Rush Hour Reality: A Spatiotemporal Analysis of Dublin Bus Delays

CSC1143/CSC1175 Data Visualisation Assignment

Group Name: **ZA**

Abstract

This project investigates the patterns of bus delays across Dublin City over a full 24-hour cycle. We aimed to test the hypothesis that "Rush Hour" is not a uniform event but a localized phenomenon affecting specific regions differently. To investigate this, we built a data pipeline to fetch real-time GTFS data from the National Transport Authority. Over a period of 13 days, we collected approximately 1.5 million records using an automated Windows Task Scheduler script. We processed this data to calculate specific delay times for every bus in the network. The final artifact is an animated geospatial map that allows users to observe the accumulation of delays. Our analysis shows that while Central Dublin maintains flow due to bus corridors, West Dublin experiences severe delays during the 07:00 - 10:00 and 17:00 - 20:00 peak windows.

What is the question you are answering or the story you are trying to tell?

We are answering a geographic question: Does the city gridlock uniformly, or are delays specific to certain routes? The story focuses on the inequality of commuting, a passenger in the City Centre might move freely, while a commuter on the D18 faces significant stagnation.

What is the conclusion that you reached?

We concluded that rush hour is geographically uneven. Our visualization proves that routes servicing West Dublin experience "Badly Delayed" status (>10 minutes) far more frequently than North or Central routes. Central Dublin shows resilience, likely due to dedicated infrastructure.

1. Dataset(s)

Source and Retrieval

We used open data provided by the National Transport Authority (NTA) of Ireland via their developer portal. We integrated two distinct datasets to build this visualization:

- **GTFS-Realtime (Dynamic):** We wrote a Python script to poll the TripUpdates and VehiclePositions API endpoints. We set up a Windows Task Scheduler job to run this script automatically every 30 minutes for a duration of 13 days. This allowed us to build a historical archive of real-time movement. Note: There are minor gaps in the data during periods where the host machine was powered down, but the volume collected (1.5 million rows) was sufficient for a robust 24-hour aggregate model.
- **GTFS Static (Reference):** We downloaded the standard schedule files (routes.txt, stops.txt) from Transport for Ireland. This dataset provided the necessary context, such as the full names of routes and the latitude/longitude coordinates of stops.

Data Description

The collected data is substantial in both volume and variety.

- **Geographic Coverage:** We filtered the data to the Greater Dublin Area (Latitude 53.15 to 53.60) to focus strictly on urban congestion.
- **Attributes**: Key fields include trip_id, stop_sequence, arrival_delay (seconds), and GPS coordinates.
- **Format**: The raw real-time feed uses Protocol Buffers (Protobuf), a binary format that requires specific decoding libraries.

Big Data Aspects (Volume, Velocity, Variety)

This project demonstrates all three V's of Big Data:

- Volume: We collected over 1.5 million records over a 13-day period. This scale required efficient memory management and could not be processed in Excel.
- Velocity: The dataset originates from a high-frequency stream. The NTA API updates vehicle positions every 30-60 seconds. Our project captures this velocity by animating the data rather than presenting a static snapshot.
- Variety: We integrated semi-structured binary data (Protobuf streams) with structured CSV files (Static Schedule). Merging these disparate formats was a key challenge in the data preparation phase.

2. Data Exploration, Processing, Cleaning and/or Integration

Preparation Pipeline

We used Google Colab for our data processing pipeline. We chose the Polars library instead of Pandas because Polars is multi-threaded and handles large datasets significantly faster.

- 1. Ingestion: We used the ‘gtfs-realtime-bindings’ library to parse the binary Protobuf messages into Python dictionaries.
- 2. Cleaning: We implemented a crucial geographic filter to exclude intercity coaches and focus solely on the Greater Dublin Area bounding box (Lat 53.15 to 53.60, Lon -6.60 to -6.00).
- 3. Integration: We performed a join operation between the live delay data and the static route shapes. This matched moving buses to their physical path on the map.

Attribute Selection & Exploratory Analysis

We focused on ‘delay_minutes’. We derived this metric by subtracting the ‘scheduled_arrival_time’ from the ‘actual_arrival_time’. Before building the final map, we performed extensive exploratory analysis to validate our hypothesis.

Figure 1: Exploratory analysis identifying the 07:00–10:00 morning rush as a critical window.

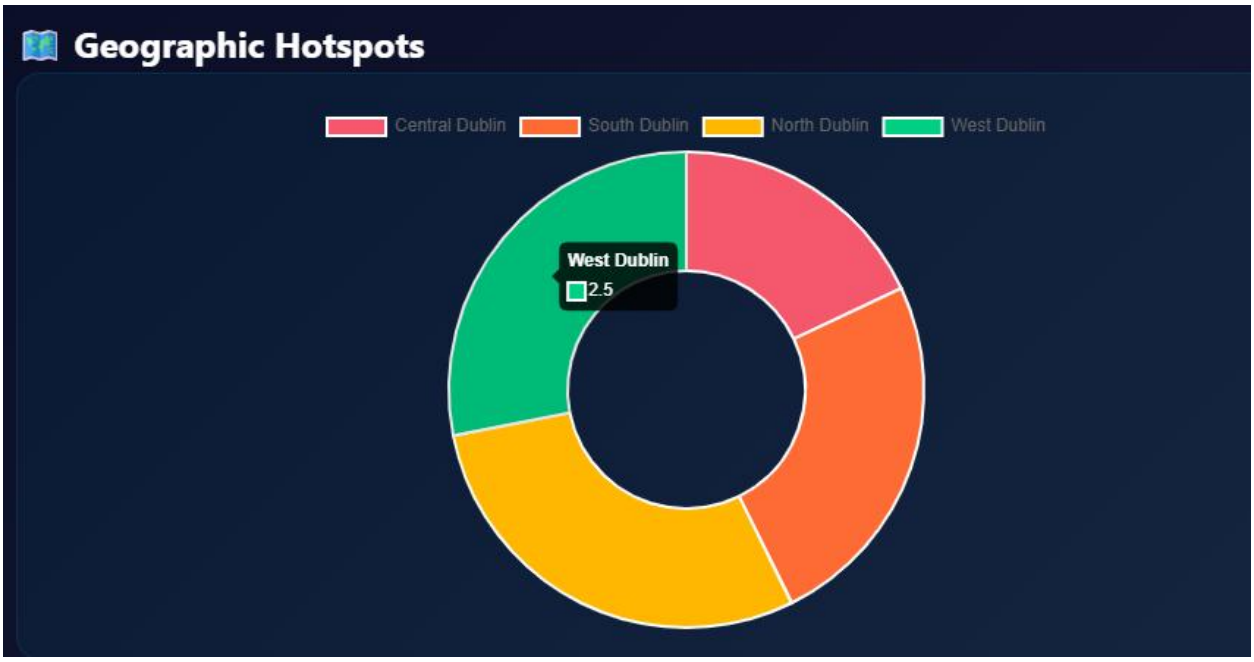

Figure 2: Regional aggregation confirming West Dublin as the primary bottleneck, guiding our geospatial focus.

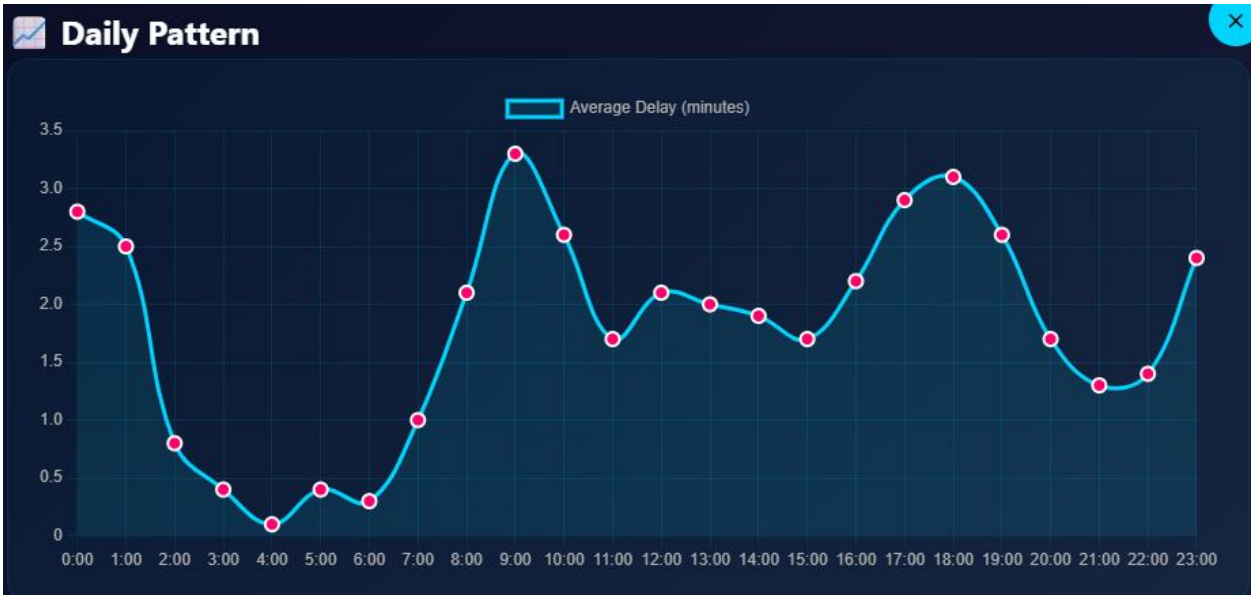

Figure 3: Temporal trend analysis used to calibrate the animation speed for the final visualization.

3. Visualisation

Design Process

We initially considered a static choropleth map (heatmap). However, a static image failed to capture the velocity of the data. Traffic is a flow, not a fixed state. Our design iteration moved towards an animated path visualization to show the accumulation of delay over time. We iterated through several sketches to plan the User Interface.

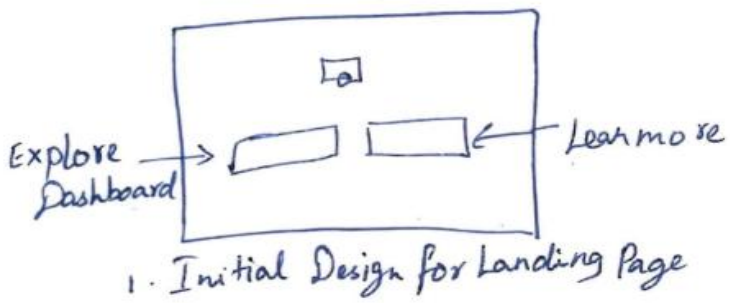

Figure 4: Initial design sketch for the Landing Page layout.

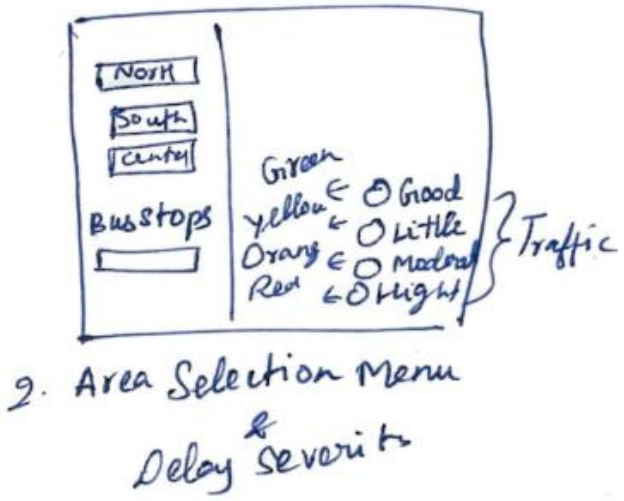

Figure 5: Sketch of the Sidebar logic, Region selection, and Color coding.

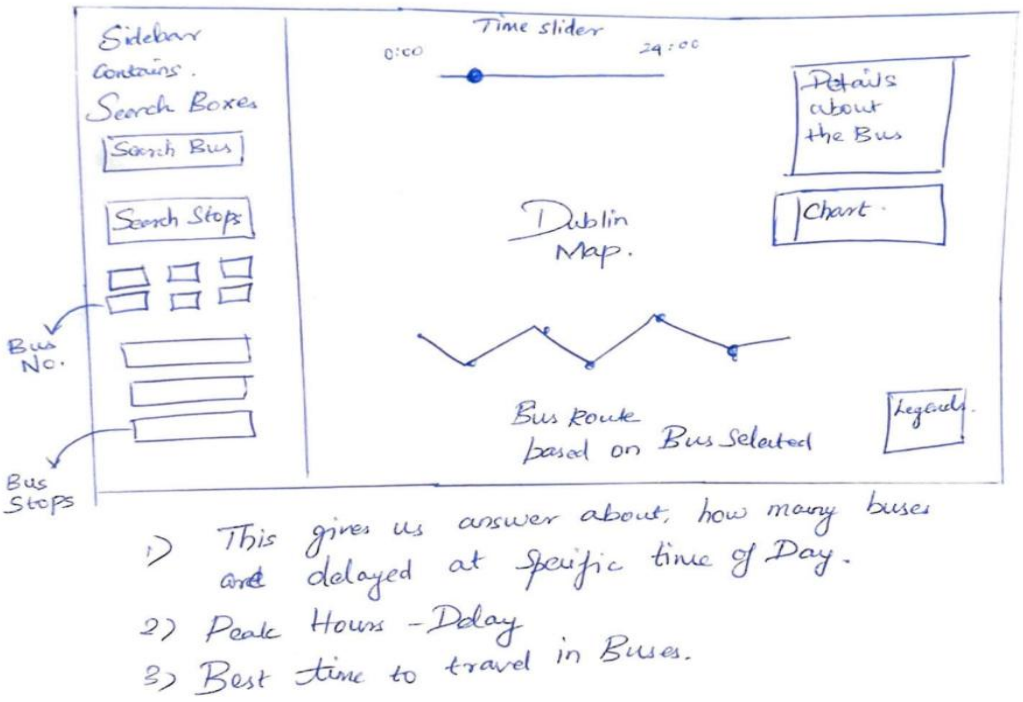

Figure 7: Final concept sketch showing the Map, Time Slider, and Sparkline Graph integration.

Chart Type & Rationale

We built an Animated Geospatial Node Link Diagram.

We used Mapbox GL JS as our rendering engine. We chose a map because traffic data is inherently spatial. A bar chart can indicate that delay is high, but only a map can show where it is high. We added a Time Slider to the interface. This turns the visualization into an exploratory tool, allowing the user to "play" the day like a video.

Design Choices & Justification

Dark Mode Map:

A dark-themed satellite base map was selected to reduce visual clutter and ensure the bright, neon-colored data points stand out clearly (luminance contrast). This makes the routes legible even in dense city areas where a light map would be too noisy.

Colour Encoding: We used a diverging "Traffic Light" scale:

- Green (#00d084): On Time (< 2 mins)
- Amber (#ffb700): Delayed Slightly (2–6 mins)
- Vibrant Orange (#ff6b35): Moderate (6-10 mins)
- Red (#f5576c): Badly Delayed (> 10 mins)

Justification:

This leverages pre-attentive processing. Users instinctively associate red with danger or stopping in a traffic context. We avoided blue or purple for delays as these do not carry the same semantic warning weight.

3D Context:

We extruded the buildings in 3D using the Mapbox building layer. This serves as a landmark system, allowing users to orient themselves (e.g., recognizing the Spire) without requiring excessive text labels.

Animation Physics:

We used Turf.js to calculate the movement of the bus icon along the route path. We linked the speed of the icon to the delay value. If a bus is "Badly Delayed," the icon physically moves slower across the screen. This allows the user to feel the delay visually.

Heads-Up Display (HUD):

To adhere to the single-graph rule while providing depth, we overlaid a sparkline chart on the map. This follows the principle of "details on demand." The user views the spatial context on the map but can glance at the HUD for the 24-hour temporal trend.

No Service State

We implemented a specific visual state for when a bus is not running (e.g., 3 AM). The route line turns Dark Grey and the bus icon fades out. This prevents misleading "On Time" readings when the API simply returns no data.

Interactivity

Users can click any white node (stop) on the route to see the specific delay at that location and view connecting routes. The sidebar allows for filtering by region (e.g., "North Dublin"), which solves the issue of overplotting where too many lines would make the map unreadable.

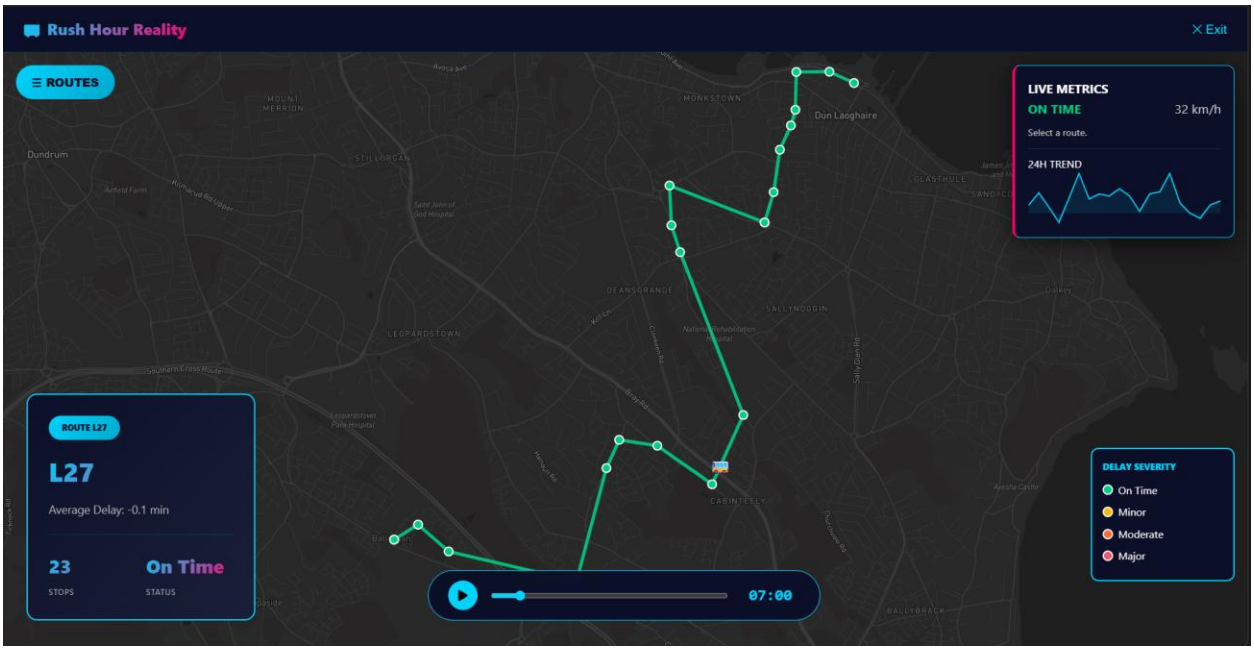

Figure 8: The Final Explanatory Visualization showing Route L27 in South Dublin with 3D buildings and HUD active.

4. Conclusion

Tools Used

- Python & Google Colab: For data collection and cleaning.
- Polars: For high-performance data aggregation.
- Mapbox GL JS: For the core mapping visualization.
- Turf.js: For geospatial calculations and animation logic.
- Chart.js: For the HUD sparklines.

Critical Analysis

The visualization successfully answers the research question. It demonstrates that West Dublin routes suffer significantly higher delays than Central routes. The animation effectively communicates the "pulse" of the network.

Technical Limitations & Improvements

- Technical Limitation: Rendering all 205 routes simultaneously created chaos in the browser. We solved this by implementing area-based filtering to render only a subset of data at a time.
- Improvement: While the current visualisation shows historical averages, a future improvement would be connecting to the live API for real-time streaming visualisation (Velocity).
- Data Gap: The system relies on the API being active; if a bus stops sending data, we have to assume "No Service," which might confuse users.

Collaboration

- Arun Narayanan: Led the project, responsible for data collection (Task Scheduler), the entire Python data processing pipeline (Polars/Google Colab), final data integration, and report writing.
- Abdul Razzaq: Assisted with initial design sketches, building the final HTML visualization artifact, testing the Map interactivity to identify bugs, and contributed to sections of the report preparation.

References

[1] National Transport Authority. (2025). Transport for Ireland Open Data API. Available at: https://developer.nationaltransport.ie/

[2] Google. (2025). GTFS Realtime Reference. Available at: https://developers.google.com/transit/gtfs-realtime

[3] Google. (2025). Google Colab. Available at: https://colab.research.google.com/

[4] Transport for Ireland. (2025). GTFS Static Data Files. Available at: https://www.transportforireland.ie/transitData/Data/GTFS_All.zip

[5] Python Software Foundation. (2025). Protobuf Library. Available at: https://pypi.org/project/protobuf/

[6] Polars Developers. (2025). Polars DataFrame Library. Available at: https://pola.rs/

[7] Mapbox. (2025). Mapbox GL JS Documentation. Available at: https://docs.mapbox.com/mapbox-gl-js/

[8] Chart.js Contributors. (2025). Chart.js Documentation. Available at: https://www.chartjs.org/

[9] Turf.js. (2025). Turf.js Geospatial Analysis. Available at: https://turfjs.org/

[10] Font Awesome. (2025). Font Awesome Free Icons (Bus, Clock, Map Icons). Available at: https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css

[11] Google Maps Platform. (2025). Geocoding API (used for verifying Dublin coordinates: 53.3498, -6.2603). Available at: https://developers.google.com/maps/documentation/geocoding/overview

[12] Mapbox. (2025). Available at: https://www.mapbox.com/

Declaration of AI Use

[13] We used Google Gemini during this project. It was used to debug JavaScript errors in the animation loop and identify the root cause of map rendering failures. We also used it to write the HTML/CSS structure for the floating 'HUD' panel and to refine the vocabulary and grammar of this report to ensure clarity. All data processing logic, visualization design choices, and conclusions are our own work.