MAPD714 – iOS Development

Final Exam
BMI Calculator App

Due: Week #14 (Friday December 17, 2021) @ midnight

Value: 20%

Approximate Time: 2 to 3 hours **Maximum Mark: 100**

Overview: Working on your own and using **Xcode** and the **Swift** programming language create a UI for a simple iOS **BMI tracker app.** The app will include at least **two screens**, the **first screen** (the **Personal Information Screen)** will initially appear when the app is first started. It will allow the user to enter their **personal information** including **Name**, **Age**, **Gender**, **Weight** and **Height**. The user should be able to enter their **weight** and **height** using either Imperial or Metric units. Once they **submit** their personal information, they will be shown their **BMI score**. The App should also display a **BMI message** that indicates what category the user falls within for their **Current BMI Score** (Use the table below as a guide).

Category	BMI range - kg/m^2
Severe Thinness	< 16
Moderate Thinness	16 - 17
Mild Thinness	17 - 18.5
Normal	18.5 - 25
Overweight	25 - 30
Obese Class I	30 - 35
Obese Class II	35 - 40
Obese Class III	> 40

The Formula to calculate a user's BMI is as follows

$$BMI = \frac{weightInPounds \times 703}{heightInInches \times heightInInches}$$

or

$$BMI = \frac{weightInKilograms}{heightInMeters \times heightInMeters}$$

A **second** screen (the **BMI Tracking Screen**) will track the user's **weight**, **BMI** and the **date** it was entered or updated. This should be displayed in a table format. The user should be able to **update** their weight by entering a new **weight** by **date**. When the user updates their **weight**, a new BMI will be calculated and the results will be recorded on the **BMI Tracking Screen**. This information should persist so that it appears any time the app is loaded and is not lost.

A user should be able switch between their Personal Information Screen and the BMI tracking Screen.

Instructions :

(32 Marks GUI, 48 Marks: Functionality, 6 Marks: Internal Documentation, 4 Marks Version Control, 10 Marks: Video Demo)

1. **Task 1:** The **Personal Information Screen** should include the following components and features **(11 Marks: GUI, 11 Marks: Functionality):**
    a. A UI that allows the user to enter their personal information including **Name**, **Age**, **Gender, Weight** and **Height.** The user should be able to **toggle** between **Metric** and **Imperial** units for their Weight and Height. This information should be recorded and stored so that it persists when the app reloads (7 Marks: GUI, 7 Marks: Functionality).
    b. A UI that calculates the users **Current BMI** and displays a **BMI message**. Please use the table above to display an appropriate message and to calculate the user's BMI properly. (3 Marks: GUI, 3 Marks: Functionality).
    c. A **Done** Button that takes the user to the **BMI Tracking Screen** (1 Mark: GUI, 1 Mark: Functionality).
2. **Task 2:** The **BMI Tracking Screen** should include the following components and features (**21 Marks: GUI, 21 Marks: Functionality)**:
    a. A **Table view** that displays the User's **Weight**, **BMI** and the **Date** they were updated on each Row. This data should persist when the app reloads (7 Marks: GUI, 7 Marks: Functionality).
    b. A **button** and/or other **controls** that allows the user to enter a new value for their **weight** and the **date** it was updated. This may require another screen or popup (7 Marks: GUI, 7 Marks: Functionality).
    c. A method to allow the user to update an entry (3 Marks: GUI, 3 Marks: Functionality)
    d. A method to allow the user to delete an entry. If the user Deletes all the entries then the app reroutes the user to the Personal Information Screen (4 Marks: GUI, 4 Marks: Functionality)
3. **Task 3: Data Persistence**. Ensure your app saves the user's data between loads (16 Marks: Functionality)
    a. Choose a Data Persistence method that you are comfortable with (e.g. CoreData, SQLite, User Defaults, Property List, Firebase, etc.) and ensure that the user's data is preserved between app reloads (8 Marks: Functionality).

    b. Your app should allow the user to enter new data (Write), Read from the data store, Update the data that is saved and remove (Delete) any data item (8 Marks: Functionality).
4. **Task 4:** Include **Internal Documentation** for your code **(6 Marks: Internal Documentation):**
    a. Ensure you include a **comment header** for your ViewController file that indicates: the **File name**, **Author's name**, **Student ID**, **Date** and **Changes made** (3 Marks: Internal Documentation).
    b. Ensure you include a **comment header** for each of your **methods** and **classes** (1 Marks: Internal Documentation)
    c. Ensure your program uses contextual variable names that help make the program human-readable (1 Marks: Internal Documentation).
    d. Ensure you include inline comments that describe your GUI Design and Functionality (1 Marks: Internal Documentation)
5. **Task 5:** Share your files on **GitHub** to demonstrate Version Control Best Practices **(4 Marks: Version Control).**
    a. Your repository must include **your code** and be well structured (2 Marks: Version Control).
    b. Your repository must include **commits** that demonstrate the project being updated at different stages of development – each time a major change is implemented (2 Marks: Version Control).
6. **Task 6:** Create a **Short Video** presentation with your favourite screen capture and streaming tool (OBS Recommended) and upload it to eCentennial. You must also include a short PowerPoint (or Google Slides) Slide Deck that includes a **single slide** to start your video **(10 Marks: Video Demo)**
    a. The first (and only) Slide of your Slide Deck must include a **current image of you** (no avatars allowed) that is displayed appropriately on the page. You must also include your Full Name, Student ID, the Course Code, Course Name, and your Assignment information. (2 Marks: video)
    b. You will **demonstrate** your app's functionality. You must show each screen working properly (2 Marks: Video)
    c. You will **describe** the code in your swift files that drives the functionality of your app (2 Marks Video).
    d. Sound for your Video must at an appropriate level so that your voice may be clearly heard. Your Screen should be clearly visible (2 Marks: Video).
    e. Your Short Video should run no more than 5 minutes (2 Marks: Video).

Optional App Features (i.e. Potential Bonus Marks).

A. A Splash Screen (1 Bonus Marks)
B. A colorful progress bar that changes in proportion to the user's BMI (2 Bonus Marks)
C. A **Reset** button that reset's the screen to its original state. (1 Bonus Mark)
D. Gestures used for CRUD operations (6 Bonus Marks)

SUBMITTING YOUR WORK

Your submission should include:

1. A zip archive of your Project files.
2. A link to your GitHub Repository

Feature	Description	Marks
GUI / Interface Design	Display elements meet requirements. Appropriate spacing, graphics, colour, and typography used.	32
Functionality	Site deliverables are me and site functions are met. No errors, including submission of user inputs.	48
Internal Documentation	File header present, including site & student name & description. Functions and classes include headers describing functionality & scope. Inline comments and descriptive variable names included.	5
Version Control	GitHub commit history demonstrating regular updates. 2 marks for simply pushing your files to GitHub once. An additional 2 marks awarded for using GitHub as you code.	4
Video Presentation	Your short video must demonstrate your app and describe your code	10
Total		100

This assignment is weighted **20%** of your total mark for this course.

External code (e.g. from the internet or other sources) can be used for student submissions within the following parameters:

1. The code source (i.e. where you got the code and who wrote it) must be cited in your internal documentation.
2. It encompasses a maximum of 10% of your code (any more will be considered cheating).
3. You must understand any code you use and include documentation (comments) around the code that explains its function.
4. You must get written approval from me via email.