Product Specification: Lightweight Dashboarding Tool

Overview

A lightweight, AI-enhanced alternative to Tableau for creating and maintaining data visualizations and dashboards. Built on the Story Analytics framework, this tool emphasizes ease of use, automated maintenance, and flexible deployment options.

Product Goals

- Replace heavy enterprise dashboarding tools with a streamlined, user-friendly alternative
- Automate dashboard maintenance through AI-powered quality monitoring
- Support both cloud and local deployment scenarios
- Enable non-technical users to create publication-ready visualizations

Target Users

Data analysts, economists, and business intelligence professionals who:

- Have basic SQL knowledge or clean CSV files ready to visualize
- Need to create and maintain multiple dashboards without constant manual oversight
- Want DataWrapper-quality visualizations with more flexibility and automation
- Require either cloud-based or local deployment options

Core User Flow

Step 1: Data Source Connection

Inputs:

- CSV file upload, OR
- Snowflake database connection credentials + pre-written SQL query

Assumptions for v1:

- Users arrive with clean, ready-to-use data

- SQL queries are pre-written by users (AI SQL editor deferred to future version)
- CSV files are properly formatted

Technical Requirements:

- Support Snowflake database connections
- Support local DuckDB for CSV processing (automatically create DuckDB instance from uploaded CSV)
- Connection validation and error handling

Step 2: Data Preview & Validation

Interface Components:

- Elegant table preview showing column headers and first 10 rows
- Column data type selector for each column
- "Edit SQL" or "Adjust Data" buttons to return to Step 1
- Data validation indicators

Technical Requirements:

- Execute SQL query against connected database
- Parse and display CSV data
- Allow manual data type assignment/override
- Validate data structure and types

Step 3: Chart Type Selection & Mapping

Supported Chart Types:

- Horizontal bar chart
- Vertical bar chart
- Column chart
- Line chart
- Area chart
- Scatter plot
- Stacked bar chart
- Pie chart

Interface Components:

- Chart type selection menu
- X-axis column selector
- Y-axis column selector
- Styling preference selector (user-defined brand style OR default DataWrapper-inspired style)

Technical Requirements:

- Apply selected styling template
- Validate column compatibility with chart type
- Generate initial chart preview

Step 4: Chart Customization & Preview

Layout:

- **Center pane:** Live chart preview
- **Left pane:** Toolbox menu with chart-specific options
- **Right pane:** AI chat interface (Cursor-style)

Toolbox Options (Chart-Specific):

Axis Controls:

- X-axis min/max values
- Y-axis min/max values
- Axis labels and formatting

Chart Enhancements:

- Smoothing options (for applicable charts)
- Trend/regression line toggle
- Data point annotations
- Range highlighting

Metadata & Context:

- Chart title
- Chart subtitle
- Source note with optional link
- Data notes/disclaimers

AI Chat Capabilities:

- Natural language chart adjustments
- Styling suggestions
- Error troubleshooting

Technical Requirements:

- Real-time chart rendering
- Dynamic option panels based on chart type

- AI integration for conversational editing

Step 5: Save, Publish & Schedule

Publishing Outputs:

- Permanent shareable URL
- Embedding code/link
- Direct link to underlying SQL/data source

Scheduling Options:

- Manual refresh
- Scheduled refresh (configurable interval)
- SQL re-execution or CSV reload based on source type

Logging & Monitoring:

- Comprehensive execution logs
- Error capture and categorization
- Performance metrics (execution time, data freshness)
- Anomaly detection

Technical Requirements:

- Persistent storage for chart configurations
- Scheduled job management system
- Structured logging framework
- URL routing and embedding infrastructure

Step 6: Dashboard Assembly

Dashboard Structure:

- Two-column responsive layout
- 2-6 charts per dashboard (recommended)
- Header with title and contextual description

Dashboard Features:

- Shareable URL
- Social sharing buttons
- Email scheduling (send dashboard to recipients on regular intervals)
- Individual chart editing links

Technical Requirements:

- Dashboard composition engine
- Email delivery system with scheduling
- Responsive layout rendering
- Dashboard-level permissions management

AI-Powered Chart Hygiene System

Purpose

Automated quality assurance and proactive error resolution to minimize manual dashboard maintenance.

Core Capabilities

1. Error Detection:

- Daily (or configurable interval) log scanning
- Error categorization and prioritization
- User alerting system

2. Proactive Error Resolution:

- Automatic schema change detection (e.g., renamed columns)
- SQL query adjustment and resubmission
- Retry logic for transient failures

3. Data Freshness Monitoring:

- Compare data timestamps against expected refresh schedules
- Detect delayed or stale data
- Alert users to upstream data pipeline issues

4. Visual Quality Checks:

- Screenshot-based chart validation using AI vision models
- Detect rendering errors, blank charts, or data anomalies
- Compare against expected visual patterns

Technical Requirements

- Scheduled job runner for hygiene checks
- AI model integration (vision + reasoning)
- Log parsing and analysis engine
- Automated remediation workflow
- Alert/notification system

Technical Architecture

Deployment Models

Local Mode:

- Runs entirely on user's machine
- DuckDB for local data processing
- No cloud dependencies for data storage

Cloud Mode:

- Hosted service with centralized chart/dashboard management
- Supports Snowflake and cloud-based databases
- Shared dashboards with team collaboration features

Database Support

Primary:

- Snowflake (cloud databases)
- DuckDB (local databases, CSV processing)

Future Consideration:

- PostgreSQL, MySQL, BigQuery, Redshift

AI Model Flexibility

Supported Providers:

- OpenAI (GPT-4, etc.)
- Anthropic (Claude)
- Google Gemini
- Open-source local models (e.g., Llama, Mistral) for privacy-sensitive deployments

Requirements:

- Provider-agnostic abstraction layer
- User-configurable model selection
- Local model support for air-gapped environments

Technology Stack Considerations

Frontend:

- Modern JavaScript framework (React, Vue, or similar)
- Chart rendering library (D3.js, Plotly, or similar)
- Responsive design framework

Backend:

- Python-based (building on Story Analytics foundation)
- SQL execution engine
- Job scheduling system
- API layer for AI integrations

Storage:

- Chart/dashboard configuration storage
- Log storage and retrieval
- User preferences and branding templates

Non-Functional Requirements

Performance

- Chart preview generation: < 3 seconds
- Dashboard loading: < 5 seconds
- SQL query timeout: configurable, default 30 seconds

Security

- Secure credential storage for database connections
- Role-based access control for dashboards
- Audit logging for all chart/dashboard modifications

Scalability

- Support 100+ charts per user
- Handle datasets up to 1M rows for preview
- Concurrent dashboard rendering

Privacy

- Option to run fully locally without internet connection
- No data transmission to cloud services when using local models
- Configurable data retention policies

Future Enhancements (Post-v1)

- AI-powered SQL query builder
- Collaborative editing and commenting
- Version control for charts and dashboards
- Advanced chart types (heatmaps, geographic maps, network diagrams)
- API for programmatic chart creation
- Mobile-responsive dashboard viewing
- Export to PowerPoint/PDF
- Data source management and cataloging