# Software Developer Course Assessment

## Quantitative Assessment Practice

Course Name: Software Design, Architecture, Testing

Current Week: November 21th, 2022

Submission date: December 2nd, 2022

Introduction:

The purpose of this assessment is to help us understand how the class is doing in terms of the review material that we have covered during the previous couple of weeks. The only purpose of this assessment is for us to improve our approach to review and ensure that what we’re currently doing is an effective strategy. Completion of this assessment is mandatory - if you don’t submit a solution, it will be marked as incomplete. You must complete a minimum of 80% of your assigned QAPs per course – otherwise you will be marked as incomplete for that course no matter how good your other grades are. If you do submit a solution, it will be marked as complete, as you will receive full marks no matter what your actual performance was – again this is a participation grade. Again, the goal here is to help you all in the best way that we can, so please do be honest when answering the questions related to how long it took, which resources you used, etc. And please ensure that you do your own work – don't just copy off a friend to get it done, earnestly do your best with it. If you can’t get it completely working, give us what you have. While it will be graded, the grade will not count against you, it’s just a way for us to see where everybody is, and to know which concepts, if any, we, as a class, may be struggling with.

Deadline: You will have until the end of the day on Wednesday June 1st , 11:59pm to submit your assessment solutions. Please ensure you answer all the questions outlined in the instructions portion of this document as well in your submission.

Marking: In this program core evaluation is marked with one of three possible marks: Incomplete, Pass, Pass Outstanding. For QAPs, though, where incomplete marks are more important for our own information as well as for the information of the student, we wanted to increase the resolution of our grading system. Therefore, QAPs are marked on a scale of 1-5. The details of this marking system are summarized in the table below.

Grade Meaning

1 Incomplete. Student shows severe lack of understanding of the material – solution is heavily incomplete, non-functional, or completely off base of what the assignment was asking for.

2 Partially Complete. Students show some understanding of the material. Solution may be non-functional or partially functional, but the approach is correct, albeit with some major bugs or missing features.

3 Mostly Complete. Student demonstrates understanding of the major ideas of the assignment. Solution is mostly working, albeit with a few small bugs or significant edge cases which were not considered. Shows a good understanding of the correct approach, and is either nearly a feature-complete solution, or is a feature-complete solution with some bugs.

4 Complete (Equivalent to: Pass.) Student shows complete understanding of assigned work and implemented all necessary features. Any bugs that are present are insignificant (for example aesthetic bugs when testing the functionality of code) and do not impact the core functionality in a significant way. All necessary objectives for the assignment are completed, and the student has delivered something roughly equivalent to the canonical solution in terms of features and approach.

5 Complete with Distinction (Equivalent to: Pass Outstanding) The student demonstrates a clear mastery of the subject matter tested by the QAP. The solution goes above and beyond in some way, makes improvements on the canonical solution, or otherwise demonstrates the student’s mastery of the subject matter in some way. A solution in this category would consider all reasonable edge cases and implement more than the necessary functionality required by the assignment.

Instructions:

Build from starch or using the template we worked on in class build a SpringBoot app with the controller pattern.

Model a Tournament object with the following attributes:

* Start date.
* End date.
* Location
* Entry Fee
* Cash Prize Amount

For this object implement the GET, POST, PUT and DELETE following the pattern we used for the city example in class.

Once this is working, build a docker image and test locally as we did in class using Postman.