DALHOUSIE UNIVERSITY
Faculty of Computer Science

CSCI 5708 (Graduate)
Tami Meredith

Mobile Computing Marking Rubric: Assignment 1

Component	Unacceptable: 1	Minimal: 2	Good: 3	Really Good: 4	Exceptional: 5
Functionality	You can enter a bill amount and it displays.	One can calculate a basic 15% tip.	It can calculate any desired tip amount.	It can calculate any tip amount, errors can be corrected, and it functions correctly.	Perfect. It works exactly like it should in all respects. It's not some other sort of thing that didn't do as expected. It is smooth, responsive, and considers local concerns
Usability	It isn't finished.	It works correctly. Buttons are obvious.	It works correctly and is mostly attractive. It has error messages if, for some reason, errors are possible.	It is correct, reports errors well and accurately. It looks good and seems attractive.	Perfect. No room for improvement. It is attractive, effective, easy to use, and isn't bloated with extra stuff. It is clean and simple, not complicated. You can use it without any effort.
Documentation	There is some internal documentation. That is, the code has decent variable, method, and class names. There are a few comments.	The readme file is there and useful.	The code is well commented. The comments add value and explain things you can't see in the code.	As well as the readme file there is some kind of external documentation – design documents, wireframes, UML diagrams, test plans, etc.	It has outstanding internal documentation for programmers. All sources are properly referenced. There is clear evidence of design and testing (QA).
Design and Code.	It has a few classes and methods. It works but feels hacked together. (or) Libraries do most of the work.	It works and has some structure, but not a lot. (or) Libraries do much of the work.	It has an abstract data type or some sort of design model described. It could be easily extended (e.g., other languages/currencies).	It has abstract data types, it is well structured and has various parts encapsulated. E.g., an input module, a computation module, a UI module.	There is a strong design that is documented in some way (e.g., UML, text description).
Other Concerns	There are no spelling or grammar errors in the comments or documentation (-1 for sloppiness in documentation). It is a .zip file, not .rar, not a git link – they sent in a .zip file of code and no code is missing (-2 for incorrect formats). It compiles and executes. (Maximum of 10/20 for an assignment that doesn't compile or execute. -5 for run time failures). The readme file is present (-5 for missing documentation). Implements bill splitting (+2 to +4 depending upon quality).				

This assignment will have a mark out of 20. I expect the average score for a component to be somewhere between a 3 and a 4 (3.5 is the 70% pass that a graduate student needs).

Notes:

1. Structure of the code is important. I want to see some form of design and not just a single monolithic class. Remember: High Cohesion, Low Coupling.
2. Testing must be performed. For an assignment this simple, it is expected that test data and/or test plans will be provided.
3. It must work the way it's supposed to work. It has to be easy to use and intuitive. Do not bother with providing "Help and Documentation." If the application needs help, it is most likely too complex or poorly designed.
4. Using libraries is OK, but no marks will be given for anything a library does.
5. There must be references for anything looked up or downloaded. Using an unreferenced resource will result in a potential discipline for violating academic integrity. References should be placed in the README file.