**School of Computer Science**

**COMP-3220 Object-Oriented Software Analysis and Design**

**Assignment 2**

**Due: November 1, 2021 – 11:59 pm**

**100 marks – 10% of grade. Group of 1-3 students allowed.**

**Objective:** Help students understand requirements capture and analysis in object-oriented information systems development. Help students understand the concepts and techniques relating to use case realization.

**Tasks:**

The Lecture 5 slides describe ways to obtain, document, and analyze requirements for information systems development. In this assignment, you will develop and realize two use cases relating to a nice restaurant that desperately needs to computerize their a) table booking and b) staff scheduling business activities.

**Table Booking Situation Description:**

A restaurant has three time-slots during which they accept customers: breakfast (7-9am), lunch (noon-2pm), and dinner (5-7pm). The restaurant has some number of tables, each having a specific number of seats. For a given time-slot (breakfast/lunch/dinner), a given table can only be booked once – that is, it cannot have two bookings in e.g. the breakfast time slot. In a given day, at most, it can have one booking in the breakfast time-slot, one in the lunch time-slot, and one in the dinner time-slot. Any given table may have no booking in a given time-slot.

The restaurant currently has a manual system (ew) for booking their tables. A receptionist writes down requests for booking a table on a booking sheet – a form showing all tables and all time slots for a given day. These booking sheets are stored in a big old binder. Each sheet looks something like this (only showing two tables, for example):

|  |  |  |  |
| --- | --- | --- | --- |
| **Bookings for October 18, 2021** | | | |
| **Time-slot** | **Info** | Table 1 (capacity 6) | Table 2 (capacity 8) |
| Breakfast | *Customer Name* | John Doe |  |
| *Phone Number* | 098-765-4321 |  |
| *Number of People* | 5 |  |
| *Time of Arrival* | 8:15am |  |
| Lunch | *Customer Name* |  | Jane Smith |
| *Phone Number* |  | 123-456-7890 |
| *Number of People* |  | 6 |
| *Time of Arrival* |  | Noon |
| Dinner | *Customer Name* | Bob | Joe |
| *Phone Number* | 777-777-7777 | 222-222-2222 |
| *Number of People* | 4 | 8 |
| *Time of Arrival* | 5:30pm | 6:45pm |

When a customer calls to make a reservation, the receptionist typically:

1. Asks how many people are in the party.
2. Asks what time the party would like to arrive.
3. Checks if a table with sufficiently many seats is available (for simplicity, assume one party may only receive one table) during the requested time-slot.
4. If a sufficient table exists, the receptionist writes down with a pencil the customer’s name, phone number, number of people in the party, and their arrival time.

In your use case and corresponding UML diagrams, consider the following:

1. The receptionist should never make a double-booking for a given table in a given time-slot on a given day. For example, in the above bookings form, only Table 2 can take another reservation for breakfast, or Table 1 for lunch. The system should check automatically for availability and decline the request if no tables are available.
2. If the receptionist receives a booking request at an invalid time or for too many people given the available tables, the system must not allow it.

**Staff Scheduling Situation Description:**

The restaurant needs to schedule its staff. It has the three shifts mentioned above – breakfast, lunch, and dinner. Staff scheduling for a given day proceeds as follows. First, the manager selects one supervisor for each shift – one for breakfast, one for lunch, and one for dinner. A supervisor may have 0-3 shifts per day – that is, the supervisor may have the day completely off, or they may have all three shifts and work a full day, or almost anything else between. However, on a given day, a supervisor should only be assigned contiguous shifts (since we only have three shifts, that means a supervisor can never be assigned only breakfast and dinner). Regardless, we must have one supervisor present per shift. No single supervisor can work more than 9 distinct shifts in a week.

After the manager allocates a supervisor to each shift, assigned supervisors can schedule general staff to their given shifts. For each shift, the restaurant must always have a receptionist, a chef, 2 cooks, and 5 waitstaff on hand. No one cook or waitstaff can have more than 6 distinct shifts in a week, but chefs and receptionists can have as many shifts as necessary. Good chefs are hard to find. The chefs are subject to the same contiguity constraint as supervisors, but the cooks, receptionists, and waitstaff are not.

Currently, schedules are hand-written on a printed form which, like the reservations, are stored in a big old binder (ew). The form looks like this:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **October 18, 2021** | | | | | |
| **Shift** | **Supervisor** | **Receptionist** | **Chef** | **Cooks** | **Waitstaff** |
| Breakfast | Aaron | Mohammad | Ryan | Steph  Steve | Richard  Joe  Kathy  Jane  Jim |
| Lunch | Jen | Keith | Ryan | Julie  Aaron | Stephen  Steve  Stefan  Stephanie  Stefanie |
| Dinner | Jen | Keith | Ryan | Steph  Joe | Joe  Joseph  Josephine  Josephina  Jo |

In addition, for payroll purposes, the total number of shifts assigned for a week per employee is written on a form and stored in a binder, and the form looks like this:

|  |  |  |
| --- | --- | --- |
| **October 18, 2021** | | |
| **Employee** | **Role** | **Shifts** |
| Ryan | Chef | 12 |
| Mary | Cook | 6 |
| Joe | Waitstaff | 4 |
| … | … | … |

Of course, the system must not allow managers to schedule supervisors in a way that violates any of the constraints above. Similarly, the system must not allow supervisors to schedule any other staff in a way that violates the constraints above.

**For each use case, feel free to make reasonable assumptions. You must state any assumptions you’ve made when documenting and realizing the use case.**

1. **(50 marks for each situation – 100 marks total)** Realize the above use cases – one use case for the Table Booking situation, and another for the Staff Scheduling situation. Your realizations of the use cases must include the following:
   1. (10 marks) Using the attached template, fill out the use case supplementary text for the use case.
   2. (10 marks) Create a Use Case Diagram for the use case. Be sure to use abstraction of actors and actions as necessary. Use reasonable names for the actors and actions. Provide a brief text description for anything necessary.
   3. (10 marks) Create a Collaboration Diagram for the use case. Use reasonable names for the objects. Provide a brief text description for anything necessary.
   4. (10 marks) Create a Communication Diagram for the use case. Use reasonable names for the objects and messages. Provide a brief text description for anything necessary.
   5. (10 marks) Create an Analysis Class Diagram for the use case. Use reasonable names for the classes and edge labels. Provide a brief text description for anything necessary.

Each realized use case should be put into a separate document (e.g. MS Word, PDF, etc. – something that supports both text and images). Use whatever software you like for making the diagrams – e.g. PowerPoint, Word, etc. - however I recommend <https://draw.io/> (it has an entire UML section with stick figures and everything, and is super-simple to use and you can save the images on your hard drive – if you save as XML(.drawio) you can reload the images for future use e.g. future editing/refining, and also save as .png or whatever to actually have an image you can embed into your document). Thus, you will have one document for the entire Table Booking use case, and another for the Staff Scheduling use case. Each document will have all five sections (a-e) outlined above.