SOEN 343 Fall 2020 D1
Grading Scheme

GINA CODY School of Engineering and Computer Science

Department of Computer Science and Software Engineering

Concordia University

SOEN 343--- Fall 2020

Project deliverable 1 grading scheme

Instructions for project delivery

You must deliver an operational version demonstrating a subset of the capacity of your system. This is about demonstrating that the code build is effectively aimed at solving specific project problems or completely implementing specific system features. The code build must not be just a "portion of the final project", but rather be something useful with a purpose on its own, that can be demonstrated by its operational usage.

You are graded according to how effectively you can demonstrate that the features are implemented. If you cannot really demonstrate the features through execution, you will have to prove that the features are implemented by explaining how your code implements the features.

To summarize, delivery 1 is comprised of the following elements:

- A written report in PDF and submitted through Moodle as a team (see report template)
- A demo recording explaining how to install, and execute the simulator, and a guided tour of the features implemented. If the size of your video is less than 250 mb, submit through Moodle. Otherwise, provide the link in Moodle
- Source code of the corresponding deliverable in a git repository of your choice. Remember to add a label in the Git repository named delivery <#> where # indicates the number of the delivery. You must provide the link and/or grant access to TA and Instructor

Grading

ITEM	DESCRIPTION	VALUE %	SCORE ACHIEVED
Report		**6**	
Presentation	The document is complete, clear and concise.	4	
Project template	The document follows the project template	2	
Programming process		**48**	
Architecture design documentation	The architectural design should be reflected in the implementation of well-separated modules and/or folders. Implement the simulator using the MVC architecture.	4	
	Use cases for the following features: House layout ● Read and load a house-layout file Simulation parameters ● Add/remove edit user profiles ● Set Date and time ● Log in using an existing user profile and set house location Context of the simulation ● Start/stop simulation ● Modify date and time ● Move the logged user to different room ● Place people in specific rooms, or outside home ● Modify temperature outside home ● Block windows movement by putting an arbitrary object	10	
	Diagrams include: ● Domain model class diagram (1, all the system)	8 10	

	● Class diagram (1, just the classes implemented) ● Sequence diagram (1, context of the simulation)	8	
Software versioning repository	Well-populated history with dozens of commits, distributed evenly among team members, as well as evenly distributed over the time allocated to D1. A tagged version should have been created for D1. Students will provide the link to the repository or grant access to TAs, and the instructor. Instructor Github user is **moar82**	2	
Javadoc API documentation	Completed for all files, all classes and all methods.	2	
JUnit tests	One-unit test class for each use case, clearly identified using the use case id from the architecture design documentation	2	
Coding Standards	· Consistent and proper use of code indenting, naming conventions and comments · Absence of “commented out” code · See https://google.github.io/styleguide/javaguide.html	2	
Functional Requirements (software prototype running)		**42**	
House layout	Read and load a house-layout file (without exception handling that will be marked in delivery 3)	10	
Simulation parameters	SHS tab with the following options: · Add/remove edit user profiles · Set Date and time · Log in using an existing user profile and set house location	10	
Context of the simulation	The edit button in the simulation group box opens a form that allows users to perform the following actions: ● Place house inhabitants in specific rooms, or outside home ● Block windows movement by putting an arbitrary object	7	

Smart Home dashboard	· Smart home simulator user interface · Simulation group completely working including on/off, edit simulation buttons, and links to change current user, location, outside temperature, and date and time	15	
Miscellaneous		**4**	
Simulator demo	· Record a demo of how to install and execute the simulator and a tour of the functionality that you developed for this delivery	4	
TOTAL		**100**	

Report template

1. Cover page including the name of the team, student names, student ids, and TA section
2. Table of contents
3. Introduction
4. Deliverable 1 scope.
5. Use Cases table format as seen in class
6. Diagrams clearly cross-referenced with use cases

Remarks

In case of late submission, you will lose 2% for each day of delay.

References

1. How to Design a Java Framework? - A simple example https://www.programcreek.com/2011/09/how-to-design-a-java-framework/
2. McLaughlin, Brett, Gary Pollice, and David West. Head First Object-Oriented Analysis and Design: A Brain Friendly Guide to OOA&D. " O'Reilly Media, Inc.", 2007. https://resources.oreilly.com/examples/9780596008673/tree/master
3. Home I/O Simulation of a smart house and surrounding environment https://realgames.co/home-io/
4. OpenSHS: Open Smart Home Simulator https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5469526/

Appendix A. Delivery Demo

The video will be divided in two sections. Follow the steps in the order they are presented.

1. Installation

 - Specify installation requirements with their corresponding versions of JDK, Build system (e.g., Ant, Maven, Graddle, etc)
- Please specify any external libraries or dependencies used.
 - Explain in a detailed manner how to clone, build and deploy your application

2. Simulator Execution & Feature Implementation

 - Provide a clear demonstration of MVC structure using the source code.
 - Provide a clear demonstration of all other Architectural design requirements.
 - Provide clear demonstration of all functional requirements in the grading scheme.
 - Demonstrate the use cases implemented for this delivery with their corresponding unit tests (JUnit)

Dr. Morales. Page