Tunisian Republic
Ministry of Higher Education and Scientific Research
University of Carthage
Higher School of Statistics and Information Analysis

Computer Science Department

INTERNSHIP REPORT

Data Analysis and Development of an Interactive Dashboard for OACA

Presented by:
Wissem SAHLI

Supervised by:
Mr Aymen KHADIMALLAH

Period:
July 01, 2024 - July 31, 2024

HOST COMPANY:
Civil Aviation and Airports Office (OACA)

Academic Year: 2024/2025

Acknowledgments

I would like to express my deep gratitude to the Civil Aviation and Airports Office (OACA) for offering me the opportunity to carry out my internship within its services. It was an extremely enriching experience both professionally and personally.

I particularly thank my supervisor Mr. Ayman Khadimallah for his guidance, availability, and the quality of his advice throughout this period. His orientation and expertise greatly contributed to the success of this work.

Finally, I express my gratitude to all those who, directly or indirectly, contributed to the smooth running of this internship and the enrichment of this experience.

Abstract

As part of my internship at OACA, I conducted an in-depth analysis of a database of flights recorded during the year 2024, comprising over 60,000 rows. The main objective of this work was to exploit this data to extract relevant information, produce clear statistical visualizations, and automatically generate reports in PDF format using Python. I also had access to other sources of information, including quarterly billing databases, which allowed me to enrich the analysis and better understand the overall functioning of air traffic and administrative management.

The interest of this work lies in a real need: making data accessible, understandable, and usable, even for people without prior knowledge of statistics or data analysis. The central problem of the internship was therefore: how to allow any user, expert or not, to explore a complex database, clean it, analyze it, and derive reliable and actionable information from it?

To address this problem, the final project consisted of developing an interactive dashboard, similar to Power BI, but fully customizable and accessible via a website. This dashboard allows importing a database, displaying a statistical summary (minimum, maximum, mean, quartiles...), managing missing data and outliers, creating different types of charts according to the nature of the variables, performing linear regression, statistical tests, and a correlation study. The tool also allows automatically generating a complete and downloadable report. To assist the user in interpreting the results, a help menu detailing the main statistical concepts and an assistance chatbot were also integrated.

All the work carried out demonstrates the importance of data science in decision support, while offering an intuitive and accessible solution for data exploration and analysis within OACA.

Contents

Acknowledgments **1**

Abstract **2**

General Introduction **5**

1 Company Presentation **7**
1.1 History and Mission 7
1.1.1 Internal Organization 7

2 Methodology and Data **8**
2.1 Data Description 8
2.1.1 Sources, Period and Volume 8
2.1.2 Key Variables of the Flight Register 8
2.1.3 Data Quality 9
2.2 Methods and Tools Used 9
2.2.1 Programming Environment and Frameworks 9
2.2.2 Data Engineering and Statistics Tools 10
2.2.3 Applied Statistical Methods 10
2.2.4 Project Management 10
Chapter Conclusion 11

3 Data Analysis and Development of Practical Tools: Valorization of the Empirical Contribution **12**
3.1 Statistical Analysis of Tunis-Carthage Air Traffic (2024) 12
3.1.1 Justification and Usefulness of the Analysis Report 12
3.1.2 Brief Data Description and Methodology 13
3.1.3 Detailed Analysis of Strategic Indicators 13
3.2 Analysis of the Billing Database with Power BI 18
3.2.1 Justification for Choosing Power BI 18
3.2.2 Importance of Visualizations in Data Analysis 19
3.2.3 Description of Billing Data 19
3.2.4 Analysis of Created Visualizations 20
3.3 Design and Realization of the Data Science Dashboard 23
3.3.1 Literature Review and Justification of Originality 23
3.3.2 Presentation and Initial User Interface 24
3.3.3 Technical Architecture and Valorization of Development 25

3.3.4 Key Features and Automatic Interpretation 25

General Conclusion **29**

Bibliography **31**

Appendices **31**

A Detail of Delay Calculation **32**

B Source Code of the Descriptive Analysis (Vols.py) **33**

C Project Tree Structure and Architecture **34**

D Technologies Used **36**

General Introduction

In a global context where **data** has become an essential strategic lever for analysis, decision-making, and optimization, the Civil Aviation and Airports Office (**OACA**) faces an ever-increasing volume of information - whether operational, administrative, or statistical. Data related to aircraft movements, daily operations, and billing constitute a precious source of knowledge to improve service quality, optimize internal processes, and **support decision-making**. However, the size, complexity, and heterogeneity of these databases make their direct exploitation difficult for non-specialist users.

It is in this context that this internship takes place. The main objective lies in the **enhancement and optimization of the exploitation** of existing data through the development of simple, automated, and accessible tools. These tools aim to analyze, visualize, and extract relevant information. The central motivation is to **democratize access to statistical analysis** within OACA: offering an environment where any user, even without training in statistics or data science, can import a database, clean it, explore it, and obtain reliable and immediately interpretable results.

Problem Statement

The problem that guided this work can be formulated as follows:

> ***How to design and implement an intuitive and fully automated solution allowing to explore a complex OACA database, perform essential statistical processing (cleaning, descriptive analysis) and generate visualizations and reports usable by all staff, thus promoting the democratization of data-driven management?***

Approach and Achievements

To address this problem, the project was structured into three main phases.

Phase 1: Exploratory Analysis with Python

This phase consisted of an **in-depth statistical analysis** of a 2024 flight database (comprising over **60,000 rows**), as well as the exploration of quarterly billing databases. Using the **Python** language, scripts were developed to automate the generation of charts, the extraction of key descriptive statistics, and the production of professional PDF reports.

Phase 2: Business Intelligence Analysis with Power BI

This intermediate phase focused on the analysis of billing data using the **Microsoft Power BI** tool. The objective was to create interactive dashboards allowing advanced visualization of key financial indicators. This approach facilitated the identification of commercial trends, analysis of revenue distribution by customer and airport, and optimization of collection processes through dynamic and interactive visualizations.

Phase 3: Development of the Interactive Dashboard

The third phase was dedicated to the design and development of an **interactive dashboard** accessible via a web application. This tool offers multiple functionalities: display of a dynamic statistical summary, proactive management of missing values and outliers, creation of custom charts, calculation of models (such as linear regression or correlation study). An essential feature allows the **automatic generation of reports** in PDF format or interactive HTML pages to archive and share the analyses performed. To guarantee total accessibility of the tool, a pedagogical help menu and an **assistance chatbot** were also integrated.

Report Structure

This report is structured as follows:

- **Chapter 1** presents OACA, its organization, and the detailed context in which the internship took place.
- **Chapter 2** describes the raw data used as well as the methodologies and tools (languages, libraries, algorithms) mobilized for their processing.
- **Chapter 3** details the work carried out, from the analysis of the databases to the construction of the interactive dashboard.
- Finally, the **General Conclusion** synthesizes the main results obtained, discusses the limitations encountered, and proposes improvement perspectives for the project.

Chapter 1

Company Presentation

1.1 History and Mission

The Civil Aviation and Airports Office (OACA) is the body responsible for the management and development of airports in Tunisia. Its main mission is to ensure the safety, fluidity, and quality of services offered to passengers and airlines. Since its creation, OACA has played a key role in the development of air transport and the modernization of Tunisian airport infrastructure.

1.1.1 Internal Organization

OACA is structured into several departments, each with specific responsibilities: airport management, air safety, infrastructure maintenance, passenger services, and general administration. Each department is headed by a manager who reports directly to the general management, thus ensuring effective coordination and optimal functioning of the entire organization.

Chapter Conclusion

This chapter has presented OACA, its history, mission, and internal organization. This presentation forms the basis for analyzing the company's operational activities and performance in the following chapters.

Chapter 2

Methodology and Data

This chapter aims to define the analytical foundation of this study by describing the datasets used by the Civil Aviation and Airports Office (OACA) and detailing the technical methodology and software environment mobilized for the development of the analysis solution and the interactive dashboard.

2.1 Data Description

The analysis was conducted on two distinct datasets, characterized by their large volume and heterogeneous nature: operational flight data and financial billing data.

2.1.1 Sources, Period and Volume

Data were extracted from OACA's internal systems.

- **Flight Database:** This is the **OACA Flight Register**, an `Vols.xlsx` file covering the year **2024**.
  - **Volume:** The initial dataset contains about **60,000 rows** (flights) and about twenty columns, representing a volume of 14 MB.
  - **Temporality:** The analysis covers the entire year 2024 and the data are of a **daily/hourly** nature (recording of landing/takeoff movements).
- **Billing Database:** The `Finance.xlsx` file contains financial records for the first quarter of **2024** (January to March).
  - **Volume:** This dataset is larger in rows, totaling about **107,000 rows** (11 MB).

2.1.2 Key Variables of the Flight Register

The Flights database was the focus of the detailed descriptive analysis. The key variables used for the PDF report generation and analysis are:

- `Operator` (Categorical): Name of the airline (e.g., TUNISAIR, NOUVEL AIR).
- `Date` and `H. Landing/ Takeoff` (Date/Time): Crucial information for calculating punctuality indicators.

- `Aircraft Type` (Categorical): Aircraft model used (e.g., A320, B738), essential for fleet analysis.
- `Continent` and `Country` (Categorical): Used for analysis of geographical distribution and main destinations.
- `Pax` (Numeric): Number of passengers, allowing quantification of real traffic per airline.

2.1.3 Data Quality

The billing database is used to link financial amounts to services. Relevant variables include `montant_ttc` and `nature_prestation` (landing charges, parking fees).

The main challenge of the analysis lay in **data quality**, notably:

- **Missing Values:** Presence of missing values in critical columns (e.g., arrival/departure times), requiring imputation techniques.
- **Format Inconsistencies:** Problems with date/time formatting and character strings (partially resolved by cleaning functions as in the `Vols.py` script).
- **Risk of Bias (Potential Fraud):** The analysis of delay data (variable `H. Landing/ Takeoff` vs. `Block Time`) may be subject to bias. Schedule manipulations are possible to artificially minimize the calculated delay of certain airlines (e.g., modification of the actual takeoff time to improve the punctuality indicator of airlines like Tunisair), requiring increased vigilance in interpretation.

2.2 Methods and Tools Used

The project revolves around a modern and specialized technology stack in data processing and the development of interactive web applications.

2.2.1 Programming Environment and Frameworks

The project relies on a Full Stack architecture integrating Python for the back-end and web technologies for the user interface.

- **Back-end / Processing: Python** is the central language. The API for the assistance chatbot is based on the integration of the **Google Gemini** model.
- **Web Development:** The interactive user interface (dashboard) is developed in **HTML**, **CSS**, **JavaScript** and the **React** framework, ensuring a dynamic and professional user experience.
- **Web Service:** The **Flask** framework (version 3.0.0) was used for back-end request routing (API, data management) between the React application and Python scripts.

2.2.2 Data Engineering and Statistics Tools

The robustness of the analysis is guaranteed by high-level scientific libraries:

Table 2.1: Libraries and Software Used

Domain	**Tools**	**Main Role**
Manipulation	`pandas`, `numpy`	Cleaning, aggregation, time series calculations.
Visualization	`matplotlib`, `seaborn`	Generation of static charts for the PDF report.
Statistics/ML	`scikit-learn`, `statsmodels`, `scipy`	Regression models, outlier detection, statistical tests.
PDF Reporting	`reportlab`, `fpdf`	Automated generation of professional summaries.
Chatbot	`google-generativeai`	Connection to the Gemini API for contextual assistance.
Business Intelligence	**Microsoft Power BI**	Creation of interactive dashboards and advanced visualizations.

2.2.3 Applied Statistical Methods

The applied methods aim to transform raw data into actionable performance indicators:

- **Advanced Descriptive Analysis:** Calculation of punctuality indicators (**average delay**) based on the difference between block times and actual times, as illustrated in the `Vols.py` script (calculation of 'New delay (min)' for flight type 'J').
- **Correlation Analysis:** Use of the **Pearson correlation coefficient** to quantify linear relationships between variables (e.g., passengers and aircraft type) and **Linear Regression** (`scikit-learn`) to model the potential impact of operational factors on passenger volume or delays.
- **Outlier Detection:** Application of the **Interquartile Range (IQR)** method to isolate extreme values of delays or passenger volumes, improving the reliability of descriptive statistics.

2.2.4 Project Management

Work coordination, essential in a complex technical project, was ensured by the use of **Git** and **GitHub**. These tools allowed rigorous version control and effective collaboration with the supervisor throughout the development cycle.

Chapter Conclusion

This chapter has established the study framework by describing the nature, complexity, and volume of the **60,000 flights** analyzed, as well as the presence of crucial challenges in terms of **data quality** and potential biases. It has also exposed the technical methodology based on the **Python/Flask/React** environment and the use of specialized libraries (`Pandas`, `scikit-learn`, `ReportLab`). This solid foundation, combining complex real data and modern Data Science tools, sets the stage for the detail of concrete achievements (analysis, dashboard creation, and Gemini API integration) that will be presented in the next chapter.

Chapter 3

Data Analysis and Development of Practical Tools: Valorization of the Empirical Contribution

Chapter Introduction

This chapter constitutes the **empirical and applicative part** of our work, following the theoretical foundations and literature review presented in the previous chapters. The objective is to materialize the acquired knowledge by applying it to real OACA datasets and developing a decision support tool. This chapter is structured into three main parts, each representing a major component of our internship project:

1. An in-depth analysis of the air traffic of Tunis-Carthage airport performed under **Python**, intended to produce strategic insights for airport management.
2. An analysis of the billing database via **PowerBI**, aiming to optimize financial processes.
3. The design and realization of a complete **Data Science Web Platform (Dashboard)**, a versatile tool for data exploration and analysis.

We will ensure to valorize all the efforts deployed, from data collection to the final interpretation of results.

3.1 Statistical Analysis of Tunis-Carthage Air Traffic (2024)

3.1.1 Justification and Usefulness of the Analysis Report

This report presents a descriptive analysis of flights operated at Tunis-Carthage airport. It covers airlines, aircraft types, delays, passengers, and destinations.

The application of data analysis methods to this flight database is essential for the Civil Aviation and Airports Office (OACA). The main usefulness of this report is to:

- **Guide Commercial Strategy:** Identify primary markets (continents, countries) for the future development of air routes.

- **Optimize Logistics and Infrastructure:** Adapt ground equipment and stand management according to the most frequent aircraft types and congestion areas.
- **Evaluate Performance:** Measure carrier punctuality, a fundamental criterion for the airport's reputation and user satisfaction.

The data comes from a detailed extract of international passenger flights.

3.1.2 Brief Data Description and Methodology

Source and Data Preparation

The analyzed database comes from a flight extract for the year **2024**. Processing and visualization production were performed in **Python** (Pandas, Matplotlib, Seaborn libraries).

Key Variables Used

Indicators are built around the variables: *Continent*, *Aircraft Type*, *Operator*, and temporal data allowing the calculation of *Delays* and *Monthly Activity*.

3.1.3 Detailed Analysis of Strategic Indicators

We focus on the four most strategic indicators, including for each an explanation of the code, the visualization, and the interpretation of the results.

Context of the Exploratory Analysis

The complete exploratory analysis of the database allowed the generation of **10 charts** covering all operational and commercial aspects of air traffic. These visualizations include:

1. **Geographical distribution of flights by continent**
2. Distribution of airlines
3. **Top 15 aircraft types used**
4. Monthly traffic distribution
5. Total passenger volume transported by the top 15 airlines
6. **Carrier Punctuality Performance (Delays)**
7. Market comparison between Tunisair and Nouvelair
8. Flight typology (excluding regular 'J' flights)
9. **Stand Occupancy (P50-P59)**
10. Main international destinations

For the purposes of this report and to provide an in-depth analysis of the most crucial elements for **OACA**, we will **focus the detailed study** on the four strategic indicators highlighted in bold above (1, 3, 6 and 9).

1. Geographical Distribution of Flights by Continent

This pie chart (Figure 3.1) highlights the geographical origin of flights, an essential indicator for the airport's commercial strategy.

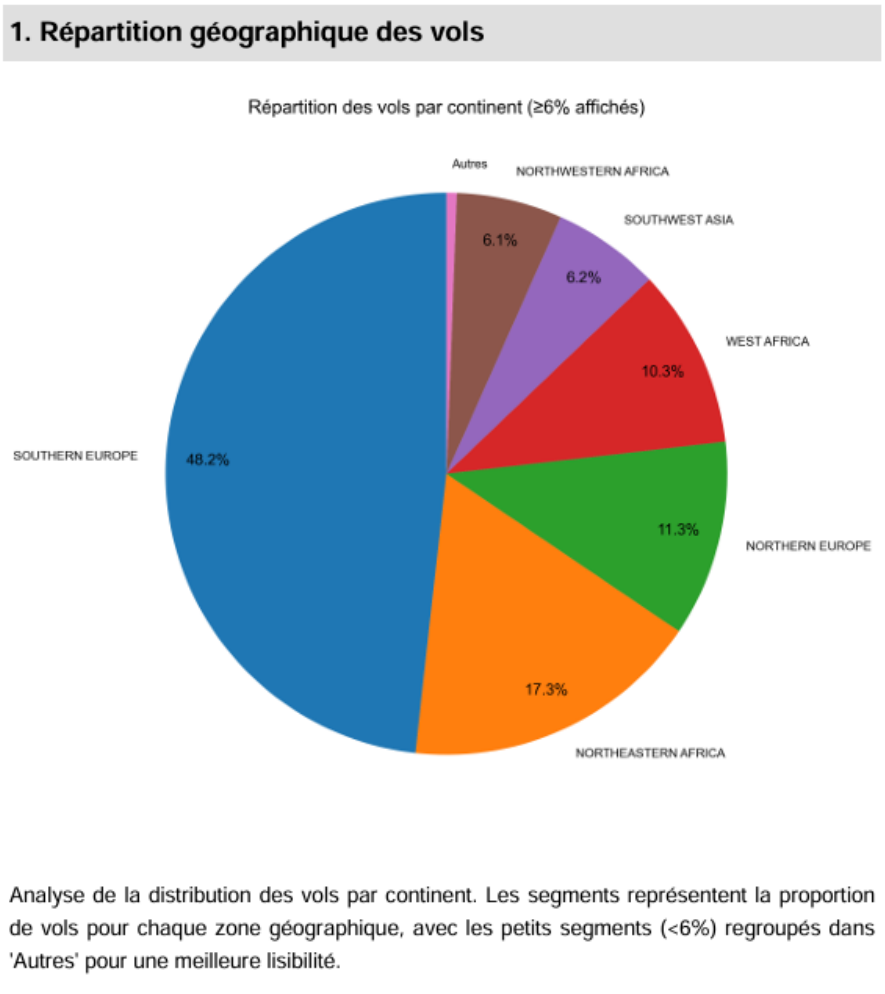


Figure 3.1: Flight distribution by continent ($\geq 6\%$ displayed)

Source: OACA Flight database, Processing: Python/Pandas/Matplotlib.

Python Code (Small Segments Treatment): The Python code was implemented to group small segments (proportion $< 6\%$) into a single 'Others' category to improve readability.

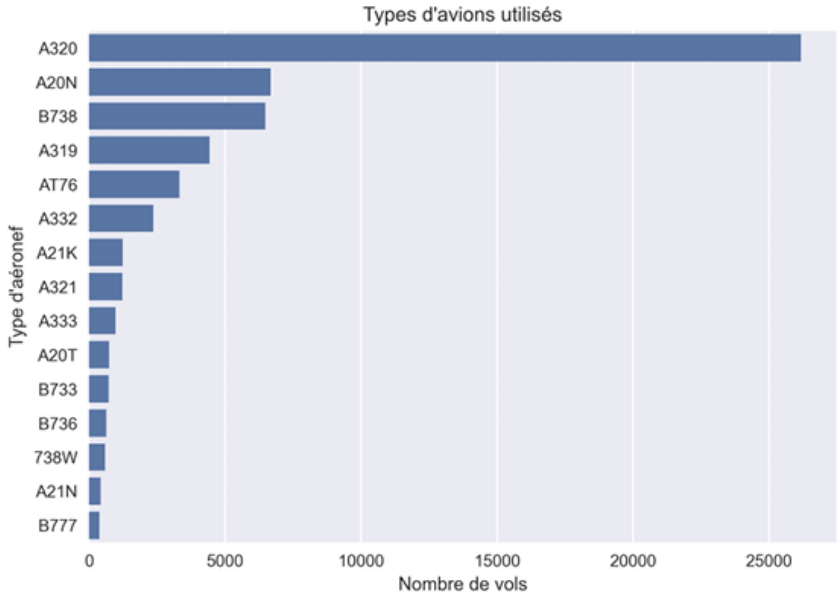
```python
continent_counts = df['Continent'].value_counts()
threshold = 0.06 * continent_counts.sum()
small_values = continent_counts[continent_counts < threshold]

if not small_values.empty:
    # Aggregation of small continents into 'Others'
    main_values['Autres'] = small_values.sum()
```

Listing 3.1: Treatment of small segments for the pie chart

Interpretation of Result: Observation of Figure 3.1 reveals the **overwhelming dominance** of Southern Europe with **48.2%** of total traffic. Next is Northeastern Africa with **17.3%**. This strong concentration confirms the tourist vocation and historical links. The interest for OACA is to primarily direct commercial development efforts towards these areas.

2. Top 15 Aircraft Types Used at Tunis-Carthage

The study of the fleet (Figure 3.2) is a key indicator for technical planning and airport equipment management.



Classement des 15 types d'avions les plus utilisés. Ce graphique horizontal montre la diversité de la flotte et la fréquence d'utilisation de chaque modèle.

Figure 3.2: Aircraft types used

Source: OACA Flight database, Processing: Python/Pandas/Seaborn.

Python Code (Extraction of Top 15): The Python code was implemented to extract and visualize the 15 most used aircraft types at Tunis-Carthage airport.

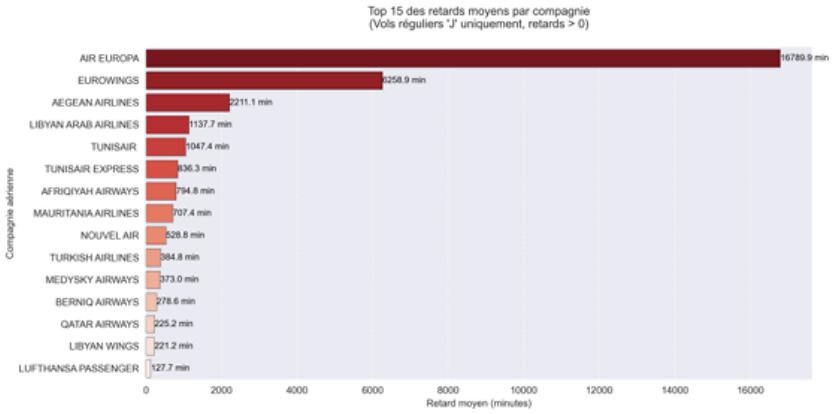
```
top_aircrafts = df["Type d'aeronef"].value_counts().head(15)
sns.barplot(y=top_aircrafts.index, x=top_aircrafts.values, ax=ax)
```

Listing 3.2: Extraction of the most used aircraft types

Interpretation of Result: The chart shows that the **Airbus A320** is by far the most frequent aircraft, followed by the modern models **A20N** and the **B738** (Boeing 737-800 family). The predominance of these single-aisle aircraft is **crucial for technical planning**: it justifies the adaptation of runway lengths and the availability of jet bridges and maintenance equipment specific to these models.

3. Carrier Punctuality Performance (Delays)

This indicator (Figure 3.3) evaluates the operational efficiency of airlines by focusing only on regular flights (type 'J') and positive delays (delays > 0 minutes).



Analyse des retards moyens pour les vols réguliers (type J) uniquement. Seuls les retards positifs sont pris en compte. Les compagnies sont classées des moins performantes aux plus performantes.

Figure 3.3: Top 15 average delays by airline (Regular 'J' flights only, delays > 0)

Source: OACA Flight database, Calculations and visualization Python.

Python Code (Delay Calculation): The Python code was implemented to calculate the punctuality indicators of airlines on regular flights.

```
# Calculation of delay in minutes
```

```
df_j['Nouveau retard (min)'] = (
    df_j['Heure bloc'] - df_j['Date']
).dt.total_seconds() / 60
```

Listing 3.3: Calculation of punctuality indicators

Interpretation of Result: The ranking reveals significant disparities. The airlines **AIR EUROPA** and **EUROWINGS** show the highest average delays (approximately 16,790 min and 6,259 min respectively). Our national airline, **TUNISAIR**, is also in the upper part of the ranking. **Consequence:** These results require immediate action to identify the causes (slots, maintenance) in order to maintain service quality.

4. Stand Occupancy (P50-P59)

This analysis (Figure 3.4) is purely operational and aims to optimize ground resources in a specific area of the airport.

Fréquence d'utilisation des postes P50 à P59. Ce graphique permet d'identifier les postes les plus sollicités sur cette zone de l'aéroport.

Figure 3.4: Frequency of use of stands (P50 to P59)

Source: OACA Flight database, Calculations and visualization Python.

Python Code (Filtering and Counting): The Python code was implemented to analyze the occupancy of stands P50 to P59.

```
postes = [f'P{i}' for i in range(50, 60)]
vols_par_poste = df[
```

```
    df['Poste de stat.'].isin(postes)
][ 'Poste de stat.'].value_counts()
```

Listing 3.4: Analysis of stand occupancy

Interpretation of Result: Figure 3.4 indicates that stands **P50 (4,474 flights)**, **P51 (4,404 flights)**, and **P56 (4,433 flights)** are the most used, while **P52** and **P53** show relative underuse (approximately 2,300 flights each).
Recommended Action: This disparity suggests that an **optimization of ground resource management** is necessary to avoid saturation of certain stands and improve operational fluidity.

Conclusion of Part I

The air traffic analysis allowed transforming raw flight data into actionable **strategic insights**. We confirmed the major importance of the European market (48.2% of flights) and the imperative to monitor the punctuality of certain airlines. The effort undertaken in Python demonstrates the ability to quickly produce detailed and reliable reports, serving as a solid basis for operational recommendations.

3.2 Analysis of the Billing Database with Power BI

Introduction of Part II

This second part of the empirical study is dedicated to the analysis of OACA's billing database using the **Microsoft Power BI** tool. The main objective is to transform raw financial data into interactive visual indicators allowing optimization of the collection process and identification of main revenue sources.

3.2.1 Justification for Choosing Power BI

The choice of Power BI was imposed for several fundamental reasons:

- **Ease of creating interactive visualizations:** The intuitive interface allows quickly generating complex charts without deep technical skills
- **Better visualization and interpretation:** Advanced graphical capabilities offer a clear and immediately understandable representation of data
- **Interactivity:** The possibility to filter and explore data in real-time facilitates detailed analysis
- **Optimal integration with data sources:** Native connection with the Excel files used by OACA

3.2.2 Importance of Visualizations in Data Analysis

The use of curves and charts is fundamental in data analysis to transform raw information into actionable knowledge. The human brain processes visual information more effectively, allowing:

- A **quick understanding** of general trends without dissecting complex data tables
- **Identification of patterns and trends** revealing relationships between variables and temporal evolutions
- **Detection of anomalies** highlighting outliers or data errors
- An **efficient comparison** between different categories and segments
- **Convincing communication** of results to non-specialist decision-makers

3.2.3 Description of Billing Data

The analyzed database contains financial records for the first quarter of 2024, with the following key variables:

- **Identifiers:** `id`, `n_fact`, `codeclient`
- **Financial amounts:** `montant`, `taxes_amount`, `total_amount_before_tax`
- **Temporal information:** `d_fact`, `arrivees_date`, `departs_date`
- **Descriptions:** `descr`, `service`, `airport_name`
- **Operational information:** `arrivees_nvol`, `departs_nvol`, `airplane_type`

3.2.4 Analysis of Created Visualizations

Amount Distribution by Client

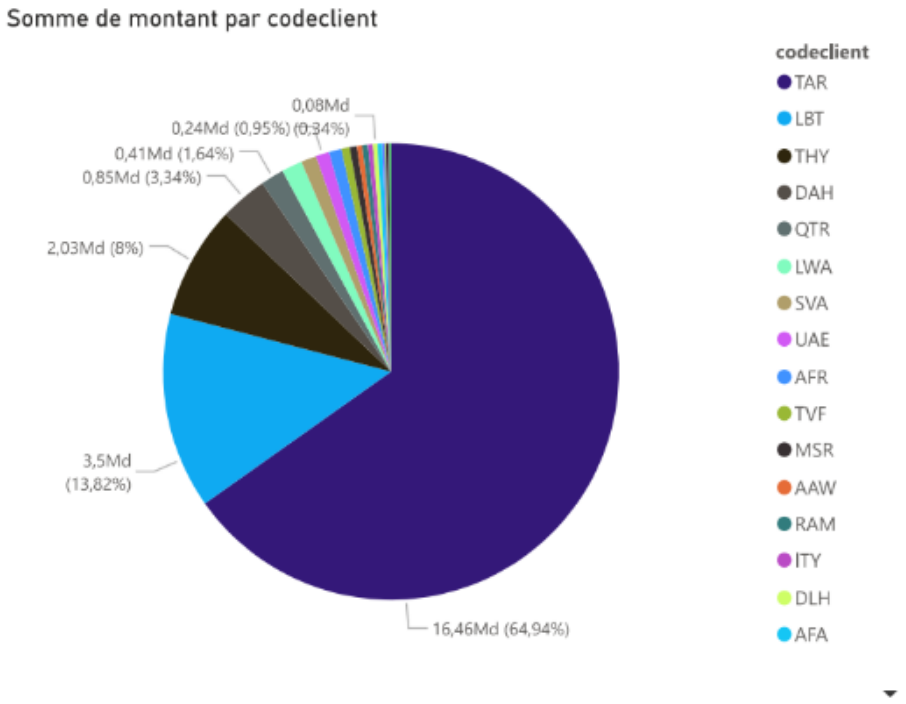


Figure 3.5: Distribution of the sum of amounts by client code

Interpretation: The pie chart reveals a **significant concentration** of revenue on a limited number of clients. The client **TAR** alone represents **64.94%** of the total amount (16.46 Bn), demonstrating strong economic dependence. The clients **LBT** (13.82%) and **THY** (8%) complete the major trio, these three clients generating more than **85%** of the total turnover.

Strategic implications:

- Need to diversify the client portfolio to reduce dependence
- Development of loyalty strategies for major clients
- Identification of growth opportunities among minority clients

Tax Distribution by Airport

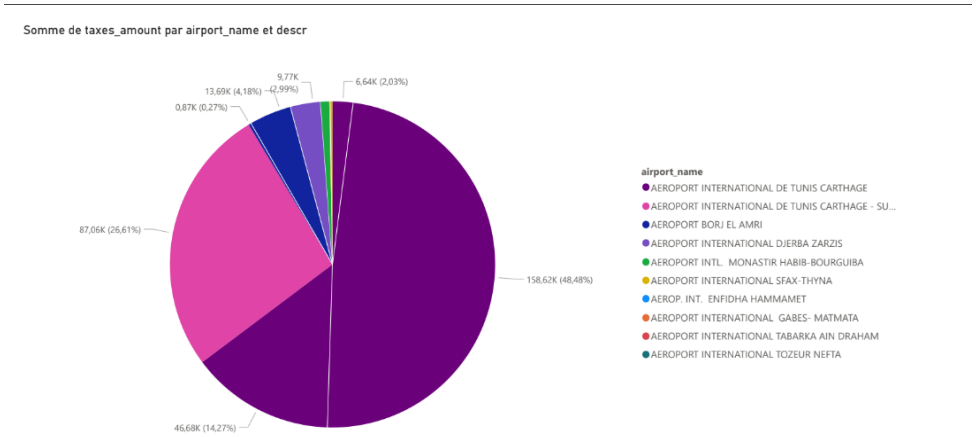


Figure 3.6: Distribution of the sum of taxes by airport

Interpretation: The tax analysis demonstrates a marked **geographical concentration**. Tunis Carthage International Airport generates **48.48%** of collected taxes (158.62K), while overflight of the same airport represents **26.61%** (87.06K). These two items thus cumulate **75%** of total taxes.

Operational implications:

- Optimization of resources on sites generating the most fiscal revenue
- Analysis of performance gaps between different airports
- Evaluation of the effectiveness of pricing policies by site

Detailed Tax Analysis by Airport and Service

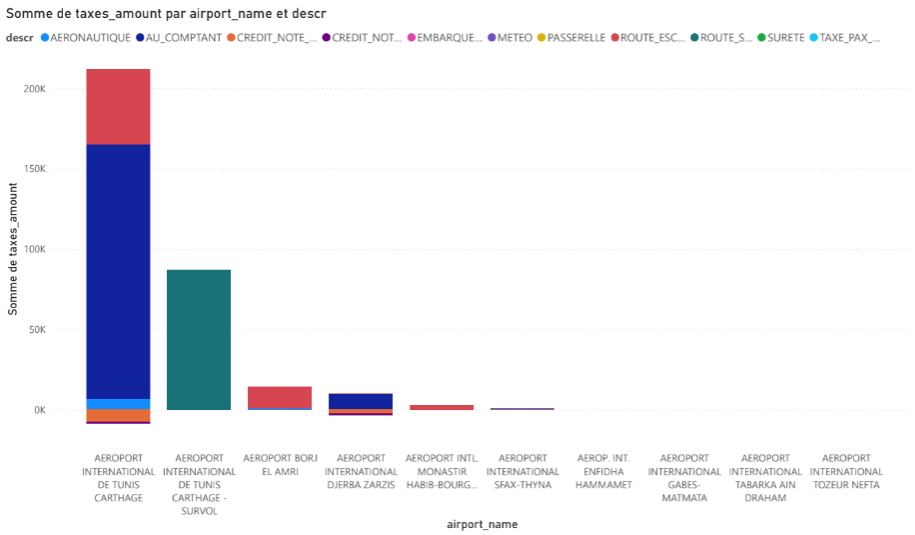


Figure 3.7: Detailed distribution of taxes by airport and service type

Interpretation: The stacked histogram allows a granular analysis of the tax composition. We observe that:

- Tunis Carthage International Airport has the highest tax burden (bar exceeding 200K)
- The service **AU_COMPTANT** (represented in dark blue) constitutes the major component for all airports
- Significant variations exist in the service composition according to airports
- Some airports present specific tax profiles requiring in-depth analysis

Managerial recommendations:

- Prioritize optimization actions on the **AU_COMPTANT** service which impacts the total taxes the most
- Analyze the causes of composition differences between airports
- Develop differentiated pricing strategies by site and by service

Power BI Part Conclusion

The Power BI analysis demonstrated its added value in transforming complex financial data into actionable strategic insights. The created visualizations allowed identifying:

- A **risky client concentration** requiring portfolio diversification
- An **uneven distribution of fiscal revenue** between different airports
- **Distinct service profiles** according to airport sites

These results provide a solid basis for the development of better-informed commercial and operational strategies, demonstrating the crucial importance of Business Intelligence tools in modern decision-making.

3.3 Design and Realization of the Data Science Dashboard

Introduction of Part III

The third part of our empirical work is the design and development of a complete web application, the **Data Science Dashboard**. It is an all-in-one data analysis platform, ranging from file import to predictive modeling and report generation. This development represents the **maximum valorization of our technical contribution** and demonstrates our ability to transform an analysis need into a robust software solution.

3.3.1 Literature Review and Justification of Originality

Positioning Relative to Existing Solutions

Many data analysis tools exist (*Jupyter Notebook*, *RStudio*, *PowerBI*, *Tableau*). However, our **originality** lies in the integration of the entire Data Science workflow (**Upload, Cleaning, Statistics, Visualization, Regression, Tests**) within a **unique and modern web interface (React)**, without requiring programming skills. We offer a simplified alternative to complex solutions.

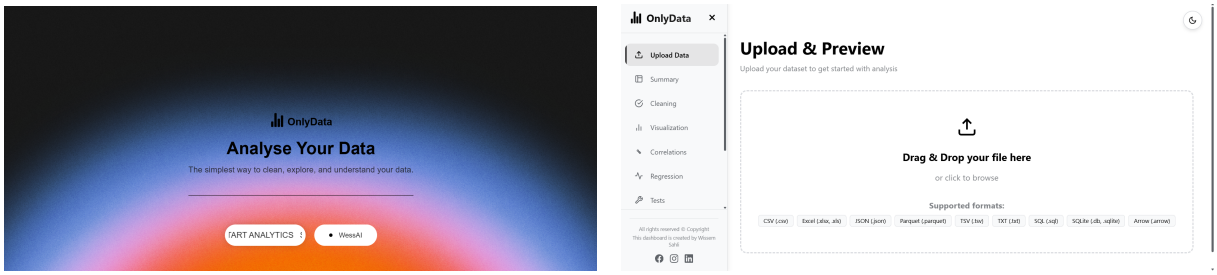
Justification of the Feasibility Hypothesis

We had emitted the **hypothesis** that the integration of a **Generative AI Chatbot** (Gemini API) could revolutionize data exploration by providing contextual interpretations and recommendations in real-time. This functionality, which is at the heart of our dashboard's innovation, was justified by the rapid evolution of conversational artificial intelligence and its potential to democratize access to data *insights*.

3.3.2 Presentation and Initial User Interface

Home Screen and Navigation

The application opens on a home screen designed for accessibility, providing a clear starting point and minimal instructions. This screen serves both as an introduction to the project and as a data upload area.



(a) Home Page (b) Upload Page

Figure 3.8: Home Screen Interface and Data Upload Module.

Source: Screenshots of the Dashboard interface.

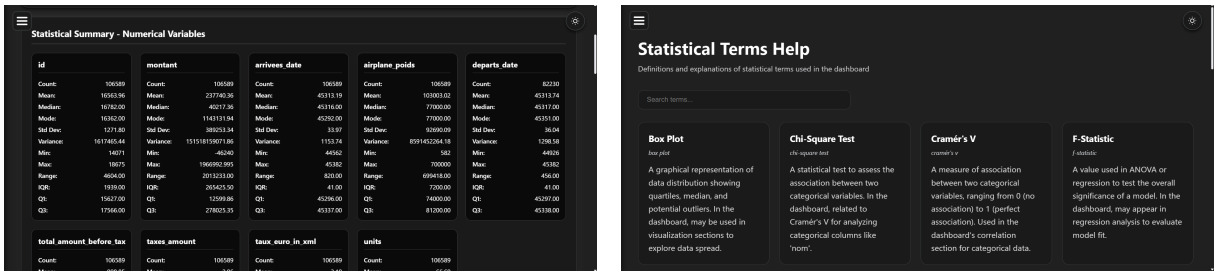
As shown in Figure 3.8a and 3.8b, the home page directly integrates the flight file upload module (CSV or Excel). This simplified approach emphasizes the initial operation.

- **Standard Navigation:** A sidebar allows access to different modules (Summary, Cleaning, Regression, Chatbot...).
- **Imposed Workflow:** The user is forced to load a dataset before being able to activate the analysis modules, guaranteeing a logical workflow.
- **Ease of Use:** The interface is clean (thanks to React/Vite), avoiding the visual complexity of BI tools, which is crucial for users unaccustomed to *Data Science* environments.

Overview of Key Views (Summary and Help)

The user experience is organized around thematic views, each with a distinct role.

- **Summary View (Figure 3.9a):** It provides a first level of understanding of the dataset. It is here that the user accesses descriptive metrics (Mean, Standard Deviation, Max, Min, Missing Values...) and the **automatic interpretation layer**.
- **Help View (Figure 3.9b):** This section ensures user autonomy. It offers succinct documentation on data prerequisites, the meaning of statistical terms, and the best way to use the different modules.



(a) Summary View (b) Help View

Figure 3.9: Interfaces of the Statistical Summary and User Help modules.

Source: Screenshots of the Dashboard interface.

3.3.3 Technical Architecture and Valorization of Development

Technologies and Software Used

The Data Science Dashboard follows a modern **Front-end/Back-end** architecture.

- **Front-end (*Software*):** Developed with **React (18.2.0)** and **Vite**, guaranteeing a reactive user interface (UI) and a smooth experience.
- **Back-end (*Software*):** Built on the micro-framework **Flask (3.0.0)** in **Python**, serving as an API gateway for intensive data processing.
- **Data Processing (*Software*):** The heart of the application relies on the power of **Pandas (2.1.3)** for manipulation, **NumPy** for numerical calculation, **SciPy** and **Statsmodels** for advanced statistical tests, and **Scikit-Learn** for modeling (**Valorization of technical effort**).

Data Flow and Functional Robustness

The work consisted of designing an architecture where the Front-end communicates via **Axios** with the Back-end (REST API). The **Data Flow** is well defined: the upload loads the dataset into memory; the pages (Summary, Cleaning, Regression) call Flask endpoints that use **Pandas** to process the dataset and return results in JSON format or base64-encoded charts.

3.3.4 Key Features and Automatic Interpretation

Data Cleaning and Descriptive Statistics

The **Cleaning** module offers advanced strategies (DROP, FILL_MEAN/MEDIAN/MODE, Interpolation) for missing values and outlier detection (**Outliers**) by IQR or Z-Score methods. The **Statistical Summary** module provides all required metrics (Mean, Standard Deviation, Max, Missing Values, IQR...) for numerical and categorical variables, with a textual interpretation layer valorizing accessibility.

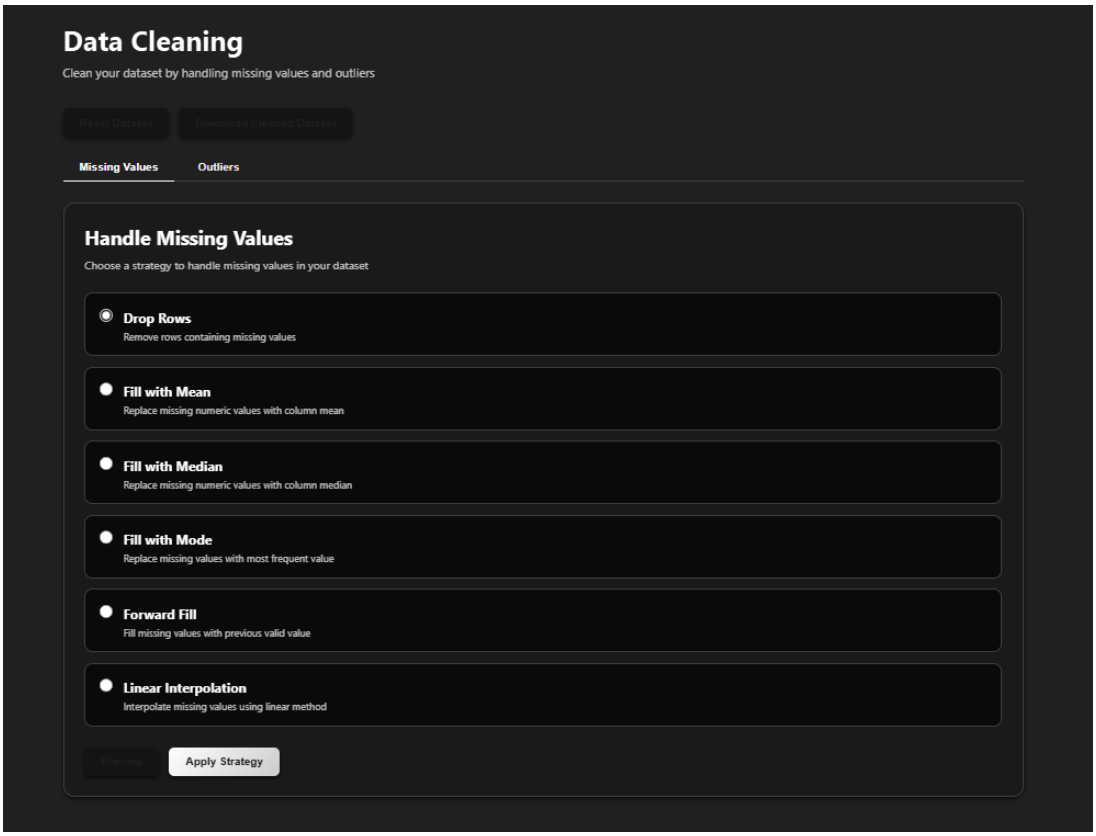
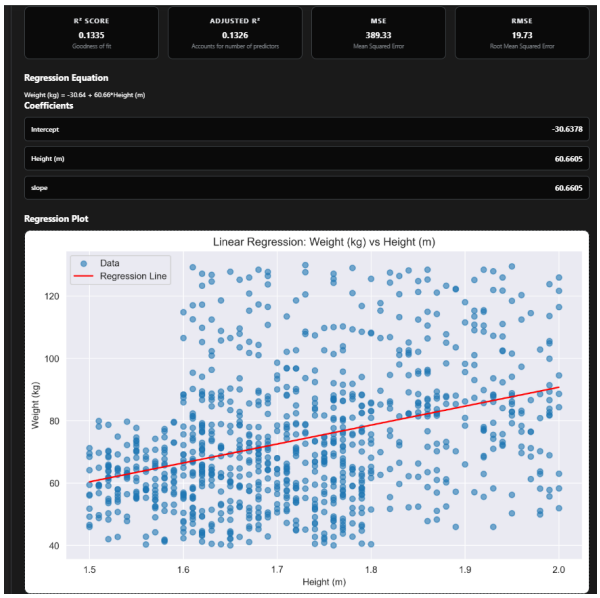


Figure 3.10: Overview of the Data Cleaning module.

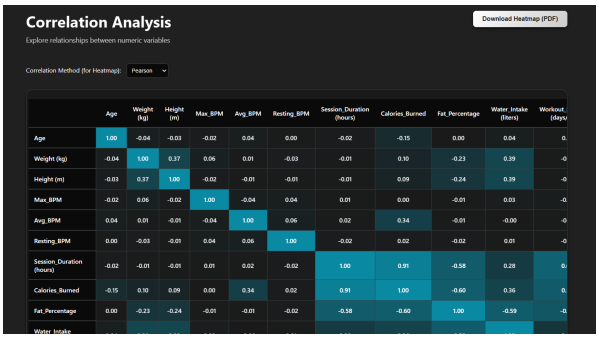
Source: Dashboard user interface.

Modeling and Interpretation of Results

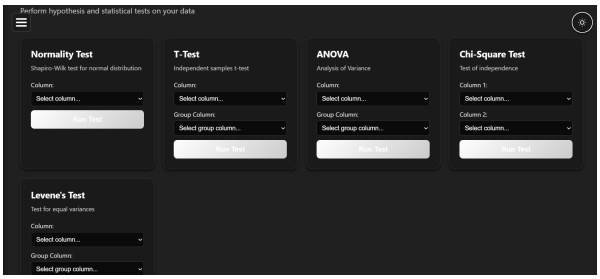
The platform implements a **Multiple Linear Regression** module (via Scikit-Learn). Results are displayed with the complete regression equation, metrics (**Adjusted R^2**, MSE, RMSE) and an **automatic interpretation** of the model's performance. Similarly, **Statistical Tests** (**Shapiro-Wilk, T-Test, ANOVA, Chi-Square**) display the p-value and the conclusion (Rejection/Non-rejection of H_0) in clear language for the user. Furthermore, the part dedicated to correlation analysis allows visualizing the relationships between variables using a Heatmap.



(a) Linear Regression



(b) Correlation with a Heatmap



(c) Valid statistical tests

Figure 3.11: Interfaces of the Correlation, Regression and Tests modules.

Source: Screenshots of the Dashboard interface.

Innovation: The Integrated AI Chatbot (Gemini API)

The integration of the **AI Chatbot** (based on **Gemini API**) represents a significant contribution (**Valorization**). This module allows the user to ask questions in natural language about the loaded dataset or the analysis results, and to obtain **insights and recommendations**. For example, after running a T-Test, the AI can provide a didactic explanation of the meaning of the obtained p-value, transforming a raw software output into immediately exploitable information.

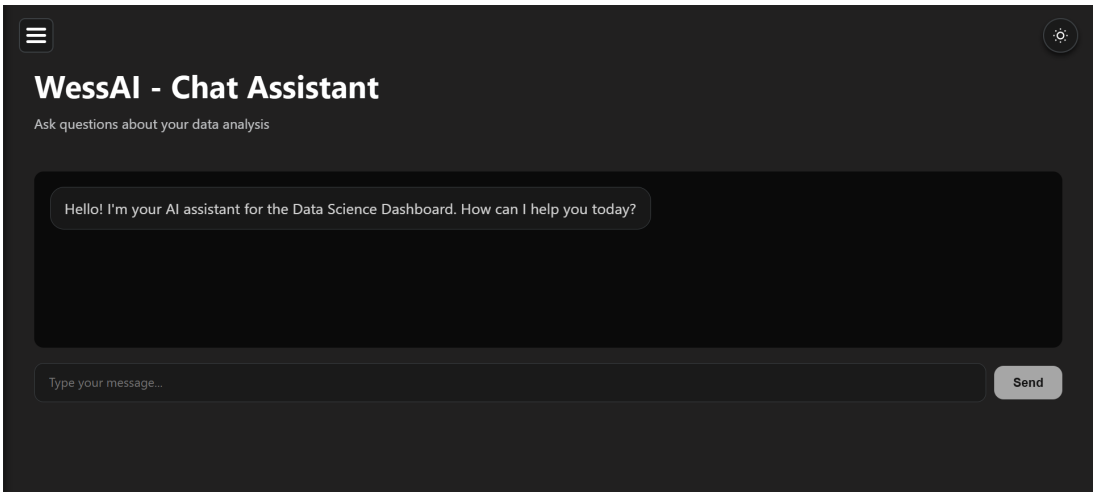


Figure 3.12: AI Chatbot: Interface of the conversational assistance for data analysis.

Source: Dashboard user interface.

Conclusion of Part III: Synthesis of Technical Development

The development of the Data Science Dashboard is the concrete demonstration of the application of **robust Data Science methodologies** to a sensitive operational environment. By relying on the efficiency of the **Front-end React / Back-end Flask** architecture and valorizing the potential of Python libraries (Pandas, Scikit-Learn), we have created a tool that goes beyond simple statistical reporting. The integration of artificial intelligence via the Gemini API is the culmination of this innovation, positioning the Dashboard as a **strategic decision support tool** capable of analyzing, modeling, and interpreting air traffic data in real-time. This project paves the way for a more proactive and data-driven management of Tunis-Carthage airport.

Chapter Conclusion

This chapter allowed moving from theory to practice by addressing three essential aspects of data engineering and decision analysis. The analysis of air traffic under Python provided actionable insights for the airport, targeting resource optimization and service quality improvement. The Data Science Dashboard constitutes the culmination of our development effort, offering a flexible and powerful tool that centralizes the data analysis lifecycle. Together, these works **directly respond to the objective** of applying statistical methods to business problems, while developing an original solution. The next chapter (e.g., **Chapter 4: Synthesis and Recommendations**) will proceed to synthesize the obtained results, draw the general conclusions of the project, and propose future perspectives based on the developed data and tools.

General Conclusion

My internship, conducted at the Civil Aviation and Airports Office (**OACA**), was part of a context of growing need to **enhance internal data** to optimize operational and strategic decision-making. The major interest of the subject lay in the **democratization of statistical analysis**, aiming to make exploration and report production accessible to all staff, without requiring deep expertise in Data Science.

Synthesis of Obtained Results

To address the problem, which was to design an intuitive and automated solution for exploring complex data, work in two parts was accomplished:

- **Descriptive Analysis and Automated Reporting:** An in-depth analysis of the **60,000 flights** of the year 2024 was carried out, allowing the establishment of key indicators on traffic distribution by continent and airline, aircraft typology, and passenger volumes. The development of a script in Python (`Vols.py`, visible in Appendix B) allowed the **automated generation of a PDF report** (cf. Chapter 3), concretely illustrating the feasibility of rapid and standardized reporting.
- **Development of an Interactive Dashboard (Full Stack):** The culmination of the project is the design and implementation of a dynamic web application. This dashboard, built on a **Python/Flask/React** architecture, offers essential functionalities: data cleaning (outlier management via IQR), application of statistical models (linear regression, correlation) and interactive visualization.
- **Integration of AI for Assistance:** The integration of the **Google Gemini API** for the assistance chatbot represents a major advancement, offering users direct contextual and pedagogical help on the interpretation of statistical results.

These achievements allowed transforming raw and complex datasets into an intuitive management tool, thus achieving the objective of democratizing access to analytical information within OACA.

Limits and Difficulties Encountered

Despite the success of the achievements, several limits were identified:

- **Quality and Reliability of Delay Data:** The main obstacle was the management of **potential biases** in punctuality data. As mentioned in Chapter 2, the possibility of adjusting flight times to minimize declared delays introduced a level of uncertainty

in the performance analysis, requiring the application of robust statistical methods like outlier detection.

- **Complexity of the Full Stack Infrastructure:** The need to integrate **Python** (back-end), **Flask**, **React** (front-end) and the Gemini API within a limited internship time represented a significant technical challenge in terms of coordination and deployment.
- **Statistical Scope:** The lack of longer historical data (limited to 2024) restricted the possibility of using **time series** models or performing robust forecasts.

Improvement and Complementarity Perspectives

This work opens the way to numerous evolutions:

- **Predictive Modeling:** Adding a functionality for **short-term traffic forecasting** (number of flights and passengers) by integrating Machine Learning models (e.g., ARIMA, Prophet) would require access to a more extensive historical database.
- **Operational Optimization:** Develop advanced indicators for resource allocation (e.g., correlation between aircraft type and optimal use of the stand, based on data from Figure 9 of the report) to optimize ground operations.
- **Chatbot Improvement:** Make the chatbot capable of generating specific queries on the data (e.g., "What is the average delay of Tunisair to France?") directly via the API, thus transforming the assistance tool into a true **natural language interrogation tool**.
- **Production Deployment:** Finalize the integration and securing of the dashboard for a real and continuous deployment on OACA's infrastructure.

This project demonstrated that an intelligent exploitation of existing data can radically transform operational efficiency, and it is essential that OACA continues this dynamic to consolidate its strategy based on factual analysis.

Appendices

Appendix A

Detail of Delay Calculation

This appendix details the formula used for calculating average delays, for the purpose of anomaly detection and management of 'J' type flights.

The delay variable R (in minutes) is calculated as follows for positive delays:

$$R = \max\left(0, \frac{\text{Block Time} - \text{Actual Date and Time of arrival/departure}}{\text{60 seconds}}\right)$$

The use of $\max(0, \dots)$ ensures that only real delays are counted, excluding early flights for the performance indicator.

Appendix B

Source Code of the Descriptive Analysis (Vols.py)

Below is an excerpt of the Python code (`Vols.py`) used for loading data, initial cleaning, and generating static charts for the PDF report.

```
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from fpdf import FPDF
import os
import matplotlib.ticker as ticker

# Data loading and column cleaning
df = pd.read_excel("Vols.xlsx", header=8)
df.columns = df.columns.str.strip().str.replace('\n', ' ')

# Date conversion
date_cols = ['Date', 'H. Landing/ Takeoff', 'Block Time']
for col in date_cols:
    df[col] = pd.to_datetime(df[col], dayfirst=True, errors='coerce')

# Delay calculation for punctuality analysis (Figure 6)
df_j = df.loc[df['Flight Nature'] == 'J'].copy()
df_j.loc[:, 'New delay (min)'] = (
    df_j['Block Time'] - df_j['Date']
).dt.total_seconds() / 60
# ... Continuation of analysis and chart generation
```

Listing B.1: Main code for OACA flight analysis

Appendix C

Project Tree Structure and Architecture

This appendix details the folder and file structure of the project. The tree structure has been simplified to exclude system dependency folders and the virtual environment (`node_modules`, `venv`) and focus on the business and technical logic.

C.1. Key File Tree Structure

```
/Mon_Projet_Dashboard/
|
|-- .gitignore
|-- index.html          <-- Front-end entry point
|-- package.json        <-- Front-end dependencies (React, Vite)
|-- vite.config.js      <-- Front-end bundle configuration
|
|-- backend/            <-- Core of the API service (Flask)
|   |-- API_STRUCTURE.md  <-- Internal API documentation
|   |-- app.py          <-- Main API logic (Flask Routing)
|   |-- run.py          <-- Server startup script
|   |-- requirements.txt  <-- Python dependencies (pandas, scikit-learn, Flask)
|   |
|   |-- charts/         <-- Folder for generated chart images
|   |-- templates/      <-- Report HTML files (Jinja/PDF)
|   |   |-- correlations.html
|   |   |-- full_report.html
|   |   |-- summary.html
|   |   |-- ...
|   |-- uploads/        <-- Temporary storage of uploaded data files
|   |   |-- 000010_Flight_Status_...xlsx
|   |   |-- base_fictive.csv
|
|-- src/                <-- Front-end source code (React)
|   |-- main.jsx        <-- React starting point
```

```
|   |
|   |-- api/           <-- API connection modules (Axios, Fetch)
|   |   |-- endpoints.js
|   |   |-- services/
|   |   |   |-- analysisService.js
|   |   |   |-- cleaningService.js
|   |   |   |-- uploadService.js
|   |
|   |-- components/    <-- Reusable UI components
|   |   |-- Sidebar.jsx
|   |   |-- ThemeToggle.jsx
|   |   |-- ...
|   |
|   |-- pages/         <-- The main views of the application
|   |   |-- AIChat.jsx       <-- Gemini Chatbot assistance page
|   |   |-- Cleaning.jsx     <-- Data Cleaning interface
|   |   |-- Correlations.jsx <-- Correlation test results
|   |   |-- Upload.jsx       <-- Import page
|   |   |-- Visualization.jsx <-- Interactive charts
|   |   |-- ...
|   |
|   |-- contexts/      <-- Global state management files (React Context)
|   |   |-- ThemeContext.jsx
```

Appendix D

Technologies Used

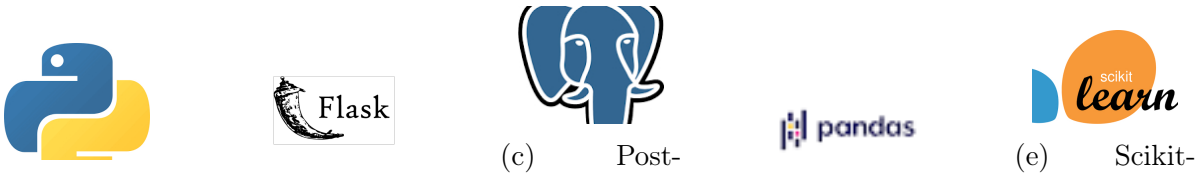
This appendix presents the different technologies, frameworks, and tools used in the context of this project.

Front-end Technologies

(a) React (b) HTML5 (c) CSS3 (d) JavaScript (e) Node.js

Figure D.1: Logos of Front-end technologies used

Back-end and Data Science Technologies



(a) Python (b) Flask (c) PostgreSQL (d) Pandas (e) Scikit-learn

Figure D.2: Logos of Back-end and Data Science technologies

Development and Versioning Tools

Cloud Services and APIs

Specialized Python Libraries

(a) GitHub

(b) Git

(c) VScode

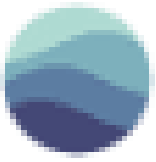
Figure D.3: Logos of development and versioning tools

(a) Google Gemini

(b) Power BI

Figure D.4: Logos of cloud services and APIs used

(a) Matplotlib



(b) Seaborn

(c) NumPy

(d) SciPy

(e) PyTorch

Figure D.5: Logos of Python libraries used