Teodora Valova 201221003

Alesandro Dragnev 201221041

**Business Intelligence – Course Project**

**Overview**

This project is a tool designed to calculate basic business information. It allows users to view, manage, and analyze key business datasets such as Sales, Inventory, and EmployeePerformance, with real-time data manipulation and automatic chart updates.

**Features**

* Computation of fundamental business metrics.
* User-friendly interface for data input and result visualization.
* Modular code structure for ease of maintenance and scalability.

**Project Structure**

The repository comprises the following key components:

* **app.py**: The main application script that orchestrates the tool's functionality.
* **utils/**: Contains utility modules supporting various operations within the application.
* **data/**: Directory designated for input datasets and output results.
* **Install\_App.bat**: A batch script to facilitate the installation process on Windows systems.

**Usage**

Upon running the application, users can input relevant business data through the interface. The tool will process this data and output calculated business metrics, aiding in decision-making processes.

**app.py – Main Application Script**

This file is the central script that runs the Streamlit-based Business Intelligence Dashboard. It provides a simple GUI for interacting with business data: **Sales**, **Inventory**, and **Performance**.

**Technologies Used**

* **Programming Language**: Python
* **Streamlit** for UI
* **Pandas** for data handling
* **Plotly** for chart visualizations
* **Automation Scripts**: Batch scripting for Windows
* **Modular code** via utils for data and chart logic
* **Version Control**: Git

**Project Sections**

**1. Sales Dashboard**

**Features:**

* View all sales records.
* Add new sales entries using dropdowns and input fields.
* Delete existing sales entries.
* Calculate and display key metrics:
  + **Total Sales**
  + **Top Product by Sales Amount**
* Filter records by date range.
* Visualize:
  + Monthly sales trends (sales\_by\_month)
  + Product-wise sales distribution (product\_sales\_pie)

**Related Utility Functions**:

* load\_sales\_data() – loads data from data/sales\_data.csv.

**2. Inventory Overview**

**Features:**

* Display the current inventory dataset.
* Add a new product (name, SKU, stock, reorder point).
* Delete existing products.
* Visualize inventory level by product (inventory\_bar)
* Warning about low-stock items (stock below reorder point).

**Related Utility Functions**:

* load\_inventory\_data() – reads from data/inventory.csv.

**3. Performance Reports**

**Features:**

* Display employee performance metrics.
* Add new employee sales performance by month.
* Delete specific entries.
* Visualize sales by employees (performance\_bar)

**Related Utility Functions**:

* load\_performance\_data() – reads from data/performance.csv.

**Data Handling**

CSV files in the data/folder act as the database. All changes made through the UI — adding or deleting sales/products/entries — are written back to disk. The updated data is then immediately reloaded and reflected in the interface.

**Sample Interaction Flow**

1. User selects “Sales” from sidebar.
2. Enters a new sale via form → Data saved to sales\_data.csv
3. The table and charts update instantly.
4. User deletes a row from table → Data reloaded and re-rendered immediately.
5. Metrics like “Total Sales” and “Top Product” update accordingly.

**Visualizations**

These are dynamically created using Plotly:

* **sales\_by\_month(df)** – Line or bar chart of sales over time.
* **product\_sales\_pie(df)** – Pie chart of product-wise sales share.
* **inventory\_bar(df)** – Bar chart of stock levels.
* **performance\_bar(df)** – Bar chart of employee performance.

**Code Modularization**

* Data Loading:
  + utils/data\_loader.py contains functions like load\_sales\_data(), load\_inventory\_data(), etc.
* Chart Drawing:
  + utils/charts.py contains all chart generation functions.

**utils/charts.py – Chart Generation Module**

This module uses Plotly Express to generate interactive visualizations for the dashboard. Each function returns a chart object for rendering in Streamlit.

**sales\_by\_month(df)**

* Purpose: Displays monthly aggregated sales as a bar chart.
* Input: DataFrame with date and sales\_amount columns.
* Output: Bar chart showing total sales per month.
* Key Detail: Converts daily dates into YYYY-MM periods.

**product\_sales\_pie(df)**

* Purpose: Pie chart showing the distribution of total sales by product.
* Input: DataFrame with product and sales\_amount.
* Output: Pie chart

**inventory\_bar(df)**

* Purpose: Shows current inventory levels for each product.
* Input: DataFrame with product and stock.
* Output: Bar chart.

**performance\_bar(df)**

* Purpose: Visualize employee performance based on sales.
* Input: DataFrame with employee and sales.
* Output: Bar chart of employee sales.

**utils/data\_loader.py – Data Loading Module**

This module provides wrapper functions to load CSV data into pandas DataFrames for use in the dashboard.

**load\_sales\_data**(path="data/sales\_data.csv")

* Purpose: Loads sales records.
* Behavior: Parses date column as datetime.

**load\_inventory\_data**(path="data/inventory.csv")

* Purpose: Loads inventory dataset (stock levels, SKU, etc.).

**load\_performance\_data**(path="data/performance.csv")

* Purpose: Loads employee performance data by month.

**Summary of Modularization**

| **Purpose** | **Module** | **Key Functions** |
| --- | --- | --- |
| UI/Interaction | app.py | UI logic, chart rendering, data sync |
| Data Source Handling | data\_loader.py | load\_sales\_data(), load\_inventory\_data() |
| Chart generation | charts.py | sales\_by\_month(), inventory\_bar() |