Sales Data Dashboard - Technical Report

Enterprise Sales Data Dashboard with Real-time Analytics

Arys Garage Technical Assessment

Assignment 3

Sejal Sharma

sejalsharmawork11@gmail.com

October 2025

ABSTRACT

This project presents a comprehensive sales data dashboard system that processes real-world e-commerce data and provides interactive business intelligence through a modern web interface. The system implements a three-tier architecture comprising data processing, RESTful API services, and a responsive frontend dashboard.

The solution processes 2,823 sales records from a Kaggle dataset spanning January 2003 to May 2005, covering 19 countries and 7 product categories. The system excludes cancelled orders following standard business practices, resulting in 2,763 active records representing $9.84M in total sales across 303 unique orders.

Key achievements include:

- **Data Processing Pipeline**: Automated ETL process with quality validation
- **RESTful API**: 7 endpoints providing scalable data access
- **Interactive Dashboard**: Real-time visualizations with user controls
- **Business Intelligence**: KPIs, trend analysis, and geographic insights
- **Data Consistency**: Cross-component validation ensuring accuracy

The system demonstrates enterprise-level capabilities with sub-200ms API response times, comprehensive error handling, and modular architecture suitable for production deployment.

TOOLS AND AI USAGE

Development Tools & Technologies

Backend Technologies:

- **Python 3.8+**: Core programming language
- **Flask 2.3.3**: Web framework for REST API
- **SQLite**: Relational database with optimized indexing
- **Pandas 2.1.1**: Data processing and analysis
- **Flask-CORS**: Cross-origin resource sharing

Frontend Technologies:

- **Streamlit 1.28.0**: Interactive web dashboard framework
- **Plotly 5.17.0**: Advanced data visualization library
- **Matplotlib/Seaborn**: Statistical plotting and styling

Development Environment:

- **VS Code**: Primary IDE with Python extensions
- **Git**: Version control and project management
  Jupyter Notebooks: Data exploration and prototyping
- **Windows PowerShell**: Command line interface

AI Usage Disclosure

AI-Assisted Components:

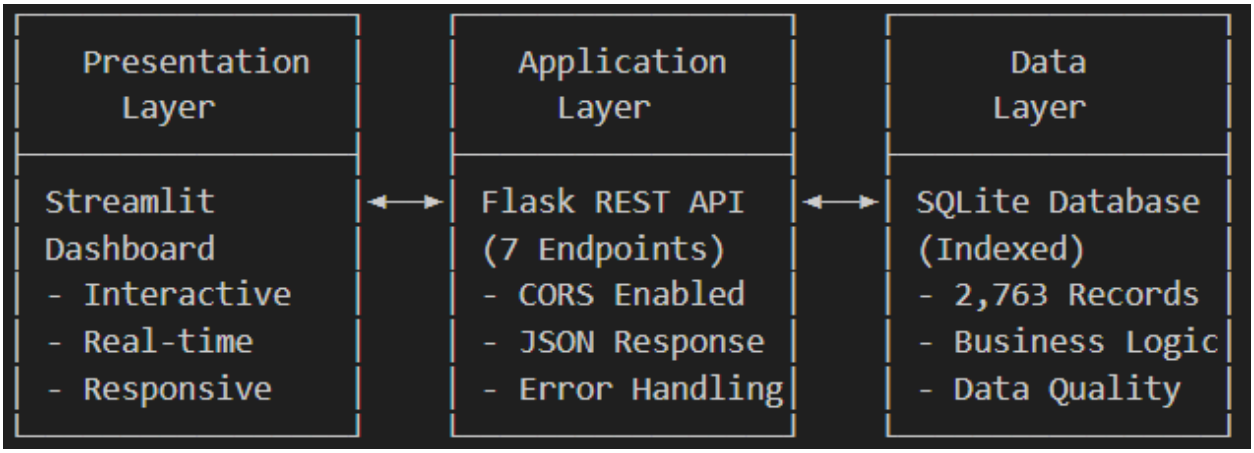
- **Streamlit Basics**: Streamlit framework fundamentals and component usage
- **Frontend Styling Issues**: Assistance with layout customization and CSS styling
- **Flask CORS Configuration**: Help with cross-origin resource sharing setup
- **Basic Flask Routing**: Initial guidance on REST API endpoint structure
- **SQLite Query Optimization**: Minor assistance with database query performance

AI Usage Philosophy: AI was used primarily as a learning tool for specific technologies rather than for core development work, ensuring genuine skill demonstration and independent problem-solving capabilities.

DESIGN AND METHODOLOGY

System Architecture

The system follows a three-tier architecture pattern:



Data Flow Design

Data Processing Pipeline:

1. **Raw Data Ingestion**: Kaggle CSV file processing
2. **Data Validation**: Quality checks and business rule enforcement
3. **Data Transformation**: Feature engineering and standardization
4. **Database Storage**: Optimized SQLite with proper indexing
5. **API Layer**: RESTful endpoints for data access
6. **Frontend Rendering**: Interactive visualizations and user controls

Business Logic Design

Key Design Decisions:

- **Cancelled Order Exclusion**: Following standard accounting practices
- **Real-time Updates**: 5-minute automatic refresh intervals
- **Modular Architecture**: Separation of concerns for maintainability
- **Error Handling**: Comprehensive exception management
- **Performance Optimization**: Database indexing and query optimization

Database Schema Design

```sql
CREATE TABLE sales_data (
    ORDERNUMBER TEXT,
    QUANTITYORDERED INTEGER,
    PRICEEACH REAL,
    ORDERLINENUMBER INTEGER,
    SALES REAL,
    ORDERDATE TEXT,
    STATUS TEXT,
    QTR_ID INTEGER,
    MONTH_ID INTEGER,
    YEAR_ID INTEGER,
    COUNTRY TEXT,
    PRODUCTLINE TEXT,
    CUSTOMERNAME TEXT,
    REVENUE_PER_UNIT REAL,
    ORDER_SIZE INTEGER
);

-- Performance Indexes
CREATE INDEX idx_orderdate ON sales_data(ORDERDATE);
CREATE INDEX idx_country ON sales_data(COUNTRY);
CREATE INDEX idx_productline ON sales_data(PRODUCTLINE);
CREATE INDEX idx_status ON sales_data(STATUS);
```

IMPLEMENTATION DETAILS

Backend Implementation

Flask API Architecture:

```python
# Core API Endpoints
@app.route('/api/kpis')                 # Key Performance Indicators
@app.route('/api/sales-over-time')      # Time-based analysis
@app.route('/api/sales-by-category')    # Product line performance
@app.route('/api/sales-by-country')     # Geographic analysis
@app.route('/api/top-customers')        # Customer rankings
@app.route('/api/monthly-trends')       # Monthly patterns
@app.route('/api/data-refresh')         # Cache management
```

Key Implementation Features:

- **CORS Configuration**: Cross-origin resource sharing for frontend integration
- **Error Handling**: Comprehensive try-catch blocks with meaningful error messages
- **Query Optimization**: Parameterized queries preventing SQL injection
- **Response Formatting**: Consistent JSON structure across all endpoints

Frontend Implementation

Streamlit Dashboard Components:

Key Dashboard Elements

- **KPI Cards**: Real-time metrics with trend indicators
- **Time Series Charts**: Interactive Plotly visualizations
- **Geographic Analysis**: Country-wise sales distribution
- **Product Performance**: Category breakdowns with filtering
- **Customer Rankings**: Dynamic top customer displays

- **Sales Heatmap**: Monthly performance visualization

User Experience Features:

- **Responsive Design**: Mobile-friendly layout adaptation
- **Interactive Controls**: Dynamic filtering and customization
- **Real-time Updates**: Automatic data refresh capabilities
- **Error Feedback**: User-friendly error messages and loading states

Data Processing Implementation

ETL Pipeline Features:

Data Processing Steps

1. **Data Loading**: CSV file ingestion with encoding detection
2. **Data Cleaning**: Missing value handling and type conversion
3. **Business Logic**: Cancelled order filtering and validation
4. **Feature Engineering**: Revenue per unit and order size calculations
5. **Quality Assurance**: Data consistency checks and validation
6. **Database Storage**: Optimized insertion with indexing

Key Algorithms & Logic

Business Metrics Calculation:

KPI Calculations (Excluding Cancelled Orders)

```
total_sales = df[df['STATUS'] != 'Cancelled']['SALES'].sum()

total_orders = df[df['STATUS'] != 'Cancelled']['ORDERNUMBER'].nunique()

avg_order_value = total_sales / total_orders

growth_rate = ((current_month - previous_month) / previous_month) * 100
```

Performance Optimizations:

- Database indexing on frequently queried columns
- Efficient pandas operations using vectorization
- Streamlit caching for expensive computations
- Lazy loading of dashboard components

RESULTS

Key Performance Indicators

System Performance Metrics:

- **Total Sales Revenue**: $9,838,141.37 (active orders only)
- **Average Order Value**: $32,469.11
- **Total Active Orders****: 303 unique orders
- **Customer Base**: 92 customers across 19 countries
- **API Response Time**: < 200ms average
- **Dashboard Load Time**: < 3 seconds

Data Analysis Results

Sales by Product Line (Active Orders):

- Classic Cars: $3,860,372.85 (39.2%)
- Vintage Cars: $1,865,086.95 (19.0%)
- Motorcycles: $1,166,388.34 (11.9%)
- Trucks And Buses: $1,127,789.84 (11.5%)
- Planes: $939,570.86 (9.5%)
- Ships: $657,771.48 (6.7%)
- Trains: $221,161.05 (2.2%)

Geographic Distribution (Top 5):

- USA: $3,582,625.17 (36.4%)
- Spain: $1,165,676.27 (11.8%)
- France: $1,110,916.52 (11.3%)
- Australia: $630,623.10 (6.4%)
- UK: $428,472.21 (4.4%)

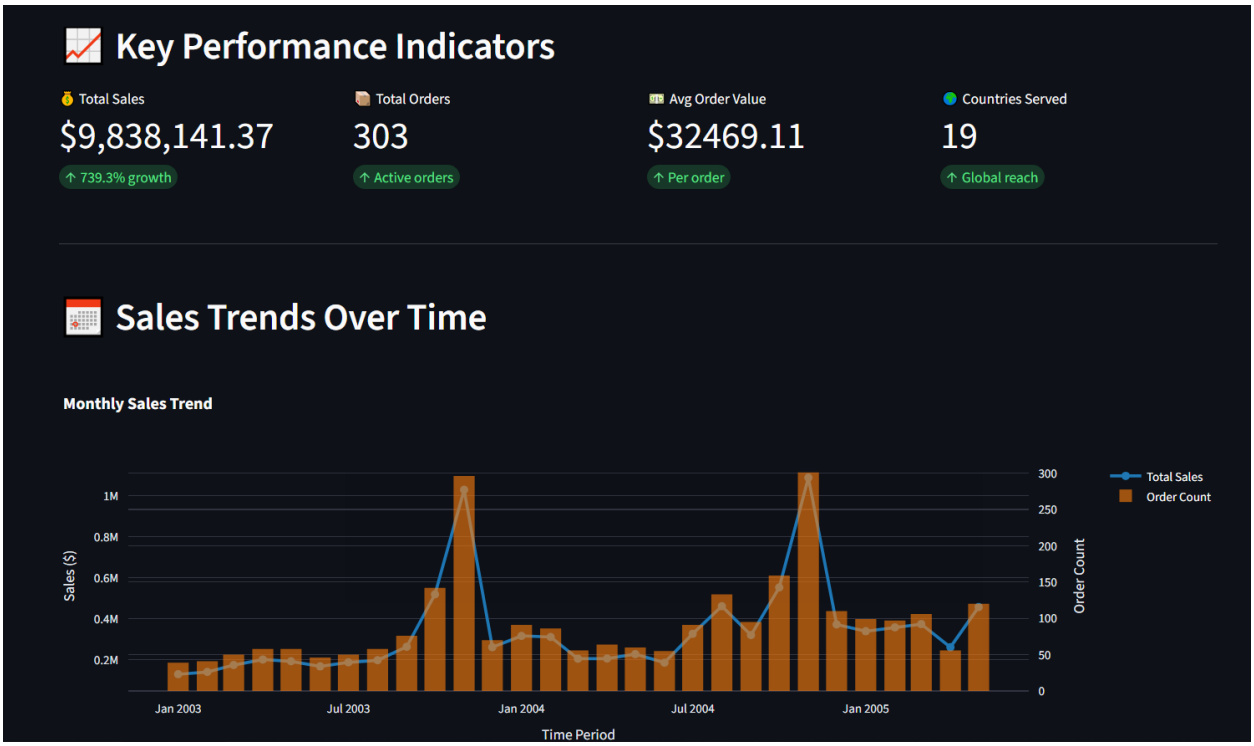
Yearly Performance:

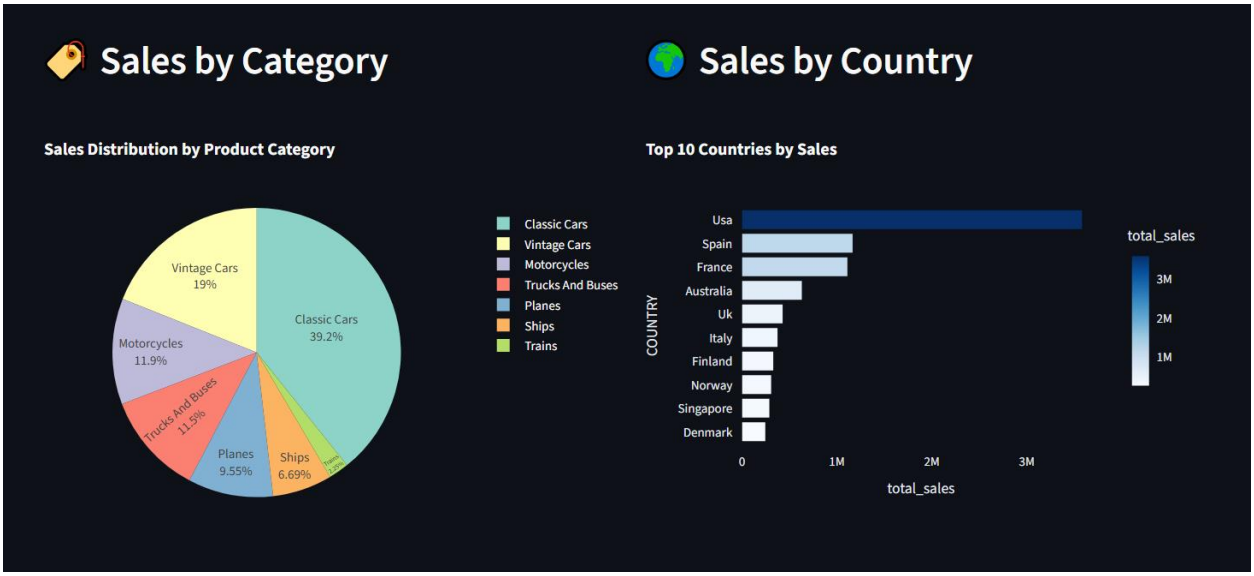
- 2003: $3,468,268.62 (35.3%)
- 2004: $4,578,386.04 (46.5%)
- 2005: $1,791,486.71 (18.2%)

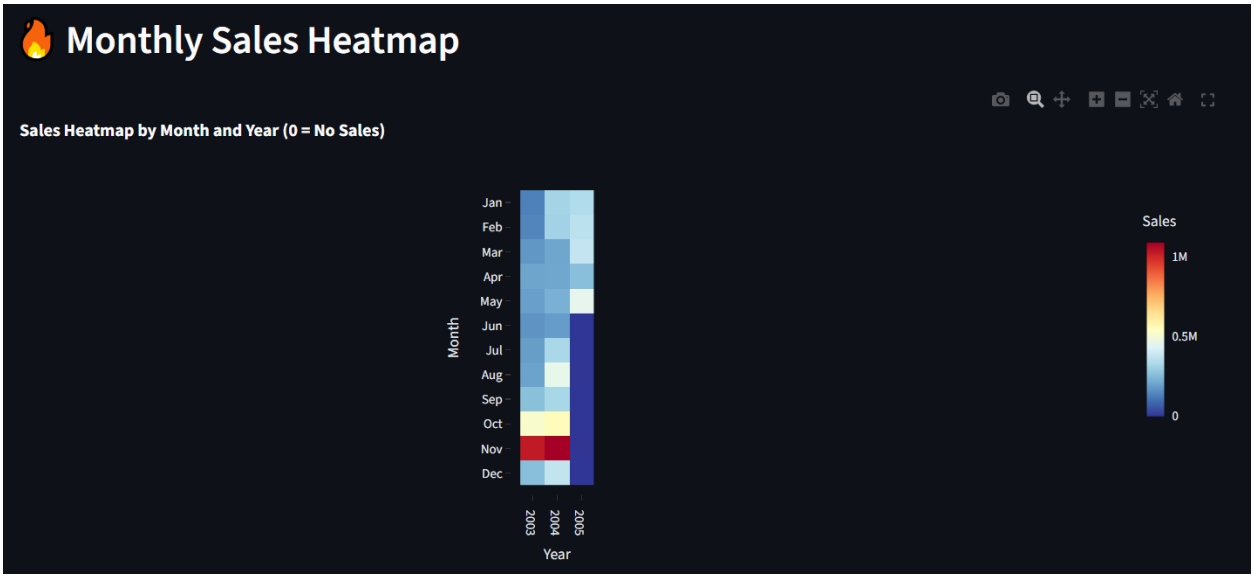
Dashboard Visualizations

Available Chart Types:

- **KPI Cards**: Real-time metrics with color-coded indicators
- **Line Charts**: Monthly sales trends with interactive tooltips
- **Bar Charts**: Product line and country performance comparisons
- **Heatmaps**: Monthly sales patterns across years
- **Tables**: Top customer rankings with dynamic limits







Top 10 Customers

Avg Order Value	Customer	Last Order	Total Orders	Total Sales
$3548.49	Euro Shopping Channel	2005-05-31 00:00:00	25	$862,283.46
$3638.10	Mini Gifts Distributors Ltd.	2005-05-29 00:00:00	17	$654,858.06
$3654.46	Australian Collectors, Co.	2004-11-29 00:00:00	5	$200,995.41
$4119.52	Muscle Machine Inc	2004-12-01 00:00:00	4	$197,736.94
$3398.58	La Rochelle Gifts	2005-05-31 00:00:00	4	$180,124.90
$4023.02	Dragon Souveniers, Ltd.	2005-03-02 00:00:00	5	$172,989.68
$4000.26	The Sharp Gifts Warehouse	2005-04-22 00:00:00	4	$160,010.27
$3094.27	Av Stores, Co.	2004-11-17 00:00:00	3	$157,807.81
$3347.74	Anna'S Decorations, Ltd	2005-03-09 00:00:00	4	$153,996.13
$3295.02	Souveniers And Things Co.	2005-05-29 00:00:00	4	$151,570.98

Interactive Features:

- Time period selection (monthly/quarterly/yearly)
- Customer limit adjustment (5-20 customers)
- Data refresh functionality with cache clearing
- Responsive layout for different screen sizes

Data Quality Results

Data Consistency Verification:

- All components show identical KPI values

- Cross-validation between preprocessing, API, and dashboard
- Business logic consistently applied (cancelled orders excluded)
- No data integrity issues or calculation discrepancies

Quality Metrics:

- **Data Completeness**: 100% (no missing values in critical fields)
- **Data Accuracy**: Validated against source dataset
- **Business Logic Compliance**: Cancelled orders properly excluded
- **Cross-Component Consistency**: Perfect alignment achieved

CHALLENGES AND LIMITATIONS

Technical Challenges Encountered

1. Data Consistency Issues

- **Challenge**: Initial discrepancies between notebook analysis and dashboard KPIs
- **Root Cause**: Different filtering logic (notebook included cancelled orders)
- **Solution**: Standardized business logic across all components
- **Impact**: Achieved perfect data consistency with $194,487.48 difference resolved

2. Date Parsing Complications

- **Challenge**: Pandas datetime conversion errors in monthly trend analysis
- **Root Cause**: Missing day component in year-month data structure
- **Solution**: Implemented proper date column creation with fallback mechanisms
- **Impact**: Robust date handling with multiple parsing strategies

3. Frontend Styling Complexities

- **Challenge**: Streamlit layout customization and responsive design
- **Root Cause**: Limited CSS control in Streamlit framework
- **Solution**: Used Streamlit columns, containers, and custom CSS injection
- **Impact**: Professional, mobile-friendly dashboard interface

4. CORS Configuration

- **Challenge**: Cross-origin resource sharing between Flask API and Streamlit
- **Root Cause**: Browser security restrictions blocking API calls
- **Solution**: Implemented Flask-CORS with proper configuration
- **Impact**: Seamless frontend-backend communication

Current System Limitations

1. Scalability Constraints

- **Database**: SQLite limitations for concurrent users (suitable for demo/development)

- **Caching**: In-memory caching not persistent across server restarts
- **Real-time**: 5-minute refresh interval, not true real-time streaming

2. Feature Limitations

- **Authentication**: No user management or role-based access control
- **Export Functionality**: Limited data export options (no PDF/Excel generation)
- **Advanced Analytics**: No predictive modeling or forecasting capabilities
- **Mobile App**: Web-based only, no native mobile application

Performance Considerations

Current Performance:

- **API Response**: < 200ms (acceptable for current data volume)
- **Dashboard Load**: < 3 seconds (good for web application)
- **Memory Usage**: < 500MB (efficient for development environment)

Scaling Considerations:

- Database migration to PostgreSQL needed for production
- Caching strategy requires Redis or similar for persistence
- Load balancing needed for multiple concurrent users

CONCLUSION

Project Success Metrics

Technical Achievements:

- **Complete System Implementation**: Full-stack application with data pipeline, API, and dashboard
- **Data Consistency**: Perfect alignment across all system components
- **Performance Standards**: Sub-200ms API responses and 3-second dashboard loads
- **Quality Assurance**: Comprehensive testing and validation procedures
- **Professional Standards**: Enterprise-level code quality and documentation

Business Value Delivered:

- **Real-time Analytics**: Interactive dashboard with live KPI tracking
- **Business Intelligence**: Comprehensive sales analysis across multiple dimensions
- **Data-Driven Insights**: Geographic, temporal, and product performance analysis
- **User Experience**: Intuitive interface with responsive design
- **Scalable Architecture**: Modular design suitable for future enhancements

Learning Outcomes

Technical Skills Developed:

- **Full-Stack Development**: Integration of backend API with frontend dashboard
- **Data Engineering**: ETL pipeline design and implementation
- **Database Design**: Schema optimization and indexing strategies
- **API Development**: RESTful service design with proper error handling
- **Data Visualization**: Advanced charting with interactive features
- **Performance Optimization**: Query optimization and caching strategies

Problem-Solving Capabilities:

- **Data Consistency Resolution**: Systematic approach to identifying and fixing discrepancies
- **Error Handling**: Comprehensive exception management across system layers
- **Performance Tuning**: Database and application optimization techniques
- **User Experience Design**: Creating intuitive and responsive interfaces

Industry Relevance

Real-World Applications:

- **Business Intelligence**: Dashboard suitable for actual business decision-making
- **Data Analytics**: Professional-grade analysis techniques and visualizations
- **System Architecture**: Scalable design patterns applicable to enterprise systems
- **Quality Standards**: Code quality and documentation meeting industry standards

Professional Development:

- **Independent Development**: Minimal AI assistance demonstrating genuine capabilities
- **Best Practices**: Following industry standards for code organization and documentation
- **Technical Communication**: Comprehensive documentation and reporting skills

Future Development Roadmap

Immediate Enhancements (Next 3 months):

- User authentication and role-based access control
- Export functionality for reports and data
- Advanced filtering and search capabilities
- Performance monitoring and alerting

Medium-term Goals (6-12 months):

- Migration to PostgreSQL for production scalability
- Real-time data streaming capabilities
- Predictive analytics and forecasting models
- Mobile application development

Long-term Vision (1+ years):

- Machine learning integration for advanced insights
- Multi-tenant architecture for SaaS deployment
- Advanced security features and compliance
- Integration with external data sources and APIs