Ira A. Fulton Schools of Engineering
Arizona State University

CSE 598: Advanced Software Analysis and Design

BMI Calculator Project Part 2

Purpose

In the second part of the BMI Calculator Project, learners will develop either an iOS app (preferred) or an Android app. Software architecture and web data integration are fundamentally important in mobile app design and implementation. This project provides learners with the opportunity to put their knowledge of MVC architecture and mobile web integration into practice. Learners will develop a simple BMI calculator application using MVC architecture, then use Web APIs to perform the same BMI calculation.

Objectives

Learners will be able to:

- Apply the MVC architecture to design a mobile app that satisfies given requirements.
- Implement an MVC architecture-based mobile app.
- Use Web API calls.
- Process JSON data in a mobile app.

Technology Requirements

- For iOS app: XCode 10, programming language Swift (no Objective-C) **(strongly preferred)**
- For Android app: Android SDK using Java
- A way to record your screen

Project Overview

Design and implement a mobile app that calculates BMI and displays the results to the user.

Project Description

In this second part of your BMI Calculator Project, you will create an application that, when given the height and weight of a person, calculates their BMI. You will use a web API call to calculate BMI and show the API call results to the user.

Directions

Develop an application that calls the calculateBMI API, which takes height and weight as parameters and returns the BMI JSON data that contains BMI, Risk Factor, and Array of web links with BMI information. Include screenshots of your app's outputs with your submission. All of this should be submitted as a single ZIP file.

Once you have created your app, create a video demonstration of your app's output. Your video must start from you opening your ZIP file, followed by compilation and run. You must demonstrate every possible test case scenario that you tested as well. Include this video in a separate ZIP file submission as either a YouTube link or an MP4.

API Call

You will be calling the calculateBMI API, which takes height and weight as parameters and returns the BMI JSON data that contains BMI, Risk Factor, and Array of web links with BMI information.

Your app should read the height and weight and call the API to calculate the BMI. See the example API call for a height of 60 inches and weight of 156 lbs in **Figure 1**.

Figure 1: Example API call for height = 60 in. and weight = 156 lbs.

http://webstrar99.fulton.asu.edu/page3/Service1.svc/calculateBMI?height=60&weight=156

The results from the API call is a JSON document structure:


```
{
"bmi":30.463333333333335,
"more":["https://www.cdc.gov/healthyweight/assessing/bmi/index.html","https:
//www.nhlbi.nih.gov/health/educational/lose_wt/index.htm","https://www.ucsfh
ealth.org/education/body_mass_index_tool/"],
"risk":"You are obese :("
}
```


Figure 1: Example API call for height = 60 in. and weight = 156 lbs.

User Interface (UI) Design

The main UI of your app must have all required components, and all components should have proper alignment and placement on the screen.

Required Components:

- Field to enter height (in inches)
- Field to enter weight (in pounds)
- BMI value display
- Risk message display
- Call BMI API button
- Educate Me button

Refer to **Figure 2: User Interface (UI) Design Guidelines**[1] to see an example of what the main UI of your app should look like. (Please note that weight should be given in **pounds**, not "inches.")

Figure 2: User Interface (UI) Design Guidelines

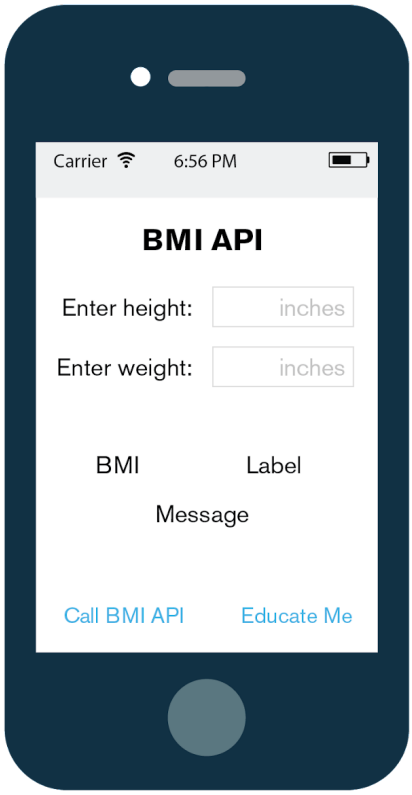

Figure 2: User Interface (UI) Design Guidelines

[1] Figure 2 is modified from Xcode 10 and originally developed using Swift 4.2.

App Requirements

Your app must process the JSON results and display BMI information to the user. Then, based on the BMI value, it must display one (1) of these risk messages to the user:

> *You are **underweight** if BMI is < 18: Blue Color*
>
> *You are **normal** if BMI is ≥ 18 and < 25: Green Color*
>
> *You are **pre-obese** if BMI is between 25 and 30: Purple Color*
>
> *You are **obese** if BMI is greater than 30: Red Color*

Additionally, when the user selects the button "Educate Me," your app should load a web page that shows additional information about BMI by using one of the web links from the JSON results (refer to **Figure 1: Example API call for height = 60 in. and weight = 156 lbs.**).

Submission Directions for Project Deliverables

You are given three (3) attempts to submit your best work. The number of attempts is given to anticipate any submission errors you may have in regards to properly submitting your best work within the deadline (e.g., accidentally submitting the wrong paper). It is **not** meant for you to receive two (2) rounds of feedback and then one (1) final submission. Only your most recent submission will be assessed. You must submit your BMI Calculator Project Part 2 as two (2) separate ZIP files in the designated submission space in the course.

Learners may not email or use other means to submit any project for review, including feedback, and grading. Your BMI Calculator Project Part 2 includes two (2) deliverables:

1. **Application ZIP file:** The application and output screenshots must be included in a **single ZIP file** with the correct naming convention: *Your Name_CSE 598 ASAD_BMI Calculator Project Part 2*.
2. **Demo Video ZIP file:** The video demonstration must be included as either a YouTube link or MP4 in a **single ZIP file** with the correct naming convention: *Your Name_CSE 598 ASAD_BMI Calculator Project Part 2_Demo.*

Evaluation

Please review the rubric for how your BMI Calculator Project Part 2 will be graded. Projects will be evaluated based on each criterion and will receive a total score. Projects missing any part of the project will be graded based on what was submitted against the rubric criteria. Missing parts submitted after the deadline will not be graded.

Review the course syllabus for details regarding late penalties.

Rubric

Rubrics communicate specific criteria for evaluation. Prior to starting any graded coursework, learners are expected to read through the rubric, so they know how they will be assessed. You are encouraged to self-assess your responses and make informed revisions before submitting your final report. Engaging in this learning practice will support you in developing your best work. Points may be deducted at the discretion of the faculty for disorganized submissions that convolute the grading process.

Component	No Attempt	Undeveloped	Developing	Approaching	Meets
User Interface (UI) Design	Provided no response.	The app's UI is missing most or all of the required components.	The app's UI is missing some of the required components.	The app's UI has all of the required components, but some or all of these do not have proper alignment or placement on the screen.	The app's UI has all the required components, and all components have proper alignment and placement in the screen.
API Call and JSON Data Processing	Provided no response.	The API call is incorrect, and the JSON processing is incorrect.	The API call is correct, but the JSON processing is incorrect.	The API call is correct, and there are no more than three (3) errors in the JSON processing.	The API call is correct, and the JSON processing is correct.
"Educate Me" Button Functionality	Provided no response.	The "Educate Me" button is not implemented.	The "Educate Me" button is present but the functionality is implemented incorrectly.		The "Educate Me" button is present and the functionality is implemented correctly.
App Demo Video	Provided no response.	Video does not demonstrate the learner's app.	Video demonstrates the learner's app, but may not begin from opening the ZIP file or show the compilation and run. Video does not include a demonstration of test case scenarios.	Video demonstrates the learner's app, beginning with opening the ZIP file and showing the compilation and run. Video includes a demonstration of very few test case scenarios.	Video demonstrates the learner's app, beginning with opening the ZIP file and showing the compilation and run. Video includes a demonstration of every possible test case scenario that the learner tested.