INSY6112A1

By: Gregory Mawman

Due Date:26 August 2025

Teacher: Uzono Ezekial

Student Number: 10476479

Question 1

The type of Database you would use for the task above would be a non-relational database or a NoSQL database. This type of database is non-Rigid as well as being designed for maximum scalability, performance and flexibility which is needed for a social media platform.

With the high-performance part being incredibly important for a social media platform to function. Since having a slow database can lead to problems with logging and in and people being timed out for waiting too long.

As well as having good scalability, as in that instead of having only one machine running anything it, it can run off multiple machines at the same time to minimize failure points and Aswell as allowing for more data since it can take in larger amounts of data.

Being able to process and store big data is also another benefit to a NoSQL database structure since with the need of the social media platform to be able to store photos, videos, texts and gifs, being able to store all that data is crucial to what a social media platform is.

Meaning the main types of data that would be stored is big data like photos, videos etc. with images most likely taking on the font of PNG or Jpeg. With either one being able to be inputted since the database isn't rigid with the specific data type. Same with videos which use MP4. The Database would also have to have time series data, to show how long it has been since the user last logged in, what current topics are going viral and past viral topics.

The database would also have to store relationships like, follows, likes and posts of certain individuals. Which also correlates with storing private messages sent from user to user or from user to groups. Which also correlates with the fact that the database must be fast enough to send said data from point to point without a delay. However, for this data to be stored, there has been a database. Type used:

1) Document-Oriented Database:
   This database is usually used to store documents like Json files (organized text file is how I would describe it). This in turn stores values like Strings, integers, Booleans and other variables. It offers a flexible data model, which is very suited to what we need for a social media platform.

2) Key-Value databases:
   This type of database is very simple, as it stores data with a unique key, making said key unique to each user, which allows it to transfer and store data at high speeds. Meaning it can be used for basic stuff, for us it can be used to update likes, saves and comments made to a post made by a user. Since it can be updated quickly with little errors.

3) Wide-Column stores:
   A wide-column store organizes data into tables with rows and columns, but unlike traditional relational databases, the structure is **flexible rather than rigid**. Each row does not need to have the same columns, and new columns can be added whenever needed. This means extra information can be stored easily without redesigning the entire database, making wide-column stores highly adaptable for big data applications.

4) Graph Database:
   It stores the data in the form of nodes and edges. Where usually, Nodes are used to store personal information on people and places. Where edges usually store the relationship between nodes. It can show the connection between a user node and post by having likes act as the edge (or connection).

However, these databases must also deal with the 3Vs of big data. These Vs being Volume, Velocity and Variety. Volume for size, Velocity for the speed data is generated and processed and Variety of the data being generated.
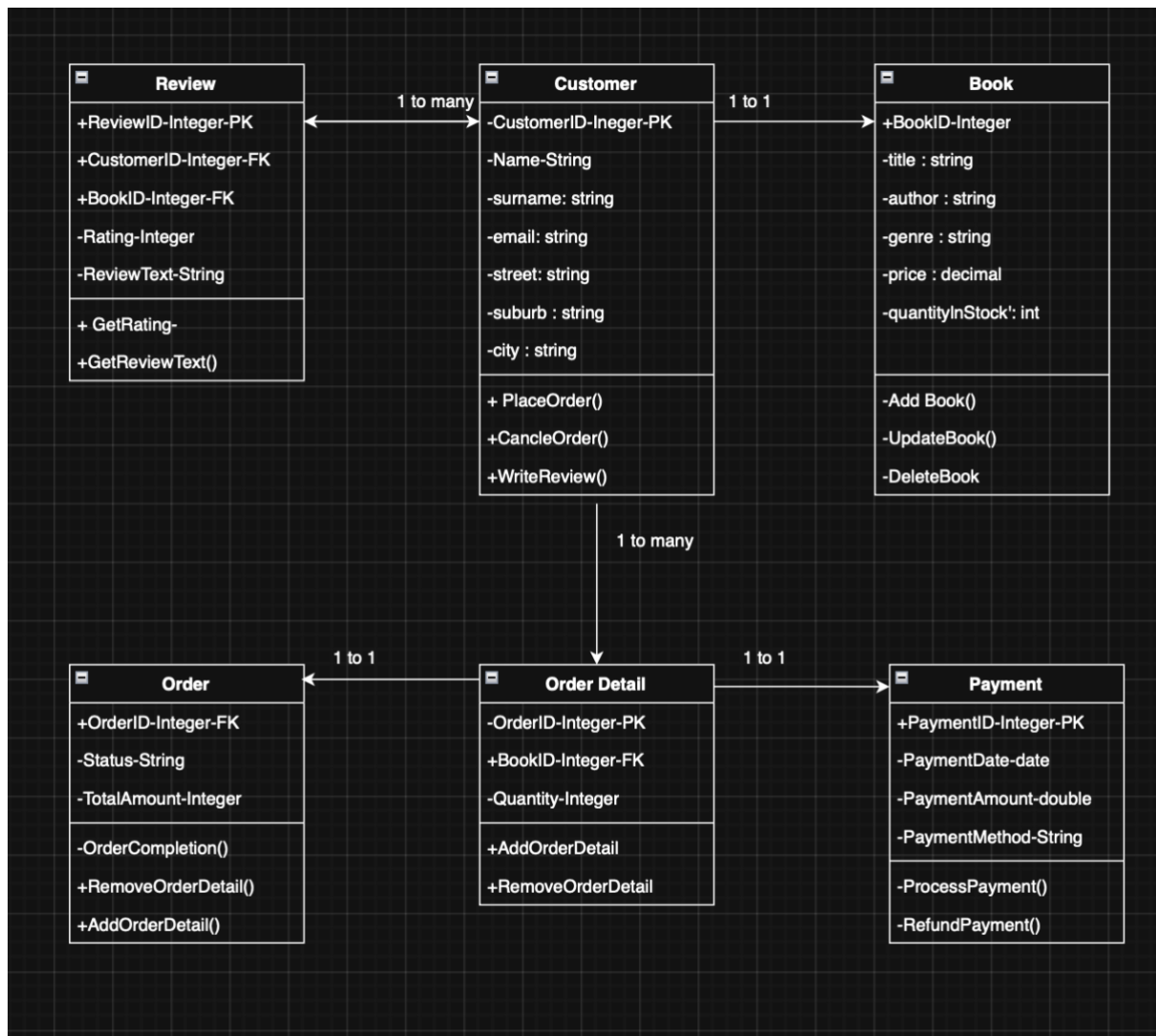
With the Volume of data being generated from the huge number of likes, posts and other interactions within the social media platform, only a non-relational database would be able to handle it, as with the amount of info generated per second could reach thousands of gigs. Meaning for the database to be able to take in the sheer volume of data, it would have to be able to scale horizontally to store the data in different databases, thus allowing for the speed to stay neutral.

Having the database be non-relational also means it can keep up with the velocity of data being produced. With following of what I said before, the social media platform would have to be able to keep up with live updates, with data constantly being updated and changed depending on the user. For every like, the database must update live and the produce the updated version of the number of likes. Which is why a non-relational would work best sine it spreads out the data rather than have all data traffic come to one storage area.

And with the variety of data being processed, the database would have to be able to process all kinds of different data types. From gifs, images and videos, too messages, live streams and likes. If it were to be a rigid SQL system, the platform would be too slow and be forced to have a database that is unable to change and allow data to be added of any kind. I.e, if someone posts a png, when jpegs are only allowed. The system would not allow it to take place and force the user to either get the jpeg changed to a png or not post it at all.

(Refsnes, et al., 2025) (ayusharma0698, 2025) (bansal_rtk_, 2025) (hiteshreddy2181, 2025) (Lauren Schaefer , 2025) (Gall, 2019) (Naudé, 2014) (BIG DATA LDN, 2018) (Roberts, 2025)

Question 2

Review
+ReviewID-Integer-PK

+CustomerID-Integer-FK

+BookID-Integer-FK

-Rating-Integer

-ReviewText-String

+ GetRating-

+GetReviewText()

Customer
-CustomerID-Ineger-PK

-Name-String

-surname: string

-email: string

-street: string

-suburb : string

-city : string

+ PlaceOrder()

+CancleOrder()

+WriteReview()

Book
+BookID-Integer

-title : string

-author : string

-genre : string

-price : decimal

-quantityInStock': int

-Add Book()

-UpdateBook()

-DeleteBook

Order
+OrderID-Integer-FK

-Status-String

-TotalAmount-Integer

-OrderCompletion()

+RemoveOrderDetail()

+AddOrderDetail()

Order Detail
-OrderID-Integer-PK

+BookID-Integer-FK

-Quantity-Integer

+AddOrderDetail

+RemoveOrderDetail

Payment
+PaymentID-Integer-PK

-PaymentDate-date

-PaymentAmount-double

-PaymentMethod-String

-ProcessPayment()

-RefundPayment()

1 to many

1 to 1

1 to many

1 to 1

1 to 1

Bibliography

Refsnes, H., Refsnes, S. & Refsnes, J. E., 2025. *W3Schools*. [Online]
Available at: https://www.w3schools.com/mysql/mysql_rdbms.asp
[Accessed 19 August 2025].

ayusharma0698, 2025. *GeeksforGeeks*. [Online]
Available at: https://www.geeksforgeeks.org/dbms/introduction-to-nosql/
[Accessed 19 August 2025].

bansal_rtk_, 2025. *GeeksforGeeks*. [Online]
Available at: https://www.geeksforgeeks.org/dbms/difference-between-relational-database-and-nosql/
[Accessed 19 August 2025].

hiteshreddy2181, 2025. *GeeksforGeeks*. [Online]
Available at: https://www.geeksforgeeks.org/dbms/types-of-nosql-databases/
[Accessed 19 August 2025].

Lauren Schaefer , 2025. *MongoDB*. [Online]
Available at: https://www.mongodb.com/resources/basics/databases/nosql-explained#:~:text=NoSQL%20databases%20(AKA%20%22not%20only,data%20and%20high%20user%20loads.
[Accessed 20 August 2025].

Gall, R., 2019. *packtPub*. [Online]
Available at: https://www.packtpub.com/en-us/learning/how-to-tutorials/different-types-of-nosql-databases-and-when-to-use-them?srsltid=AfmBOoqLGnXJDhASNOSRZ6ZY8FBecC8UzVFk8DLQWT6QC3bDM2h3sRI8
[Accessed 20 August 2025].

Naudé, R., 2014. *bizcommunity*. [Online]
Available at: https://www.bizcommunity.com/Article/196/662/119834.html
[Accessed 20 August 2025].

BIG DATA LDN, 2018. *BIG DATA LDN*. [Online]
Available at: https://www.bigdataldn.com/en-gb/blog/data-engineering-platforms-architecture/big-data-the-3-vs-explained.html
[Accessed 20 August 2025].

Roberts, S., 2025. *theknowledgeacademy*. [Online]
Available at: https://www.theknowledgeacademy.com/blog/big-data-3v/
[Accessed 20 August 2025].