November 2022

Hands-On Project for: Java Backend Developer

Background

This document describes a backend developer hands-on project.
Maximum submit time is **3 days** since the project was assigned, while taking into consideration that there may be other commitments in place.
The project should be implemented using **java** only, **do not** use third party libs unless specified.
This document contains 2 questions.
In each question, please consider the following guidelines:

- Code syntax best practices (readable)
- Code design best practices (OOP, design patterns…)
- Code efficiency
- Integration efficiency
- Generated outputs should be readable
- Unit Testing with Junit
- Logging with Java Logger

At the end, please send the source code to **vladimirp@continuitysoftware.com** as a zip file (or a git repo link)
Feel free to contact us with any questions/clarifications using the above mail.

Project Details

1. Create a program that collects information using REST API from https://jsonplaceholder.typicode.com
   The JSONPlaceholder web provides usable REST API with free fake data. The available resources are:

/posts	100 posts
/comments	500 comments
/albums	100 albums
/photos	5000 photos
/todos	200 todos
/users	10 users

   NOTE: you need to create representative objects for each resource.

   a. Create a method that returns the summary for each user, his/her uncompleted tasks (todos)
      i. Returns: Collection
   b. Create a method that returns the uncompleted tasks of a **given user id**
      i. Returns: Collection
      ii. Param #1: user id
   c. Create a method that returns the summary for each user, the email of each replier (in a comment) per each post that the user has posted. If the post had 0 replies, do not show it.
      Example: user 1 posted 8 posts and for each post a few users commented, except for one post that had no response.
      i. Returns: Collection
   d. Create a method that returns all albums of a specific user that contains more photos than a given threshold
      i. Returns: Collection
      ii. Param #1: user id , Param #2: photos threshold
2. Create a main program that uses the classes you developed in the above question. The program should create scenarios that will test and utilize the above implementations and will be used as a proof of concept.

3. OOP - Create a program to manage Issue tickets and calculate their statistics.
   Background:

A ticketing system is using tickets to deliver its findings to the customer.
Design a program capable of creating and managing tickets including providing statistics regarding the open tickets.
Keep in mind the following guidelines:

- There is **only one** manager in a running system that can manage tickets
- All tickets should use a shared id sequence (incremental and unique number to identify each ticket).
- There are 3 types of ticket:
    - Security ticket – represent security weaknesses
    - Configuration ticket – represent weaknesses in systems configuration
    - BestPractice ticket – represent objects/systems that are failing best practices guidelines
- Only Security and BestPractice tickets can use CVE (best practice guideline provided by approved companies)
- Use the below interfaces as a design/requirements guideline
    - Feel free to change the naming convention if you see fit
    - You may (and should) add accessory methods to your design
- Keep the following best practice – when instantiating (creating new object instance) an object, the declarative type must be the interface while the actual implementation will be the actual object.
    - Example:
        - Good: `Set<T> mySet = new HashSet<>();`
        - Bad: `HashSet<T> mySet = new HashSet<>();`

Task:

Create a statistics manager that can provide statistics based on:

- Severity - Will show how many tickets opened for each severity
- CVE - Will show how many tickets opened for each CVE

The program should create 1000 tickets for the statistics analysis.