Data Engineering Part:

- I have used Spark streaming to ingest csv files.
- The project contains a Readme file which has instructions to run the code.
- I have used spark streaming to ingest the data.
- Since the instructions given said to ingest csv files every one hour, I think the csv files will be uploaded at some data lake like hdfs. For optimizing the program and reading different files simultaneously, I have assumed the structure of the csv files directory to be
    - data/
      ads/
        Ads.csv
      users/
        Users.csv …. Etc.
- The program will read all the files inside this directory for the first time and then at each hour when the program reruns, it will only read the file created in the next hour. This is done based on the assumption that new files are created every hour and modified to add new rows to the same file.
- This is done by retrieving the creation time of the files in the directory and subtraction from the current time. The code is written in the file but not implemented as for the ease of implementing locally the project, the storage for csv files for the moment is in the directory data.
- I have used spark streaming as it allows ingestion of million of rows and since we might need some analysis, it can be done using spark sql as well.
- I have also used a custom schema to validate the input type of data.
- The process to automate the running of spark job can be done using cron process which is not implemented in the project as it is a system file.
- The docker file in the project creates a docker image with spark.
- Docker compose.yml creates a spark cluster with 1 master and 2 slaves both will run on local machine. Instructions to build and run the cluster is given in README.md

Data Analysis Part:

- I have not been able to do the data analysis part.
- Since I have already used SQLAlchemy to do the connection while ingesting the data, I will use the same objects to read the data.
- I would also ggplot or matplotlib to create different visualisations.
- I would also use flask to create a http dashboard and fast api to serve the data.