School of Computing Sciences

Module: **CMP-5012B Software Engineering**
Assignment: **Reassessment 001**
Set by: Jaejoon Lee (MO) – jaejoon.lee@uea.ac.uk
Checked by: Temitope Alade – T.Alade@uea.ac.uk
Date set: 15 July 2024
Value: 45%
Date due: 09 August 2024 15:00
Submission: Blackboard (report in **PDF** format)

Description:

The application is a **Mental Health Tracker** app designed by using a MVC design pattern. **It is an INDIVIDUAL assignment and should not be done in group.**

Analyse the Mental Health Tracker project specifications below and develop a project requirements and design report.
Your report should contain the following (at minimum):

1. Analysis of two similar mental health apps (similar systems analysis) with a clear feature matrix that includes your system.
2. One use case diagram and two use case tables about the system.
3. 'Shall' statements that clearly state the functional and non-functional requirements. They should cover all the project specifications.
4. Test case table for the identified shall statements.
5. A UML class diagram showing all identified classes and their relationships with multiplicity. Also classify them into three group: Model, View, Controller.

Deliverables:

- A requirements and design report of no more than 2,000 words but no (word) limit on pages, diagrams, references, tables and figures. You should use the 'Requirements and Design Report_template_reassessment.docx' template.

Mental Health Tracker project specifications:

A customer wants to have an app with the following features. The app should:

1. Support registration of new users and authentication of existing users with a unique user ID and encrypted password permanently stored in a database or csv file.
2. Record and store the gender, DOB, age, height in cm and weight in kg of a user at registration.
3. Record and store the weight of an existing user at a certain date:

3.1 Make sure that the date is not in the future and does not backdate more than a certain reasonable period (e.g., one week).
3.2 Make sure the weight is in kg.

4. Record and store the mood of an existing user at different times for a certain date:
4.1 Make sure that the date is not in the future and does not backdate more than a certain reasonable period (e.g., one week).
4.2 Use an emoji and/or colour to categorise five different moods ranging from very bad to very good.
5. Record and store the hours and minutes (e.g., five hours and thirty minutes) of sleep of an existing user at a certain date:
5.1 Make sure that the date is not in the future and does not backdate more than a certain reasonable period (e.g., one week).
6. Display three bar plots all with a common horizontal axis that cover a period ranging from a user-selected past date to the latest recorded date that shows per day on each of the three following vertical axes:
1) your average mood (averaged over the number of times you recorded it during that day).
2) BMI = weight in kg / (height in m)^2.
3) hours of sleep.
with the date on the horizontal axis.

Marking scheme:

1. Introduction and Project Ideas [10% marks]
1.1 Main objective and overview
1.2 Analysis of similar systems
1.3 Feature matrix of similar systems

2. Requirements Analysis [50% marks]
2.1 Use Case Diagram
2.2 Use Case Table
2.3 Shall-Statements
2.4 Test Cases

3. Object-Oriented Design [40% marks]

Guidelines for marks:

70%+: Exemplary range and depth of attainment of the learning objectives related to the item. The writing is very clear and concise. (Excellent/Outstanding/Beyond Expectation)

60-70%: Conclusive attainment of nearly all learning objectives related to the item. The writing is well structured and expressed. (Good – Very Good)

50-60%: Clear attainment of most of the learning objectives related to the item, some more securely grasped than others. (Average – Good)

40-50%: Acceptable attainment of the learning objectives related to the item but shows clear weaknesses in describing the item. (Poor)

< 40%: Little, no or very poor attempt to describe the item. (Very Poor)

A note on use of external sources and plagiarism

Please note that while use of texts, or online sources, is encouraged in order to 'learn' design and programming principles, use of functions, lists, etc.; they are not a substitute for completing the work yourself. It is not appropriate to find solutions or part solutions to assignments and submit them as your own work. Neither is it allowed to post questions on online forums requesting help or solutions to specific assignment tasks. To do either (copying/requesting) would be in breach of the university's regulations on plagiarism and collusion (General Regulation 18).

In the instances where you do use code (or any other work) copied from any source, you must acknowledge the source (e.g., including a comment with the URL and author alongside the copied sections). If in doubt, approach the coursework setter to discuss what is appropriate.