PulseCast: Turning a Noisy Footfall Index into a Weekly Staffing Plan

Author: Noel Sabia (ORCID 0009-0008-3064-7573)
Affiliation: —
Date: 14 November 2026
Artifact: https://pulse.noelsabia.dev · **API:** https://api-pulse.noelsabia.dev · **Code:** to be released after the hackathon

Abstract

Inner-city store managers must decide next week's shifts without knowing next week's footfall. We present PulseCast, a decision-first staffing copilot built on eight passive Bluetooth footfall counters in a ~580 m × 280 m cluster in Heilbronn, Germany. The system fuses five months of 15-minute device counts with Open-Meteo weather and Baden-Württemberg public holidays, trains a global gradient-boosted model on a time-based split, and converts the forecast into a weekly shift plan through a transparent staffing heuristic. On a 28-day holdout, the weather-aware 7-day model reaches MAE 3.32, WAPE 27.5%, and R^2 0.777, versus 5.40, 44.7%, and 0.462 for a seasonal-naive baseline; the day-ahead model reaches 3.36, 27.8%, and 0.763. The plan is generated in 19 ms cold on v1.1 (contract < 2 s), and the deployed system handles 120/120 concurrent SSE clients with zero errors and a p95 event latency of 331 ms. A code-level audit of the public 2026 firmware commit e8be8bf shows a ~10-second scan, a 16-bit dedup key, and a 0.7 multiplier; the deployed 2025 binary may differ, and the data confirm the 0.7 scaling and per-wake export, not the scan duration or truncation. We therefore frame the result as a relative activity index that supports staffing decisions, not a city-wide people counter, and state what calibration and privacy work absolute claims would require.

1. Introduction

A store manager on Kaiserstraße knows the weather forecast and the staff roster. She does not know how many people will walk past next Saturday. Raw counts are not a decision; the manager needs a forecast, an explicit staffing rule, and a way to test a scenario. A dashboard that shows history does not close that gap.

The Schwarz Digits
Pulse of the City

challenge asks teams to turn privacy-preserving pedestrian counts from Heilbronn into predictive tooling for advertising, store locations, and staffing [1]. We chose staffing: eight fixed locations, five months of multi-sensor history, and a clear daily/weekly rhythm. Store-location ranking would need city-wide coverage that eight sensors in a 580 m cluster cannot provide; a city-wide heatmap would be a visualization, not a measurement.

PulseCast makes three contributions. It treats the decision as the product: the forecast feeds a weekly shift plan. It uses a global, weather-aware gradient-boosted model with a time-based holdout and a seasonal-naive baseline, rather than a model the data cannot support. It is deployed as a three-service system on SKE with a cached fallback and a live scenario layer. It also reports a code-level audit showing that the published value is a short-window count of randomized BLE addresses scaled by 0.7, not a 15-minute count of people.

The report also serves the Schwarz Digits criteria: **problem and benefit** (a manager gets a plan, not a dashboard), **originality** (decision-first tooling plus a measurement audit), **feasibility and future** (live on SKE, 120-client load test, calibration roadmap), **reliability** (120/120 connections, zero errors, cached fallback), **understanding** (Sections 2.2 and 6 state what the index does and does not measure), **teamwork** (data/ML, Go API, Nuxt UI, and infra workstreams against a written contract, with human integration), **documentation** (code, contract, metrics, and this report), and **technology** (Vue/Nuxt + Go + Onyx + FastAPI on namespace-only SKE). For the final pitch, the technical evidence supports the 50% idea/implementation weight, the honest limits support the storytelling, and the calibration/privacy roadmap is the 20% vision.

We do not claim a production people counter, a city-wide heatmap, annual seasonality, live sensor data, or a proven return on investment. We claim a working decision loop, a credible relative forecast, and an honest account of the measurement limits.

2. Data

2.1 The delivered PAX data

The dataset contains 192,464 CSV rows: 96,232 density readings and 96,232 battery readings. There are 12 device IDs; after merging the two `density-01` IDs and excluding three test sensors, eight usable locations remain in a ~580 m × 280 m cluster: `density-01`, `hn-goetzentuem-01`, `hn-hagenbuchersee-01`, `hn-hagenbuchersee-02`, `hn-marahaus-01`, `hn-neckarufer-01`, `hn-soleo-01`, and `hn-stadtgalerie-01`.

The 12 June model window (12 June–7 November 2026) has a median cadence of 15.3 minutes; values range 0–292.6, with median 8.4, p95 49.7, p99 79.8. Across all 96,232 density rows, the median is 7.7, p95 46.9, p99 78.4 and the multi-sensor mean is ~14.4. In the 12 June model window, the inner city peaks 18:00–21:00 local, top 19:00; Saturday is busiest on average (mean 19.1) [2]. The sensors are LoRaWAN-connected PAX counters that passively sniff Bluetooth Low Energy (BLE) advertisements and transmit aggregated device counts [1, 2]. The counts are estimates, not a census: device penetration, Bluetooth settings, and the scanning design all matter.

2.2 What one value actually is

A code-level audit of the public 2026 Code for Heilbronn `lorax` firmware (commit e8be8bf) changes the interpretation of the numbers [3, 4]. The node scans BLE passively for about 10 seconds, counts distinct locally administered addresses, truncates the last two address bytes into a 16-bit ID, multiplies by 0.7, and sleeps for 900 seconds. The deployed 2025 binary may differ; the data confirm the 0.7 scaling and per-wake export, not the scan duration or truncation.

In the public 2026 firmware, the published value is `0.7 × U`, where `U` is the number of distinct randomized BLE addresses seen in a ~10-second scan. The 15-minute cadence is the sleep cycle, not the integration window; the effective duty cycle is ~1.1% in that code. Every one of the 83,156 values in the **12 June model window** is a multiple of 0.7; the reconstructed count has median 12, p95 71, max 418, and 2.70% of those values are zero. A hardware replacement creates a scale break in `density-01`; the legacy and current series are not pooled [5].

This is a relative activity index, not a count of people. A 30-second passer-by has a ~4.4% chance of overlapping a scan; a one-hour lingerer can be counted up to four times. The index conflates flow and dwell, cannot distinguish phones from watches/beacons/cars, and the people-per-device ratio is unknown. The 0.7 factor is a provisioning default, not a validated penetration correction.

> **Practical consequence.** The model predicts the index well enough to compare relative peaks and drive a staffing rule. It does not predict people, and it does not support absolute footfall or revenue claims without calibration.

2.3 Coverage, gaps, and context

Coverage is uneven. In the **18 June coverage panel (82,170 readings)**, only 28.2% of 15-minute slots have all eight sensors; the modal count is seven (4,117 slots). Aggregating to 30, 60, or 120 minutes raises complete coverage to 63.4%, 69.7%, and 71.0% [5]. Across all **96,232 density rows**, 5.18% are zero, 10.36% are <1, and 39.57% are <5; in the **12 June model window** the shares are 2.70%, 8.32%, and 35.86%. The worst month falls to 23.3% coverage at one location in the 18 June panel. Gaps reach four to six days; the cause is undocumented, but missing periods differ systematically from full-coverage periods [2, 5]. Weather comes from Open-Meteo: historical archive for training, current/7-day forecast for the dashboard. Temperature and precipitation are the useful features; cloud and wind did not improve the short test. Baden-Württemberg holidays are a feature and badge; they added little and are context, not a large driver [2].

Sensor	Readings	Mean	Median	p95	Max	Max daily gap
hn-stadtgalerie-01	8,982	21.1	8.4	67.2	137.2	6 days
hn-neckarufer-01	9,527	19.2	11.9	61.6	255.5	5 days
hn-hagenbuchersee-02	10,099	19.3	11.2	64.4	200.9	4 days
hn-marahaus-01	9,254	13.7	9.1	38.5	194.6	6 days
hn-hagenbuchersee-01	12,345	13.0	6.3	49.7	292.6	4 days
hn-goetzentuem-01	9,952	11.1	8.4	29.4	210.7	5 days
hn-soleo-01	10,327	10.6	8.4	28.0	111.3	4 days
density-01	12,670	9.9	7.7	28.0	63.0	35 days

Table 1. Per-sensor coverage and value statistics. Except for the `density-01` max-gap entry, statistics use the 12 June–7 November 2026 model window; the `density-01` 35-day gap spans the longer history and hardware replacement.

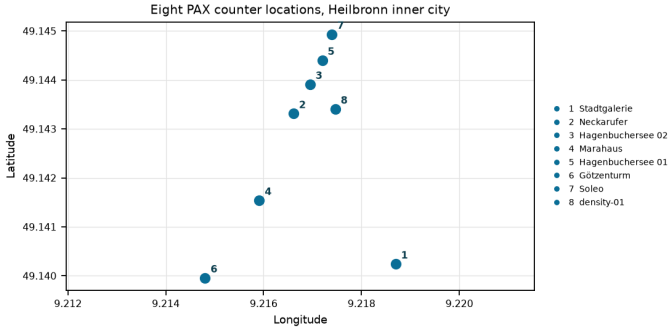


Figure 1. The eight sensor locations in the ~580 m × 280 m inner-city cluster. Measured points only; no city-wide surface.

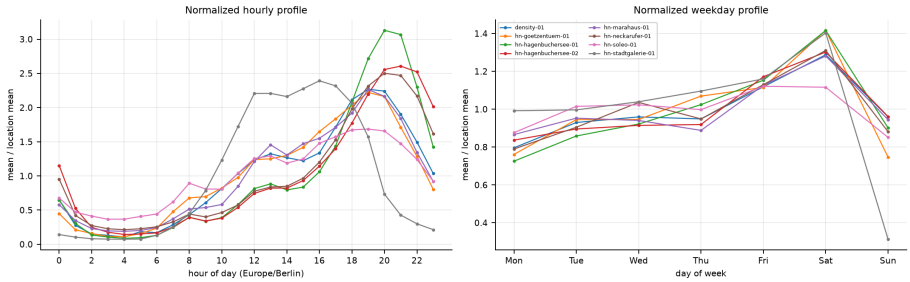


Figure 2. Mean index by hour and weekday across the eight sensors. The peak is 18:00–21:00 local; Saturday is busiest on average.

3. Method

3.1 Preprocessing and features

We parse timestamps as UTC, convert to `Europe/Berlin`, handle daylight saving, treat the two `density-01` IDs as one logical location but do not pool the legacy integer series with the 0.7-scaled replacement (the model uses the current device only), exclude the three test sensors, and aggregate to hourly means. The target is the hourly mean of the published index, not people.

Features are calendar (hour, weekday, month, weekend, holiday), lags (24, 48, 168 hours, plus 24-hour and 7-day rolling means), temperature and precipitation from Open-Meteo, and sensor identity. All lags and rolling windows end at the forecast issue time; the time-based split prevents leakage.

3.2 Model and validation

We train a global `HistGradientBoostingRegressor` across all eight sensors. Train 12 June–10 October 2026; test the last 28 days, 10 October–7 November. Baseline: seasonal naive, same hour one week earlier. We report MAE, WAPE, and R^2. The global model pools strength across sensors while keeping sensor identity; with eight series and five months, a deep sequence model is not justified.

The model serves 15 min, 1 h, 48 h, and 7 d. The 15 min/1 h horizons are nowcast/replay because the CSV is static; 48 h/7 d are planning horizons, with 7 d the hero.

3.3 From forecast to shift plan

The ML model predicts the published index; a deterministic heuristic turns it into staff:

```
forecast index per hour
  → calibration factor k (unvalidated)
  → visitors per staff per hour (service standard)
  → required staff = ceil(index × k / standard × safety factor)
  → minimum staff, shift length, opening hours
  → shift blocks + coverage score + estimated cost
```

The assumptions are visible and adjustable: calibration factor `k`, visitors per staff hour, labor cost per hour, minimum staff, shift length, and safety factor. `k` converts index units to visitors and is **unvalidated**; setting `k = 1` would treat one index unit as one visitor, which the audit says is not justified. The manager can edit assumptions and regenerate the plan with a scenario. This is decision support, not an optimizer with proven ROI; we have no POS, sales, queue, or labor-cost data.

4. System

PulseCast runs as three namespace-scoped Deployments on STACKIT SKE plus a Cloudflare Tunnel connector: **Nuxt 4 SSR + Onyx** for the server-rendered shell, map, charts, forecast, plan, and scenario controls; a **Go API** for sensor/series/forecast/staffing endpoints, SSE fan-out, scenario state, and a cached fallback; and a **FastAPI ML service** for `/predict` and `/simulate`, loading a model baked into the image at build time.

The namespace does not allow Ingress, Jobs, CronJobs, cert-manager, or external-dns. We expose the app through a Cloudflare Tunnel, terminate TLS at Cloudflare, and run the batch at build time. The model and fallback JSON are baked into the images; there is no database or PVC. If FastAPI is unavailable, Go serves the cached forecast and labels it. SSE carries scenario changes with a 15-second heartbeat; polling is the fallback. The plan is generated in 19 ms cold on v1.1; the contract is under two seconds.

The public app is at https://pulse.noelsabia.dev and the API at https://api-pulse.noelsabia.dev. The v1.1 images are 16.9 MB (api), 249 MB (web), and 952 MB (ml). In the v1.1 120-client Cloudflare k6 run, 120/120 SSE connections opened and closed with zero connection errors; p95 event latency 331 ms, p95 scenario latency 646 ms, presenter POST 49.60 ms, over 4m57.9s. The system runs non-root with resource limits and a read-only root filesystem. The public dashboard is read-only; `/control` uses basic auth. We store no PII or MAC addresses in the application. This is a hackathon deployment: one API replica, no SSO, no HA, no live ingestion. We document those gaps.

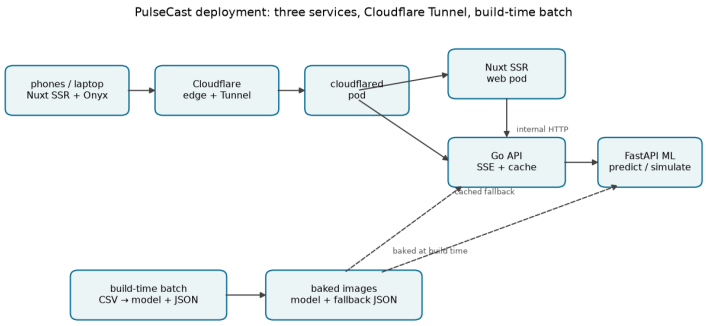


Figure 3. Three services plus `cloudflared`: Nuxt SSR, Go API/SSE and cache, FastAPI ML. Build-time batch; cached fallback; no Ingress, cert-manager, or Jobs.

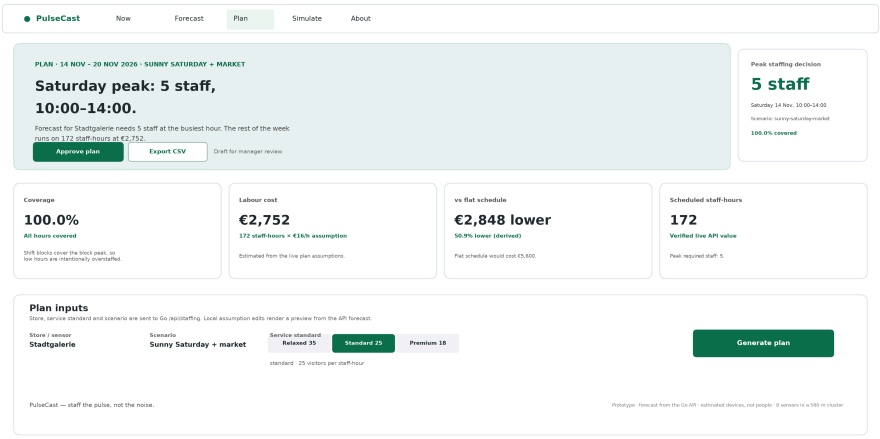


Figure 4. The v1.1 plan screen for `scenario=sunny-saturday-market`, `standard=standard` (live API query, 14 Nov 2026; reconstructed for print): Saturday peak 5 staff, 172 scheduled staff-hours, €2,752 labour cost, 100% coverage, versus a €5,600 flat schedule. The live screen also includes a "why this plan" context card; the app is at https://pulse.noelsabia.dev.

5. Evaluation

5.1 Forecast accuracy

Table 2 reports the 28-day holdout from the generated `metrics.json`. The weather-aware 7-day model reduces MAE by about 19% relative to the no-weather 7-day model (4.08 → 3.32) and by about 39% relative to seasonal naive (5.40 → 3.32). WAPE falls from 44.7% to 27.5%, and R^2 rises from 0.462 to 0.777. The day-ahead weather model reaches 3.36, 27.8%, and 0.763; the longer horizon is slightly stronger (3.32 vs 3.36 MAE) and remains well above the baseline.

Model	MAE	WAPE	R^2
Seasonal naive (same hour last week)	5.40	44.7%	0.462
Day-ahead, no weather	3.96	32.8%	0.706
Day-ahead, + temperature/rain	3.36	27.8%	0.763
7-day-ahead, no weather	4.08	33.8%	0.697
7-day-ahead, + temperature/rain (production)	**3.32**	**27.5%**	**0.777**

Table 2. Model comparison on the last 28 days, from the generated `metrics.json` (`n = 5,350`). The mission brief's 3.36/27.8/0.779 was the quick baseline; the canonical generated metrics are the production 7-day 3.32/27.5/0.777 and day-ahead 3.36/27.8/0.763. The target is the published index, not people.

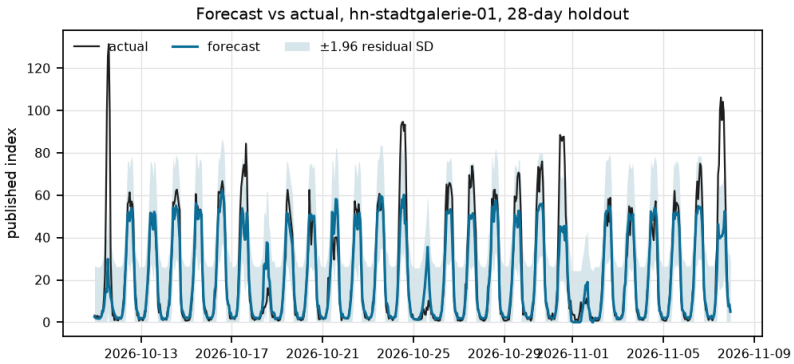


Figure 5. Forecast versus actual for one sensor over the 28-day holdout. The model captures the daily/weekly shape but misses spikes; the band is a planning aid, not a calibrated 95% interval unless coverage is reported.

5.2 Ablation and per-sensor behavior

Weather is the main driver. In the day-ahead ablation, removing it raises MAE 3.36 → 3.96. The holiday flag changes 3.96 → 3.84 (0.12); it is context, not a large lift. The production 7-day model is 3.32 with weather and 4.08 without. These are single-site, 28-day results.

Per-sensor errors vary (Table 3): Stadtgalerie has the highest MAE (5.83); Hagenbuchersee-01 the highest WAPE (38.6%); density-01 the lowest MAE (2.26). The global model is pooled; local errors are not.

Sensor	MAE	WAPE	R^2
density-01	2.26	30.8%	0.722
hn-goetzentuem-01	2.40	26.9%	0.718
hn-hagenbuchersee-01	2.47	38.6%	0.602
hn-marahaus-01	2.67	29.8%	0.671
hn-soleo-01	3.19	20.2%	0.548
hn-neckarufer-01	3.63	28.2%	0.695
hn-hagenbuchersee-02	4.12	31.1%	0.731
hn-stadtgalerie-01	5.83	25.3%	0.765

Table 3. Per-sensor holdout error for the production 7-day model ($n \approx 669$ per sensor).

5.3 System metrics and decision evidence

Metric	Value
Plan generation latency	19 ms cold on v1.1 (contract < 2 s)
Concurrent SSE clients	120/120 opened and closed; 0 connection errors
SSE p95 event latency	331 ms
SSE p95 scenario latency	646 ms
Presenter POST latency	49.60 ms
Duration	4m57.9s
Image sizes (api / web / ml)	16.9 MB / 249 MB / 952 MB
Cold deployment time	not measured

Table 4. System metrics from the v1.1 deployment and the v1.1 120-client Cloudflare k6 run. Cold deployment time was not measured.

The v1.1 120-client Cloudflare run is the measured result: 120/120 connections opened and closed, zero errors, p95 event 331 ms. The plan screen compares the recommended schedule to a flat schedule on coverage and cost under stated assumptions; this is a heuristic comparison, not an ROI claim.

5.4 What the evaluation does and does not show

The evaluation shows that a global, weather-aware model beats a seasonal-naive baseline on a time-based holdout in one cluster. It does not show that the model predicts people, generalizes to another city, that weather is causal, or that the plan improves profit. The 28-day test is short, weather is observed, and the target is a ~10-second BLE index. It is a relative decision-support signal, not a breakthrough.

6. Discussion and Limitations

The main limitation is the measurement. The sensor samples ~10 seconds out of every 15 minutes, counts only randomized BLE addresses, truncates to 16 bits, multiplies by 0.7, and cannot distinguish people from devices. The value is a relative index, not a census. Penetration is unknown; a single factor cannot absorb device ownership, multiple devices per person, Bluetooth settings, and the short scan window.

Coverage is also a limitation. Eight sensors in 580 m × 280 m cannot support a city-wide heatmap or store-location recommender. Only 28.2% of 15-minute slots have all eight sensors; gaps are not random. The 28-day holdout is short, weather is observed, and there is no event log, POS, or labor-cost data. The deployment has one API replica, no SSO/HA, and no live ingestion; 15 min/1 h are replay/nowcast.

Privacy. The app uses aggregate counts only, stores no MAC addresses or PII, and does not redistribute the raw challenge CSV. The sensing layer is upstream of this application. A production deployment would need a DPIA and a documented legal basis before collecting MAC-derived counts.

What to keep: decision-first framing, global model, time-based holdout, seasonal-naive baseline, transparent heuristic, cached fallback, build-time batch, honest limitations. **What to replace:** sensing (thermal/radar/camera/probe/manual), calibration (site-specific, with uncertainty), privacy (no raw MAC retention, central DP, k-threshold), spatial model (network-aware hierarchical field), forecast (quantile GBM, forecast weather, conformal calibration), platform (auth, observability, SLOs) — in that order. Fix the measurement before scaling the model.

7. Related Work

Passive Bluetooth/Wi-Fi sensing consistently finds a small, biased sample: counts track relative trends, absolute levels need local calibration [6, 7]. MAC randomization reduces linkability but does not make passive counting a census, and fingerprinting can defeat it [7]. Our audit is consistent and makes the sampling design explicit. Gradient-boosted trees remain a strong baseline for tabular, covariate-rich, small-data problems [8]. Weather affects pedestrian activity, but its magnitude is context-dependent [9]; we treat it as predictive, not causal.

8. Conclusion

PulseCast shows that a small, weather-aware global model can turn a noisy inner-city footfall index into a weekly staffing plan that beats a seasonal-naive baseline and runs in 19 ms cold on a real deployment. It also shows that the index is not a people counter: each value is a short BLE scan scaled by 0.7, sampled every 15 minutes, with unknown penetration and incomplete coverage. The honest conclusion is narrower and more useful than a city-scale claim: the decision loop works, the relative forecast is credible, and the next step is calibration and a privacy-first sensing layer. We would rather ship a plan a manager can use and state what it does not measure than promise a city-wide heatmap the data cannot support.

Artifacts, data, and AI disclosure

The live app is https://pulse.noelsabia.dev (API: https://api-pulse.noelsabia.dev); the code repository will be released after the hackathon. The raw challenge CSV is not redistributed. Derived aggregates, the model card, and the metrics JSON can be shared subject to the organizers' permission. The app stores no MACs or PII; the sensing caveats are in Section 2.2 and the privacy box is in Section 6.

AI tools were used for drafting and editing. The authors checked each number and citation; some sources are project documentation or estimates. AI tools are not authors and take no responsibility for the content.

References

[1] Schwarz Digits / Code for Heilbronn. *The Pulse of the City*, challenge brief, 2026. Local: `OKF_Vault/Schwarz Digits/FCH Challenge The _Pulse of the City_.pdf`.

[2] PulseCast project. `OKF_Vault/mission/data.md`, `mission/mvp.md`, `mission/architecture.md`, 2026.

[3] Code for Heilbronn. "Passantenzahlen messen mit LORA," 18 September 2025. https://codeforheilbronn.de/blog/2025/09/passantenzahlen-messen-mit-lora/

[4] Code for Heilbronn. `lorax` / `sensor-pax` firmware and backend, commit `e8be8bf5c11e1b76c88c3f176382c6df99eea989` (develop, 18 June 2026). https://codeberg.org/cfhn/lorax/commit/e8be8bf5c11e1b76c88c3f176382c6df99eea989; upstream `ESP32-Paxcounter` and `libpax`, https://github.com/cyberman54/ESP32-Paxcounter and https://github.com/dbinfrago/libpax.

[5] PulseCast data-quality analysis, `paper/research/07-data-quality.md`, 2026.

[6] A. Kurkcu and K. Ozbay. "Estimating pedestrian densities, wait times, and flows with Wi-Fi and Bluetooth sensors." *Transportation Research Record* 2644(1):72–82, 2017. DOI: 10.3141/2644-09.

[7] J. Martin, T. Mayberry, C. Donahue, L. Foppe, L. Brown, C. Riggins, E. C. Rye, and D. Brown. "A study of MAC address randomization in mobile devices and when it fails." *Proceedings on Privacy Enhancing Technologies* 2017(4):365–383. DOI: 10.1515/popets-2017-0054.

[8] J. H. Friedman. "Greedy function approximation: a gradient boosting machine." *Annals of Statistics* 29(5):1189–1232, 2001. DOI: 10.1214/aos/1013203451.

[9] L. F. Miranda-Moreno and A. C. Lahti. "Temporal trends and the effect of weather on pedestrian volumes: a case study of Montreal, Canada." *Transportation Research Part D* 22:54–59, 2013. DOI: 10.1016/j.trd.2013.02.008.