HIGH LEVEL DESIGN (LLD) DOCUMENT

ANALYZING SWIGGY

Version:	**1.0**
Last Revised Date:	**16-09-2021**
Written By:	**Balaji Mummidi**

Document Version Control:

Date Issued	Version	Author
16th September,2021	1.0	Balaji Mummidi

Contents

I. Abstract

Online Food Delivery is emerged as one of the most fast-paced development in the e-commerce space. This sector has revolutionized the entire outlook towards the food industry as consumers now have the privilege to choose from a wide range and variety of cuisines, anywhere, anytime from a range from a restaurant listed online. Revenue in the online food delivery segment amounts close to US$ 10,196 Mn in 2020.The revenue is expected to show an annual growth rate (CAGR 2020-2024) OF 9.5% resulting in a market volume of US$14,670 Mn by 2024.

1. Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessarydetail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, andcan 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 databasearchitecture, 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

Online Food Delivery is emerged as one of the most fast-paced development in the e-commerce space. This sector has revolutionized the entire outlook towards the food industry as consumers now have the privilege to choose from a wide range and variety of cuisines, anywhere, anytime from a range from a restaurant listed online. Revenue in the online food delivery segment amounts close to US$ 10,196 Mn in 2020.

Things to be done:

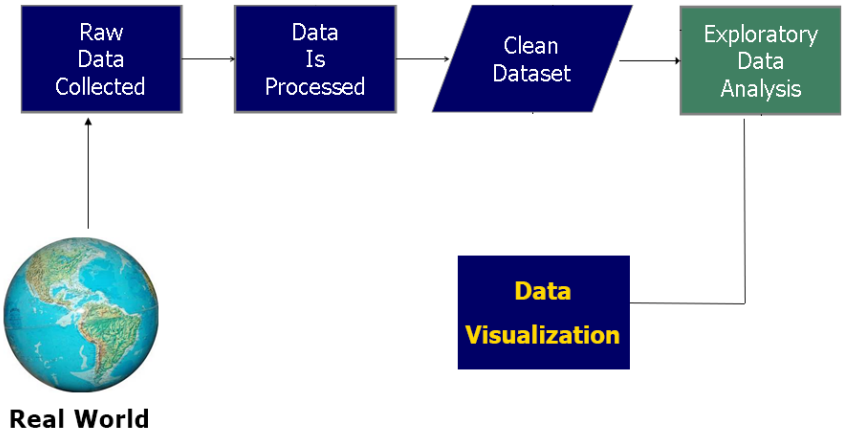
- Do ETL Operations - Extract-Transform-Load the dataset and find for me some information from this large data. This is form of data mining. What allinformation can be achieved by mining this data, would be brainstormed bythe interns.
- Find key metrics and factors and show the meaningful relationships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

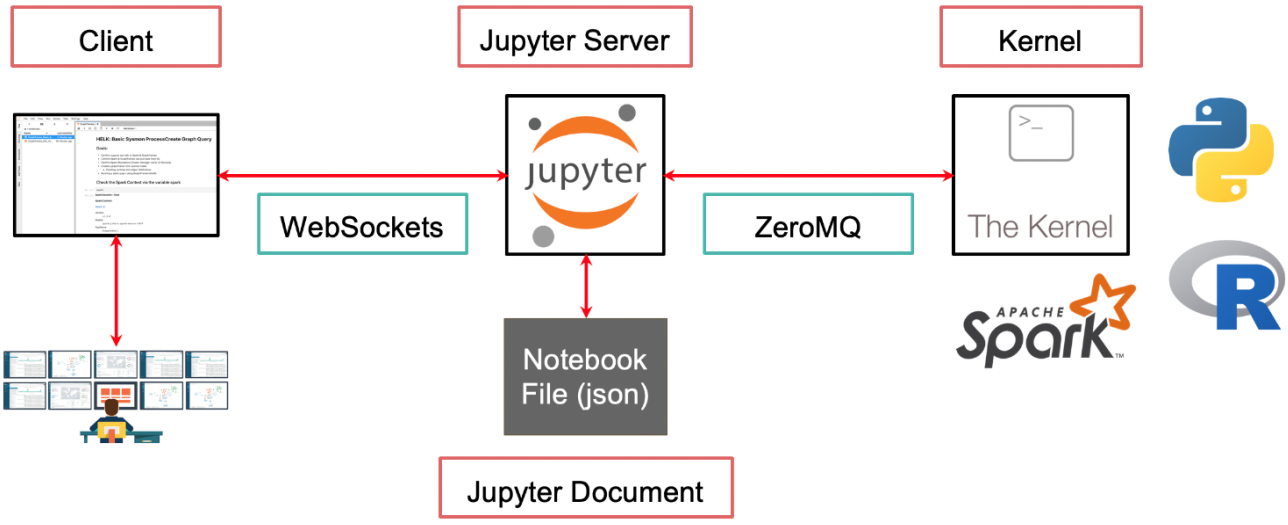
2.2 Tools Used:

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 Python, NumPy, Pandas, Seaborn & Matplotlib.

3. Design Details:

3.1 Functional Architecture:





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 filtersto 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

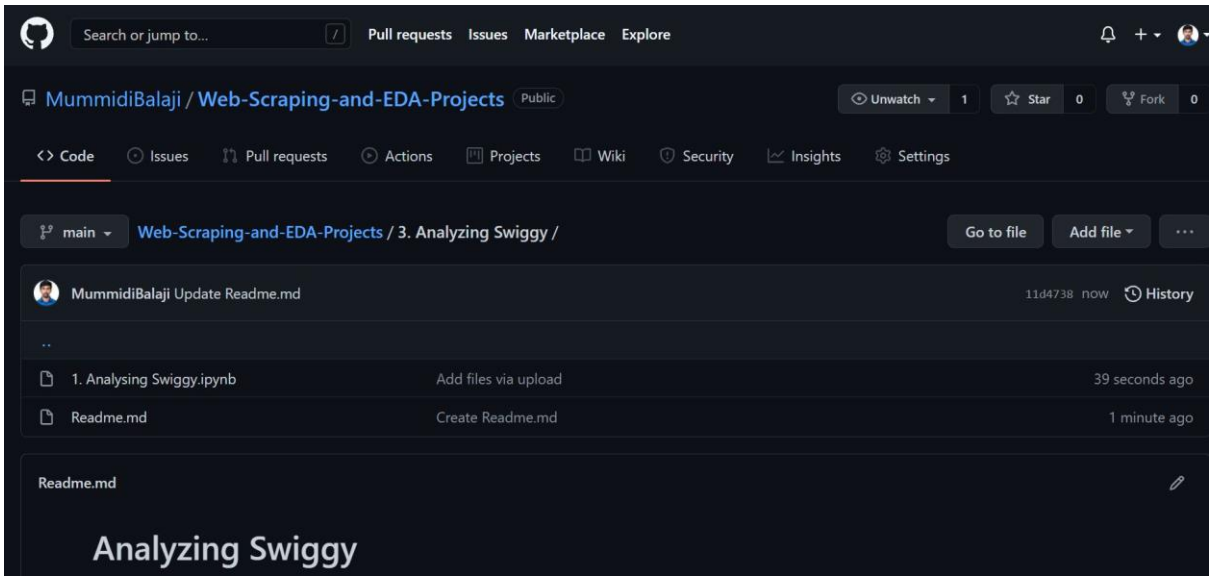
- Reduce the number of filters in use. Excessive filters on a view will createa more complex query, which takes longer to return results. Double-check your filters and remove any that aren't necessary.
- Use an include filter. Exclude filters load the entire domain of a dimension,while include filters 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 arefaster 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 workacross data sources).

4. Optimize and materialize your calculations

- Perform calculations in the database
- Reduce the number of nested calculations.
- 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 sameresult with MIN, MAX, or AVG.
- Make groups with calculations. Like include filters, calculated groups loadonly 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. Deployment

We can deploy the report in the form of Jupyter Notebook in GitHub and we can share that report with others. Here we had successfully deployed the report in GitHub.



TYPE PROS CONS

- They’re great for showcasing your work. You can see both the code and the results. The notebooks at Kaggle is a particularly great example of this.
- It’s easy to use other people’s work as a starting point. You can run cell by cell to better get an understanding of what the code does.
- Very easy to host server side, which is useful for security purposes. A lot of data is sensitive and should be protected, and one of the steps toward that is no data is stored on local machines. A server-side Jupyter Notebook setup gives you that for free.

Downsides of Jupyter notebooks

When we’re writing code in cells instead of functions/classes/objects, you quickly end up with duplicate code that does the same thing, which is very hard to maintain.

Consequences of duplicate code:

- It’s hard to actually collaborate on code with Jupyter — as we’re copying snippets from each other it’s very easy to get out of sync
- Hard to maintain one version of the truth.