Publicbikesystem API

During the development of the web app (November 1, 2024), a strategic decision was made to replace the original CityBik [44] realtime API with the official Barcelona bikesharing API [38]. This transition was driven by the need for more comprehensive data specific to the city of Barcelona, which would enable richer analysis and better functionality for users of the web app.

The CityBik API offered several advantages that initially made it highly convenient, primarily because it aggregated station information (such as station names and coordinates) and status data (number of bikes available, number of docks free) into a single call. This streamlined approach made it particularly effective for presenting information on a map, minimizing the number of API requests, and thereby reducing latency.

However, CityBik is designed to be a global service that aggregates APIs from bike-sharing systems around the world. Consequently, it focuses solely on providing essential and standardized data to ensure consistency across diverse bike-sharing networks. Although this approach is excellent for broad-based applications requiring uniformity, it does not allow for the depth of data that was necessary for this project.

In contrast, the official API of the City of Barcelona offers a richer and more detailed dataset, including additional metrics that provide deeper insight into the bikesharing system. To utilize this data, it was necessary to combine two of its endpoints—`station_status` (Listing 1) and `station_information` (Listing 2)—to present the comprehensive data needed for our purposes. Although this required more complex integration compared to the CityBik API, it ultimately allowed for a more accurate, real-time representation of station statuses, thereby enhancing the user experience. This additional level of detail is crucial for features such as providing more reliable information to users.

4.5.2 Optimizing Data Storage: From JSON to Parquet

On November 29, 2024, when attempting to allocate files to a server (Google Drive) for reading, we found that uploading these files was neither practical nor efficient, and the reading process was slow. This led us to reconsider the data structure that we were using in search of a more efficient solution. After evaluating various options, we decided to transition to the **Parquet format** [45] due to its advantages in both performance and storage efficiency.

To validate the compression efficiency of Parquet, we conducted a compression test using 7 full days of data. The results were as follows:

- **Number of records in the dataset:** 10,078 records.
- **Parquet file size:** 31.11 MB.
- **Size of JSON files:** 1,668.64 MB.
- **Compression ratio (JSON/Parquet):** 53.64x.

As shown by these results, Parquet significantly outperforms JSON in terms of storage efficiency, achieving a compression ratio of **53.64x**. This dramatic reduction in file size not only reduces storage requirements, but also accelerates data access and processing times, which is critical to improving the overall performance of the system.

Parquet is a columnar file format designed for efficient data processing and storage. Its primary advantages over **JSON** include:

- **Compression efficiency:** Parquet files are highly compressed, which significantly reduces storage requirements and speeds up read/write operations compared to JSON, which can be more verbose and less optimized for large datasets.
- **Columnar storage:** Unlike JSON, which stores data in a row-based format, Parquet stores data in columns. This allows for faster access to specific columns, especially when dealing with large datasets where only a subset of the data is needed.
- **Schema evolution:** Parquet supports schema evolution, meaning it can adapt to changes in data structure without requiring significant changes to the underlying system or data model.

The performance of both solutions was also compared. This was done by calculating the processing time for the most demanding operation: calculating the flow of incoming and outgoing bicycles at stations throughout the city of Barcelona over a week (see Listing 7).

Listing 7: Output of performance test comparing JSON vs Parquet

```
Running performance test with 5 iterations...
Time period: 2023-09-01 00:00:00 to 2023-09-08 00:00:00
Found 10078 JSON files and 1 Parquet files

Testing JSON performance...
JSON Iteration 1/5 completed in 32.35 seconds
JSON Iteration 2/5 completed in 32.82 seconds
JSON Iteration 3/5 completed in 32.93 seconds
JSON Iteration 4/5 completed in 33.07 seconds
JSON Iteration 5/5 completed in 33.07 seconds

Testing Parquet performance...
Parquet Iteration 1/5 completed in 18.20 seconds
Parquet Iteration 2/5 completed in 17.33 seconds
Parquet Iteration 3/5 completed in 17.44 seconds
Parquet Iteration 4/5 completed in 17.39 seconds
Parquet Iteration 5/5 completed in 17.35 seconds

JSON Performance:
Mean time: 32.848 seconds
Min time: 32.348 seconds
Max time: 33.074 seconds
Std dev: 0.300 seconds

Parquet Performance:
Mean time: 17.542 seconds
Min time: 17.331 seconds
Max time: 18.196 seconds
Std dev: 0.368 seconds

Performance Comparison:
--------------------------------------------------
JSON average time:    32.848 seconds
Parquet average time: 17.542 seconds
Speed difference:     15.307 seconds
Parquet is 46.6% faster than JSON
Parquet processes data 1.9x faster than JSON
```

4.6 Findings

4.6.1 User Satisfaction and Altitude

The visualization of stations that were frequently empty revealed that the stations most likely to be empty were those located farther from the sea and, consequently, at higher altitudes. To identify the impact of this factor on user experience, district user experience data was reviewed [46].

An observation identified throughout the development of this research is the presence of a statistically significant negative correlation between user satisfaction [46] and both the mean altitude of the district's stations and the capacity-weighted average altitude Table 2 [47]. This negative correlation can be attributed, as discussed in the Introduction, to the logistical difficulties associated with the restocking of bicycles in areas of higher elevation. These challenges are likely to affect the efficiency of bicycle availability, ultimately leading to a decrease in user satisfaction, particularly in districts characterized by considerable average altitude.

Table 2: Correlation factors over the years, comparing simple and weighted altitude calculations from 2019 to 2023.

Year	Simple	Weighted
2019	-0.766	-0.769
2020	-0.794	-0.799
2021	-0.837	-0.839
2022	-0.664	-0.669
2023	-0.745	-0.746

Table 3: Detailed view of the most recent recorded year (2023).

District Code	District Name	Weighted Altitude	Simple Altitude	Satisfaction
01	Ciutat Vella	6.40	6.36	6.7
02	Eixample	24.51	25.17	6.2
03	Sants-Montjuïc	25.39	25.50	6.2
04	Les Corts	60.79	60.29	6.0
05	Sarrià-Sant Gervasi	87.40	87.20	6.0
06	Gràcia	79.45	79.77	6.1
07	Horta-Guinardó	89.72	88.64	6.2
08	Nou Barris	64.87	66.62	6.3
09	Sant Andreu	26.90	27.11	6.5
10	Sant Martí	7.43	7.66	6.5

5 Conclusions and future work

5.1 Evaluation of Results

The evaluation of the project highlights key outcomes in three main areas: the **API system**, the **web interface**, and **general operational results**. The **API system** provided a robust backend for data analysis and processing, leveraging **historical data** to generate insightful metrics and facilitate smooth integration with other components. The **web interface** offered an intuitive and responsive platform for users, integrating both the **analytical API** and the **real-time API** provided by the **Open Data system** of the city council. Together, these components significantly enhanced data management and user engagement, providing reliable **real-time updates** and actionable insights. Despite certain challenges, such as computational demands and limited scope of some analyses, the project established a strong foundation for future improvements.

5.2 Achievement of Objectives

The objectives outlined for the project were met with varying degrees of success.

1. **General Objective**:

   Successfully developed a web-based system to optimize the management of the Bicycle Sharing system of Bicing in Barcelona. The web application provided intuitive interfaces for both operators and users, fulfilling the central goal.

2. **Specific Objectives**:

   - **Analysis of Station Usage Patterns**: This objective was not fully achieved, but the foundations were laid to enable its future achievement.
   - **Frontend Development**: Delivered a user-friendly web application with real-time data visualization and analytical tools, significantly enhancing user interaction.
   - **Backend API Implementation**: Successfully implemented a robust API that facilitates data management and analysis with high performance and security.
   - **Experimentation with Models**: Conducted extensive trials using notebooks, leading to the integration of predictive models that improved resource allocation and system reliability.
   - **Optimization of Service Efficiency**: Achieved through a hybrid system design that combines predictive analytics with real-time monitoring, leading to demonstrable improvements in operational efficiency.

Although the objectives were largely achieved, certain areas that will be discussed later can be improved. Overall, the project aligns well with its initial goals of optimizing Barcelona's bike-sharing system while contributing to sustainability and equitable mobility.

5.3 Future Opportunities

5.3.1 React Google Maps

An opportunity to improve the user interface was identified by considering the use of **React Google Maps** [48] instead of **OpenStreetMap**; however, **React Google Maps** was introduced when the map functionality had already been developed in the app and was still in an initial version. Although **React Google Maps** potentially offered a more refined user interface, opting for a first version of a technology is often not advisable for production-level environments due to **stability** and **support concerns**.

5.3.2 Migration to AIWell Servers

The migration to **AIWell servers** remains incomplete, presenting a clear opportunity for future system improvements. Specifically, this migration could encompass the transfer of the following components to the **AIWell infrastructure**:

1. The **data capture module**, currently managed via Firebase.
2. The **data storage system**, currently located in Firestore.
3. The **data compression process**, which is currently an ad hoc procedure to convert JSON to Parquet.
4. The **analytical API** developed using FastAPI is currently being implemented in an interim environment.
5. The **web application** developed with Next.js, which is currently hosted on Vercel.

Migrating these components to AIWell servers would centralize the system architecture, significantly improving **security** by consolidating data management in a unified security framework. This consolidation would reduce the vulnerabilities associated with disparate systems, improve the integrity of **data**, and streamline access control and monitoring processes. By bringing all components under a cohesive infrastructure, it would also facilitate more robust **data protection protocols**, leading to a more resilient overall system.

5.3.3 Pre-computation of Metrics

To improve the user experience, pre-computing metrics in advance is crucial. This strategy reduces computational load during analytical API operations, as ranking and final aggregation become the primary focus. Shifting intensive tasks such as station metrics or flow calculations to offline processes minimizes delays and increases scalability.

A challenge encountered during the API deployment was its high computational consumption, both in RAM and CPU usage. Precomputing metrics would resolve this issue, enabling the API to serve more users efficiently and dramatically reducing response times compared to the current on-demand calculation approach.

6 Glossary

- **API (Application Programming Interface)**: A set of protocols and definitions that allow different software applications to communicate with each other, specifying how software components should interact to provide seamless integration and data exchange.

- **Bicing**: Barcelona's public bike-sharing system.

- **Docking Station**: A designated location where bicycles are securely parked, picked up, or returned by users. These stations are strategically placed across urban areas to maximize convenience and accessibility for users.

- **FastAPI**: A Python web framework used for building high-performance APIs.

- **GBFS (General Bikeshare Feed Specification)**: A standardized format used to provide real-time data about bike-sharing systems, including station information, bike availability to ensure consistent data-sharing across platforms.

- **GeoPandas**: An open-source Python library designed to facilitate the manipulation and analysis of geospatial data.

- **IoT (Internet of Things)**: A network of interconnected physical devices embedded with sensors, software, and other technologies that allow them to collect, exchange, and act on data over the internet. In the context of bike-sharing, IoT devices help monitor bike status, usage, and location.

- **JSON (JavaScript Object Notation)**: A lightweight data-interchange format that is easy for humans to read and write and simple for machines to parse and generate. It is commonly used to structure data for transmission between a server and a web application, ensuring efficient data handling.

- **KPI (Key Performance Indicator)**: Metrics used to evaluate the success or performance of a specific process or activity. In bike-sharing, KPIs may include metrics like bike availability, station usage rates, and user satisfaction to monitor service efficiency.

- **MDS (Mobility Data Specification)**: A standard data format in JSON, developed to manage and regulate shared mobility services such as bike-sharing and scooters. It provides a framework for cities to manage data collection and ensure smooth integration of shared mobility systems.

- **Open Data**: Data made available to the public without legal or technical restrictions. It is intended to promote transparency, encourage innovation, and facilitate public research, often used in smart city initiatives to improve urban services.

- **Scalability**: The capacity of a system to cope with a growing amount of work or expand in response to increased demand. Scalability is crucial for bike-sharing services to adapt to fluctuating usage patterns and larger user bases.

- **Vercel**: A cloud platform used for deploying and managing web applications. It provides tools to handle server-side rendering and serverless deployment, making web applications fast and easily scalable.

- **Web-Based Application**: Software that is hosted on remote servers and accessed through a web browser over the internet.

- **Weighted Altitude**: The average altitude of bike stations, adjusted by station capacity to better reflect the impact of elevation on logistical operations, such as bike redistribution.

- **Docker**: A tool that uses containerization to package applications and their dependencies into standardized units called containers. This ensures consistency across different computing environments, facilitating easier deployment and scaling.

- **Backend**: The server-side part of a software application that processes data, runs business logic, and communicates with databases. In a bike-sharing system, the backend handles data storage, analytics, and API management.

- **Frontend**: The client-side part of an application that users interact with directly. It includes all visual elements, such as forms, maps, and dashboards in the bike-sharing system's web interface.

- **Authentication**: The process of verifying the identity of a user or device, typically through credentials like usernames and passwords, to ensure that only authorized users can access certain resources or data.

- **Deployment**: The process of making an application available for use. In the context of Bicing, this refers to setting up the web application and backend services on servers to make them accessible to users.

7 Appendices

7.1 BCN Bicing Analytics API

The **BCN Bicing Analytics API** provides analytics and statistics for Barcelona's bike-sharing system. It enables users to query various data endpoints for detailed information about bike station usage, flow metrics, and historical trends. Below, the key features, endpoints, and schemas are outlined in detail.

Features:

- Access station statistics over customizable time periods.
- Analyze bike flow (incoming and outgoing) for individual stations.
- Optimized for efficiency using the Parquet data format.
- Flexible time aggregation options for advanced flow analysis.

Endpoints:

- **API Welcome Endpoint:**
  - **Method:** GET
  - **Path:** `/`
  - **Description:** Returns a welcome message for the API.
  - **Example Response:**

```
{
  "message": "Welcome to BCN Bicing Analytics API!"
}
```

- **Get Data Timeframe:**
  - **Method:** GET
  - **Path:** `/timeframe`
  - **Description:** Provides the available timeframe of the Bicing dataset.
  - **Parameters:** None
  - **Example Response:**

```
{
  "start_date": "2023-01-01",
  "end_date": "2023-12-31"
}
```

- **Get Station Statistics:**
  - **Method:** GET
  - **Path:** `/stats`
  - **Description:** Retrieves statistics for a specific station.
  - **Parameters:**
    - `station_id` (required): ID of the station.
    - `start_date` (optional): Start date for the query.
    - `end_date` (optional): End date for the query.
  - **Example Response:**

```
{
  "station_id": 123,
  "total_bikes": 458,
  "average_flow": 25.6
}
```

- **Get Flow Statistics:**
  - **Method:** GET
  - **Path:** `/flow`
  - **Description:** Provides bike flow metrics such as incoming and outgoing bikes at specified stations.
  - **Parameters:**
    - `station_id` (required): ID of the station.
    - `start_date` (optional): Start date for the query.
    - `end_date` (optional): End date for the query.
    - `aggregation_level` (optional): Aggregation level (e.g., hourly, daily).
  - **Example Response:**

```
{
  "station_id": 123,
  "incoming_flow": 120,
  "outgoing_flow": 95
}
```


Schemas: The API supports the following schemas:

- **FlowOutput:** A string output containing bike flow metrics.
- **HTTPValidationError:** Object detailing validation errors for HTTP requests.
- **ValidationError:** Object describing a specific validation error.

Contact Information:

- **Author:** Andrés Santos Sanz
- **Website:** GitHub Profile

Bibliography

[1] Elliot Fishman and Vaughn Allan. Chapter six - bike share. In Elliot Fishman, editor, *The Sharing Economy and The Relevance for Transport*, volume 4 of *Advances in Transport Policy and Planning*, pages 121–152. Academic Press, 2019. doi: https://doi.org/10.1016/bs.atpp.2019.05.003. URL https://www.sciencedirect.com/science/article/pii/S2543000919300058.

[2] Barcelona City Council. Bicing data - shared bike service statistics, 2024. URL https://www.bicing.barcelona/es/datos-bicing. Accessed in October 2024.

[3] Ajuntament de Barcelona. Bicing system in barcelona, 2024. URL https://www.bicing.barcelona/es/node/205. Accessed: 2024-10-09.

[4] Barcelona City Council. Data portal - municipal statistics, 2024. URL https://portaldades.ajuntament.barcelona.cat/es/estad%C3%ADsticas/ks3r3iy9tj?view=chart&chart=bar-chart. Accessed in October 2024.

[5] MovementBC. Movementbc - bicing solutions, 2024. URL https://www.movementbc.com/soluciones/bicing/. Accessed: 2024-10-10.

[6] United Nations. The 17 goals — sustainable development, 2024. URL https://sdgs.un.org/goals. Accessed: 2024-10-10.

[7] Overleaf. Overleaf, online latex editor, 2024. URL https://overleaf.com/. Accessed: 2024-10-10.

[8] Google. Google sheets - online spreadsheets for collaborative work, 2024. URL https://workspace.google.com/products/sheets/. Accessed: 2024-10-10.

[9] Python Software Foundation. Welcome to python.org, 2024. URL https://python.org/. Accessed: 2024-10-10.

[10] TypeScript. Typescript: Javascript with syntax for types, 2024. URL https://www.typescriptlang.org/. Accessed: 2024-10-10.

[11] FastAPI. Fastapi - modern, fast (high-performance) web framework for building apis with python, 2024. URL https://fastapi.tiangolo.com/. Accessed: 2024-10-10.

[12] Vercel. Next.js by vercel - the react framework, 2024. URL https://nextjs.org/. Accessed: 2024-10-10.

[13] Supabase. Supabase — the open source firebase alternative, 2024. URL https://supabase.com/. Accessed: 2024-10-10.

[14] Vercel. Vercel: Build and deploy the best web experiences with the frontend cloud, 2024. URL https://vercel.com/. Accessed: 2024-10-10.

[15] Andrés Santos Sanz. bcn-bicing-dashboard: A dashboard for barcelona's bicing system. https://github.com/santos-sanz/bcn-bicing-dashboard, 2024. Accessed: 2024-12-19.

[16] Cursor. Cursor - the ai code editor for extraordinary productivity, 2024. URL https://cursor.com/. Accessed: 2024-10-10.

[17] AIWELL Research Group. Ai well - advancing ai research for human well-being, 2024. URL https://aiwell.uoc.edu/. Accessed: 2024-10-10.

[18] David Herlihy. *Bicycle: The History*. Yale University Press, 2004. ISBN 0-300-10418-9. Retrieved 2009-09-29.

[19] Zack Furness. *One Less Car: Bicycling and the Politics of Automobility*. Temple University Press, Philadelphia, 2010. ISBN 978-1-59213-613-1. Archived from the original on 27 May 2010. Retrieved 8 July 2010.

[20] Citi Bike. Citi bike nyc, 2024. URL https://citibikenyc.com/. Accessed: 2024-10-15.

[21] UNEP Copenhagen Climate Centre. Public bike sharing program in hangzhou city, 2024. URL https://c2e2.unepccc.org/kms_object/public-bike-sharing-program-in-hangzhou-city/. Accessed: 2024-10-15.

[22] Mobike. Mobike, 2024. URL https://mobike.com. Accessed: 2024-10-15.

[23] GB Times. What is open data?, 2024. URL https://gbtimes.com/what-is-open-data. Accessed: 2024-11-16.

[24] Open Government Partnership. How is open data changing the world? key findings from open data impact case studies, 2024. URL https://tinyurl.com/2ur9kxjj. Accessed: 2024-11-16.

[25] Kelly Easterday, Tim Paulson, Proxima DasMohapatra, Peter Alagona, Shane Feirer, and Maggi Kelly. From the field to the cloud: A review of three approaches to sharing historical data from field stations using principles from data science. *Frontiers in Environmental Science*, 6, October 2018. doi: 10.3389/fenvs.2018.00088. URL https://www.frontiersin.org/journals/environmental-science/articles/10.3389/fenvs.2018.00088/full. Accessed: 2024-11-16.

[26] Barcelona City Council. Open data bcn, 2024. URL https://opendata-ajuntament.barcelona.cat/en. Accessed: 2024-11-16.

[27] Greater London Authority. London datastore, 2024. URL https://data.london.gov.uk/. Accessed: 2024-11-16.

[28] City of New York. New york city's open data, 2024. URL https://opendata.cityofnewyork.us/. Accessed: 2024-11-16.

[29] Government of Singapore. Singapore's data.gov.sg, 2024. URL https://data.gov.sg/. Accessed: 2024-11-16.

[30] MobilityData Association. General bikeshare feed specification, 2023. URL https://gbfs.org/specification/. Accessed: 2024-10-18.

[31] Open Mobility Foundation. About the mobility data specification (mds), 2023. URL https://www.openmobilityfoundation.org/about-mds/. Accessed: 2024-10-18.

[32] GTFS. Overview of general transit feed specification (gtfs), 2023. URL https://gtfs.org/documentation/overview/. Accessed: 2024-10-18.

[33] Transmodel. Siri – service interface for real time information, 2023. URL https://transmodel-cen.eu/index.php/siri/. Accessed: 2024-10-18.

[34] Damilola Oladimeji, Khushi Gupta, Nuri Alperen Kose, Kubra Gundogan Kose, Linqiang Ge, and Fan Liang. Smart transportation: An overview of technologies and applications. *Sensors*, 23:3880, 04 2023. doi: 10.3390/s23083880.

[35] Oliver O'Brien. Barcelona bike share map, 2024. URL https://bikesharemap.com/barcelona/#/12/2.179/41.3939/. Accessed: 2024-10-18.

[36] Citymapper. Citymapper barcelona, 2024. URL https://citymapper.com/barcelona. Accessed: 2024-10-18.

[37] Paul DeMaio. Barcelona bike sharing world map, 2024. URL https://bikesharingworldmap.com/#/barcelona/. Accessed: 2024-10-18.

[38] Barcelona Public Bike System. Barcelona public bikesystem api, 2024. URL https://barcelona-sp.publicbikesystem.net. Accessed: 2024-11-03.

[39] Google. Firebase — google's mobile and web app development platform, 2024. URL https://firebase.google.com/. Accessed: 2024-10-10.

[40] Ismael Benito Altamirano. Bicidata: Repository. https://github.com/bicidata/bicidata, 2023. URL https://github.com/bicidata/bicidata. Accessed: 2024-10-25.

[41] Andrés Santos Sanz. bcn-bicing-api: An api for accessing barcelona's bicing system data. https://github.com/santos-sanz/bcn-bicing-api, 2024. Accessed: 2024-12-19.

[42] Railway. Railway: Cloud infrastructure simplified, 2024. URL https://railway.app/. Accessed: 2024-12-09.

[43] Amazon Web Services. Amazon s3 - cloud object storage, 2024. URL https://aws.amazon.com/s3. Accessed: 2024-12-09.

[44] CityBik. Citybik api v2, 2024. URL https://api.citybik.es/v2/. Accessed: 2024-11-03.

[45] Apache Software Foundation. Apache parquet, 2024. URL https://parquet.apache.org/. Accessed: 2024-12-01.

[46] City Council of Barcelona. User satisfaction by district, 2023. URL https://portaldades.ajuntament.barcelona.cat/es/estadsticas/ks3r3iy9tj. Accessed: 2024-11-05.

[47] Andrés Santos Sanz. bcn-bicing-tools: Tools for analyzing barcelona's bicing system data. https://github.com/santos-sanz/bcn-bicing-tools, 2024. Accessed: 2024-12-19.

[48] visgl. React google maps. https://visgl.github.io/react-google-maps/, 2024. URL https://visgl.github.io/react-google-maps/. Library for integrating Google Maps with React applications.

List of Figures

List of Tables