Theme Park Simulation

Rodrigo Calderón, Álvaro Aguilar, Jaime Carrera

04/04/2025

Contents

1 BUSINESS ORIENTATION

The development of a theme park simulator not only holds academic or technical value, but also offers direct application within the business and operational environment of such facilities. This project is aimed at providing a decision-support tool for leisure managers, industrial engineers, operations supervisors, and experience designers in the tourism sector.

1.1 Project Justification and Context

Theme parks face numerous operational challenges: managing visitor flow, optimizing limited resources (ticket booths, turnstiles, staff, attractions), maximizing customer satisfaction, and adapting to fluctuations in demand. Implementing real-world solutions to evaluate decisions is costly, risky, and time-consuming. This tool enables a realistic simulation of a full week of park operations, accounting for thousands of individual interactions between visitors, facilities, and timeframes. It allows experimenting with different configurations at no real cost—adjusting entry prices, extending or reducing opening hours, adding turnstiles, or modifying the range of available attractions. Additionally, it provides a controlled environment to analyze how these changes affect user behavior and the overall performance of the park.

1.2 Simulator Impact and Usefulness

The simulator has been designed to meet the following core objectives defined by the development team: Reduce average waiting time by identifying operational bottlenecks. Identify the most frequently used attractions and analyze saturation patterns. Optimize the distribution of available resources (infrastructure, access points, staff). Maximize visitor satisfaction by tracking the full experience. Analyze the effects of specific decisions regarding flow, capacity, and ticketing. A key functionality is its ability to detect improvement opportunities within the target audience.

1.3 Practical Application in Real Environments

This simulator can be applied to any type of amusement park, whether permanent or temporary. Its modular and configurable design allows it to adapt to large, medium, or small parks, regardless of their location or organizational structure. Typical implementation scenarios include: Permanent theme parks with multiple zones and thematic areas.

- Mobile or seasonal parks with variable operations.
- Family-oriented urban parks looking to expand their offer for children.
- Mixed leisure spaces with technological and digital attractions. Thanks to its flexible architecture, the simulator can be customized based on:
- User profiles (age, ticket type, waiting time tolerance).

- Types of attractions and their specific configurations.
- Adaptable schedules, special promotions, and events.

1.4 Future Projection of the Simulator as a Product

This software is conceived as an evolving platform. In future versions, the following enhancements are planned:

- Integration of machine learning modules for flow prediction.
- Real-time data visualization through interactive dashboards.
- Simulation of marketing strategies or campaigns during high-traffic periods.
- Expansion of behavioral logic based on psychological and social profiles.
- Development of a visual version with virtual visitor paths for presentations.

The ultimate goal is to consolidate the simulator as a professional-grade tool capable of improving, optimizing, and validating operational decisions in real amusement parks. A dynamic, adaptable, and data-driven system to transform both the visitor experience and operational efficiency.

2 GENERAL SYSTEM DESCRIPTION

The developed system is a discrete event simulator that represents the complete operation of a theme park over seven consecutive days. To achieve a realistic representation, the SimPy simulation environment was used, allowing the modeling of concurrent processes and shared resources—fundamental in environments with multiple points of interaction such as theme parks. This simulator is aimed at both internal analysis and operational improvement of the park, enabling managers to evaluate visitor behavior under different configurations. One of its main objectives is to identify saturation points such as long queues at ticket booths or attractions, and to analyze the distribution of usage across the park's various resources. The simulation includes a fully dynamic configuration layer via a .csv file, which allows the following parameters to be defined:

- Number of ticket booths and entry turnstiles.
- Daily operating hours (opening and closing time).
- Total simulation duration per day.
- Ticket price.
- List of attractions, including capacity, duration, number of modules, and target audience.

Each day begins with the loading of this configuration, the initialization of the simulation environment, and the random generation of visitors. These visitors have specific characteristics (type, satisfaction level, fatigue, ticket type), and their behavior evolves throughout the day based on their experiences within the park. Modular Structure of the System The simulator has been designed following a modular and object-oriented approach, composed of the following main components:

- ThemePark: Manages the complete flow of visitors, from arrival to departure, recording all intermediate events.
- TicketOffice: Manage entry resources (ticket booths and turntiles) and handle ticket purchase/validation processes, considering waiting times and potential dropouts.
- Attraction: Models the behavior of each individual attraction, including capacity, duration, queueing, wait-time penalties, and maintenance cycles.
- DataCollector: Logs all events occurring during the simulation, including visitor data, journey logs, usage times, wait times, ticket type, and exports the information to CSV files for analysis.
- Helper functions: Support functions for dynamic visitor generation, time formatting, and daily summary printing.

Daily Simulation Focus and Analytical Objectives The park is simulated day by day, replicating realistic operational conditions. The environment enables:

- Detection of the most popular attractions and assessment of their operational performance.
- Measurement of average time spent and usage by visitor profile.
- Evaluation of queue-reduction strategies through resource adjustment.
- Study of the impact of expanding the park's offer for children and its effect on family attendance.

Each visitor may enter using an online ticket or through physical ticket booths. During their visit, they choose which attractions to visit, how long to stay, and when to leave the park, based on their fatigue and satisfaction levels. Every step of their journey is recorded for later analysis. This system offers a solid foundation for future expansions and customizations, enabling its transformation into a high-level simulation tool applicable to a variety of real-world scenarios.

3 USE CASES AND FUNCTIONALITIES

The simulator has been designed with a realistic and practical approach to cover the main behavioral flows that occur in a real theme park. By modeling the various actors (visitors, attractions, ticketing) and shared resources, a system has been built that allows for observation, analysis, and data-driven decision-making.

3.1 Main Use Cases

The following use cases have been implemented in this first version of the simulator: Visitor arrival simulation: Random generation with variable arrival rates depending on the time slot (peak hours during the first two hours, and decline at the end of the day). Ticket purchase and validation: Management of entry flow through physical ticket booths and online ticket turnstiles. Waiting penalties and early exits if satisfaction drops too low. Autonomous park navigation: Visitors randomly select valid attractions based on their profile, energy level, and satisfaction. Interaction with attractions: Access control using SimPy resources (modules and staff), queuing, service usage, and penalties for delays. Includes maintenance cycles after certain usage thresholds. Comprehensive event logging: Every visitor action is recorded (entry time, visited attraction, wait time, satisfaction, exit), enabling complete traceability. Data analysis and export: At the end of each day, data is saved to .csv files for further analysis using tools like Excel or Pandas.

3.2 Specific Features of the Simulator

- Capacity control in attractions based on modules and per-module limits.
- Queue management for ticket booths, turnstiles, and attractions, with realistic timing and stochastic variability.
- Early exit simulation when visitors experience excessive wait times or low satisfaction.
- Daily revenue calculation based on ticket booth sales.
- Impact measurement by visitor profile (children, teens, adults) to analyze attraction usage trends.
- Planning of critical events such as attraction maintenance, triggered under defined conditions (based on usage count).

3.3 Strategic Applications of the Simulator

The simulator acts as a diagnostic tool applicable to various operational needs in a theme park:

- Validate new pricing policies or entry configurations (discounts, extra ticket booths).
- Assess the potential of adding more child-focused attractions to attract family audiences.
- Optimize human resource allocation, especially in high-demand attractions.
- Forecast the impact of special promotions or events on park operations. This event-driven simulation approach allows not only precise modeling of internal processes but also testing of different configurations to enhance the visitor experience and operational profitability.

4 ARCHITECTURE AND TECHNICAL DESIGN

The simulator's architecture is based on a modular, object-oriented model that enables the representation of different actors and processes involved in the operation of a theme park. Each module has a clearly defined responsibility and is designed to work in harmony with the SimPy simulation environment, which facilitates parallel process execution through discrete events. The main components of the architecture, their relationships, and functional logic are detailed below.

4.1 ThemePark Class

- Acts as the core class responsible for managing the complete visitor flow.
- Coordinates park entry through the TicketOffice.
- Controls individual visitor behavior throughout the day.
- Assesses each visitor's fatigue and satisfaction to determine when they leave.
- Interacts with available attractions and logs events in the DataCollector.

Simplified pseudocode:

```
for visitor in visitors:
    if visitor.entra:
        while not fatiga or baja_satisfacción:
            elegir_atracción()
            usar_atracción()
            registrar_evento()
        registrar_salida()
```

Figure 1: Simplified pseudocode of visitor behavior during the simulation.

4.2 TicketOffice Class

- Manages two SimPy resources: ticket booths (for purchasing tickets) and turnstiles (for entry validation).
- Implements queue logic and satisfaction penalties for excessive waiting times.
- Controls access based on ticket type (online or on-site).
- Records successful sales, dropouts, and calculates daily revenue.
- Key aspects:

- Adaptive logic to prevent entry of visitors with low satisfaction levels.
- Stochastic waiting times ranging between 0.5 and 2 minutes.
- Ability to configure the number of ticket booths and turnstiles via the CSV file.

4.3 DataCollector Class

- Stores two main data structures:
  - `visitor_data`: summary of each visitor (note: deprecated in current version).
  - `journey_log`: detailed log of all events during the simulation.
- Exports the journey log to a CSV file for further analysis.
- Records the following:
  - Successful and failed entries.
  - Waiting and usage times.
  - Attendance by ticket type.

```
{
  'id': 'V0034',
  'type': 'child',
  'ticket_type': 'online',
  'attraction': 'Montaña Rusa',
  'wait_at_attraction': 3.5,
  'ride_duration': 1.8,
  'satisfaction_status': 72,
  'fatigue': 2,
  'time': 678,
  'day': 'Wednesday'
}
```

Figure 2: Example of a visitor event record in the simulation log.

4.4 Attraction Class

- Controls access per module, duration per ride, and maintenance cycles.
- Logs wait times, usage duration, popularity, and performance statistics.
- Implements maintenance logic when a visitor usage threshold is reached.
- Included logic:
  - Satisfaction penalty if the wait time exceeds 5 minutes.
  - Random maintenance triggered after 400 to 600 uses.
  - Staff and module resources allocated per attraction.

4.5 Helper Functions (utils.helpers)

- `generate_visitors`: generates visitors stochastically based on profile and time of day.
- `format_time`: converts minutes to HH:MM format (useful for logs and visualization).
- `print_summary`: generates end-of-day reports including sales, timing, and attendance.

Module Relationships:

```
generate_visitors --> ThemePark --> TicketOffice
                                    |
                                    --> Attraction (usage)
                                    --> DataCollector (log)
```

This architecture provides a solid foundation for future expansions, allowing the introduction of new visitor types, behavioral rules, economic models, or visual layers. Its modular design ensures maintainability and adaptability of the system in response to any operational change.

5 WORK METHODOLOGY (AGILE)

The development of the simulator was carried out using an iterative and incremental work approach, based on the principles of Agile methodology. It was structured into a series of weekly sprints, with functional deliveries at the end of each cycle and a constant system of review and continuous improvement.

5.1 Work Phases

(The detailed table of phases can be added here if needed.)

5.2 Tools Used

- Version control: GitHub, to ensure traceability of changes.
- Planning: Trello and Gantt charts for task assignment and scheduling.
- Development environment: Visual Studio Code and Jupyter Notebooks.
- Peer review: Code reviews were conducted at the end of each sprint.

5.3 Meetings and Team Dynamics

- Sprint planning meetings at the beginning of each week.
- Progress review meetings every 3 days.
- Weekly retrospectives to improve work processes.
- Continuous communication via shared repositories and collaboration spaces.

5.4 Adaptability and Iterative Improvements

During the development process, improvements were identified and incorporated iteratively:

- Implementation of maintenance logic after identifying recurring congestion.
- Enhancement of visitor generation based on time-of-day adaptation.
- Removal of the `visitor_data.csv` file to simplify the system.

This approach enabled the creation of a functional product from the early stages, rapid adaptation to changes, improved code quality, and a clear vision of the final objective: to build a flexible, professional tool for decision-making in amusement park management.

6 TEAM ROLES

This project was developed through the coordinated participation of a multidisciplinary team, where each member assumed clear responsibilities in different phases of the process. The role structure allowed for effective time management, prioritization of deliverables, and assurance of product quality.

Role	Responsibilities	Specific Contributions per Sprint
Project Owner (José Luis Salmerón)	Defines the vision and scope of the project, ensures	Requirement validation, feedback loops, supervision c
Project Manager (Jaime Carrera)	Overall planning, team coordination, goal setting	Timeline design, task organization, progress tracking
Software Architect (Álvaro Aguilar)	Technical design, modular structure definition, desig	Class modeling, architecture and dependency choices
Lead Developer (Rodrigo Calderón)	Core simulation logic implementation and key entiti	Development of ThemePark, TicketOffice, Attraction, a
Data Analyst (Jaime Carrera)	Processing and analysis of exported results after ea	Summary generation, visualizations, metric validatior
QA Tester (Rodrigo Calderón)	Consistency testing, bug detection, and edge case a	Invalid input review, error handling in events

Figure 3: Team roles and specific contributions during development sprints.

7 TECHNICAL COMPLEXITY AND DESIGN CONSIDERATIONS

Implementing the simulator involved several technical challenges, requiring in-depth analysis of software design, process modeling, event control, and data handling. This section outlines the key aspects that define the complexity of the project and the design decisions made to ensure its robustness and scalability.

7.1 Technology Selection

SimPy was chosen as the simulation engine due to its ability to model discrete event processes in a simple yet powerful way. Unlike more visual but rigid libraries, SimPy allows for customized logic, queue handling, wait times, limited resources, and concurrency.

Additional tools and technologies used:

- Python: Main language for its versatility, readability, and large ecosystem.
- CSV: Lightweight format to parameterize the park and export simulation results.

7.2 Input Validation and System Robustness

All core classes include validations for data type and structure. Examples include:

- Verifying non-empty lists when instantiating attractions.
- Ensuring numeric and positive values for capacities, prices, or durations.
- Capturing and handling exceptions when using SimPy resources.

7.3 Concurrency and Shared Resource Handling

Using `simpy.Resource` enabled accurate simulation of simultaneous access to:

- Ticket booths
- Entry turnstiles
- Attraction modules
- Attraction staff (operators)

These resources generate queues and wait times based on real demand. The design considers:

- Limited and configurable capacity from the CSV.
- Satisfaction penalties for long waits.
- Retries, early exits, and blocked access when applicable.

7.4 Dynamic Penalties and Adaptive Behavior

Visitor satisfaction is affected by environmental variables:

- Waiting times over 5 minutes reduce satisfaction based on duration.
- Fatigue increases after each attraction, influencing park stay.
- Visitors with low satisfaction may leave before completing their journey.

This logic creates emergent, unpredictable behaviors that enrich the simulation experience.

7.5 Maintenance Control and Attraction States

Attractions alternate between "operational" and "maintenance" states. This process is triggered when a configurable visitor threshold is reached. During maintenance:

- The attraction is temporarily disabled.
- A random repair time is simulated.
- After completion, maintenance counters are reset.

7.6 Export, Persistence, and Post-Simulation Analysis

Currently, the system exports the `visitor_journey.csv` file, which records all events in detail. This file can be analyzed using tools such as Pandas, Power BI, or Excel.

Example of exported CSV content:

```
id,type,ticket_type,attraction,wait_at_attraction,ride_duration,satisfaction_status,fatigue,time,day,time_of_entry,time_left
V0056,adult,ticket_office,Roller Coaster,4.2,1.9,65,3,702,Saturday,10:35,11:18
```

Figure 4: Example of exported visitor data in CSV format.

7.7 Future Considerations

- Use of relational databases to manage simulation histories.
- Real-time dashboards for integrated visualization.
- Modularization into packages to facilitate maintenance and reusability.
- Multi-scenario support to compare configurations and evaluate improvements.

8 CONFIGURATION AND RESULTS ANALYSIS

This section presents a detailed analysis of the data collected during the weekly simulation of the theme park. The objective is to identify key behavioral patterns among visitors, evaluate the effectiveness of operational configurations, and provide data-driven recommendations for improving the overall park experience.

A variety of statistical and visualization techniques were used, including correlation analysis, distributional plots, segmentation by visitor attributes, and clustering.

8.1 Correlation Analysis

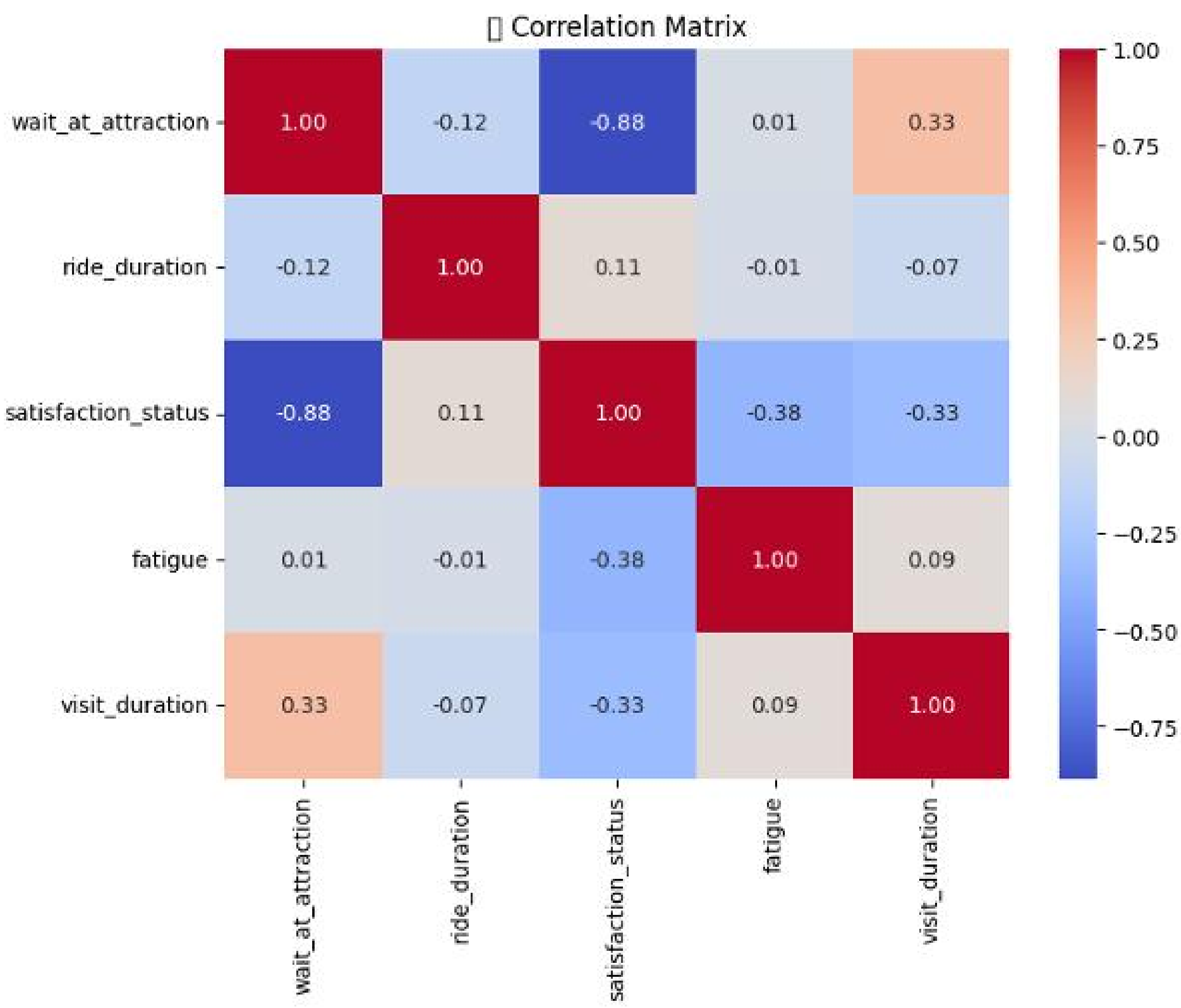

Figure 5: Correlation Matrix

The following correlation matrix summarizes the relationships between key visitor experience variables, such as wait time, satisfaction, fatigue, visit duration, and ride duration.

- **Wait Time and Satisfaction (r = -0.88):** A strong negative correlation indicates that longer wait times significantly reduce visitor satisfaction. Reducing queues should be a priority.
- **Wait Time and Visit Duration (r = 0.33):** Visitors who remain longer in the park tend to experience higher wait times. This could be due to visiting during peak hours or engaging in a larger number of attractions.
- **Satisfaction and Fatigue (r = -0.38):** Higher fatigue levels are associated with lower satisfaction, reinforcing the importance of visitor comfort.
- **Satisfaction and Visit Duration (r = -0.33):** Longer visits tend to correlate slightly with reduced satisfaction, likely due to cumulative fatigue or reduced novelty.
- **Ride Duration Correlations:** Ride duration has weak or negligible correlations with other metrics, indicating that it is not a strong predictor of satisfaction or fatigue.
- **Fatigue and Other Variables:** Fatigue shows weak correlation with most variables, except for satisfaction. This suggests that fatigue is more complex and influenced by individual pacing or experience.

8.2 Distribution of Visitor Metrics

The following histograms illustrate the distribution of key metrics to better understand general visitor behavior and experience throughout the week.

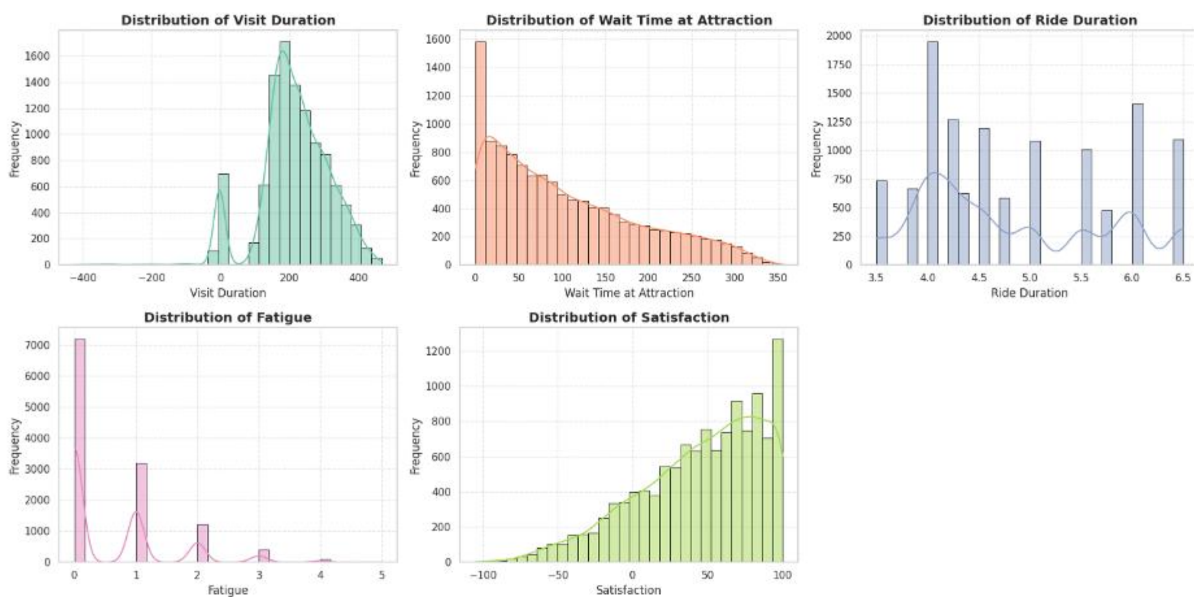

Figure 6: Distribution of Visitor Metrics

- **Visit Duration:** The distribution is right-skewed, with the majority of visitors spending between 100 and 300 minutes in the park. Outliers with extremely short or negative values suggest data quality issues.

- **Wait Time at Attractions:** Most visitors experience relatively low wait times. However, the long right tail indicates that some visitors experience significantly longer queues, particularly during peak hours.
- **Ride Duration:** A multimodal distribution reflects predefined ride durations. Peaks indicate standardized categories (e.g., 4–6 minutes), consistent with fixed attraction schedules.
- **Fatigue:** Most visitors report low levels of fatigue, with minimal dispersion. A few higher values may correspond to more intensive usage patterns.
- **Satisfaction:** Satisfaction scores are concentrated above 70, with some negative outliers pointing to possible recording or processing errors.

8.3 Comparison by Visitor Type (Adults vs. Children)

This section compares experience metrics between adult and child visitors, focusing on duration of stay, satisfaction levels, and fatigue.

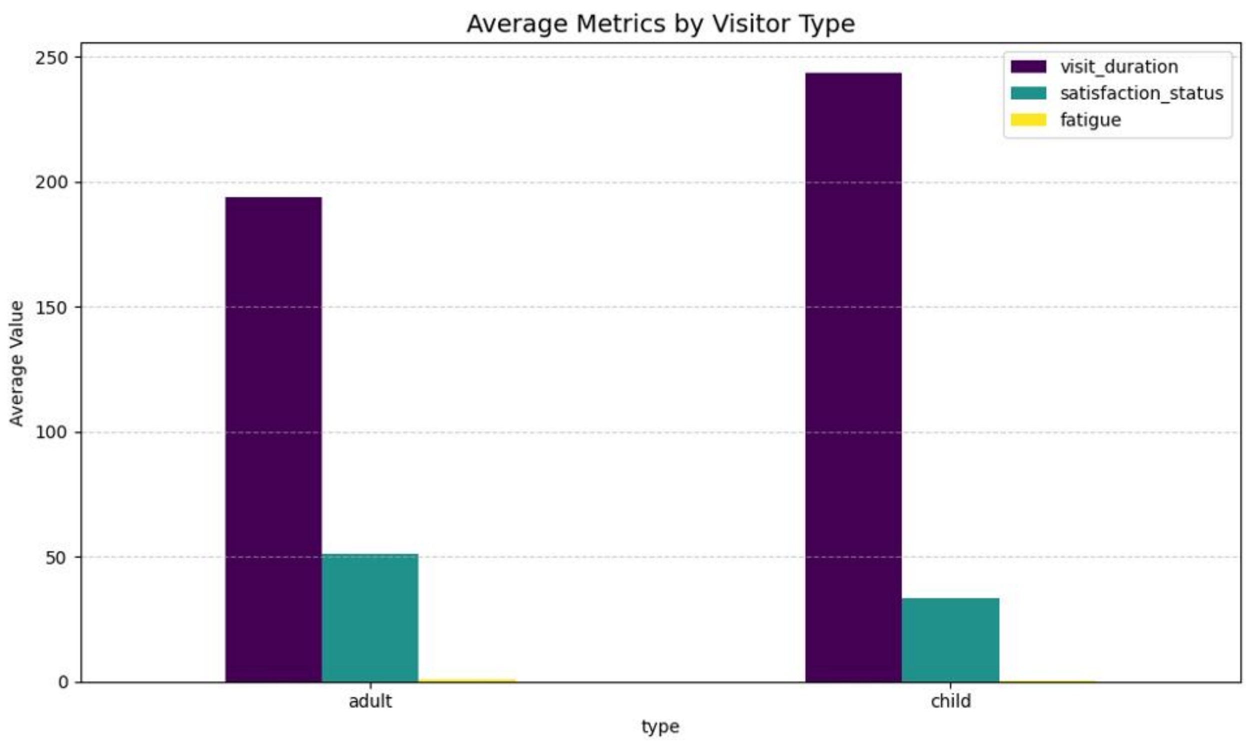

Figure 7: Average metrics by visitor type

- **Visit Duration:** Children tend to stay longer than adults. This may be due to broader engagement with child-specific attractions or family visits.
- **Satisfaction:** Adults report significantly higher satisfaction. This may indicate that existing services are better aligned with adult expectations, or that children are more affected by discomfort or waiting.

- **Fatigue:** Both groups report low fatigue levels, with children showing slightly less fatigue, possibly due to different pacing.
- **Conclusion:** The experience is currently more favourable for adults. Improvements in child-specific engagement and comfort may help balance satisfaction across demographics.

8.4 Impact of Ticket Type

This analysis compares the impact of ticket acquisition method (online vs. ticket office) on visit duration and satisfaction.

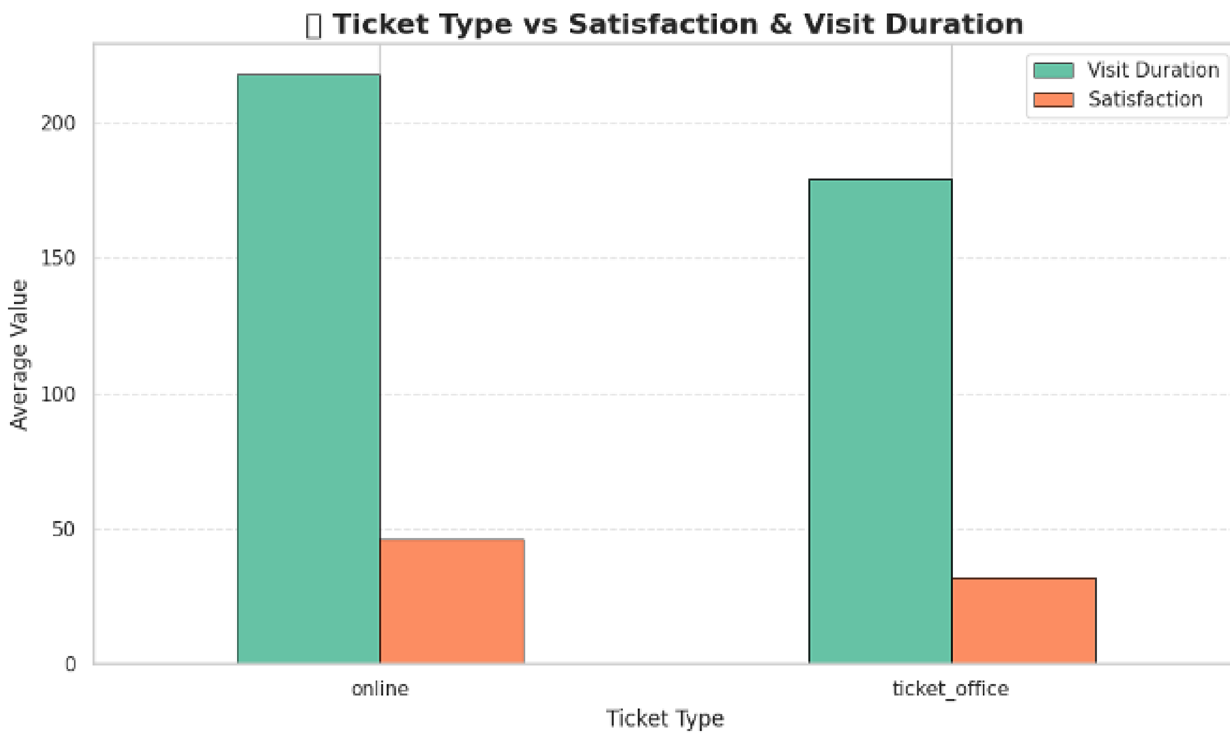

Figure 8: Comparison of ticket type with average visit duration and satisfaction level.

Visit Duration: Online ticket holders tend to remain in the park longer, possibly due to streamlined entry and better planning. Satisfaction: Visitors who purchased tickets online report higher satisfaction, likely due to reduced entry friction and greater convenience. Conclusion: The data highlights a clear benefit in promoting online ticketing: It correlates with longer engagement in the park. It is associated with higher visitor satisfaction. Improving the digital experience and incentivizing online sales could further enhance operational flow and visitor sentiment.

8.5 Visitor Clustering (K-Means Analysis)

A behavioral segmentation was conducted using K-Means clustering based on standardized metrics such as visit duration, wait time, satisfaction, and fatigue.

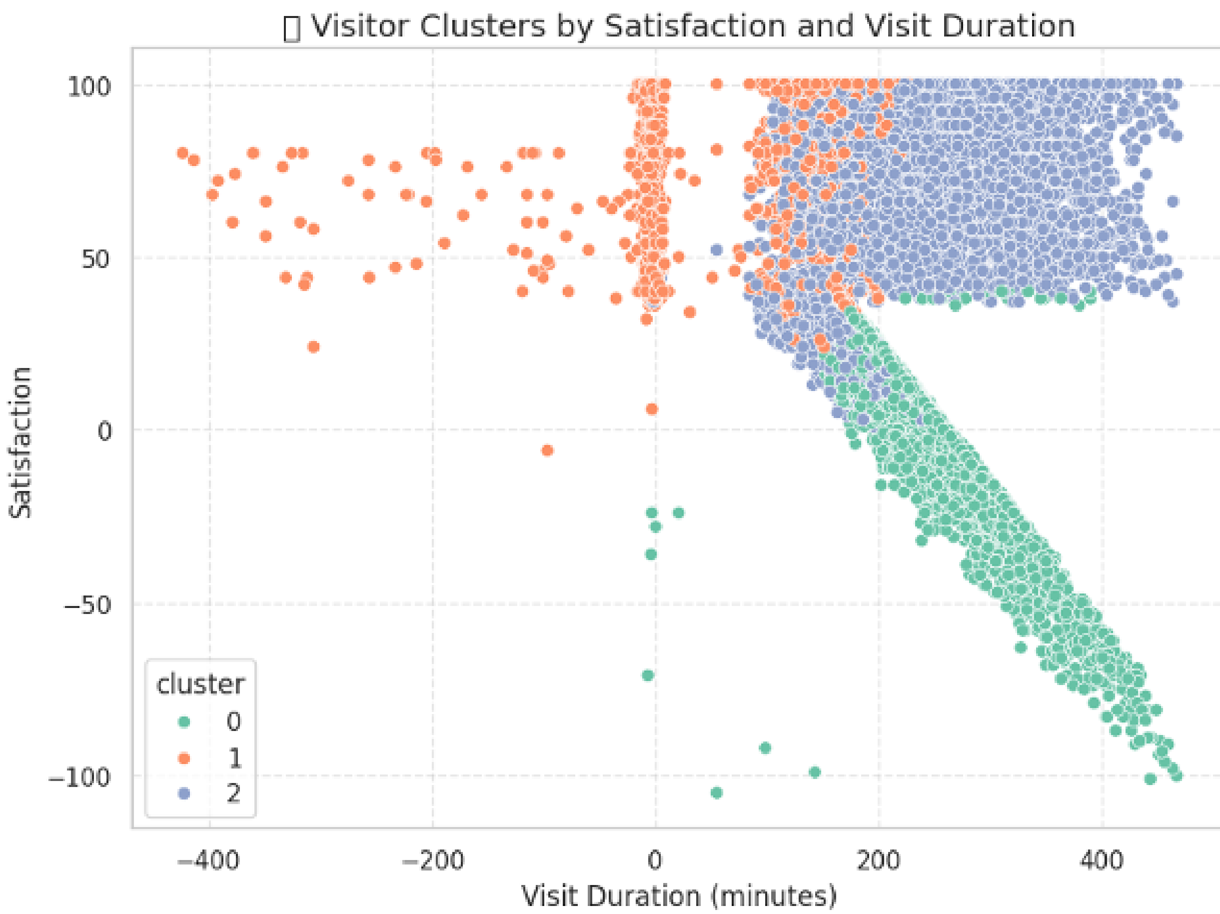

Figure 9: Segmentation of visitors using clustering based on visit duration and satisfaction level.

Cluster Descriptions: Cluster 0 (Short Visit, High Satisfaction): Visitors in this group have short visits but report high satisfaction. They likely focus on a few targeted experiences. Cluster 1 (Long Visit, Low Satisfaction): This group stays the longest but reports low or even negative satisfaction. Factors may include long queues or exhaustion. This group represents a risk for negative feedback. Cluster 2 (Balanced Experience): Visitors show moderate-to-long durations with consistently high satisfaction. This cluster represents the target behavior pattern for optimal park flow. Observations: Data anomalies (e.g., negative values) were present and require correction. Longer stays do not necessarily correlate with higher satisfaction. Segmentation allows for differentiated strategies: improve Cluster 1, maintain Cluster 2, and optimize efficiency for Cluster 0.

8.6 Boxplot Analysis by Age Group

Boxplots were used to compare visit duration, satisfaction, and fatigue between adults and children.

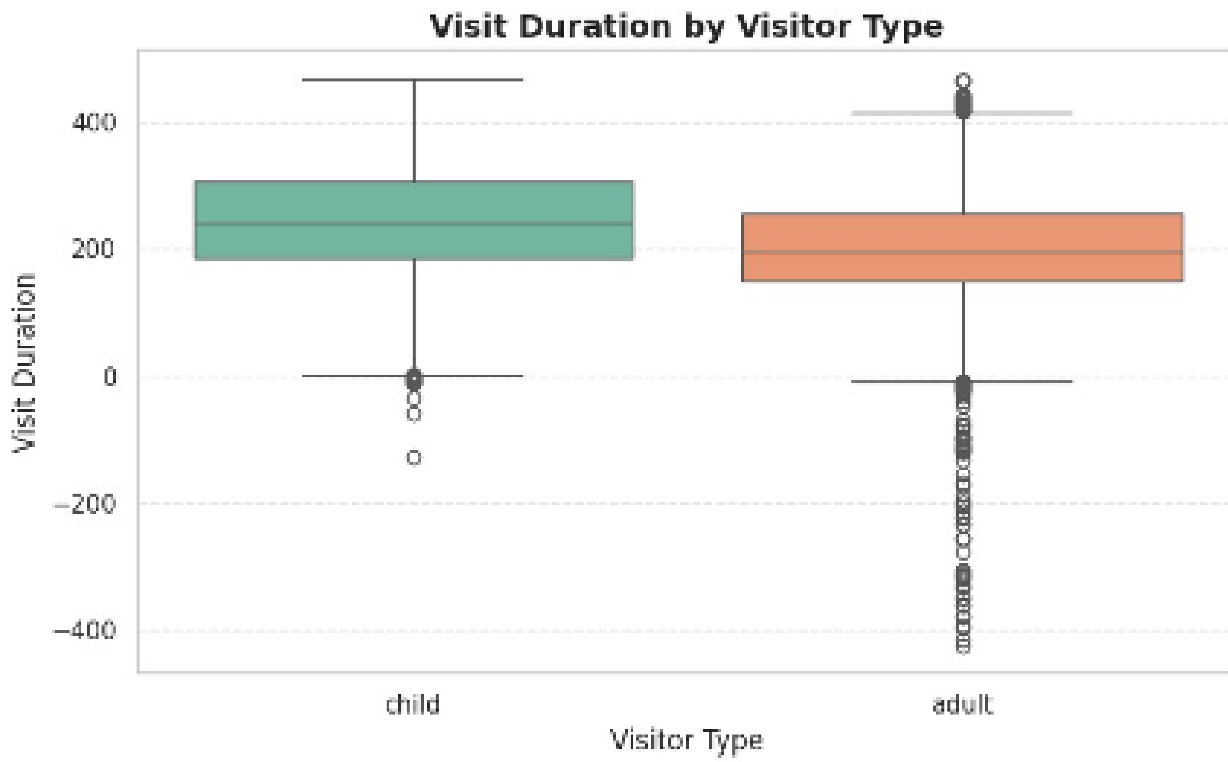

Figure 10: Visit duration by visitor type (children vs. adults).

Visit Duration: Children stay longer on average, with a wider interquartile range. Outliers, including negative durations, should be addressed through data cleaning.

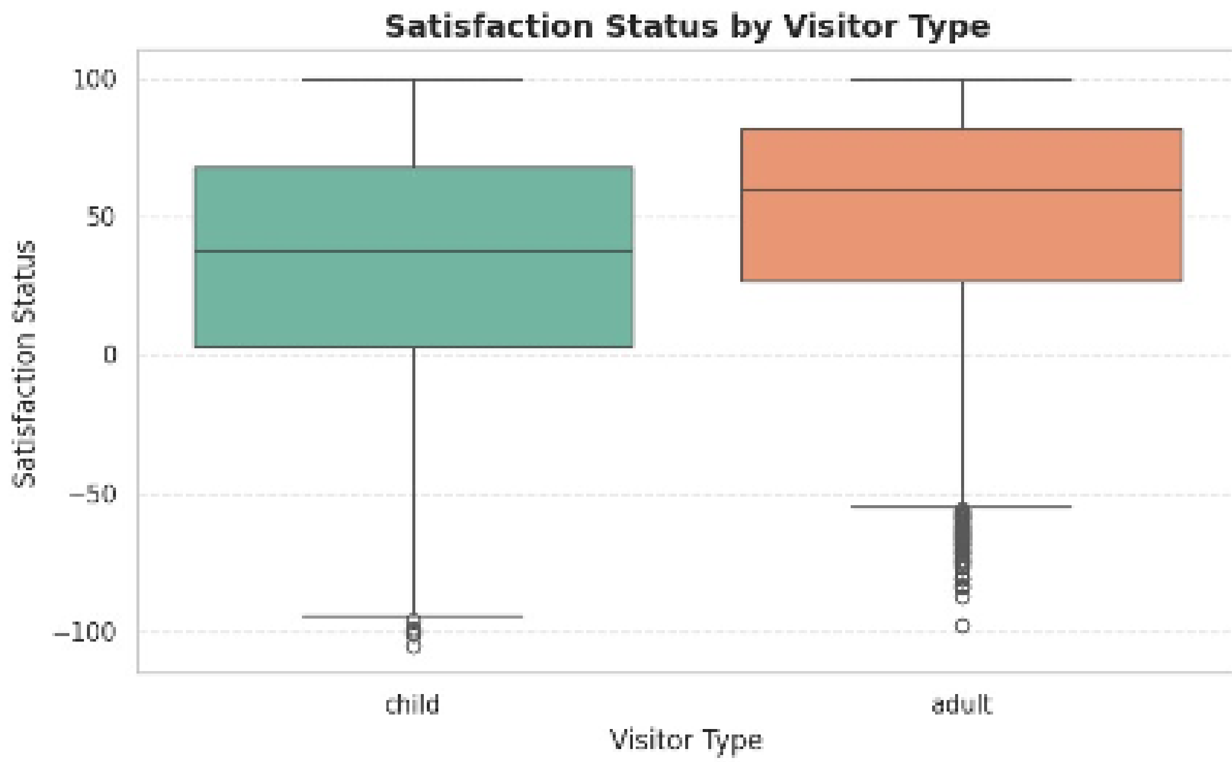

Figure 11: Satisfaction level by visitor type (children vs. adults).

Satisfaction: Adults consistently show higher satisfaction. Children's results are more dispersed, indicating a greater sensitivity to negative factors.

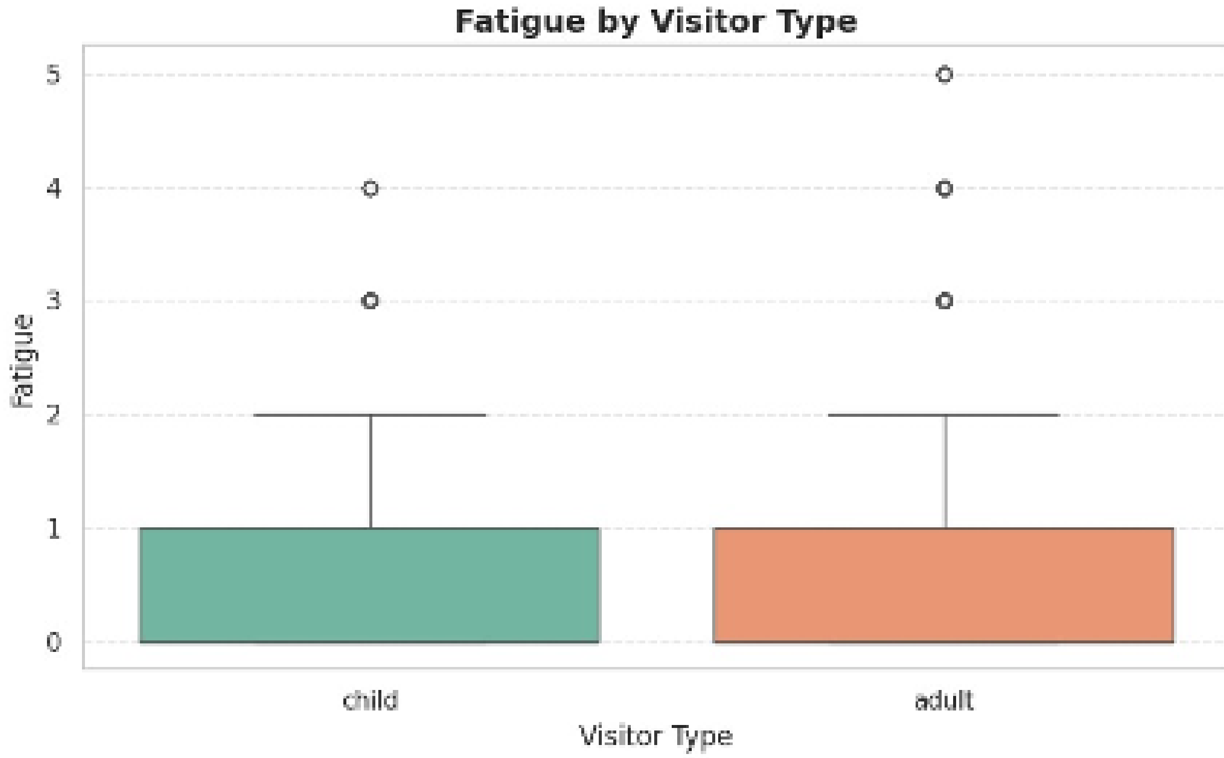

Figure 12: Fatigue levels by visitor type (children vs. adults).

Fatigue: Both groups exhibit low levels of fatigue, suggesting balanced pacing. Isolated high outliers may be linked to individual behavior. Conclusion: The experience is more optimized for adult visitors. Additional efforts should be made to enhance the child experience, reduce dissatisfaction, and maintain low fatigue levels.

8.7 Weekly Wait Time Patterns

A violin plot was used to visualize how attraction wait times fluctuate by day of the week.

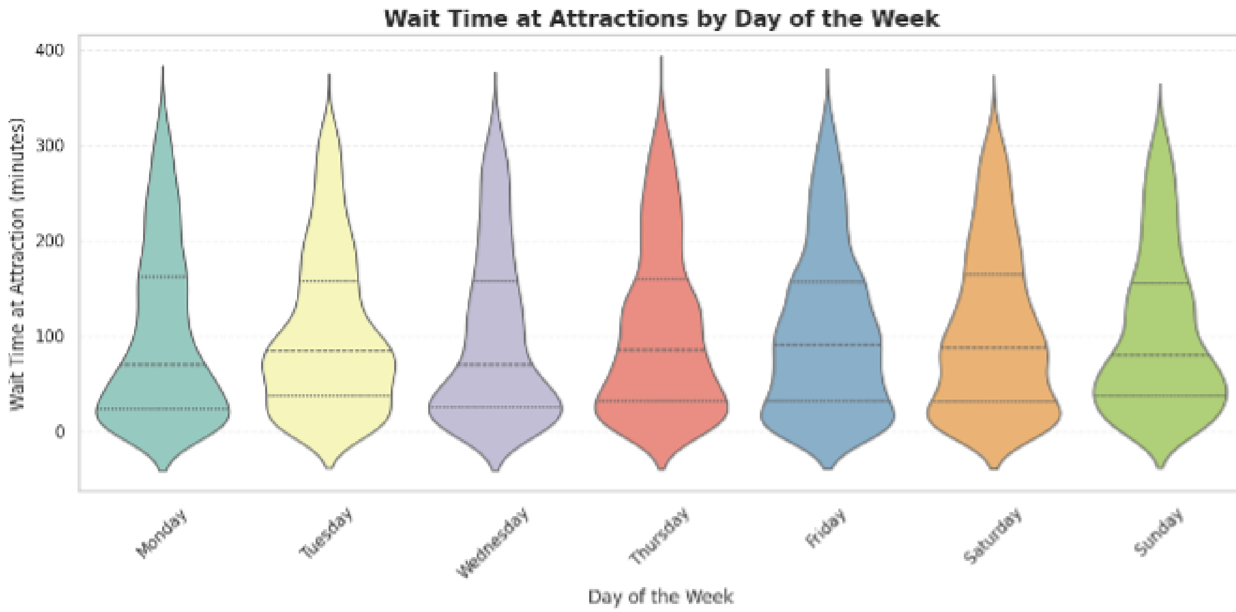

Figure 13: Wait time at attractions by day of the week.

Weekends (Saturday–Sunday): Highest variability and longest wait times due to peak visitor volume. Tuesday: Lowest and most consistent wait times, making it the optimal day for a balanced park experience. Monday and Wednesday: Moderate wait times and stable distributions, suggesting efficient management. Thursday and Friday: An upward trend in wait times, signalling the beginning of weekend congestion. Conclusion: Visitor load follows a predictable weekly cycle. Operational planning should prioritize weekend crowd management while leveraging midweek periods—particularly Tuesday—for smoother and higher-quality visitor experiences.