Coding task (JAVA)
A retailer offers a rewards program to its customers, awarding points based on each recorded purchase.

A customer receives 2 points for every dollar spent over $100 in each transaction, plus 1 point for every dollar spent over $50 in each transaction (e.g. a $120 purchase = 2x$20 + 1x$50 = 90 points).

Given a record of every transaction during a three-month period, calculate the reward points earned for each customer per month and total.

Checklist

a) **Goal of the assignment**: showcase great craftsmanship skills in area of JAVA, services, testing, building readable, maintainable, resilient systems. If you have personal project (comparable in complexity) that you would like to present instead of the assignment –let us know
b) **Key elements**: tests, good coding practices, well thought-out design –if requirements change your code should be easily fixable. Your assignment solution should not contain shortcuts that violate good coding practices e.g. class having multiple responsibilities; magic numbers in code; methods, variables names not giving any clue what they are for; etc.
c) I**mplement RestAPI's for all CRUD operations** – in this case – creating/updating transactions, calculating and providing reward information for a User.
d) **Time required**: anywhere between 4-8h depending on proficiency. It's best to focus on solving the problem first with appropriate tests and benefit from features of recent versions of language and frameworks (autogenerate boilerplate code) to complete the assignment quickly. Check "key elements" that are evaluated.
e) **Technical guidelines:**
    i. Used libraries should be up to date (for example: Junit5 not 4, Spring boot 2.6.* not 1.5.*)
    ii. Code should be clean, nice formatted, easy to read and understand
    iii. Code must compile and run, **there should be an instruction in readme.md file how to run the application**
    iv. Provide documentation of an application REST API
    v. All errors should be handled with appropriate HTTP status codes (200, 400, 422, 500 etc.) and with human readable messages.
    vi. Points calculation logic should be separated from infrastructure (REST, DB) and should be easy to test with unit tests
    vii. Please show the ability to write testable code. It is important.
    viii. Unit tests for points calculation logic should be written (please remember about corner cases)
    ix. Integration tests should not only check if application starts but also call REST endpoints and check at least if HTTP status code is correct
    x. Use of appropriate logging levels, framework
    xi. The solution must be checked into Github (provide a public Github url)