**ea-april-2024-spring-data-2 Report**

Student: Ngoc But Bui, Tanusha Ayer, Prayash Shrestha

Sample data

|  |  |  |
| --- | --- | --- |
| User | Product | Review |
| 100 | 1,000 | 1,000,000 (1,000 for each product) |

Populate the database with sample data

* Write an application to create CSV files for user, product and review.
* Import CSV files to database.

Performance of getting all products

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Fetch mode | Fetch | No of SQL query | Execution time (s) | Memory usage | Data Transfer |
| SELECT | EAGER | 1001 | 12.02 | 1.55GB | 35.13MB |
| SELECT | LAZY | 1001 | 10.84 | 1.91MB | 35.13MB |
| JOIN | EAGER | 1 | 17.76 | 1.74GB | 35.13MB |
| SUBSELECT | EAGER | 2 | 15.02 | 1.57GB | 35.13MB |
| SUBSELECT | LAZY | 2 | 14.79 | 8.82MB | 35.13MB |
| BATCH (100) | EAGER | 11 | 10.63 | 1.54GB | 35.13MB |
| BATCH (100) | LAZY | 11 | 10.39 | 9.7MB | 35.13MB |
| BATCH (10) | EAGER | 101 | 10.19 | 1.54GB | 35.13MB |
| BATCH (10) | LAZY | 101 | 9.61 | 10.73MB | 35.13MB |

Conclusion

* **Select:**
  + The SELECT fetch mode, it indicates that the associated entities are loaded using separate SELECT queries, often referred to as N+1 queries.
  + It can be better option for the non-associated and smaller data but it is very much discouraged for the larger data.
  + **Eager Fetch:**
    - All associated entities are fetched immediately along with the main entity. Each associated entity is fetched separately through individual SELECT queries. As a result, the retrieved data for each associated entity is stored in memory concurrently with the main entity.
    - Since, all associated entities are loaded eagerly, the total amount of data fetched and stored in memory can be significant, especially if there are mainly associated entities or if they contain a large amount of data.
    - So, the memory usage is very high than the Lazy loading.
  + **Lazy Fetch:**
    - Associated entities are not fetched immediately when the main entity is retrieved.
    - Each associated entity is fetched separately through individual SELECT queries, but these queries are deferred until the associated entities are accessed.
    - This means that when using EAGER loading with SELECT fetch mode, each associated entity is fetched separately through individual SELECT queries.
    - This means that when using EAGER loading with SELECT fetch mode, each associated entity is fetched separately through individual SELECT queries.
    - Therefore, the lower initial memory usage in LAZY loading with SELECT fetch mode can be attributed to the delayed loading and storage of associated entities, which occurs only when they are accessed.
* **JOIN (**always EAGER)**:**
  + The number of SQL queries executed is significantly lower (1 query) compared to using SELECT fetch mode (1001 queries).
  + This reduction in the number of queries indicates that the JOIN operation efficiently retrieves all necessary data in a single query by joining the Product and Review tables.
  + However, the execution time appears to be higher (17.76 seconds) compared to SELECT fetch mode. As we are performing this test in small data, so we don’t see any higher performance.
  + But even for the larger data set we won’t see any optimized performance if the query is not well optimized and if the join is complex.
  + The memory usage is like SELECT fetch mode with EAGER loading, indicating that the amount of data transferred and stored in memory is comparable.
* **SUBSELECT**
  + only 2 SQL queries are executed which is significant reduction compared to the 1001 queries.
  + **EAGER:**
    - loading might be slower than SELECT or JOIN queries due to the overhead of executing multiple queries, increased database round-trips.
    - As it also uses the resource at once to get the data so that the resource utilization is very high.
  + **LAZY:**
    - The execution time is almost same but the memory is well optimized as the data is not loaded at once.
* **Batch:**
  + It just splits data into smaller chunks, the performance depends upon the size of the batch.
  + It has (total-number-of-data/batch-size) number of SQL queries.

@Fetch(FetchMode.SELECT)->Eager

A white background with black text

Description automatically generated

A screenshot of a computer

Description automatically generated

@Fetch(FetchMode.SELECT) -> Lazy

A white screen with text

Description automatically generated

A screenshot of a computer

Description automatically generated

@ Fetch(FetchMode.JOIN)

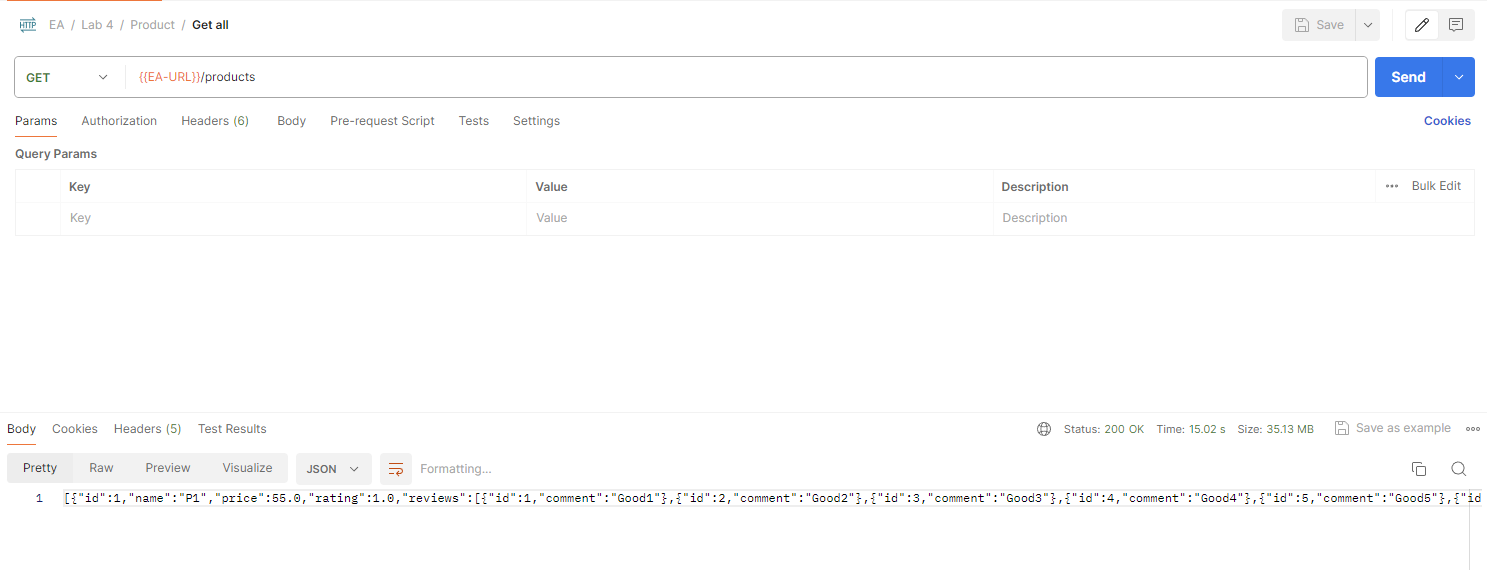
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

@Fetch(FetchMode.SUBSELECT)->Eager



A screenshot of a computer

Description automatically generated

@Fetch(FetchMode.SUBSELECT)->LAZY

A close up of a computer screen

Description automatically generated

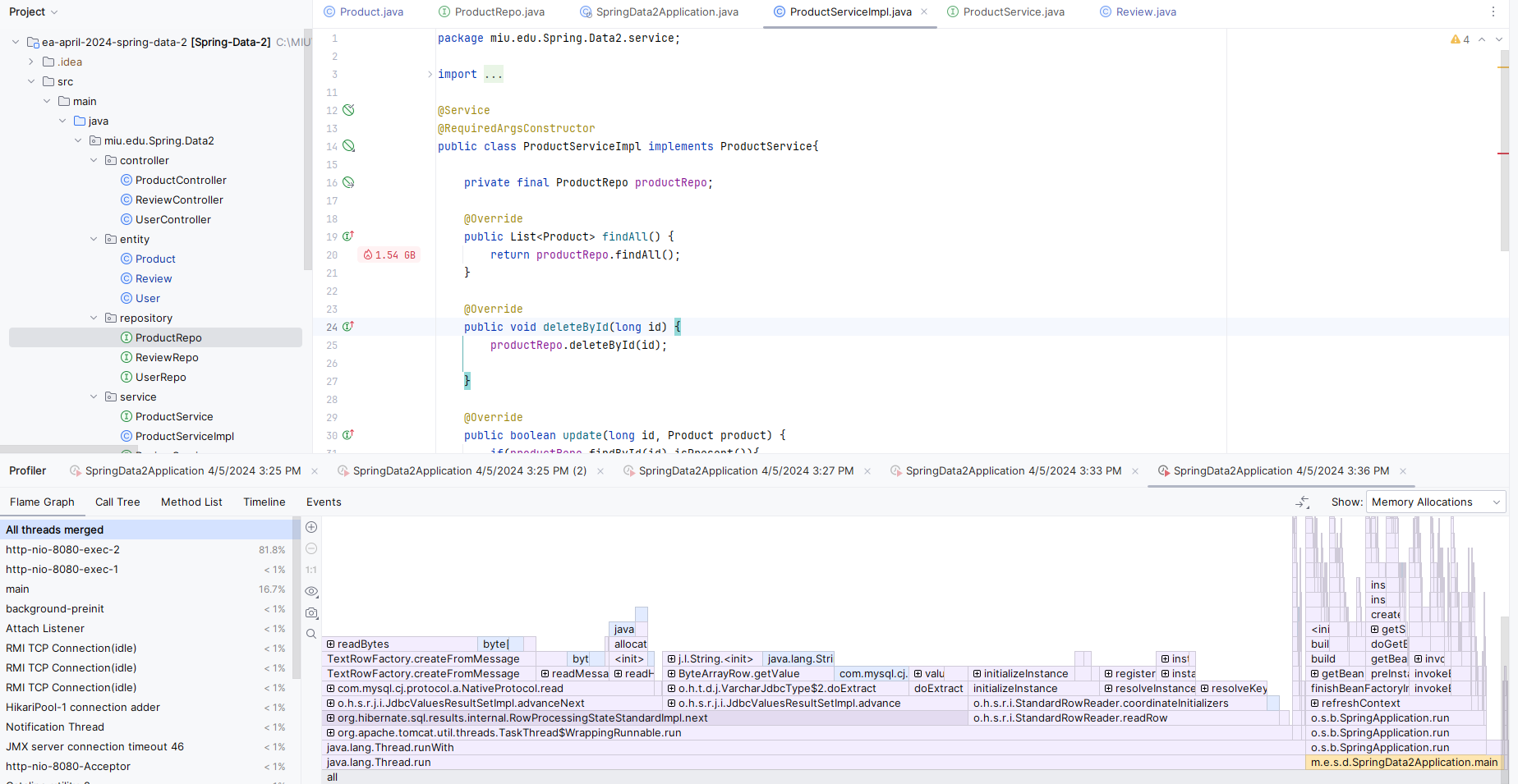
A screenshot of a computer

Description automatically generated

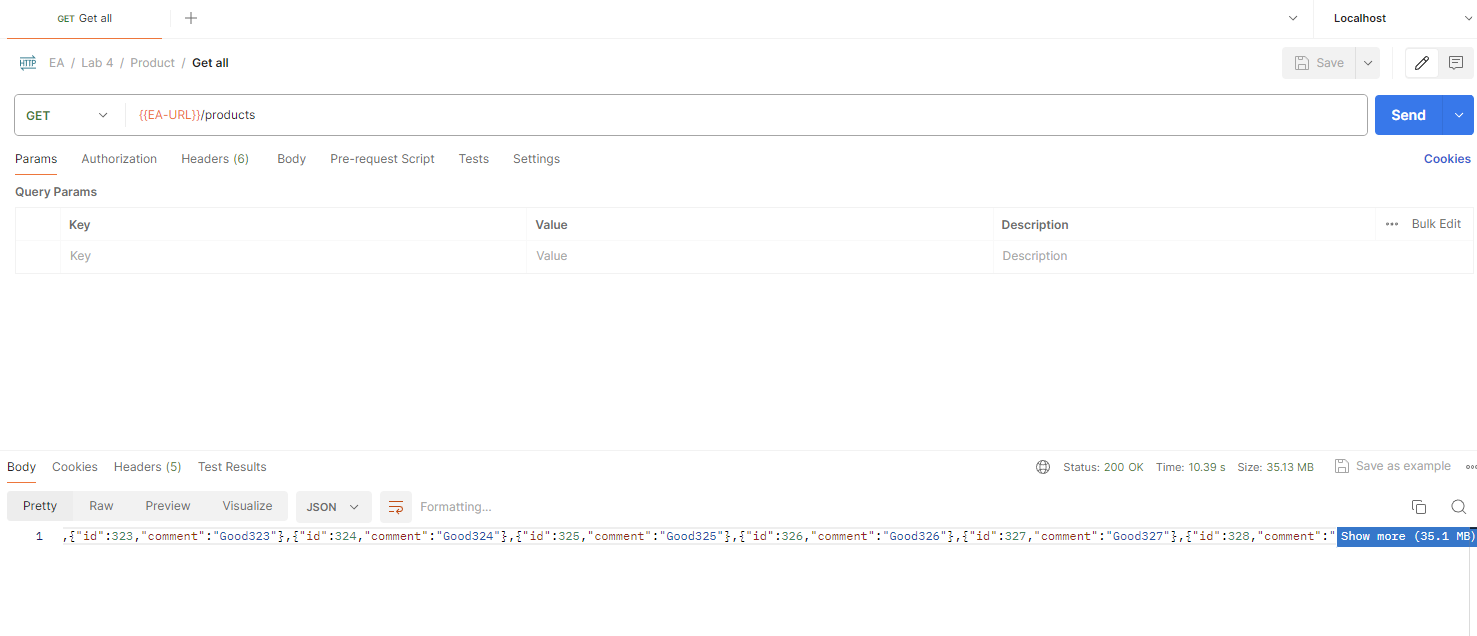
@Fetch(FetchMode.BATCH)->EAGER BATCHSIZE= 100

A screenshot of a computer

Description automatically generated



@Fetch(FetchMode.BATCH)->LAZY BATCHSIZE= 100



A screenshot of a computer

Description automatically generated

@Fetch(FetchMode.BATCH)->EAGER BATCHSIZE= 10

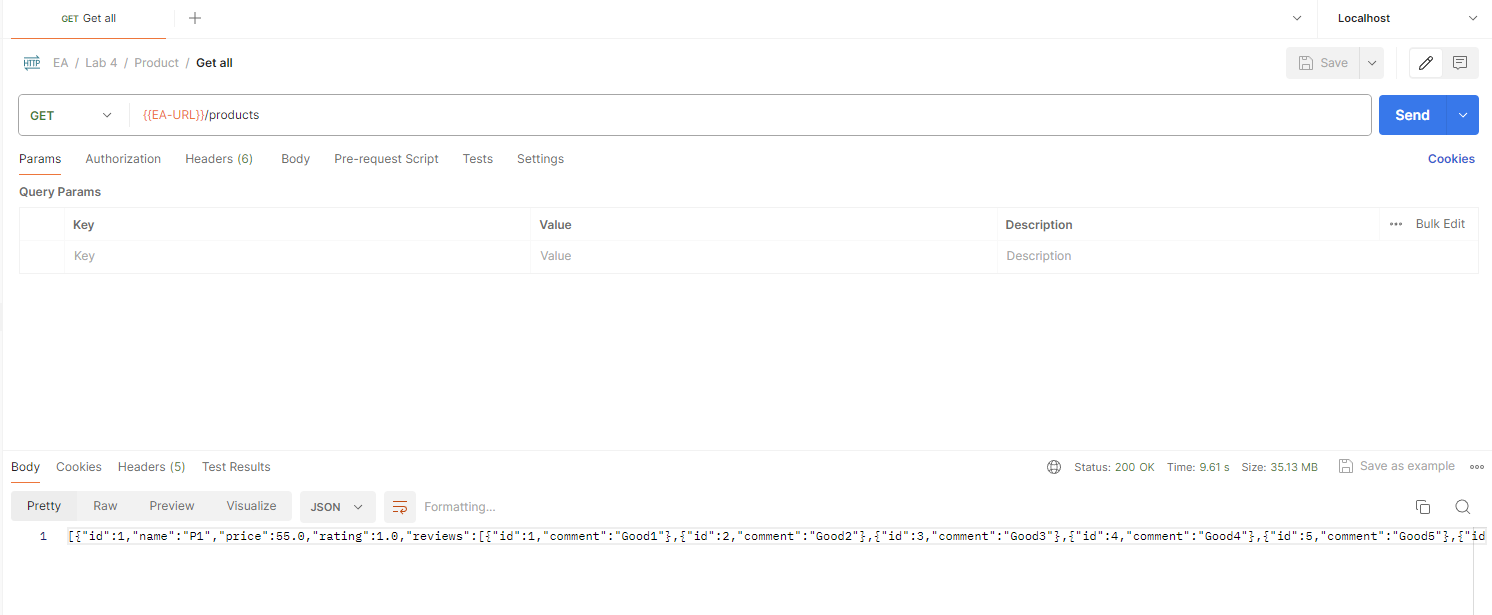
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

@Fetch(FetchMode.BATCH)->LAZY BATCHSIZE= 10



A screenshot of a computer

Description automatically generated