

Part 1 - Analysis, Architecture Design and Detailed Design

Due 1 Dec 2023 by 17:00 **Points** 100 **Submitting** a file upload
Available until 1 Dec 2023 at 17:00

This assignment was locked 1 Dec 2023 at 17:00.

Weighting %:	50	Submission deadline (for students):	See Canvas
Authorship:	Individual	Target date for returning marked coursework:	12 January 2024
Tutor setting the work:	Dr John Kanyaru	Number of hours you are expected to work on this assignment:	50

This Assignment assesses the following module Learning Outcomes (from Definitive Module Document):

1. Be able to define software architecture.
2. Have knowledge and understanding of common software architecture patterns, Conway's law and its implications.
3. Understand the advantages of a three-tier architecture.

Assignment Tasks:

Introduction

This is an individual piece of work. Any form of cheating, including collusion or plagiarism will not be tolerated and will be reported as per UH assessment regulations.

The work is given in stages, so that you can start as soon as possible. You only hand in your final version for marking.

Working in stages, you can develop your deliverables iteratively and incrementally. Develop your analysis models, designs and code a little bit at a time. Use a version control system such as Git & GitHub so you don't lose your work, and also, it is good practice to use version control systems in software engineering. As part of your submission, you are required to submit a screen grab of your commit log with names and email matching the details that are held for you by the University.

You have to make it obvious that your analysis models reflect the provided scenario, and that your architecture and detailed design are a realistic match of your analysis model. The extent to which your designs are achievable will be tested by your implementation of them. Hence the constructs in your implementation(e.g., classes, methods) should closely match your design model.

If you run out of time, submit a working program which implements some significant aspects of your design. You will get very few marks for handing in lots of code that does not run correctly.

Read this document several times before starting work, to ensure that you understand what is involved, and so also you can make informed decision on where to start.

Overview of the scenario

Various people take part in a competition. You can choose the type of competition – it could be anything, from ice skating, tennis, football (soccer), athletics or some electronic game.

A system is required to keep details of a competition, which has various categories for competitors. Before the competition, competitors register by filling in their details using an online form or using an app on a mobile device. On the day of the competition, competitors are awarded scores for their performance. These scores are typed into the system by a member of staff, who first searches for the competitor using their number.

When all the people in a given category have competed, a staff member requests details of the results..

After the competition is over, competitors and staff can search for a particular competitor using their number, and view their details, including the basic and overall scores. They can also print out various summary reports.

Here is a bit more detail about the registration process. Competitors enter their name, email, date of birth, category and level, and are supplied with a unique competitor number. If any fields are omitted, an error message is displayed and the competitor is asked to resubmit the form. If a competitor already exists with the same email address and the same category, the registration is refused. If a competitor already exists with the same email address and for a different category, their registration is accepted and they are allocated a different competitor number for this category. If a competitor's age is incompatible with the level, the competitor is offered the opportunity to resubmit the form for a different level. If everything is ok, the competitor is allocated a customer number and the registration is accepted.

Again, each competitor gets a fixed number of scores, and a calculation is done with these scores to determine their overall score. The staff managing the competition require an intuitive interactive GUI-based that has a web and mobile interface. Typically, staff record competitor details, the competition they are taking part in, and their scores. Moreover, not all staff have the same access level to the system. Also, officials should be able to register competitors on arrival, or remove competitors who have not showed up, or failed to meet competition rules, or amend competitor details should anything have changed since their initial registration. On-site fans and other audience can watch proceedings as competitions progress, and those not close enough can follow proceedings displayed on screens strategically located around the competition venue. Besides referees and data entry staff, there are emergency response services present to provide support to the competitors, staff and audience.

In this assignment, you will design and create a realistic application to hold details of competitions, competitors, staff, and produce a report including details of all competitors, the winner, and some summary statistics. The reports can be in the form of a chart (visualization) or console output. Data persistence is important just in case the sports administration team want to perform some performance analysis after the event. You should not use any proprietary database technology for data persistence - CSV or JSON files should suffice.

Example of input data pertaining to competitors:

100, Keith John Talbot, Novice, 5,4,5,4,3

101, Jane Macdonald, Novice, 3,3,3,2,4

. . . .

106, Karen Harding, Expert, 5,5,5,4,4

107, Sarah Green, Expert, 4,4,5,4,3

Example report:

Competitor	Level	Scores	Overall
100 Keith John Talbot	Novice	5 4 5 4 3	4.2
101 Jane Macdonald	Novice	3 3 3 2 4	3.0

. . . .

106 Karen Harding	Expert	5 5 5 4 4	4.6
107 Sarah Green	Expert	4 4 5 4 3	4.0

Full details for 100:

Competitor number 100, name Keith John Talbot.

Keith is a Novice and received these scores : 5,4,5,4,3

This gives him an overall score of 4.2.

Short details for 106:

CN 106 (KH) has overall score 4.6.

STATISTICAL (*you have some choice as to what to include here*)

There are 9 competitors

The person with the highest score is Karen Harding with a score of 4.6.

The following individual scores were awarded:

Score : 1 2 3 4 5

Frequency: 2 7 10 12 8

Stage One – Analysis Model and Class design and setup

This is a domain analysis stage. The outline of the scenario provides initial cues to guide you gain some insights into the nature of the problem. You are to provide a comprehensive use case diagram depicting:

- 1) The use cases (functions) provided to the actors of the system. Consider possibility for using relationships among use cases (<<extends>>, <<include>>) in order to structure your use cases in a flexible and more maintainable way.
- 2) The actors(external system users) interested in the use cases.

Now that you have a use case model, consider the domain and solution-oriented classes that will form part of the system design. This is a design class diagram. As an example , it is expected that one of the classes is the **Competitor class**, with instance variables:

- Competitor number
- Competitor name
 - Consider a name to be a user defined class rather than firstname, middle name, and surname being attributes inside Competitor class

(Don't have any scores at all at the moment)

- Level of competitor – Invent your own levels. E.g. Novice, Standard, Veteran or 1, 2, 3 ,4. These are fixed values for each person, they should not be derived from their overall score.
- Other relevant attributes (e.g. age, country.....)

You should provide a design class diagram depicting class name, main operations and attributes, including relationships among the classes.

Stage Two: Architecture

You now need to consider the way in which your system will be realized, in a big picture manner. Here, you need to consider the subsystems that will be part of your system and how they will interact (dependencies between them). You are to produce a 3-tier architecture of the system by considering what subsystems will be part of each tier and how they interact across and within tiers. Construct a 3-tier architecture of the system showing clearly what subsystems are in the data-tier, logic and presentation tiers. Use appropriate notation (UML) ensuring relationships between subsystems within a layer are clearly indicated, and relationships between subsystems across layers are also clearly delineated.

Submission Requirements:

Submit your work as a single pdf document. Your diagrams should be converted into pdf format and submitted together in one file bearing the name following this pattern **your_student_number.surname_part1.zip**.


Marks awarded for:

- **Use case diagram - 15**
 - expected use cases identified, and named intuitively
 - relationships among use cases used where appropriate
 - actors identified and named intuitively
- **Architecture model - 20**
 - subsystems in the 3 tiers identified, and have suitable names and dependencies between them
 - subsystems are informed by requirements (analysis model)
- **Class diagram - 15**
 - class names are intuitive, and expected detail (attributes and methods)
 - Class relationships are present
 - Also, an indication (brief textual description suffices) of which classes comprise which subsystems in the architecture

Type of Feedback to be given for this assignment:

Feedback will include scores for each part as outlined above and comments from the marker.

Additional information:

- Regulations governing assessment offences including Plagiarism and Collusion are available from
https://www.herts.ac.uk/_data/assets/pdf_file/0007/237625/AS14-Apx3-Academic-Misconduct.pdf 
(https://www.herts.ac.uk/_data/assets/pdf_file/0007/237625/AS14-Apx3-Academic-Misconduct.pdf)(UPR AS14).
- Guidance on avoiding plagiarism can be found here:
<https://herts.instructure.com/courses/61421>
(<https://herts.instructure.com/courses/61421>)(see the **Referencing** section)
- For **undergraduate modules**:
 - o a score of 40% or above represents a pass performance at honours level.
 - o late submission of any item of coursework for each day or part thereof (or for hard copy submission only, working day or part thereof) for up to five days after the published deadline, coursework relating to modules at Levels 0, 4, 5, 6 submitted late (including deferred coursework, but with the exception of referred coursework), will have the numeric grade reduced by 10 grade points until or unless the numeric grade reaches or is 40. Where the numeric grade awarded for the assessment is less than 40, no lateness penalty will be applied.

Use case diagram

Criteria	Ratings		Pts
<p>Use cases, use case relationships, actors</p> <p>The expected use cases are identified, and named intuitively. The use cases match the scenario.</p> <p>Relationships among use cases (e.g., <<extend>>, <<includes>>) used where appropriate.</p> <p>Also actors identified and named intuitively.</p>	<p>15 Pts</p> <p>Full marks</p>	<p>0 Pts</p> <p>No marks</p>	<p>15 pts</p>
<p>3-tier architecture</p> <p>The subsystems in the 3 tiers are identified, and have suitable names and dependencies between them</p> <p>The subsystems are informed by requirements (analysis model - the use case diagram)</p>	<p>20 Pts</p> <p>Full marks</p>	<p>0 Pts</p> <p>No marks</p>	<p>20 pts</p>
<p>Class Diagram</p> <p>Class names are intuitive, and has expected detail (attributes and methods).</p> <p>Class relationships are present.</p> <p>Also, an indication (brief textual description suffices) of which classes comprise which subsystems in the architecture.</p>	<p>15 Pts</p> <p>Full marks</p>	<p>0 Pts</p> <p>No marks</p>	<p>15 pts</p>
Total points: 50			