COE528 (Title)
Course Project – Summer 2020

Final Project Submission Deadline: Friday, July 24, 2020 at 11:30 PM

General Rules

The course project is used to model a real-world problem using an object-oriented analysis and design approach. This project will take place in groups of ***four*** students. Choose the appropriate team members and work to breakdown structures to solve the given problem below. One student in each group should act as a group facilitator to facilitate routine communication with the instructor and the submission site. All the java files in this project should have the following package declaration: package coe528.S2020.project

Problem Description

A parking lot or car parking facility is a dedicated cleared area intended for parking vehicles. In most countries where cars are a significant mode of transport, parking lots are a feature of every city and suburban area. Shopping malls, sports stadiums, megachurches, and similar locations also have parking lots in large areas.

There are two kinds of users of this application: Manager and Customer. We will assume there is only one manager with a specific manager view and zero or more customers who can access the application through a portal/kiosk on every floor. The manager can login, logout, update, modify, and delete parking tickets. The manager has username: admin, password: admin, and role: manager.

The information about each purchased ticket is stored in separate files — one file per ticket; the filename can have the ticket-number. The ticket-number is a number of 10 digits that are generated randomly. When a manager tries to login through the user interface, the manager's username, password, and role are authenticated. Only the manager of the parking lot has the authority to add, update, modify, or delete a ticket. When a customer purchased a ticket, the application must create the corresponding ticket file and store it. When the manager deletes a ticket, the associated ticket file should get deleted. It is assumed that no two tickets can have the same number.

Your team will create the above multi-floor parking application. The application must have a graphical user interface (GUI). **The GUI of the application has to be developed using JavaFX.**

System Requirements

You will focus on the following set of requirements while designing the parking lot:

1. The parking lot should have multiple floors where customers can park their cars.
2. The parking lot should have multiple entry and exit points.
3. Customers can collect a parking ticket from the entry points and can pay the parking fee at the exit points on their way out.
4. Customers can pay the tickets at the automated exit panel or to the parking attendant.
5. Customers can pay via both cash and credit cards.

6. Customers should also be able to pay the parking fee at the customer's info portal on each floor. If the customer has paid at the info portal, they do not have to pay at the exit.
7. The system should not allow more vehicles than the maximum capacity of the parking lot. If the parking is full, the system should be able to show a message at the entrance panel and on the ground floor's parking display board.
8. Each parking floor will have many parking spots. The system should support multiple types of parking spots such as Compact, Large, Handicapped, Motorcycle, etc.
9. The Parking lot should have some parking spots specified for electric cars. These spots should have an electrical panel through which customers can pay and charge their vehicles.
10. The system should support parking for different vehicle types like car, truck, van, motorcycle, etc.
11. Each parking floor should have a display board showing any free parking spot for each spot type.
12. The system should support a per-hour parking fee model. For example, customers have to pay $4 for the first hour, $3.5 for the second and third hours, and $2.5 for all the remaining hours.

Work Items

1. Develop the UML diagrams (Both "**Use Case** Diagram" and "**Class** diagram(s)") for your project using Violet UML Editor
2. **For any one class**, you must:
   a) write the Overview clause stating the responsibility of the class and whether the class is mutable or not. Provide this as **javadoc** comments.
   b) write the abstraction function and the rep invariant as **javadoc** comments.
   c) provide the necessary clauses (e.g. effects, modifies and requires) for each method as **javadoc** comments for the method
   d) implement the abstraction function in the toString() method
   e) implement the representation invariants in the repOk() method
3. Your project must use **design patterns** in your implemented code for the application.
4. A written **report** must be generated. The report should not exceed 3 pages. Use Times New Roman font size 12. You **must** write your **name**, **student ID** and **section number** at the **top of the report**. Your report should **include** the following parts:
   - Describe your Use Case Diagram in a paragraph.
   - Describe your Class Diagram in a paragraph.
   - Mention the class you have selected to address point #2 (mentioned above).
   - Refer to your UML class diagram and indicate the part(s) that form the selected design pattern.
   - If you use any external sources such as books or papers, you must list them under a section named "References".
   - For references (if any) – Please use either 'Harvard' or 'APA' referencing system. You can find a description of both online. Further, you will find the following website useful in creating your references: http://www.citethisforme.com.

- The report will be assessed not only on their technical or academic merit, but also on the communication skills of the authors as exhibited through the report.
- You must give a demonstration of your project to your TA. There will be a schedule for code demonstration over Zoom posted on D2L. Your demo must include successful compilation and execution of your project application. **Project Demo Dates: Friday, July 24, 2020 at 09:30 AM. Every demo session will be up to 30 min for each group.**

5. **PowerPoint slides specs** - the presentation slides will be compiled out of the report and should respect the following specs:
   - Presentation Content
     - ➢ Make sure to INCLUDE A SLIDE on the logical partitioning of the work among the team members: who championed which technical part?
     - ➢ Create *as many slides as necessary* to do your task well. There is no limit on the number of slides you can have. However, your video presentation (described below) must respect the time constraint which is 10 mins (at most); thus, you can selectively present particular slides from your slide set.
     - ➢ Presentation format is important (style, use of fonts, images)
       - Do *not* include fancy, glittering, slide designs – keep them simple, please. Black text on transparent (white) background. Multi-coloured text is OK, e.g., to highlight certain terms.
6. **Video presentation specs** - the vide presentation will be compiled out of the report as well as the PowerPoint slides and should respect the following specs:

   - Logical partitioning/structuring of the video presentation matters.
   - Equitable presentation-workload sharing matters.
   - Theory knowledge and principles identified and EXPLAINED in the slides are weighted.
   - The presentation must be in MP4 format. The presentation should not exceed 10 minutes long (**DO NOT GO OVER 10 minutes**).

Submission

- Your submitted files must be consistent with **D2L** submission specs.
- Submission on the dedicated dropbox on **D2L**. Zero marks for this project if report, slides and video are NOT submitted even if the presentation and report are made.
- For the report, you must include the duly filled and signed standard cover page with your submission. The cover page can be found on the departmental web site: Standard Assignment/Lab Cover Page: https://bit.ly/AsnCovPg
- You need to submit ONE zipped file includes the final report, the PowerPoint slides and the video presentation and according to the instructions given below.
- It is your responsibility that the zipped file of your submitted work, is not empty or damaged. You can submit your file more than once. If you submit it more than once, I assume that the latest submission is the one you want to be graded.

- The project is to be documented digitally where the files should be readable/executable by the following applications:
  - MS WORD, PowerPoint, and/or Excel.
  - Acrobat Reader (PDF files)
  - QuickTime or VLC players (Video)
  - NetBeans
- If the submitted documents/files that are not readable by the above-listed applications, it will not be graded and, consequently, will receive a mark of **zero** for that portion. Please get this right, first time!
- The project folder should be named: Team <number>_Project Documentation_2020.
- The project folder should be zipped. However, it is your responsibility that the Zip file of your project, is not empty or damaged. If that happened, you would receive a mark of zero for the project.
- The ZIP file should contain:
  - The **report**
  - **Violet UML files**
  - The **implementation** of the project (e.g. source code including a main file)
  - **Javadoc for the class** that you have selected.
- Please consult with the course TA, that the source code files can be opened, compiled and executed after submission. Please ensure that the Violet UML files and the report can be opened and read.

Group Work, and Responsibility

- This project is a teamwork. Individuals in the team collaboratively create different parts of the system into an integrated whole. All the members of a group are thus expected to contribute earnestly according to the plan.
- If a team member has a complain such that in his/her interpretation:
  - There is a significant complaint from the rest of the team (not only one person);
  - The complaint is made to the instructor in writing – by the rest of the team -- at the time of the occurrence of the negative situation;
  - There will be a peer review form distributed among the team members; and
  - After a preliminary inquiry by the instructor, the instructor is “convinced” that the situation warrants a review. The instructor’s decision is final and may not be appealed.
- The project mark given to the individuals in the team is the base project mark adjusted by peer-review feedback received from the group.
  - The peer-review penalty is as follows -- % grade of the overall project mark reduced:
    - Minor infraction: 10%
    - Significant infraction but not considered major: 30%
    - Major infraction: 70%
- Note:
  - All interpretations will be made by the instructor and may not be appealed.
  - A second-time infraction for the same person will automatically lead to “major infraction” if the previous case was assessed at “minor or significant” levels.

    - If the previous case was assessed as "major infraction," then this will automatically result in a zero mark for the project for the person concerned.
    - The best policy is to collaborate for the team's success.

Evaluation

The overall percentage points gained will be weighted according to the project's weight in the course:

- Project report (20%), PowerPoint (20%), video presentation (20%), and executable code is 40% of the project marks.

Note:

- There will be a penalty of 10% for the delay delivery.

Good Luck!