Low Level Design

House Price Prediction

Written By	Author 1
Document Version	0.1
Last Revised Date	

DOCUMENT CONTROL

Change Record:

VERSION	DATE	AUTHOR	COMMENTS
0.1	19- May - 2021	Author 1	Introduction and architecture defined Architecture & Architecture description appended and Updated. Unit tests cases are added.

Approval Status:

VERSION	REVIEW DATE	REVIEWED BY		APPROVED BY	COMMENTS

Contents

1. Introduction

1.1 What is Low-Level design document?

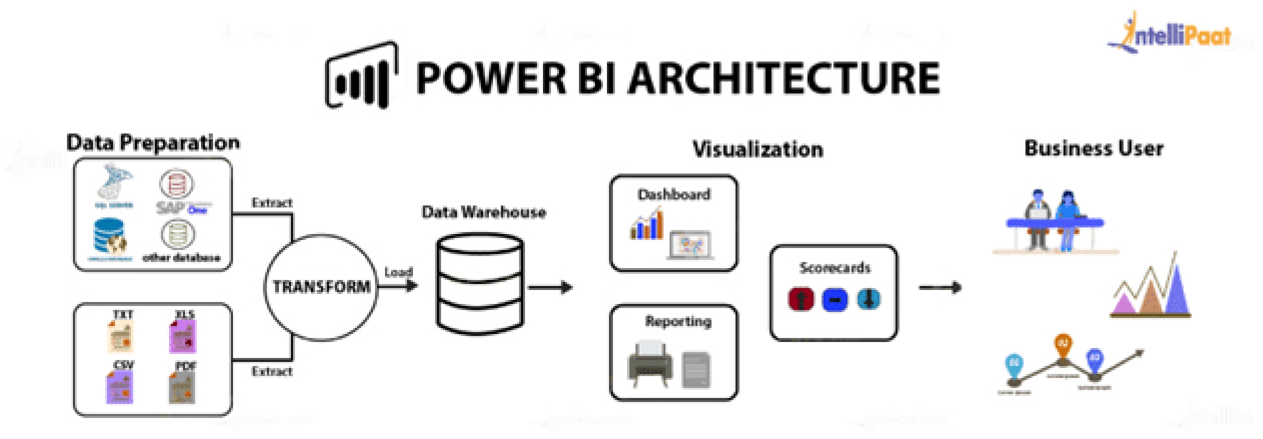
A Low-Level Design (LLD) document describes the comprehensive design of each system module and component, as well as how they will be implemented and integrated. It focuses on each module's underlying logic and interactions with other modules, including algorithms, data structures, and any necessary interfaces.

1.2 Scope

This LLD paper describes the comprehensive architecture of the ETL (Extract-Transform-Load) process and data analysis for Swiggy's Bangalore delivery outlet. The scope includes:

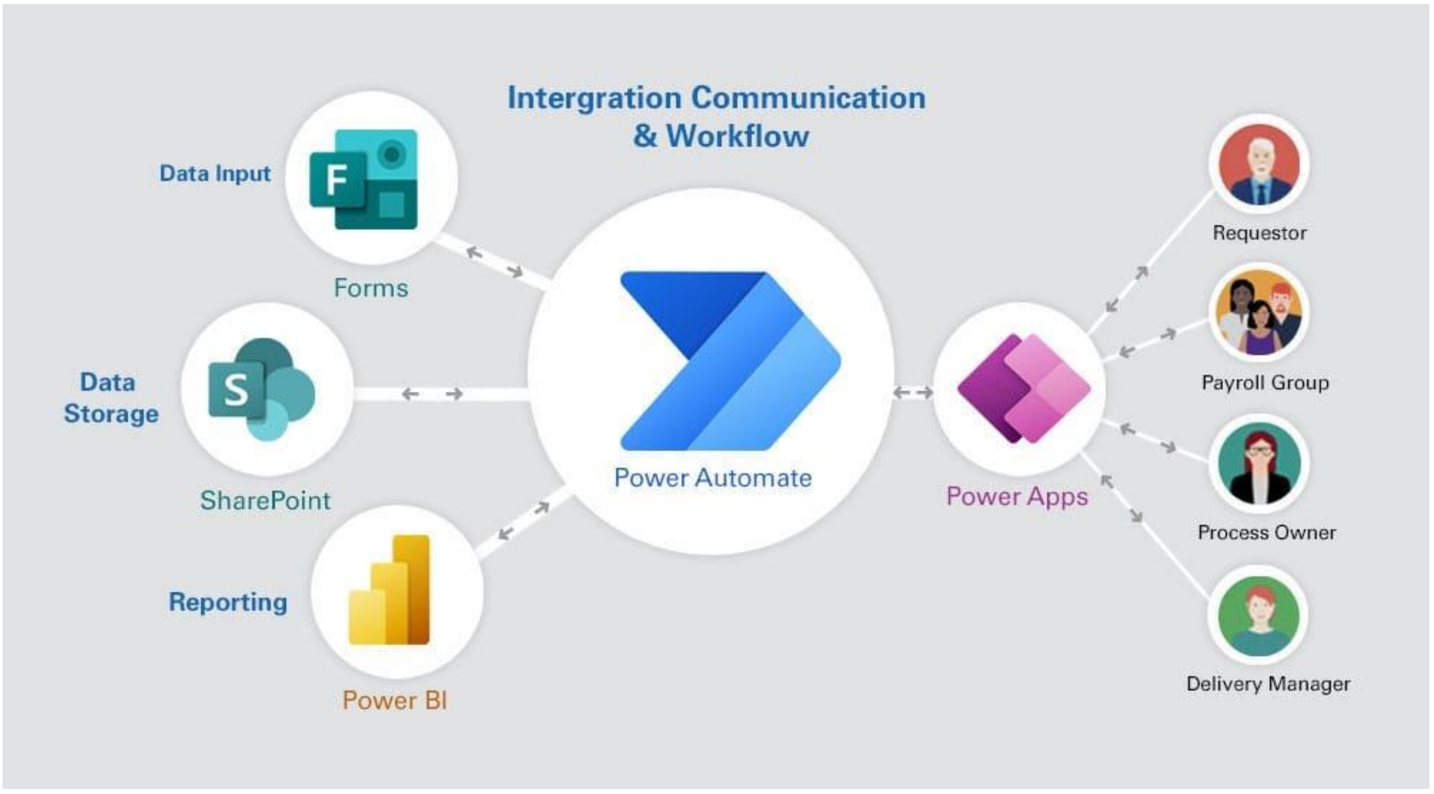
- Data Extraction: Collecting raw data from the supplied dataset and via web scraping to supplement the dataset with extra information such as real-time reviews or updated menu details.
- Data Transformation: Cleaning, normalizing, and converting extracted data to make it appropriate for analysis. This includes resolving missing numbers, updating data types, deleting duplicates, and aggregating data to provide useful metrics.
- Data loading is the process of inserting converted data into a relational database to allow for efficient searching and analysis. This includes defining the database structure, establishing tables, and building data-insertion procedures.
- Data analysis is the process of mining data to discover essential measurements and determinants and analyzing the links between various aspects.
- Data Visualization: The use of business intelligence technologies such as Tableau or Power BI to generate interactive dashboards and visualizations that convey the insights gained from data analysis. This enables stakeholders to make data-driven decisions.
- Deployment entails setting up and automating the ETL pipeline so that it can run on a regular basis or demand. This entails containerizing the pipeline using Docker and scheduling ETL tasks with Airflow.
- Testing: Ensuring the accuracy and dependability of the whole ETL process and data analysis scripts through thorough unit testing. This comprises tests for data extraction, transformation, and loading.

2. Architecture



PowerBI Server Architecture

The Power BI Server Architecture consists of three major components: the Web Front End (WFE) for user authentication and request processing, the Back-End for report production and data processing, and data storage via on-premises or cloud-based solutions such as SQL Server. Power BI Gateway secures data transfers between on-premises sources and the cloud. This design The following diagram shows Power BI Server's architecture:



Power BI Communication Flow chart is internally managed by the by the multiple server processes

- Users interact with Power BI using the online portal or mobile app, requesting reports, dashboards, or data.
- The Web Front End (WFE) handles user authentication and initial request routing. It checks the user's credentials and routes the request to the relevant back-end service.
- The gateway handles incoming requests from users. It sends authenticated queries to back-end servers while ensuring safe data transmission between on-premises data sources and the cloud.
- Back-end services comprise a variety of services, including:

  1)Data Model Service: Manages data models and guarantees that queries are run against the appropriate data models.

2)Report Rendering Service: Processes and renders reports and visualizations as required by the user.

3)Query Processing: Runs data queries against the data models to get the required data.

- Data storage consists of both on-premises databases (e.g., SQL Server) and cloud-based storage (e.g., Azure SQL Database). The storage system maintains both static and dynamic data for reports and dashboards.

- Power BI Gateway: Serves as a bridge for safe data transmission between on-premises data sources and the Power BI service in the cloud, guaranteeing data accessibility without jeopardizing security.

- Data Sources: These can be SQL databases, Excel files, cloud services, and others. The gateway and back-end services enable smooth integration and data retrieval from these sources.

- Response to User: The processed data, reports, and dashboards are provided back to the user through the WFE, providing interactive visualizations and insights.

This chain of processes guarantees that user requests are handled efficiently, data is transmitted securely, and data is processed robustly, enabling Power BI to provide extensive business intelligence capabilities.

3. Architecture Description

3.1. Data Description

Data Source: The dataset is provided in the form of CSV files containing information about Swiggy's delivery outlets in Bangalore. The dataset includes attributes such as outlet ID, outlet name, location, ratings, number of deliveries, and more.

Data Attributes:

- outlet_id: Unique identifier for each delivery outlet.
- outlet_name: Name of the delivery outlet
- location: Location of the delivery outlet
- ratings: Customer ratings for the delivery outlet
- num_deliveries: Number of deliveries made by the outlet
- delivery_time: Average delivery time
- delivery_fee: Delivery fee charged by the outlet

3.2. Web Scrapping

Objective: Extract additional data points that may not be included in the current dataset, such as real-time reviews or menu information.

Tools: Scrapy.

Process:
Determine target websites (for example, Swiggy's official website).
Create web scraping programs to retrieve useful information.
Save the scraped data in a structured format such as CSV or JSON.

3.3. Data Transformation

In the Transformation Process, we will convert our original datasets with other necessary attributes format. And will merge it with the Scrapped dataset.
Objective: To clean, normalize, and transform the raw data into a usable format for analysis.

Tools: Python (Pandas, NumPy)

Process:
1)Data Cleaning:

- Handle missing values.
- Correct data types.
- Remove duplicates.

2)Data Normalization:

- Normalize ratings and delivery times.

3)Data Aggregation:

- Summarize data to compute metrics like average delivery time per location, total deliveries per outlet, etc.

3.4. Data Insertion into Database

Objective: To load the transformed data into a relational database for efficient querying and analysis.

Tools: SQL, PostgreSQL/MySQL

Process:

1. Design database schema based on data attributes.
2. Create tables in the database.
3. Write scripts to insert data into the database tables.

3.5 Make the SQL connection and set up the data source

Step 1: Configuring Power BI

Objective: Establish a secure and efficient connection between Power BI and the SQL Server to facilitate data retrieval for analysis and reporting.

Tools and Technologies:

- Power BI Desktop: For creating reports and dashboards.
- SQL Server: As the primary database to store and manage data.
- Power BI Gateway: For secure data transfer between on-premises SQL Server and Power BI service in the cloud.

Process:

1)SQL Server Preparation:

- Ensure SQL Server is running and accessible: Verify that the SQL Server instance is up and running, and accessible from the network where Power BI Desktop or Gateway is installed.
- Configure firewall settings: Ensure that the firewall allows incoming connections on the SQL Server port (default is 1433).

2)Power BI Desktop Configuration:

- Open Power BI Desktop: Launch Power BI Desktop on your machine.
- Get Data: Click on "Get Data" on the Home ribbon.
- Select SQL Server: Choose "SQL Server" from the list of data sources.
- Enter Server Details: Input the server name and database name in the connection dialog.
- Choose Data Connectivity Mode: Select either "Import" (to import data into Power BI) or "DirectQuery" (for live queries against SQL Server).
- Authentication: Provide authentication details (Windows or SQL Server authentication) to connect to the database.

- Navigator: Select the tables and views you want to load into Power BI.

3)Power BI Gateway Configuration (for on-premises SQL Server):

- Install Gateway: Download and install the Power BI Gateway on a server within the same network as your SQL Server.
- Configure Gateway: Open the gateway application, sign in with your Power BI account, and configure the gateway.
- Add Data Source: In the Power BI Service, navigate to "Manage Gateways", select your gateway, and add a new data source.
- Data Source Type: Select "SQL Server".
- Connection Information: Enter the server and database name.
- Credentials: Provide the necessary authentication details.
- Test Connection: Ensure the connection is successful.

4)Report Publishing:

- Create Reports: Use Power BI Desktop to create reports and visualizations based on the SQL Server data.
- Publish to Power BI Service: Publish the reports to the Power BI Service.
- Set up Scheduled Refresh: In the Power BI Service, configure scheduled refresh for datasets using the gateway to ensure data is updated regularly.

Key Considerations:

- Performance: DirectQuery mode should be used for real-time data access but might impact performance. Import mode improves performance but might not reflect real-time data.
- Security: Ensure that database credentials and connections are securely managed.
- Data Refresh: Regularly schedule data refreshes to keep the data up-to-date in Power BI.

This setup allows Power BI to access SQL Server data seamlessly, enabling powerful data analysis and visualization capabilities.

3.6 Export Data from Database

Objective: To extract data from the database for analysis and visualization.

Tools: SQL, Python (Pandas)

Process:

- Write SQL queries to fetch data.
- Load the fetched data into Pandas Data Frames for further analysis.

3.7 Deployment.

Objective: To deploy the ETL pipeline and the data analysis process so that it can run periodically or on-demand.

Tools: Docker, Airflow

Process:

- Containerize the ETL pipeline using Docker.
- Set up Airflow DAGs to schedule and monitor the ETL jobs.
- Test the deployment in a staging environment before moving to production.

4 Unit Test Cases

TEST CASE DESCRIPTION	EXPECTED RESULTS
Pie Chart for Hotel Name and Cuisine	A pie chart should display the distribution of cuisines across various hotels.
Donut Chart for Hotel Name and Location	A donut chart should display the distribution of hotels across different locations.
Cards for the Number of Hotels, Locations, and Areas	Cards should show the total count of hotels, locations, and areas in the dataset.
Stacked Bar Chart for Hotel Names and Ratings	A stacked bar chart should show the distribution of ratings for each hotel.
Slicer for Price Range	When clicked, the slicer should filter data based on selected price ranges.
Clustered Bar Chart for Hotels by Location	A clustered bar chart should show the count of hotels grouped by location.