High Level Design (HLD)

Analysing Google Apps Store

Revision Number: 1.0

Last Date of revision: 25/04/2025

Swaraj Ranjan Behera

Document Version Control

Date Issued	Version	Description	Author
26-04- 2025	1.0	First Version of Complete HLD	Swaraj Ranjan Behera

Contents

Abstract – Google Play Store Data Analysis Project

The Google Play Store Data Analysis project focuses on exploring and analyzing various mobile applications available on the Play Store. This analysis helps uncover trends related to app downloads, ratings, reviews, pricing strategies, categories, and user preferences. By leveraging data from real-world apps, this project provides meaningful insights that can assist developers, marketers, and business analysts in making informed decisions.

The dataset is processed using data cleaning and Exploratory Data Analysis (EDA) techniques in Python. Additionally, insights are visualized using tools like Power BI, enabling interactive dashboards and reports. This high-level analysis supports stakeholders in identifying top-performing apps, optimal pricing strategies, and content trends across the Google Play ecosystem.

1. Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to expand upon the initial project overview and provide a comprehensive design model for analyzing data from the Google Play Store. This document serves as a blueprint for development and implementation phases, helping identify possible contradictions or limitations early in the process. It also acts as a reference for how various modules will interact at a higher level throughout the analytics pipeline.

This HLD will:

- Present all key design components of the project and define them in detail.
- Describe the visual interface of the dashboards being implemented using Power BI.
- Outline the software tools and libraries used for data cleaning, transformation, and analysis (e.g., Python, Pandas, Matplotlib, Power BI).
- Highlight performance requirements for loading and processing large datasets.
- Include overall system architecture, including ETL workflows and visualization pipelines.
- List and explain the non-functional attributes such as:
    - **Security**: Handling data with integrity and responsible access.
    - **Reliability**: Ensuring consistent and accurate results from analysis.
    - **Maintainability**: Structured modular codebase for future enhancements.
    - **Portability**: Compatibility across systems where Python and Power BI are available.
    - **Reusability**: Modular functions and visual templates that can be reused across similar projects.
    - **Application compatibility**: Integration with standard data formats (CSV, Excel).
    - **Resource utilization**: Efficient memory and compute usage while processing the dataset.
    - **Serviceability**: Easy debugging, documentation, and support for project scaling.

1.2 Scope

This High-Level Design (HLD) document provides an overview of the system's structure for analyzing Google Play Store data. It includes the overall architecture, data flow, and tools used. The document is written in simple terms to help stakeholders understand how the system works, from data processing to visualization.

2 General Description

2.1 Product Perspective & Problem Statement

With the rapid growth of mobile applications, understanding trends in app downloads, ratings, and user engagement is crucial for developers and businesses. The Google Play Store hosts a massive number of apps across various categories, making it challenging to track performance and user preferences.

This project focuses on performing **ETL (Extract, Transform, Load)** on a Google Play Store dataset to uncover trends and insights. The key objective is to analyze:

- App download patterns
- Rating distributions
- Category-wise performance

We aim to identify **trends month-wise, year-wise, and yearly month-wise**, providing a clear picture of how user behavior and app popularity change over time. These insights will support data-driven decision-making for app development, marketing, and business strategy.

2.2 Tools used

Business Intelligence tool Power BI is used to build the whole framework.

3 Design Details

3.1 Functional Architecture

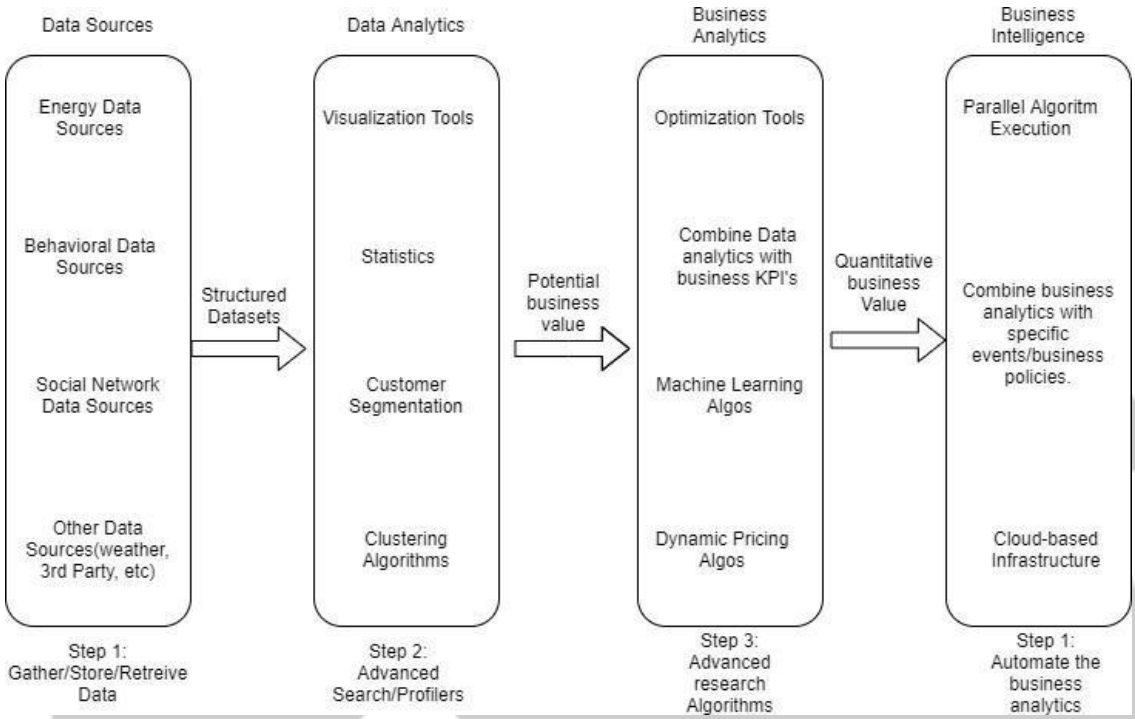
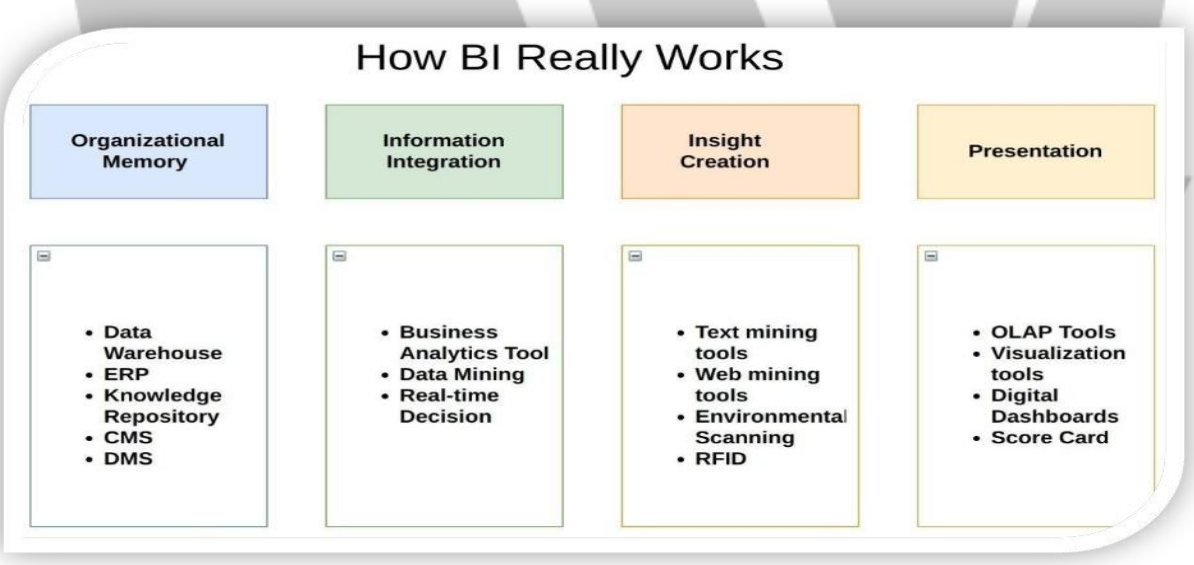


Figure 1: Functional Architecture of Business Intelligence



3.2 Optimization

To ensure smooth performance and faster insights during the analysis and visualization of Google Play Store data, the following optimization strategies are applied:

- **Minimize Data Fields & Records**: Only essential columns (like App Name, Category, Rating, Installs, etc.) are kept. Unnecessary columns are dropped during the data cleaning phase to reduce data size.
- **Pre-processed Extracts**: Calculations (e.g., average ratings, total installs) are materialized during the transformation phase to reduce real-time computation.
- **Efficient Visual Design**:
    - Dashboards are designed with **guided analytics**, showing only relevant insights per page.
    - Related views are connected using filters or slicers for smooth navigation.
    - Repeated or unnecessary dimensions and filters are avoided to prevent complex queries.
- **Filter Optimization**:
    - Only **essential filters** are used.
    - **Include filters** are preferred over exclude filters for better performance.
    - **Continuous date filters** are used to take advantage of indexing and speed.
    - Wherever possible, **Boolean or numeric filters** are applied instead of strings.
- **Use of Parameters**: Parameters and action filters are used in Power BI/Tableau to dynamically update visuals without reloading data sources, reducing load time.

These strategies help maintain high performance across ETL, analysis, and dashboard layers, delivering quick and meaningful insights.

3.3 Calculation Optimization

To improve dashboard and query performance, the following calculation optimization strategies are applied:

- **Perform Calculations Early**: Basic calculations (like average ratings, total installs, etc.) are done during the **data transformation stage** using Python or SQL, reducing the load during visualization.
- **Avoid Complex/Nested Calculations**: Nested or layered formulas are minimized to speed up processing time in tools like Power BI or Tableau.
- **Reduce Granularity**:
    - Limit the number of dimensions involved in calculations.
    - Avoid highly detailed level-of-detail (LOD) expressions or dense table calculations when simpler alternatives work.
- **Prefer MIN or MAX Over AVG**: Where applicable, using `MIN` or `MAX` instead of `AVG` can reduce processing load without affecting insight quality.
- **Use Booleans or Numeric Types**: Boolean and integer-based logic is preferred over string-based calculations, as it processes faster and uses fewer resources.
- **Group with Calculated Fields**: When grouping data, calculated fields (like categorizing apps based on installs or ratings) are more efficient than default group functions.

These practices ensure a faster, smoother analytical experience for end users and reduce computational burden during dashboard interactions.

4 KPIs & Charts

Interactive dashboards will be implemented to visualize key metrics and trends related to Google Play Store apps. These dashboards will provide quick, clear insights into various performance indicators across time and categories.

4.1 KPIs (Key Performance Indicators)

Key indicators summarizing the Google Play Store data and helping track app performance include:

1. **Total Number of Apps**
2. **Average App Rating**
3. **Total Installs (Downloads)**
4. **Top App Category by Count**
5. **Number of Paid vs Free Apps**
6. **Apps with Highest Reviews**

4.2 Charts

Various visualizations will be used to represent patterns, trends, and relationships in the data:

1. **App Categories Distribution** – *Pie Chart*
2. **Installs by App Category** – *Bar Chart*
3. **Ratings Over Time** – *Line Chart*
4. **Top 10 Most Installed Apps** – *Horizontal Bar Chart*
5. **Free vs Paid Apps Comparison** – *Column Chart*
6. **Rating vs Reviews** – *Scatter Plot*

These KPIs and charts will enable stakeholders to easily interpret trends, identify top-performing app types, and draw insights on how user feedback and app installs correlate.

5. Deployment

As the demand for data-driven decision-making continues to rise, businesses are increasingly turning to advanced analytics to drive performance and innovation. With the exponential growth of data from platforms like the Google Play Store, deploying scalable and interactive data analysis tools has become essential.

In this project, data is processed through an **ETL pipeline (Extract, Transform, Load)** and analyzed using visualization tools like **Power BI**. The pipeline ensures that data from the Google Play Store is cleaned, transformed, and structured to extract insights on app performance, user behavior, and emerging trends.

.

Deployment Options for Power BI

Type	Pros	Cons
Power BI Service (Cloud)	Easy to share and access from anywhere, no setup needed	Requires internet, limited custom control
Power BI Report Server (On-Premises)	Full control, better for sensitive data	Needs server setup, IT maintenance
Hybrid	Best of both worlds – secure + scalable	Complex to manage and integrate

Strategy

- **Default**: Use **Power BI Service** for cloud-based deployment.
- **Alternative**: Organizations with strict data policies can use **Report Server** or go for a **hybrid** setup to balance flexibility and control.