Note for AI Assistants: This document is the project's single source of truth. It must be maintained with the following rules: 1. All new changelog entries must be technical, detailed, and added chronologically just above the V2.0 Roadmap section. 2. The V2.0 Roadmap section must always remain at the very bottom of the document. 3. NEVER edit or remove previous entries to ensure a complete historical log.

### **Project 'Market Pulse' - Changelog**

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

**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: Monday, July 7, 2025 - 9:55 PM CEST* **Objective:** Finalize the investigation into the OAuth invalid\_scope error and stabilize the V2.0 authentication flow based on the findings.

* **Key Files Modified:** server.js
* **Investigation Summary & Root Cause Analysis:**
  + A definitive, step-by-step isolation test was conducted to identify the exact cause of the persistent invalid\_scope error returned by the Cloudbeds API. The test confirmed the following:
    - Requesting only read:user -> Success.
    - Requesting read:user and read:hotel -> Success.
    - Adding read:data-insights-reservations to the scope -> Immediate Failure.
    - Adding read:data-insights-occupancy to the scope -> Immediate Failure.
  + This testing provides conclusive evidence that the V2.0 authentication failure is caused exclusively by the data-insights scopes.
* **Final Conclusion: External Blocker Identified**
  + The root cause has been identified as an external issue with the Cloudbeds platform. While the Cloudbeds developer portal shows the read:data-insights-reservations and read:data-insights-occupancy permissions as enabled for the application, their authorization server does not recognize them and rejects the request before user consent. This discrepancy is the sole blocker preventing the completion of the V2.0 multi-tenant functionality.
* **Current Stable Status & Next Steps:**
  + To create a stable baseline, the application's OAuth flow has been reverted to request only the scopes that are confirmed to work: read:user and read:hotel. In this state, the entire authentication and onboarding flow is successful: a user can authorize the app, their details are fetched from the /userinfo and /me endpoints, a new record is created in the users table, and they are correctly redirected to the dashboard.
  + All V2.0 development is paused pending resolution of this external dependency. It might be that these scopes get only enabled for apps that pass certification, pending clarification

*Entry: Tuesday, July 8, 2025 - 2:16 PM CEST* **Objective:** Complete the full V2.0 multi-tenancy backend refactor, transitioning the application from a single-account tool to a robust, user-aware platform.

* **Summary of V2.0 Migration Actions:**
  + **OAuth Onboarding Flow:** A complete, end-to-end OAuth 2.0 user onboarding flow was successfully implemented. This involved a deep debugging session to resolve API endpoint issues, culminating in the discovery of the correct /datainsights/v1.1/me/properties endpoint for fetching a user's property\_id.
  + **Database Schema Refactor (Phase 1):**
    1. The users table was updated with all necessary columns (first\_name, email, property\_id, access\_token, etc.) to store complete user profiles.
    2. The daily\_metrics\_snapshots table was modified by adding a cloudbeds\_user\_id column and updating its primary key to be a composite of (hotel\_id, stay\_date, cloudbeds\_user\_id). This was a critical step to ensure data integrity for multiple users.
  + **Backend Cron Job Refactor (Phase 2):**
    1. Both background sync scripts (daily-refresh.js and initial-sync.js) were fundamentally refactored. They now loop through every active user in the users table and perform data synchronization using each user's specific credentials and property ID, rather than using a single master API key.
  + **API Server Refactor (Phase 3):**
    1. The express-session library was installed and configured in server.js to enable session management.
    2. An isAuthenticated middleware function was created and applied to all dashboard-related API endpoints, securing them from unauthorized access.
    3. All protected API endpoints were updated to be fully "user-aware," using the userId from the active session to query the database and return data specific to the logged-in user.
* **Current Status:**
  + The V2.0 backend refactoring is **complete**. The application is now a functional multi-tenant platform, capable of securely handling multiple users, each with their own data. The core architectural goals of the V2.0 roadmap have been met.
* **Immediate Next Steps:**
  + The immediate priority is to build the user-facing login system for **returning users**. This involves:
    1. **Creating a Login Page:** A new login.html page with a simple email form for returning users, alongside the existing "Connect to Cloudbeds" button for new users.
    2. **Building a Login API Endpoint:** A new /api/login endpoint in server.js that will find a user by email in the database and create a session for them.
    3. **Implementing Frontend Login Logic:** A new login.js script to handle the form submission, call the new API endpoint, and redirect the user to the dashboard on success.

### **V2.0 Roadmap & Feature Ideas**

*This section outlines the strategic direction and planned features for the next major version of the Market Pulse application. This section should always remain at the bottom of the changelog.*

* **Major Feature: Advanced Reporting Module**
  + **Objective:** Build a dedicated "Reports" section to provide users with powerful, customizable, and exportable data views.
  + **Key Tasks (in manageable chunks):**
    - **Phase 1 (Report Builder UI):** Create a new page for the reporting module. Design a UI with date pickers, granularity controls, and a series of tickboxes that allow users to select which data columns to include in their report (e.g., "Market ADR," "ADR Delta").
    - **Phase 2 (Configurable API):** Create a new, highly configurable backend endpoint (e.g., /api/generate-report) that accepts the user's selections and dynamically builds a complex SQL query to generate the requested data.
    - **Phase 3 (Advanced Analytics):** Implement the logic for advanced calculations, starting with "Market Revenue Adjusted for Hotel Size." This will involve creating new backend functions to normalize market data against the user's specific hotel capacity.
    - **Phase 4 (Exporting):** Add functionality to export the generated reports. This should be done in stages, starting with the simplest format.
      * **Chunk 4a (CSV):** Implement a server-side function to convert the JSON data to CSV format for download.
      * **Chunk 4b (PDF/Excel):** Investigate and integrate server-side libraries (e.g., pdf-lib, exceljs) to enable PDF and .xlsx exports.
    - **Phase 5 (Report Scheduler):** Implement a system for users to schedule recurring reports. This is a major task that will require its own database tables to store schedules and a robust background job processor to run the reports and email them to users.
* **User Experience & Onboarding**
  + **Objective:** Improve the first-time user experience to increase adoption and reduce support requests.
  + **Key Tasks:**
    - **Guided Product Tour:** Implement a "first-login" guided tour that uses spotlights and tooltips to walk a new user through the key features of the dashboard.
    - **User Profile & Settings:** Create a page where users can manage their account settings.
* **Access Control & Permissions**
  + **Objective:** Create different permission levels within the application to support both hotel staff and internal administrators.
  + **Key Tasks:**
    - **Implement Role-Based Access Control (RBAC):** Add a role column to the new users table (e.g., 'admin', 'user').
    - **Superadmin View:** The "Admin Panel" link in the sidebar should only be visible to users with the 'admin' role. The backend must also protect all admin API endpoints to ensure they can only be accessed by authenticated admin users.