**Note for Future Updates: This document is the primary source of truth for the project's state, designed to be consumed by AI assistants. It must be as technical and detailed as possible. NEVER edit or remove previous entries. All new updates must be appended to the bottom with a new timestamp to maintain a complete, chronological history log.**

**Project 'Market Pulse' - Changelog**

**Initial document generated: Saturday, July 5, 2025**

1. **Core Project Specification Objective: A web application to provide hotel performance metrics, including a live dashboard and a "vs. The Market" comparison tool. Frontend: public/index.html and public/script.js. Vanilla JavaScript. Backend: server.js (Node.js/Express). Serves the frontend and provides a JSON API. Database: Neon PostgreSQL. Deployment: Vercel, connected to the main branch of the GitHub repository. Sensitive Data: All secrets (API keys, database URLs) are managed via a local .env file and as Environment Variables in the Vercel project settings.**

**Entry: Saturday, July 5, 2025 - Morning Session Objective: Migrate the application from a local-only setup to a fully operational, cloud-native solution on Vercel. Key Files Modified: server.js, vercel.json, daily-refresh.js. Backend Changes: server.js: Logic updated to use process.env.DATABASE\_URL for the production database connection. daily-refresh.js: Converted to a Vercel Serverless Function format using import / export and a default handler function. Deployment & Configuration: vercel.json: Created to define build outputs and configure a cron job. Cron Job: path: "/api/daily-refresh", schedule: "0 1 \* \* \*" to run daily-refresh.js daily at 1 AM UTC.**

**Entry: Saturday, July 5, 2025 - Afternoon Session (Data Seeding) Objective: Solve the "cold start" problem for the "Us vs. The Market" feature by creating and importing a rich set of mock competitor data. Key Files Modified: seed-market-data.js (new file). Database Schema Changes: hotels table created to store static hotel information (name, city, coordinates). Foreign key constraint added between daily\_metrics\_snapshots.hotel\_id and hotels.hotel\_id. Backend Changes: seed-market-data.js: New Node.js script created to parse daily\_metrics\_snapshots\_5hotels.csv and perform a batch INSERT ... ON CONFLICT operation into the daily\_metrics\_snapshots table. Data Artifacts: daily\_metrics\_snapshots\_5hotels.csv: CSV file created with 3,650 rows of mock data for 5 competitor hotels.**

**Entry: Saturday, July 5, 2025 - 2:40 PM CEST (UI & Backend Aggregation) Objective: Transform the raw competitor data into a true "like-for-like" comparison view and improve the UI for direct comparison. Key Files Modified: index.html, script.js, server.js. Backend Changes (server.js): /api/competitor-metrics: The SQL query in this endpoint was fundamentally changed. It now uses AVG(), SUM(), and GROUP BY stay\_date to perform data aggregation directly in the database. This ensures the API returns a single, averaged "market" row per day. Frontend Changes: index.html: Restructured to use a two-column div layout (<div class="comparison-container">) to place "Your Hotel" and "Competitor Market" tables side-by-side. script.js: The renderCompetitorMetricsTable function was simplified. It no longer needs to handle multiple rows per day or a hotel\_id column. Headers were updated to "Market ADR", etc.**

**Entry: Saturday, July 5, 2025 - 3:15 PM CEST (Granularity Feature) Objective: Implement a feature allowing users to view the comparison data with daily, weekly, or monthly granularity. Key Files Modified: index.html, script.js, server.js. Backend Changes (server.js): /api/metrics-from-db & /api/competitor-metrics: Both endpoints were enhanced to accept a granularity query parameter. A logic block was added to sanitize the input (e.g., 'weekly' -> 'week') and dynamically build the SQL query. For non-daily views, the SQL queries now use DATE\_TRUNC('<granularity>', stay\_date) and GROUP BY DATE\_TRUNC(...) to perform the aggregation at the requested time interval. Frontend Changes: index.html: Added a div with class toggle-group containing three buttons: [ Daily ] [ Weekly ] [ Monthly ]. Changed the Number of Days input to an End Date picker (<input type="date" id="master-end-date">). script.js: Added a global variable currentGranularity to hold the state. Added setGranularity(granularity) function to handle button clicks and update the state. loadAllDbData() function updated to read from the new master-end-date input and pass the currentGranularity to the API calls. Table rendering functions updated to change the date column header based on the selected granularity.**

**Entry: Saturday, July 5, 2025 - 3:35 PM CEST (UX & Cosmetic Refinements) Objective: Polish the user interface and improve the overall user experience. Key Files Modified: index.html, script.js. Frontend Changes: index.html: The CSS <style> block was completely overhauled with a new, more modern design system (CSS variables, refined color palette, improved typography and spacing) for a "dashboard-like" feel. The "Live API Forecast" section was moved above the "Master Controls" box. The container for property information was changed to an empty div with a button: <div id="hotel-info-container"><button onclick="fetchHotelDetails()">...</button>...</div>. script.js: The DOMContentLoaded event listener that automatically called fetchHotelDetails() was removed. The function is now only triggered by the new button's onclick event. A new block was added to the DOMContentLoaded listener to programmatically set the master-start-date to today and master-end-date to one month in the future. The DATASET\_7\_MAP constant was updated to rename the occupancy metric's name property from "Occupancy (Direct)" to "Occupancy".**

**Entry: Saturday, July 5, 2025 - 8:27 PM CEST (Vercel Deployment Fix) Objective: Diagnose and resolve a critical deployment failure on Vercel where the application was non-functional despite working locally. Problem Summary: The deployed application on Vercel was serving a 404: NOT\_FOUND error for the root page and all static assets (e.g., script.js, dashboard.html). This resulted in a non-interactive page where JavaScript-driven features, such as the date pickers and data loading buttons, did not work. The root cause was a vercel.json configuration that did not correctly instruct the Vercel build system on how to handle the static frontend files. Debugging & Resolution: Initial analysis of the vercel.json file suggested the routing rules were incorrectly sending all traffic to the server.js backend, which does not have access to the static files in a serverless environment. After an initial attempt to fix the routes failed, a detailed review of the Vercel Build Logs revealed the true issue: Vercel was incorrectly compiling frontend files (like script.js) as if they were backend functions, instead of deploying them as static content. The key log line was Compiling "script.js" from ESM to CommonJS.... The final, successful fix involved adding an explicit build rule to vercel.json: { "src": "public/\*\*", "use": "@vercel/static" }. This rule forces the Vercel build system to recognize the /public directory's contents as static assets and deploy them correctly, resolving the 404 errors. Key Files Modified: vercel.json**

**Entry: Saturday, July 5, 2025 - 10:43 PM CEST (Dashboard UI/UX Overhaul) Objective: Transition from the proof-of-concept index.html to a professional, interactive, and modern dashboard UI using dashboard.html as the new foundation. Key Files Modified: dashboard.html (renamed to index.html), script.js (new version created as dashboard.js), index\_old.html (backup created). Feature 1: New Visual Design A clean, light-themed design was implemented with a professional color palette and typography (Inter for UI, Manrope for data). The layout was updated to a two-panel system with a persistent sidebar. Feature 2: Interactive KPI Cards The main KPI cards (Occupancy, ADR, RevPAR) were made interactive. Clicking a KPI card now updates the charts and tables below to display data for the selected metric. The active KPI card is highlighted with a distinct visual style to provide clear user feedback. Feature 3: Advanced Charting Synchronized Tooltips: Hovering over a data point on one chart now simultaneously displays the corresponding tooltip on the other chart, allowing for direct visual comparison. Conditional Shading: The area between the "Your Hotel" and "The Market" lines on the comparison chart is now subtly shaded green when your performance is higher and red when it is lower. Dynamic Y-Axis: The Y-axis for the Occupancy chart is now fixed from 0-100% for clarity, while other metrics use an auto-scaling axis. Feature 4: Table Enhancements Synchronized Highlighting: Hovering over a row in one table now highlights the corresponding row in the other table. Dynamic Delta Column: A "Delta" column was added to the market table, which dynamically calculates and displays the difference between your hotel and the market for the currently active KPI. Bug Fixes: Fixed a bug where RevPAR was always displaying as 0 by dynamically calculating it in the backend API. Corrected a data scaling issue where the Occupancy chart line was not displaying correctly.**

**Entry: Sunday, July 6, 2025 - Morning Session Objective: Debug a failing Vercel cron job and refactor its core logic to meet new business requirements for daily forecast refreshing. Key Files Modified: daily-refresh.js, package.json, vercel.json.**

* **\*\*1. Problem Summary\*\***
* **The Vercel cron job configured to run `daily-refresh.js` was failing to execute. Once triggered manually, it produced a cascade of errors, starting with module system conflicts, followed by multiple API payload errors, and finally a database write error. Furthermore, the script's fundamental purpose was misaligned with the project's goal; it was only updating yesterday's data, not the future forecast.**
* **\*\*2. Debugging & Resolution Narrative\*\***
* **The resolution process was iterative, tackling each layer of the application stack from the runtime environment down to the data processing logic.**
* **\*\*Part 1: Resolving the Module System Conflict (`ERR\_REQUIRE\_ESM`)\*\***
* **\* \*\*Problem:\*\* The initial failure was an `ERR\_REQUIRE\_ESM` error in the Vercel logs.**
* **\* \*\*Diagnosis:\*\* The script was written using ES Module (`import`) syntax for the `node-fetch` package, but the project's Vercel environment defaults to the CommonJS module system.**
* **\* \*\*Solution:\*\***
* **\* The `node-fetch` dependency in `package.json` was downgraded from v3+ (ESM-only) to a CommonJS-compatible version (`^2.7.0`).**
* **\* All `import` statements in `daily-refresh.js` were converted to `require()`.**
* **\* The `export default` statement was changed to `module.exports` to make the file fully CommonJS compliant.**
* **\*\*Part 2: Correcting API Payload Errors (`400 Bad Request`)\*\***
* **\* \*\*Problem:\*\* Once the script was running, the Cloudbeds API consistently rejected the request with `400 Bad Request` errors.**
* **\* \*\*Diagnosis & Solution:\*\* By inspecting the specific error messages from the API, several issues in the request payload were fixed sequentially:**
* **\* \*\*Missing `property\_ids`:\*\* The required key was added to the payload.**
* **\* \*\*Incorrect Operator:\*\* The filter operator was corrected from `"equal"` to `"equals"`.**
* **\* \*\*Invalid Column Name:\*\* The requested column `"occupancy\_direct"` was corrected to `"occupancy"`, aligning it with the project's data map.**
* **\* \*\*Missing `group\_rows`:\*\* The mandatory key was added, which is required by the API when fetching the `occupancy` metric.**
* **\*\*Part 3: Fixing the Database Write Error (`Invalid Input Syntax`)\*\***
* **\* \*\*Problem:\*\* After the API call succeeded, the script failed at the final step with a PostgreSQL error: `invalid input syntax for type numeric: '-'`.**
* **\* \*\*Diagnosis:\*\* The Cloudbeds API was returning a hyphen (`-`) for certain metrics that could not be calculated. The script was attempting to insert this non-numeric string into a numeric database column.**
* **\* \*\*Solution:\*\* A `sanitizeMetric` helper function was added to `daily-refresh.js`. This function ensures that any value received from the API is parsed, and if it's not a valid number, it defaults to `0` before being sent to the database.**
* **\*\*Part 4: Refactoring the Core Forecast Logic\*\***
* **\* \*\*Problem:\*\* It was identified that the script's logic was fundamentally misaligned with the business need. It was only updating yesterday's data, while the requirement was to refresh the full 365-day future forecast daily.**
* **\* \*\*Solution:\*\* The script was completely refactored.**
* **\* The date logic was changed to calculate a \*\*rolling 365-day window\*\* from the current day forward.**
* **\* A new `aggregateForecastData` function was implemented, adapting the robust aggregation logic from `initial-sync.js`. This new function correctly handles multi-day bookings by summing data points for each day instead of overwriting them, fixing the final data integrity bug.**
* **\* The database operation was placed inside a `for` loop to process each of the 365 days returned by the API, performing an `UPDATE` for each day.**
* **\*\*Part 5: Final Timezone Configuration\*\***
* **\* \*\*Problem:\*\* The cron schedule in `vercel.json` was not set to the desired local time (6 AM Poland).**
* **\* \*\*Solution:\*\* The schedule was updated to `"0 4 \* \* \*"`, which corresponds to 4:00 AM UTC, ensuring the job runs at the correct local time.**