Peer Review

Arin Arora's Approach:

- Used pymongo to make a connection to the mongodb database.
- Used subprocess library to create a new collection inside of mongodb using python.
- Used insert one function to insert into a collection.
- Created separate classes for analyzing the comments, movies and theaters dataset.Modularity was maintained and written in OOPS format only required field were projected using the python functions
- Using objects and classes can make your code more organized, modular, and easier
- For the code was written in menu driven interactive mode as possible
- Created separate functions for each query and mentioned the query inside of a pipeline
- Generalized queries were used
- The code is really well-structured and modular. It's easy to follow and understand
- Managed to create such a flexible and adaptable system using modular design principles.
- The stages of the aggregation pipeline can be used to filter, transform, group, and sort data were used in the most of the questions

Shubham Jhawar's Approach:

- Used pymongo to make a connection to the mongodb database.
- Used *subprocess* library to create a new collection inside of mongodb using python.
- Used insert one function to insert into a collection.
- Created separate files for analyzing the comments, movies and theaters dataset.
- Created separate functions for each query and mentioned the query inside of a pipeline, executed this pipeline using the aggregate function and printed the final output.
- Each function is written generically, where if any query requires top n results or results from a given year, all these attributes are passed in as function parameters
- Used indexes where required to optimize the query.