School of Computing Technologies

COSC2391 Further Programming, sem 1 2021

Assignment 2

✳	Assessment Type: **Individual assignment; no group work permitted**. Submit online via Canvas. Marks are awarded for meeting requirements as closely as possible according to assignment specifications and the supplied rubric. Clarifications/updates will be made via announcements/relevant discussion forums.
📅	Due date: **Multiple milestone submissions. Final submission due 11:59 pm June 06, 2021.** No late submissions accepted unless Special Consideration is granted in advance. This submission date is only for final code submission (not milestone reports)
⏬	Weighting: 50% of your final grade. In addition, multiple milestones through the semester to a total of 15 marks.

1. Outline

NOTE: Carefully read this document. In addition, regularly follow the Canvas assignment discussion board for assignment related clarifications and discussion.

In this assignment you are required to work individually to develop a Hotdesking System using Java. Using Java as a vehicle, you are required to demonstrate your understanding of object-oriented design principles,design patterns, generics, graphical user interfaces, unit testing and object relational mapping building on the foundations laid by the Programming Fundamentals course and further developed in COSC2391. The system you develop should be maintainable and extensible as well as produce substantial value for the stakeholders by improving the level of patient care. The marks will therefore be awarded both for your functionality as well as justifying the design decisions and code quality. During the final face to face assessment you will also be required to justify your design decisions as well as explain how your program can be extended to meet other related requirements.

This assignment is structured to let you incrementally build your product throughout the semester by incorporating the OO design principles, Java constructs and technologies introduced in the notes and videos on Canvas, and developed in the lecturorial and prac sessions. You are required to ensure you make steady progress and marks will be awarded in the prac classes for completing specific milestones. The main purpose of this assignment is to develop and demonstrate your problem-solving skills by applying the concepts taught for a reasonably complex open-ended problem (and to create a complete working system based on agile specifications). You should not be overly concerned about every minute detail and are free to make reasonable assumptions (you may check with your lab-tutor if in doubt, who will also mark your final assignment).

Disclaimer: the specification in this assignment is intended to represent a simplified version of a system in real life and thus is not meant to be an accurate simulation of any real system or service that is being used commercially.

2. Assessment Criteria

This assessment will determine your ability to implement Object-Oriented Java code according to the specifications shown below. In addition to functional correctness (i.e. getting your code to work) you will also be assessed on code quality. Specifically:

- You are required to demonstrate your understanding of object-oriented programming constructs encapsulation, inheritance and polymorphism.
- You are required to promote performance and robustness by using well tested collections and by catching and handling all exceptions.
- You are required to demonstrate that you have done adequate unit testing considering various business rules using the JUnit framework.
- You are required to demonstrate all the objects stored in memory can be persisted to a file and restored when the program starts again.
- Your interaction design must anticipate what users might need next and include appropriate JavaFX elements to make the system accessible, intuitive, usable and efficient (fewer clicks/keystrokes)
- You are required to ensure all historic data can be archived to a relational database allowing access by external software and the current system.
- Your object-oriented design should provide high cohesion and low coupling following well established design principles and patterns. You should justify your design decisions that led to the final product using detailed comments in the beginning of each class.
- You should reflect on the difficulties/limitations of the technologies used and suggest how these can be improved by researching and identifying other technologies and tools in a separate one-page report.

Further specifications and assessment criteria will follow throughout the semester.

3. Learning Outcomes

This assessment is relevant to the following Learning Outcomes:

CL01: use the Java programming language in the implementation of small to medium sized application programs that illustrate professionally acceptable coding and performance standards.

CL02: demonstrate knowledge of the basic principles of the object-oriented development process and apply this understanding to the analysis and design of solutions for small to medium scale problems.

CLO3: describe and apply basic algorithms and data structures, in particular simple searching and sorting of data stored in data structures and manipulating data.

CLO4: implement basic event-driven programming and graphical user interfaces in Java.

4. Product Description

You are required to develop a system to manage 'Hotdesking' methodology to allocate tables and seats to Arub employees. It also serves the purpose of preventing the same users from sitting in the same spot for multiple days so that they can socially interact with different people every time. The 2020 Pandemic has led to a new problem of distancing each other, which can be solved using this solution by essentially preventing booking of certain seats.

The implementation must be accessible on the user's desktop for ease of use and availability.

Functional Requirements to be Carried out through a Graphical User Interface

- User profiles:
    - login

    Already has been implemented on base-code. However you should update the Employee table, model and controller based on requirement in register (below)

    - register

    A form that letting Employee to register with filling by below data:

    Employee id, First name, Last name, Role, Username, Password, Secret question, Answer for Secret question

    - reset password

    An option for login, if user forget the password, can choose reset password and reset password will ask a secret question. If user answer is correct then it shows a new random password so user can login. (Hint: new password should be updated in database)

    - booking

    User can book a sit for any date.

    Conditions:

    User cannot have multiple date booking, only one booking at the time, however the day/time can be any date in the future.

    User cannot book the same sit that has been booked previously. (check only previous one).

    You can use a Whitelist technique for this one to have a whitelist for each employee means, which desks/sit they can book at each time. (This is an optional technique, if you have a better way to handle this condition, you can use your own way)

    Out of scope: It's not important if other are booking in the way that can sit beside each other again.

    Admin user can add/delete/update/deactivate Employee accounts

    - Booking management for tables for users
    Make booking

Cancel Booking

Update booking at 48 hours before, after 48 hours, user cannot change the booking

User can check-in via this system when they sit behind the desk. (no check-out)

- Admin Profile:
  - o Admin booking management

Admin user can confirm booking. There is time validation framework for each booking (release them), means each booking can be confirmed a day before and if any tomorrow booking has not been confirmed by midnight before tomorrow then it would be cancelled automatically.

Admin user can cancel/reject booking.

Admin can make booking with a one desk space between employee based on COVID conditions. Admin can lockdown selected sits to achieve this goal.

Admin can lockdown all sits based on COVID Lock down

Hint: you have normal booking (everyone can sit on all desks), COVID condition that everyone should sit with an empty space in between, and COVID lockdown which doesn't let anyone book a sit.

  - o Admin account management

Admin can add/update/delete other admin users (Employee id,name, surename, username, password, role, secretquestion, answerforsecretquestion)

Admin can add/update/delete Employee

Admin can manually change the whitelist (if you are using this technique to avoid same sit booking)

  - o Admin Report

Generate a CSV report from booking with date

Generate a CSV report from all Employee information

- View current table allocation for Booking

  - o Graphical Visual

    A graphical visualization of the room and desks should be used for booking and manage booking. Look at the figures(on next page)

Both Employee and Admin use the same visual
- User for booking as above green or red box explanation
- Admin for having an overal view of the room, cancel booking(red turns to green) or lock a sit (green turns to red)
- Admin can lock sits for COVID reason (the color turns to orange). There is no covid locked in above figure.

Available desks in COVID condition with social distance space. In this example no-one booked a table yet.

System Architecture

*Hint: please ignore the firebase, that would be SQLite for this assignment. This diagram is **showing an overview of the system** (means not all details), and its behavioral diagram, which **is not looking into class relationship or data structure**. Extend and Include in here have different meaning and does not mean inheritance. Include mean a behavior that is always happening as a result of main use-case(inside oval), and extends mean based on some condition. All these behaviors has been explained in previous page.
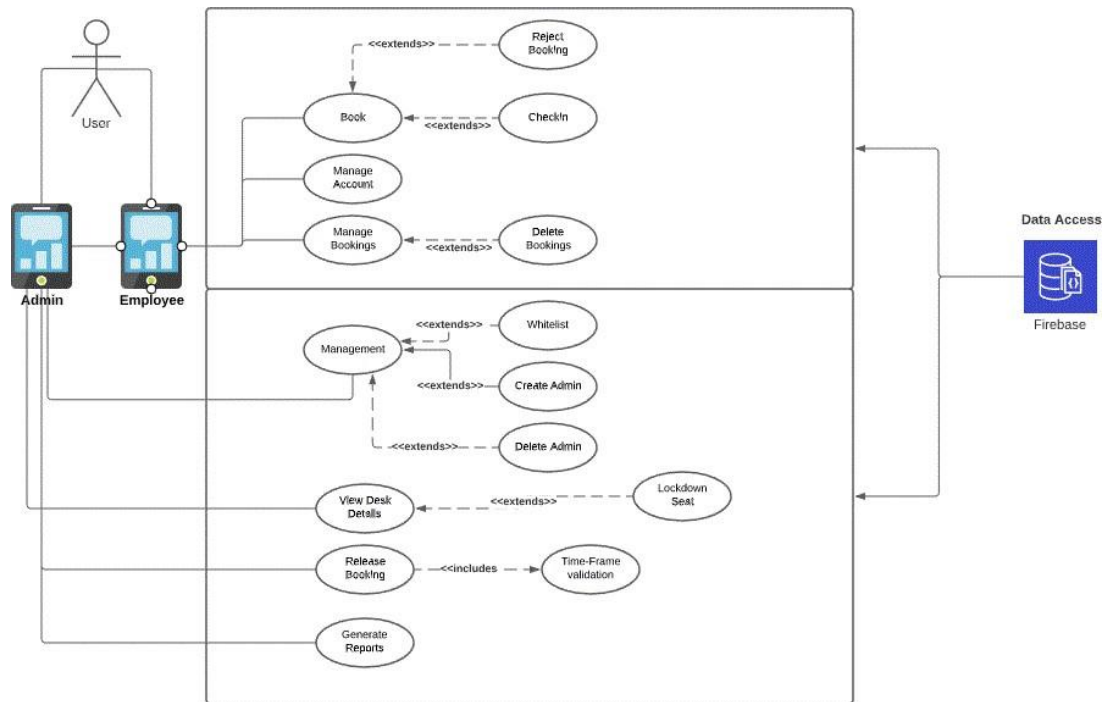
Process, Quality, Security, Testing, Personal-Reflection and Technology Related Requirements
- The program submitted should be refactored to promote code maintainability by using the object-oriented concepts including encapsulation, inheritance and polymorphism.
- Use of design patterns such as singleton and/or adherence to design principles such as SOLID.
- Well-designed test cases showing main business rules are implemented. For example, an attempt to add a new resident to a vacant bed should succeed while an attempt to add a resident to an already occupied bed should fail with appropriate diagnostic messages.
- Catching and handling all error conditions to create a robust system.
- A one-page document justifying all the design changes and lessons learned, this can be added to Readme as well.

Progress Milestones (+ video report)

You will be required to demo your progress in the lab regularly to demonstrate progress through 3 reports, two for among development and one after submission. The specific milestone requirements and associated dates is below:
You should submit your code/image/db/readme for each milestone.
Week 8 (3 marks) 2 May 11:59 pm expected:
Your Github class-room repository has been made, cloned and project is working on your local machine.
You also make a video report and explain your work and submit through milestone submission. Marks for milestone are separated from the assignment mark.
Work fellow and UI design sketch is completed. You can use lucid chart or wireframe for front-end sketch.
Database has been established and tables are created. (It can be changed)
Week 11 (4 marks) 21st May 11:59pm
Expected working in progress:
Back-end and data structure and model classes has been designed and implemented.
There are unit-test for model classes.
Front-end implementation started and around 50% of front-end is ready (not 100% completed or validated is fine) and integrated with back-end and ready for test.
Week 14 7th June 11:59pm Final report based on whole application (4 marks) (this submission is different to assignment final submission it would be a day after assignment).

It has been expected that students work on the assignment incrementally and show their work or weekly basis to their Prac teacher to get early feedback.

There will be deduction for final project if you cannot submit the progress report.
There will be 15% deduction of your final submission you miss week 8 video report or week 11 video report.
There will be 40% deduction of your final submission if you miss week 8 and week 11 progress video submission.
There would be 25% deduction if you only miss week 11 submission.
There will be no mark for the assignment 2 if you would not submit Week 8,11 and 14 video report.

5. Use of Github Classroom and IDE

You are required to use Github repository management to manage your code.

- Create your repository via the Github Classroom link provided in Canvas Announcements on April 13 2021.
- You are expected to constantly update your code repository continuously, e.g,. after each significant file modification (this may be several times per day, or even per hour).
- You are required to work with "IntelliJ IDEA" on your local machine and create a local Git from the beginning of your work ~~or if you prefer online IDE, you can start working with CR VSCode. These are only options.~~ (I found VSCode in general has many issues with JavaFX, will not be supported by us, so the no use of VSCode in Codingrooms or local)
- There will be video guide on assignment page by end of week 7 about using IntelliJ IDEA ~~and CR VSCode for this project.~~

6. Submission format for final submission

- You will submit your assignment by uploading a URL of you to your Github Classroom repository. Marks will be deducted if your Github repository shows development activity after the due date/time.

- There would be no mark if student uploads any other private repository than Github classroom for this course.
- You will also submit any related documents, e.g., a README detailing how to run your program, an outline of your design and design decisions made, and a brief refactoring report.

7. Academic integrity and plagiarism (standard warning)

Academic integrity is about honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge and ideas. You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e. directly copied), summarized, paraphrased, discussed or mentioned in your assessment through the appropriate referencing methods,
- Provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites.

If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own. RMIT University treats plagiarism as a very serious offence constituting misconduct. Plagiarism covers a variety of inappropriate behaviors, including:

- Failure to properly document a source
- Copyright material from the internet or databases
- Collusion between students

For further information on our policies and procedures, please refer to the University website.

8. Referencing guidelines

You are free to refer to textbooks or notes and discuss the design issues (and associated general solutions) with your fellow students or on Canvas; however, the assignment should be your OWN WORK.

The submitted assignment must be your own work. For more information, please visit http://www.rmit.edu.au/academicintegrity.

Plagiarism is treated very seriously at RMIT. Plagiarism includes copying code directly from other students, internet or other resources without proper reference. Sometimes, students' study and work on assignments together and submit similar files which may be regarded as plagiarism. Please note that you should always create your own assignment work **even if you have very similar ideas.**

Plagiarism-detection tools will be used for all submissions to check similarities. Penalties may be applied in cases of plagiarism and will be reported to the school.