HIGH LEVEL DESIGN (HLD) DOCUMENT
CROP PRODUCTION DATA ANALYSIS

Revision Number: 1.0

Last date of Revision: 25/09/2021

Submitted by:

Dattatreya Thunuguntla

Document Version Control:

Date Issued	Version	Description	Author
25th September,2021	1.0	First Version of Complete HLD	Dattatreya Thunuguntla

Contents:

Abstract:

The agriculture business domain, as a vital part of the overall supply chain, is expected to highly evolve in the upcoming years via the developments, which are taking place on the side of the Future Internet. This paper presents a novel business-to-Business collaboration platform from the agri-food sector perspective, which aims to facilitate the collaboration of numerous stakeholders belonging to associated business domains, in an effective and flexible manner.

This dataset provides a huge amount of information on crop production in India ranging from several years. Based on the Information the ultimate goal would be to predict crop production and find important insights highlighting key indicators and metrics that influence the crop production.

Make views and dashboards first. Make a story out of it.

1. Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

- Present all of the design aspects and define them in detail
- Describe the user interface being implemented
- Describe the hardware and software interfaces
- Describe the performance requirements
- Include design features and the architecture of the project
- List and describe the non-functional attributes like:
    - ✓ Security
    - ✓ Reliability
    - ✓ Maintainability
    - ✓ Portability
    - ✓ Reusability
    - ✓ Application compatibility
    - ✓ Resource utilization
    - ✓ Serviceability

1.2 Scope:

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

2. General Description

2.1 Product Perspective and Problem Statement

Technology is the rapidly increasing now-a-days and applicable everywhere. One of the advanced features of Technology is Android operating systems. We all use in our daily life. Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablets.

Things to be done:

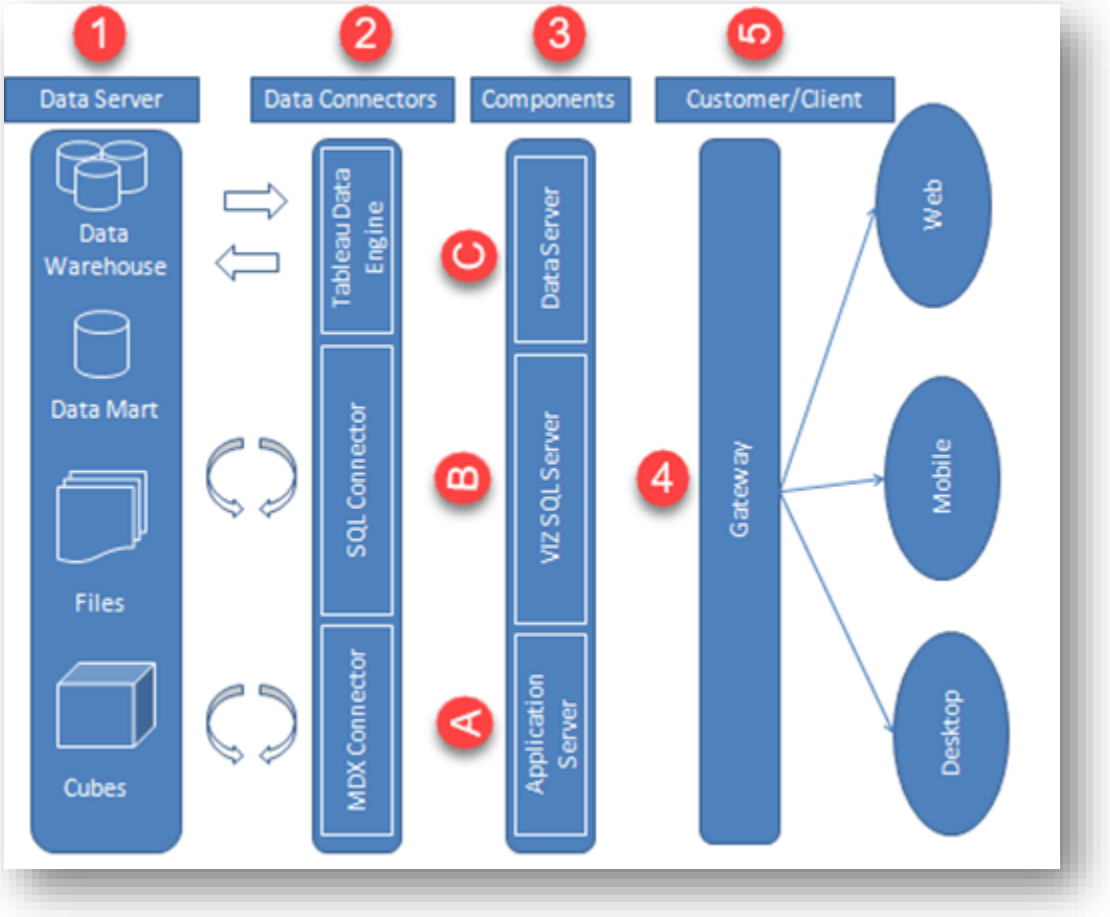
- Do ETL Operations - Extract-Transform-Load the dataset and find for me some information from this large data. This is form of data mining. What all information can be achieved by mining this data, would be brainstormed by the interns.
- Find key metrics and factors and show the meaningful relationships between attributes.
- Do your own research and come up with your findings.

2.2 Tools Used:

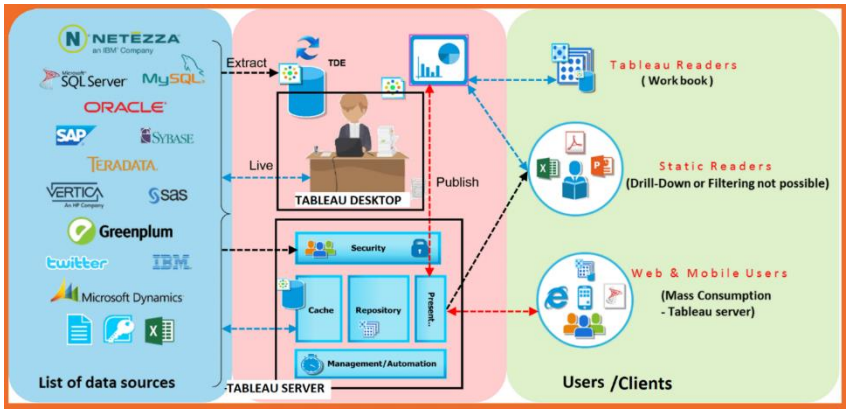
Business Intelligence tools and libraries works such as NumPy, Pandas, Excel, R, Tableau, Power BI are used to build the whole framework. For this project I had exclusively used Power BI Desktop.

3. Design Details:

3.1 Functional Architecture:



Functional Architecture of Business Intelligence



3.2 Optimization:

1. Your data strategy drives performance

• Minimize the number of fields

• Minimize the number of records

• Optimize extracts to speed up future queries by materializing calculations, removing columns and the use of accelerated views

2. Reduce the marks (data points) in your view

• Practice guided analytics. There's no need to fit everything you plan to show in a single view. Compile related views and connect them with action filters to travel from overview to highly-granular views at the speed of thought.

• Remove unneeded dimensions from the detail shelf.

• Explore. Try displaying your data in different types of views.

3. Limit your filters by number and type

• Reduce the number of filters in use. Excessive filters on a view will create a more complex query, which takes longer to return results. Double-check your filters and remove any that aren't necessary.

• Use an include filter. Exclude filters load the entire domain of a dimension, while include filters do not. An include filter runs much faster than an exclude filter, especially for dimensions with many members.

• Use a continuous date filter. Continuous date filters (relative and range-of-date filters) can take advantage of the indexing properties in your database and are faster than discrete date filters.

• Use Boolean or numeric filters. Computers process integers and Booleans (t/f) much faster than strings.

• Use parameters and action filters. These reduce the query load (and work across data sources).

4. Optimize and materialize your calculations

• Perform calculations in the database

• Reduce the number of nested calculations.

• Reduce the granularity of LOD or table calculations in the view. The more granular the calculation, the longer it takes.

o LODs - Look at the number of unique dimension members in the calculation.

o Table Calculations - the more marks in the view, the longer it will take to calculate.

• Where possible, use MIN or MAX instead of AVG. AVG requires more processing than MIN or MAX. Often rows will be duplicated and display the same result with MIN, MAX, or AVG.

• Make groups with calculations. Like include filters, calculated groups load only named members of the domain, whereas Tableau's group function loads the entire domain.

• Use Booleans or numeric calculations instead of string calculations. Computers can process integers and Booleans (t/f) much faster than strings. Boolean>Int>Float>Date>Date Time>String

4. KPIs:

Dashboards will be implemented to display and indicate certain KPIs and relevant indicators for the App Store Data Analysis. As and when, the system starts to capture the historical/periodic data for a user, the dashboards will be included to display charts over time with progress on various indicators or factors

5. Deployment:

Prioritizing data and analytics couldn't come at a better time. Your company, no matter what size, is already collecting data and most likely analyzing just a portion of it to solve business problems, gain competitive advantages, and drive enterprise transformation. With the explosive growth of enterprise data, database technologies, and the high demand for analytical skills, today's most effective IT organizations have shifted their focus to enabling self-service by deploying and operating Tableau at scale, as well as organizing, orchestrating, and unifying disparate sources of data for business users and experts alike to author and consume content.

Tableau prioritizes choice in flexibility to fit, rather than dictate, your enterprise architecture. Tableau Server and Tableau Online leverage your existing technology investments and integrate into your IT infrastructure to provide a self-service, modern analytics platform for your users. With on-premises, cloud, and hosted options, there is a version of Tableau to match your requirements. Below is a comparison of the three types:

1. Tableau Server - On Premises:

- Full control of hardware and software.
- Infrastructure and data remain behind your firewall.
- Need dedicated administrators to manage hardware and software.
- Additional infrastructure needed to access off-network (mobile, external)

2. Tableau Server - Public Cloud (IaaS):

- Full control of software on managed hardware
- Puts infrastructure in same place as data (for migration to cloud)
- Flexibility to spin up/down hardware as needed
- Need dedicated administrators to manage software
- Additional infrastructure needed to access off-network (mobile, external)

3. Tableau Online (SaaS):

- Fully hosted solution (hardware, software upgrades)
- Fast to deploy
- Easy for external audience to access
- Single-site in multi-tenant environment

- Cubes are not supported
- No guest account access

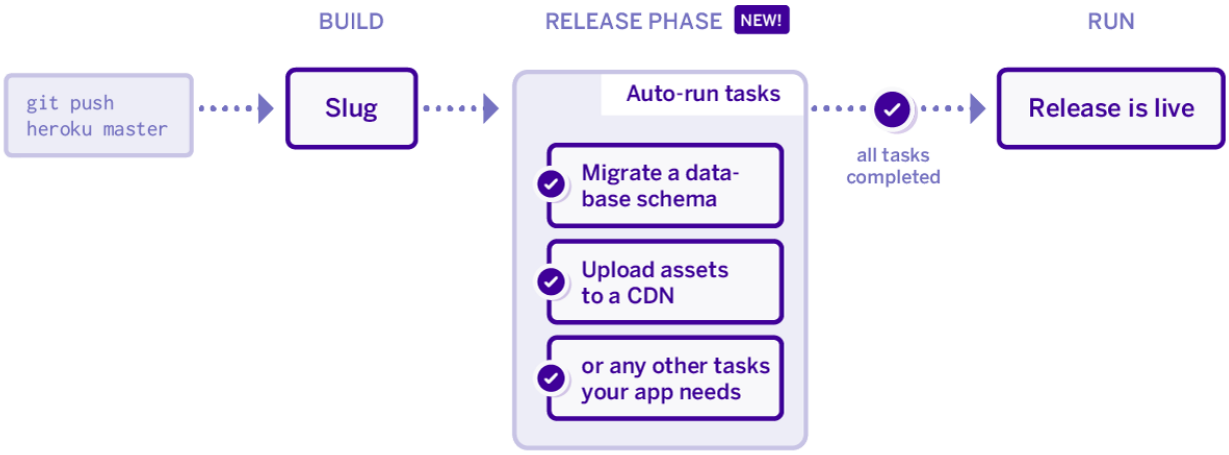
6. Deployment in Heroku:

The deployment in the Heroku consists of 3 types like

1. Heroku Git
2. Personal GitHub
3. Container Registry

In this project we had used Personal GitHub Deployment method. You can check the source files in the personal GitHub repository in the following link: https://github.com/dattu-98/crops_india

- The GitHub repository consists of Procfile, app.py, requirements.txt, runtime.txt, setup.sh



Heroku Webapp Link: https://cropsindia.herokuapp.com/