REPORT ON CHANGES MADE AND THEIR IMPACT ON PERFORMANCE

Overview

The original code for reading and filtering the `startup_funding.csv` file was working functionally but had repeated logic, redundant loops, and low modularity. The refactored version simplifies and optimizes this by improving structure, reusability, and clarity.

Original code :
https://github.com/lamchau/refactoring-exercise/blob/master/java/funding-raised/src/main/java/com/checkr/interviews/FundingRaised.java

---

Changes Made

1. **Introduced a Common HEADERS Array:**
   - Created a `HEADERS` array to represent the CSV column names.
   - Avoided repeated use of hard-coded index values.
2. **Modularized CSV Reading:**
   - Moved the CSV reading logic to a separate `readCSV()` method.
   - This method reads all rows at once and removes the header.
3. **Created a General `mapRowToData()` Function:**
   - Centralized the mapping of CSV row arrays to `Map<String, String>`.
4. **Introduced a Generic `matchesOptions()` Method:**
   - Dynamically checks whether each row matches all the filters.

5. **Simplified `where()` and `findBy()` Logic:**
    - Loops through the CSV and applies the `matchesOptions()` check.
6. **Used Try-With-Resources:**
    - Ensures `CSVReader` is properly closed automatically.
7. **Added Custom Exception Message:**
    - `NoSuchEntryException` includes a clear default message.

Impact on Performance and Maintainability

- **Better Readability & Maintainability:** Easier to read and update.
- **Reduced Redundancy:** Shared logic avoids repetition.
- **Slightly Improved Performance:** Especially in `findBy()` with early exit.
- **Scalable for Future Fields:** `HEADERS`-driven structure is extendable.
- **No Major Computational Optimization:** Still O(n) for searches.

Conclusion

The refactored version improves the code's structure, reduces complexity, and makes future changes easier. While the performance benefits are modest for small files, the maintainability and readability are significantly better.