Object-Oriented Systems Development

Session 2017-18 Group Coursework

Module Code	:	UFCFB6-30-2
Due Date	:	Group assignment hand in and demo Thursday March 1, 2018
Total Marks	:	This coursework is worth 100 marks representing 40% of your total course grade.
Type	:	Group (3 students)
Instruction	:	This assignment is to be completed in groups of three students. Do not show your design and code to any other group and do not look at any other group's design and code.
Aim of the courework	:	identify situations where object oriented problem analysis, system design and programming paradigms are applicable and to create software that exploits them.
The learning outcomes	:	• Understand and use a UML modelling tool and a Java-based Interactive Development Environment (IDE) to develop object-oriented software system • Design and implement Graphical User Interfaces • Apply good practice in code design/testing
Submission	:	Please submit a portfolio as a ZIP file with your code and a PDF document containing project vision and scope, Use Case diagram, Class diagram, Sequence diagram, Agile practices, Test cases used in your program in the form of a table, suitable program running screenshots capturing success/failure scenarios.
Demonstration	:	All the group members must be present during the coursework demonstration.
Marking scheme		Please see at the end of this document.

All the tasks are based on the following scenario. Please read carefully all information contained within the following passage of text.

UWE Bristol Accommodation System

The main responsibility of the UWE accommodation office is to provide the necessary help for all the registered students who are entitled and require their accommodation on-campus. The on-campus accommodation comprises of many halls of residence, and each hall has a number of rooms. The accommodation office has a hall manager who supervises the operation of the halls. Each hall has a warden who oversees the regular cleaning and maintenance of all the rooms in that hall. Each hall of residence has a name, number, address, telephone number. The halls provide only single rooms which have a room number and monthly rent rate. The total number of rooms provided by the accommodation office should also be available. The hall number uniquely identifies each room in all of the halls controlled by the accommodation office, and is used when renting a room to a student.

The UWE Bristol Accommodation Services allow students renting rooms for the entire 12-month academic year from September to August. Each individual rental agreement between a student and the accommodation office is uniquely identified using a lease number. The data stored on each lease includes the lease number, duration of the lease (in months), address details of the hall, room number, student's name and ID number.

The room scheduling is coordinated by the hall manager. That is, the hall manager generates and maintains the room schedule, keeps a record of all the students staying in the halls and reviews applications for future bookings. The hall manager can edit and view room details. The view should show all the detailed information about a single room in a hall, including its room number, room status (namely, occupied or unoccupied), monthly rent rate and other details describing the room. The hall manager should be able to determine from this description whether a particular room is available and its suitability for a new student to occupy. On the other hand, a warden can view the room details but he has limited editing privileges and can only change a room’s cleaning status which can be “clean”, “dirty”, or “off-line”. An off-line room is one that cannot be occupied because it requires maintenance beyond a normal cleaning.

You are the leader of a team of three developers who have been asked to design and implement a system which enables the hall manager and wardens to schedule hall activities and keep track of the hall rooms in a simpler and easier way.

Attributes	Marks Allotted
1. Project description	8
2. Use Case diagram	8
3. Class diagram	8
4. Sequence diagram	8
5. Agile practices	8
6. Coding and testing	60
Total	100

1. **Project description**
   Write the corresponding project description.

2. **Use Case**
   Provide a graphical UML use case diagram of this system. Your diagram must contain at least 3 actors, 6 use cases, 2 include, and 2 extend relations.

3. **Class diagram**
   Identify all the potential classes and draw a class diagram. Your diagram will show appropriate relationships between the classes, multiplicity specifications, and other model elements that you find appropriate.

4. **Sequence diagram**
   Draw one substantial sequence diagram that adds information.

5. **Agile practices**
   Outline the typical steps that you would follow to develop the system in relation to an agile development episode.

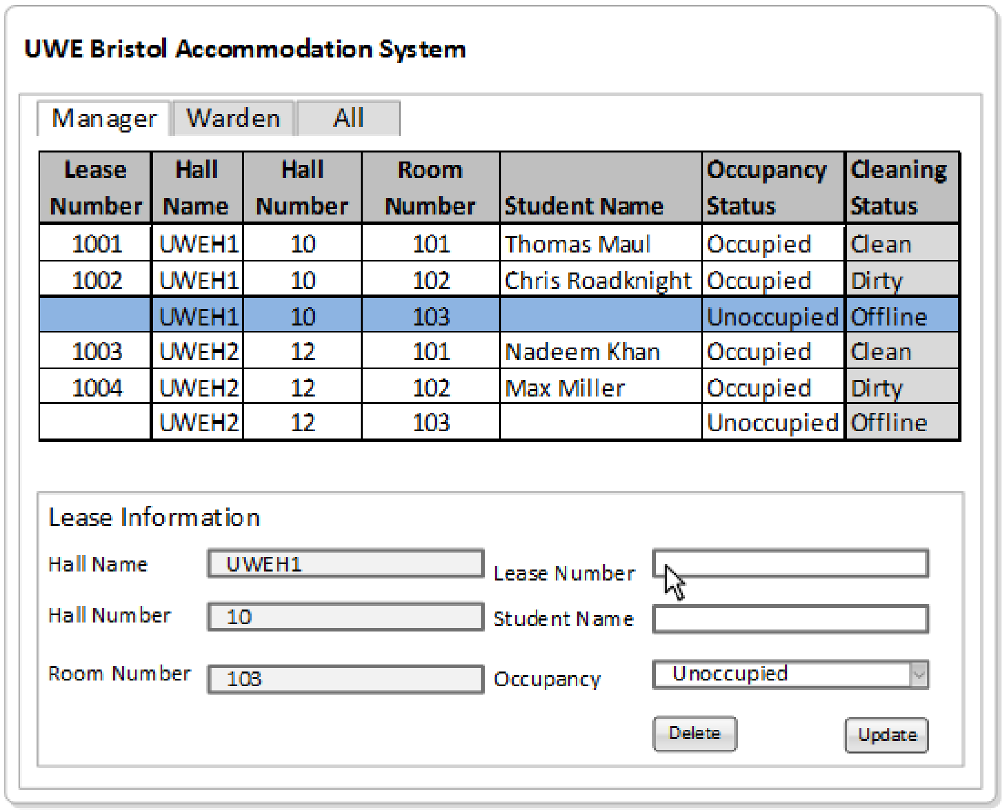
6. Coding and testing

Develop the above system using Java (and JavaFX/Swing) considering the following functionalities. You should be creative and come up with your own User Interface Design (following design shows just an example).

A user interface that lets

1. the manager to view and edit room details. For example, assume that there are 2 halls, each having 3 rooms, and rooms are either occupied or off-line at the moment.

   When the manager selects and edits a particular room, GUI will look like the following (you can add more fields if you like, e.g., student ID, lease duration etc.):



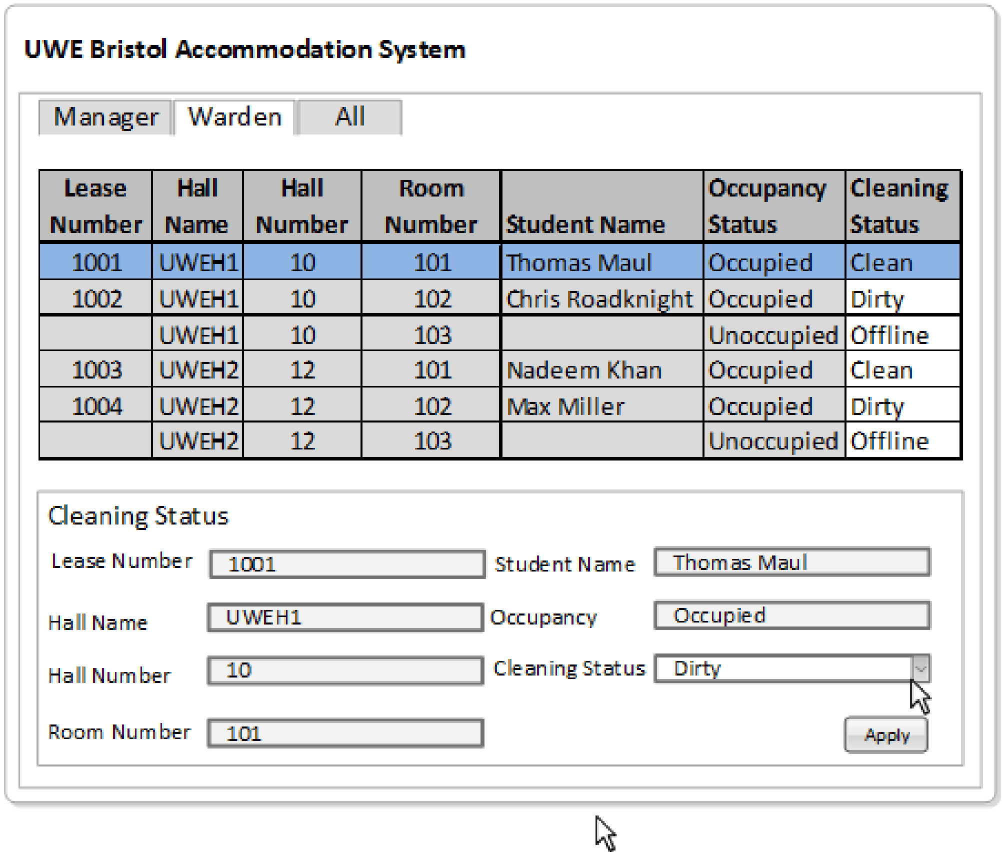
Lease Number	Hall Name	Hall Number	Room Number	Student Name	Occupancy Status	Cleaning Status
1001	UWEH1	10	101	Thomas Maul	Occupied	Clean
1002	UWEH1	10	102	Chris Roadknight	Occupied	Dirty
	UWEH1	10	103		Unoccupied	Offline
1003	UWEH2	12	101	Nadeem Khan	Occupied	Clean
1004	UWEH2	12	102	Max Miller	Occupied	Dirty
	UWEH2	12	103		Unoccupied	Offline

Lease Information

Hall Name	UWEH1	Lease Number	
Hall Number	10	Student Name	
Room Number	103	Occupancy	Unoccupied

He can then enter the required information and update accordingly. Please note that in the above selection, the manager cannot create a lease because the room is currently “off-line”. A lease can only be created if the occupancy status is “unoccupied” and cleaning status is eithr “clean” or “dirty”. Similarly, the manager can select and delete a lease. This will make the lease number and student name fields null/empty and the occupancy status as “unoccupied”. However, the other fileds will remain the same.

2. a warden to view and edit room details. The view and edit should be similar to the above; however, a warder can only change the cleaning status of a room and cannot delete a lease. For example, he can select and change the cleaning status from “clean” to “dirty” of the lease 1001.


UWE Bristol Accommodation System

Lease Number	Hall Name	Hall Number	Room Number	Student Name	Occupancy Status	Cleaning Status
1001	UWEH1	10	101	Thomas Maul	Occupied	Clean
1002	UWEH1	10	102	Chris Roadknight	Occupied	Dirty
	UWEH1	10	103		Unoccupied	Offline
1003	UWEH2	12	101	Nadeem Khan	Occupied	Clean
1004	UWEH2	12	102	Max Miller	Occupied	Dirty
	UWEH2	12	103		Unoccupied	Offline

Cleaning Status

Lease Number	1001	Student Name	Thomas Maul
Hall Name	UWEH1	Occupancy	Occupied
Hall Number	10	Cleaning Status	Dirty
Room Number	101		

3. (**optional , but highly encouraged**) for the case of "All" as is labelled it on the GUI design, you may wish to consider this view contains a superset of the other views and allows a user to perform all actions.

Testing:

Testing is typically a part of the program development – you should use a test strategy to test your program thoroughly. You should identify all the suitable test cases for all the classes implemented. When you test your code, you should make sure that your program does not allow bad data to be stored into your objects. Deliberately feed in your program out of range or wrong data and try to make it fail, see if you can swipe bad values into the member variables. One example test case shown below.

Test Case	Purpose	Expected result
occupancyStatus is occupied	To create a lease, occupencyStatus must be unoccupied	Appropriate error message

Marking scheme

1. The project description describes how the idea is turned into a product

None	*Inadequate*	*Adequate*	*Complete*	
0 %	**3 %**	**5 %**	**8 %**	
☐	☐	☐	☐	=

2. The use case diagram captures functionalities of the system

None	*Inadequate*	*Most*	*All*	
0 %	**3 %**	**5 %**	**8 %**	
☐	☐	☐	☐	=

3. The class diagram provides a static view of the system

None	*Inadequate*	*Most*	*All*	
0 %	**3 %**	**5 %**	**8 %**	
☐	☐	☐	☐	=

4. The sequence diagram describes interaction among classes

None	*Inadequate*	*Most*	*All*	
0 %	**3 %**	**5 %**	**8 %**	
☐	☐	☐	☐	=

5. The Agile practices outlined

None	*Inadequate*	*Adequate*	
0 %	**3 %**	**8%**	
☐	☐	☐	=

6. Suitable test cases have been identified and documented

None	*Inadequate*	*Most*	*All*	
0 %	**5 %**	**7 %**	**10 %**	
☐	☐	☐	☐	=

7. COMPLETED program compiles and runs

None	*Only partially*	*All*	
0 %	**3 %**	**10 %**	
☐	☐	☐	=

8. GUI displays input and output messages correctly

None	*Inadequate*	*Most*	*All*	
0 %	**3 %**	**7 %**	**10 %**	
☐	☐	☐	☐	=

9. Overall group demonstration

Absent	*Poor (Did not know how to run the system)*	*Good (Able to run the system)*	*Excellent (Able to show additional/new ideas/codes)*	
0 %	**3 %**	**7 %**	**10 %**	
☐	☐	☐	☐	=
Group assessment percentage (out of 80%):				

10. **Individual Q&A**

Absent	*Inadequate (barely able to explain the codes and/or work done)*	*Good (good explanation of codes and/or work done)*	*Excellent (excellent explanation of codes and/or work done)*	
0 %	**7 %**	**15 %**	**20 %**	

Individual assessment percentage (out of 20%)

Student's name	**%Marks**
1.	
2.	
3.	

1.	**Mark (out of 100) =**	**Mark (out of 40) =**
2.	**Mark (out of 100) =**	**Mark (out of 40) =**
3.	**Mark (out of 100) =**	**Mark (out of 40) =**

Comments: