CS2102 Project (20 marks)
Deliverables due on **April 12, 10pm**

The objective of this team project is for you to apply what you have learned in class to design and develop a web-based database application. The project is to be done in teams of four students.

1 Application Topics

Each project team will be randomly assigned to one of five application topics. The following provides a brief description of each application topic and some example concrete application deployments. As the topic descriptions are not meant to be complete specifications of the application topics, each team has the freedom to decide on the application's data requirement as well as the application's functionalities/features to be implemented. Your database design and implementation for the application should demonstrate non-trivial usage of SQL and database features.

Topic A: Task sourcing. This is a task matching application (e.g., https://www.taskrabbit.com) to facilitate users to hire temporary help to complete certain tasks. Tasks are general chores as such washing a car at Kent Vales car park on Sunday or delivering a parcel on Tuesday between 17:00 and 19:00. Users of this application are either looking for help to complete some task or bidding to complete some freelance task. The application provides templates for generic common tasks to facilitate task requesters to create new tasks. The successful bidder for a task could either be chosen by the task requester or automatically selected by the system based on some criteria. Each user has an account, and administrators can create/modify/delete entries.

Topic B: Restaurant Reservation. This is a restaurant reservation application that allows diners to book reservations at restaurants (e.g., https://www.chope.co). Restaurants can advertise their availability (e.g., cuisine type, branch locations, opening hours, menu prices) and diners can search for restaurants to book reservations by providing various information (e.g., date and time, cusine type, number of people, budget, preferred locations) and rate restaurants based on their dinning experience. Each booking can be confirmed based on various criteria (e.g., booking time, availability, number of diners). Dinners could cancel or make changes to their reservations. The system could provide various incentives (e.g., award points for each confirmed booking) to retain users. Each user has an account, and administrators can create/modify/delete entries.

Topic C: Stuff Sharing. This is a stuff sharing application that allows people to borrow or loan stuff that they own (e.g., tools, appliances, furniture or books) either for free or for a fee (e.g.,

https://www.peerby.com). Users can advertise their available stuff for loan (what stuff, where to pick up and return, when it is available, etc.) or can browse the available stuff and bid to borrow some stuff. The successful bidder for an advertised item could either be chosen by the item owner or automatically selected by the system based on some criteria. Each user has an account, and administrators can create/modify/delete entries.

Topic D: Car Pooling. This is a car pooling application that allows car drivers to advertise opportunities for car pooling and passengers to search for car rides (e.g., https://www.sharetransport.sg). A car ride advertisement specifies (among other information) an origin to a destination on a certain date and a given time. Both users and cars have a profile. Drivers advertise rides and passengers bid for the rides. Users can play both roles of drivers and passengers. The successful bidder for an advertised ride could either be chosen by the car driver or automatically selected by the system based on some criteria. Each user has an account, and administrators can create/modify/delete entries.

Topic E: Pet Caring. This application allows pet owners to search for care takers for their pets for certain periods of time (e.g., https://dogvacay.com). Both users and pets have a profile. Care takers can advertise their availability (when they can take care of a pet, for how long, the kind of pet they can take care of and other constraints and requirements), and pet owners can browse/search for care takers and bid for their services. The successful bidder could either be chosen by the care taker or automatically selected by the system based on some criteria. Each user has an account, and administrators can create/modify/delete entries.

2 Application Requirements & Functionalities

The data models of your application must satisfy the following requirements:

- The ER model have must be at least 10 entity sets (including at least one weak entity set) and at least 10 relationship sets.
- There must be at least three non-trivial application constraints that cannot be enforced using column/table constraints (i.e., they must be enforced using triggers).

Each application must provide at least the following functionalities.

- Support the creation/deletion/update of data.
- Support the browsing and searching of data.
- Support at least three interesting complex queries on the data. An example of an interesting query is one that performs some data analysis that provides some insights about your application.

Your application should not be limited by the functionalities of its brief description given in the previous section. You are free to introduce interesting functionalities to make your application non-trivial (e.g., requiring complex queries, transactions, triggers, etc.). You are also free to use any of PostgreSQL's features and other SQL constructs beyond what are covered in class.

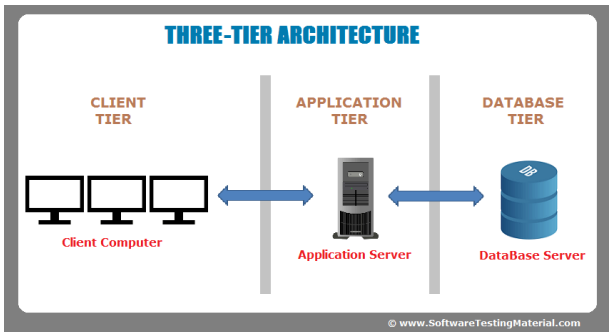

Figure 1: Three-Tier Web-based Application Architecture

For the final project demo, your application's database should be loaded with reasonably large tables. To generate data for your application, you can use some online data generators (e.g., https://mockaroo.com, https://www.generatedata.com) or write your own program.

3 Software Tools

The architecture of your application follows the typical three-tier (clients, application server, database server) architecture of a Web-based database application shown in Figure 1 with Web browsers as the clients and a Web server as the application server.

To get started, download the following two files.

- **install-postgresql.pdf** from IVLE's SQL Workbin
- **start_guide.pdf** from IVLE's Project Workbin

The first document describes how to install PostgreSQL, an open-source database server, and the second document provides an introduction on how to develop web applications using NodeJS and Bootstrap. You are welcomed to use other frontend/backend development stacks for your web application such as Angular JS, React JS, PHP frameworks (e.g, Laravel, Symfony), and Apache.

4 Project Deliverables

The project has two deliverables which are due on April 12, 10pm.

- Project report (up to a maximum of 20 pages in pdf format with at least 10-point font size). The report should include the following.
  1. Names and student numbers of all team members, project team number, and project topic (on the first page).

2. A listing of the project responsibilities of each team member.
3. A description of your application's requirements and functionalities. Highlight any interesting/non-trivial aspects of your application's functionalities/implementation.
4. The ER model of your application (which must satisfy the requirements in Section 2). If your ER model is too large (spanning more than a page), you may want to include a single-page simplified ER model (with non-key attributes omitted) before presenting the detailed ER model.
5. The relational schema derived from your ER data model; i.e show the DDL statements of all your database tables. If there is any table that is not in 3NF/BCNF, justify your design decision.
6. A English description of each of the three non-trivial constraints that are enforced using triggers. Show the code of your trigger implementation.
7. A English description of each of the three interesting queries. Show the SQL code of these queries.
8. Specification of the software tools/frameworks used in your project.
9. Two or three representative screenshots of your application in action.
10. A summary of any difficulties encountered and lessons learned from the project.

- Project source code including a README file describing how to deploy and run your application.

5 Project Deadlines

The following table lists the five project deadlines.

Due Date	Project Task
Week 8 (March 11-15)	5-min Alpha Demo (during tutorials)
April 12, 10pm	Report & Code Submission
Week 13 (April 15-19)	15-min Final Demo (during tutorials & evenings)

Alpha Demo (Week 8). Each team is to present a 5-minute demonstration to show the progress made in the project. The alpha demonstration will take place during Week 8's tutorial hours. Each team is to show the ER data model and any working functionalities. The booking of the alpha demo time slots will open during week 6.

Final Demo (Week 13). Each team will present a 15-minute demonstration of their project showing the main features and answering questions from the evaluators. The final demonstration will take place during Week 13's tutorial hours and in the evenings. The booking of the final demo time slots will open during week 11.

6 Evaluation Criteria

The maximum score for the project is 20 marks with the following breakdown:

- Alpha demonstration: 1 mark
- Report & Final Demonstration: 19 marks
  - Database design (6 marks)
    - ER data model (3 marks)
    - Relational schema (3 marks)
  - Interestingness/Complexity of application (6 marks)
    - Interesting complex queries (3 marks)
    - Non-trivial constraints enforced with triggers (3 marks)
  - Application design, functionalities, & interface (3 marks)
  - Report organization & presentation (2 marks)
  - Final demo presentation (2 marks)
- The top-10 projects will each receive 1 additional mark (if the total marks is below 20).

7 How to Submit

- Submit your source code by creating a zip file named codeNN.zip, where `NN` is your project team number. Upload this file into the IVLE Workbin named Project-Code-Submission.
- Submit your report by creating a pdf file named reportNN.pdf, where `NN` is your project team number. Upload this file into the IVLE Workbin named Project-Report-Submission.

The deliverables are due on **April 12, 10pm**.

For late submissions, submit to the IVLE Workbins Late-Project-Code-Submission and Late-Project-Report-Submission. One mark will be deducted for each late day up to two late days; projects submitted after the second late day will receive zero marks and will not be graded.