1. What would you change in the solution if the ranking will be based on most viewed items or the most/best reviewed?

Answer: I'd track additional metrics (views, reviews, ratings) in MongoDB, sync them into Elasticsearch, and use them as scoring signals during search. The ranking function would change to boost by views or average rating instead of sales.

2. How would you think about linking relevant products together?

Answer:

- *Keep a **separate** `related_products` **collection** (pivot-like) to store all relations, sharded by product id for scale.*
- *In each product document, **embed only the top 5–10 related product IDs** for fast lookups.*
- *Updates come from batch jobs that recompute relations and refresh both collections.*

3. How would you think about securing such service?

Answer:

- *Authentication & authorization at the API gateway*
- *Role-based access to MongoDB & Elasticsearch*
- *API rate limiting & request validation to prevent abuse*
- *Network isolation (private subnets, security groups)*

4. What would be the type of servers needed for such service? Would it be RAM or CPU optimized types of machines and why?

Answer:

- ***Elasticsearch**: memory/RAM optimized (to keep indices and search cache in memory, reducing latency)*
- ***API service**: CPU optimized (handles query parsing, JSON serialization, request throughput)*
- ***MongoDB**: balanced (RAM for working set, CPU for queries, SSD for storage IO)*