Architecture Design Document- Amazon Project

- KAINAT NAQVI

1. Introduction

- **Purpose**:
    - The purpose of this document is to provide a comprehensive overview of the architecture behind the sales management dashboard built using Tableau Public. This architecture serves the critical goal of enabling businesses to gain insights into their sales data, improve distribution methods, reduce operational costs, and enhance overall profitability.
- **Problem Statement**:
    - Sales management has gained significance as businesses face increasing competition, driving the need for better methods of distribution. Effective sales management is essential to maintain profitability while minimizing distribution costs. Our solution involves using a data visualization tool to display key metrics and uncover meaningful relationships between sales factors. This document outlines the architecture behind the solution, which has been developed using Tableau Public and uses extract-based data sources to power real-time insights.

2. What is an Architecture Design Document?

- An **Architecture Design Document (ADD)** provides a detailed view of the system architecture that governs how various components of a solution interact. In the context of business intelligence and data visualization, this document outlines the technical architecture, including the infrastructure, software components, and communication flow. This ADD ensures that the system is scalable, secure, and meets the functional and non-functional requirements defined by the business.
- **Why It Is Important**:
    - The ADD helps stakeholders understand how different components interact, where the system's strengths lie, and what potential bottlenecks or areas of improvement exist. For this Tableau-based solution, the architecture plays a vital role in ensuring that the dashboard is responsive, data is updated efficiently, and users can interact with the visuals without delay.

3. Scope

- **Functional Scope**:
    - The architecture detailed in this document focuses on deploying a sales management dashboard that captures key performance indicators (KPIs) such as total revenue, total profit, and sales distribution across various regions and channels. The dashboard provides insight into the relationships between sales attributes and business performance.
- **Technical Scope**:
    - This document details the infrastructure used to support the Tableau Public dashboard, covering the data connections, Tableau's in-memory data engine, the architecture behind Tableau Public, and the communication flow between different Tableau components.
- **System Boundaries**:
    - The system boundary is restricted to the Tableau ecosystem, focusing on **Tableau Public** for hosting the dashboard and **Tableau Desktop** for designing it. Since the project uses **extract-based connections**, there are no live data connections or Tableau Server integration.

4. Architecture Overview

This section provides a high-level overview of the system architecture, explaining the interaction between various Tableau components and how the data flows through the system.

The architecture revolves around three main areas:

1. **Data Layer**:
    - The data comes from an external source provided by PWSkills, containing sales data that includes columns such as region, country, item type, order date, units sold, unit price, and total profit.
    - The data is preprocessed in Tableau Desktop and converted into an **extract** for efficient querying.
2. **Processing Layer**:
    - The Tableau Public infrastructure processes the data using **VizQL**, Tableau's core query engine, which translates user interactions (filtering, clicking, etc.) into optimized SQL queries.
    - The data is stored in Tableau's **in-memory engine (Hyper)**, ensuring quick retrieval and interaction.
3. **Presentation Layer**:
    - The Tableau Public dashboard, accessible through a public URL, allows users to interact with the data through various visualizations, such as KPIs, bar charts, maps, and sales trends.

- This layer delivers the processed visual output back to users, who can interact with filters and explore the data.

5. Tableau Architecture

Tableau operates on a highly scalable and distributed architecture designed for handling large-scale data visualizations. Below is a breakdown of the core components that play a role in this architecture:

1. **Tableau Desktop**:
   - This is the authoring and design environment where the dashboard was developed. Tableau Desktop is where the data was imported, processed, and transformed into a Tableau extract. The dashboard visuals and KPIs were designed here, leveraging drag-and-drop functionalities to create meaningful visualizations of the sales data.
2. **Tableau Public**:
   - Tableau Public is a free cloud-based platform that allows users to publish and share dashboards with the public. In this project, Tableau Public is used as the hosting platform, where the dashboard is published and made accessible to users via a public link.

6. Tableau Public Architecture

Tableau Public offers a streamlined, cloud-based architecture that allows dashboards to be hosted on Tableau's cloud infrastructure. This differs from the Tableau Server architecture, where the server infrastructure is managed by the client. Below are the main components of the Tableau Public architecture:

1. **User Interface Layer**:
   - This layer consists of the browser or device where the end users view the Tableau dashboard. The UI is interactive, allowing users to filter data, explore different regions or sales channels, and drill down into specific details.
2. **Extract Connection**:
   - Since the data is provided as an **extract** (not a live connection), the entire dataset is stored in Tableau's in-memory engine (Hyper) once it is uploaded. Extract-based connections are highly efficient for querying pre-loaded data.
3. **Storage Layer**:
   - The Tableau Public cloud infrastructure stores the published data and dashboards. When a new extract is uploaded or the dashboard is updated, it is reflected in the storage.
4. **Security**:

- Tableau Public uses Tableau's cloud security protocols to protect user data. However, because Tableau Public is meant for public consumption, all published data is available for anyone with the dashboard link.

7. Gateway/Load Balancer

- **Purpose**:
  - The **gateway** acts as a load balancer, distributing incoming user requests across Tableau Public's cloud infrastructure. It ensures that user queries are efficiently routed to the appropriate VizQL Server instance to handle visualization rendering and data querying.
- **Role in Tableau Public**:
  - When a user accesses the dashboard through the public link, the request is routed via the gateway, which distributes the load across multiple VizQL Server instances to balance the computational workload and ensure consistent performance.

8. VizQL Server

- **Purpose**:
  - **VizQL Server** is the backbone of Tableau's visualization query engine. It handles query generation, execution, and rendering of visualizations. Every time a user interacts with a filter or clicks on a visualization, VizQL Server generates the necessary SQL queries, retrieves the data, and translates it into a visual representation.
- **Role in Tableau Public**:
  - Tableau Public's VizQL Server manages user interactions on the published dashboard. It receives requests, processes the required data from the extract stored in the Hyper engine, and returns visual outputs to the user interface. The VizQL Server ensures that dashboard interactivity, such as filtering by region or drilling into sales channels, happens smoothly.

9. Data Engine (Hyper)

- **Purpose**:
  - Tableau's **Hyper** is an in-memory data engine designed for fast querying and processing of large datasets. It enables users to interact with their data in real-time, offering quick responses to filters, calculations, and other operations on large datasets.

- **Role in Tableau Public**:
    - Since the data for this project is stored as an extract, the Hyper engine plays a critical role in processing queries. When a user interacts with the dashboard, the Hyper engine retrieves data directly from memory, ensuring that queries are executed rapidly.
- **Key Feature**:
    - Hyper allows for high-performance querying even with large datasets, which is crucial for maintaining interactivity on the dashboard when users explore metrics such as revenue, cost, and units sold across regions.

10. Backgrounder

- **Purpose**:
    - **Backgrounder** in Tableau Server architecture is responsible for executing scheduled tasks like data extracts and refreshing data. However, in Tableau Public, the data refresh happens manually whenever a new extract is uploaded.
- **Role in Tableau Public**:
    - Since Tableau Public does not support automatic refreshes, the dashboard needs to be republished with a new extract to reflect updated data. In this architecture, Backgrounder is not directly involved, but the platform ensures stability when new data is uploaded.

11. Data Server

- **Purpose**:
    - The **Data Server** in Tableau manages the centralized storage of data extracts and ensures that data connections are secure and reliable. It allows multiple Tableau workbooks to share a common data source.
- **Role in Tableau Public**:
    - In Tableau Public, the data server ensures that the uploaded extract is stored securely and is available for querying through the VizQL server.

12. Tableau Communication Flow

The communication flow in Tableau Public is essential for ensuring that user interactions with the dashboard translate into accurate, timely responses. Here's how it works:

1. **User Interaction**:

    - Users interact with the dashboard by applying filters, drilling down into data, or selecting specific sales channels.
2. **Request Processing (Gateway)**:
    - User actions are processed by the **gateway**, which directs requests to the appropriate VizQL server.
3. **VizQL Query Generation**:
    - The **VizQL Server** translates the user's actions into SQL queries and sends them to the Hyper engine.
4. **Data Retrieval (Hyper)**:
    - The **Hyper engine** retrieves the relevant data from memory and returns the results to VizQL Server.
5. **Visualization Rendering**:
    - The VizQL Server processes the data and renders the updated visualizations on the user's browser, allowing them to view the filtered or detailed data.

13. Conclusion

The architecture behind the Tableau Public sales dashboard ensures that the system operates smoothly, efficiently handling user interactions with large datasets. By leveraging Tableau's robust infrastructure—VizQL Server, Hyper engine, and efficient communication layers—this architecture supports real-time analysis of sales data, providing businesses with actionable insights into their distribution, costs, and profitability.

This architecture design document serves as a foundational guide for understanding how the Tableau ecosystem functions in creating interactive, data-driven solutions for sales management.