1) External Source on boarding to HDFS
   a. Download Zomato restaurant data into zomato_raw_files folder

      zomato.zip

2) Converting un-structured data into csv data
   a. Copy first three files from **zomato_raw_files (Unix)** to **zomato_etl/source/json** folder
   b. Write an **application** to convert each json file into csv file with suffix as *_filedate.csv and store them in **zomato_etl/source/csv/** (Unix) folder

      For example:
      json1→ zomato_20190609.csv
      json2→ zomato_20190610.csv
      json3→ zomato_20190611.csv
      json4→ zomato_20190612.csv
      json5→ zomato_20190613.csv

   c. Note: "zomato_*.csv" should have below fields only
      1. Restaurant ID
      2. Restaurant Name
      3. Country Code
      4. City
      5. Address
      6. Locality
      7. Locality Verbose
      8. Longitude
      9. Latitude
      10. Cuisines
      11. Average Cost for two
      12. Currency
      13. Has Table booking
      14. Has Online delivery
      15. Is delivering now
      16. Switch to order menu
      17. Price range
      18. Aggregate rating
      19. Rating text
      20. Votes
3) Creation of External/Internal Hive table
   a. Move these csv files from **zomato_etl/source/csv** folder to HDFS ***<HDFS LOCATION>***
   b. Create External Table named "**zomato**" partitioned by fildedate and load **zomato_<filedate>.csv** into respective partition:
      i. Table should have all columns as per csv file Analyzing Big Data with Hive
      ii. New partition should be created whenever new file arrives
   c. Create Hive Managed Table named "**dim_country**" using **country_code.csv** file as per below details:
      i. Table should have all columns as per csv file
   d. Create zomato_summary_log table , schema given below:
      i. Job id
      ii. Job Step

iii. Spark submit command
iv. Job Start time
v. Job End time
vi. Job status

4) Transformation using Hive and Spark
   a. Write a **spark application** in scala to load summary table named “zomato_summary” and apply the following transformation
      i. Create “zomato_summary” table partitioned by **p_filedate**,**p_country_name**
      ii. **Schema for zomato_summary table is mentioned below:**
         1. Restaurant ID
         2. Restaurant Name
         3. Country Code
         4. City
         5. Address
         6. Locality
         7. Locality Verbose
         8. Longitude
         9. Latitude
         10. Cuisines
         11. Average Cost for two
         12. Currency
         13. Has Table booking
         14. Has Online delivery
         15. Is delivering now
         16. Switch to order menu
         17. Price range
         18. Aggregate rating
         19. Rating text
         20. Votes
         21. **m_rating_colour**
         22. **m_cuisines**
         23. p_filedate
         24. p_country_name
         25. **create_datetime**
         26. **user_id**

      iii. Data in this hive table should be in ORC format
      iv. Add audit columns “create_datetime” and “user_id” in zomato_summary table
      v. Derive a column “Rating Colour” based on the rule listed below

Rating Text	Aggregate Rating	m_rating_colour
Poor	1.9-2.4	Red
Average	2.5-3.4	Amber
Good	3.5-.39	Light Green
Very Good	4.0-4.4	Green
Excellent	4.5-5	Gold

      vi. Derive a column “**m_cuisines**” and map the Indian (Andhra,Goan,Hyderabadi,North Indian etc.) cuisines to **“Indian”** and rest of the cuisines to **“World Cuisines”**
      vii. Filter out the restaurants with NULL/BLANCK **Cuisines** values
      viii. Populate “NA” in case of Null/blank values for string columns

        ix. There should be no duplicate record in the summary table

   b. Spark application should be able to perform the following load strategies
      1. Manual → should be able to load the data for a Particular **filedate & country_name**
      2. Historical → should be able to load the data historically for all the **filedate & country_name**

5) Create a shell script wrapper to execute the complete flow as given
   a. ***Spark application and shell script should be parametrized (dbname, tablename, filters, arguments etc.)***
   b. ***Check if already another instances is running for the same application***
      i. ***Exit and send notification if already an instances is running or the previous application failed***
   c. ***User should be able to execute each module separately AND all the modules together***
   d. **Module 1 :**To call application that converts json file to csv
      i. Capture Logs in a file
      ii. Check execution status
      iii. If failed then add a failure entry into log table, send failure notification and exit
      iv. If pass then add a success entry into log table and move to next step
   e. **Module 2 :**Execute command to load the csv files into Hive external/managed table (New partition should be created whenever new file is being loaded with new **filedate)**
      i. Capture Logs in a file
      ii. Check execution status
      iii. If failed then add a failure entry into log table, send failure notification and exit
      iv. If pass then add a success entry into log table and move to next step
   f. **Module 3 :**To call spark application to load the zomato summary table
      i. Capture Logs in a file
      ii. Check execution status
      iii. If failed then add a failure entry into log table, send failure notification and exit
      iv. If pass then add a success entry into log table and send a final notification
   g. Purge last 7 days of logs from the log directory
   h. Write a beeline command and insert log details for each successful and un-successful execution containing below detail
      i. Job id
      ii. Job Step
      iii. Spark submits that got triggered
      iv. Job Start time
      v. Job End time
      vi. Job status

6) Once the execution for 3 json file is completed, move these files into archive folder
7) Add new source file into source folder and execute complete workflow again
8) Schedule the job to run daily at 01:00 AM using crontab
9) Execute the complete job and perform the unit testing to check the complete ETL flow and data loading anomalies
10) Document execution statistics **(Start time,End time, Total time taken for execution,No.of executors, No. of Cores, Driver Memory)** along with application URLs
11) Create a unit test case document and reports bugs/observations.

12) Standard Coding guidelines e.g.
    a. Variable Names - Variable names will be all lower case, with individual words separated by an underscore.
    b. For each Function/Procedure add Comments in code
    c. Code Alignment and indentation should be proper
    d. Perform Exception Handling
    e. in-built functions, Code, location, table, database name, etc should be in lower cases

13) Folder Structure on Linux:

- zomato_etl
    - source
        - json
        - csv
    - archive
    - hive
        - ddl
        - dml
    - spark
        - jars
        - scala
    - script (shell scripts and property files)
    - logs (log_ddmmyyyy_hhmm.log)
- zomato_raw_files

14) Folder Structure on HDFS

- zomato_etl_<username>
    - log
    - zomato_ext
        - zomato
        - dim_country