CS2102 project

In this team project, you will develop a typical Web-database application. The objective of this project is for you to familiarize yourself with database technologies and to apply the concepts and techniques learned in class for the design and programming of a database application.

The project is done in groups of 4 students.

The evaluation of the project considers the scope of concepts and techniques used and their relevance. For instance, you should try to appropriately use the simple and advanced SQL constructs that you have learned: simple queries, aggregate queries, integrity constraints, views, etc. Feel free to extend the application requirements and add features in order to demonstrate interesting usages of the techniques learned. Consequently, the amount of data in the database should be sufficient for a complete and realistic demonstration of the system.

TOPICS

The manager of your company, ThinkCan Pte Ltd, a Singaporean software house, asked your group to design and implement the prototype of an online (Web -based) CRUD (create, read, update and delete) application on the theme of the new sharing economy. The prototype should be realistic in order to convince a major customer to commission your company to develop the system, but should also illustrate the use of relational database technology in order to serve as an in-house showcase application for engineers in your company.

It is left to your creativity to design a realistic model for the description of the items and ancillary information in the system. The design can be kept simple but should be sufficiently rich to allow the meaningful demonstration of SQL and DBMS features. Similarly, you should also populate the database with sufficiently enough data to both make the demonstration realistic and to illustrate the use of interesting SQL and DBMS features.

Your group is assigned one of the following five topics.

**Topic A, Task sourcing**: the system is a catalogue of tasks submitted by users. Users can either submit a task or pick a task. Tasks are general chores such as washing a car on Kent Vale's car park on Sunday or delivering a parcel on Tuesday between 17:00 and 19:00. Users who want to pick a task can bid. Some generic common tasks may be available for task requesters to instantiate. The user who submits a task or the system (your choice) chooses the successful bid. Each user has an account. Administrators can create, modify and delete all entries. Please refer to www.taskrabbit.com for examples and data.

**Topic B, Crowdfunding**: the system is a catalogue of projects looking for crowdfunding. Entrepreneurs can advertise their projects (title, description, start date, duration, keywords or categories, the amount of funding sought). Users can browse the projects and fund projects. Users can play both roles of entrepreneurs and investors. The system tracks the current amount of funding raised, brings the project to the status of "funded" and advertises this success on a page of funded projects. Each user has an account. Administrators can create, modify and delete all entries. Please refer to www.globalgiving.org, fundanything.com or other crowdfunding sites for examples and data.

**Topic C, Stuff Sharing**: the system allows people to borrow or lend stuff that they own (tools, appliances, furniture or books) either free or for a fee. Users advertise stuff available (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 stuff owner or the system (your choice) chooses the successful bid. Each user has an account. Administrators can create, modify and delete all entries. Please refer to www.snapgoods.com, www.peerby.com or other stuff sharing sites for examples and data.

**Topic D, Car Pooling**: the system allows commuters to search or advertise car rides. A car ride is an offer to ride in a car with the driver offering the ride from an origin to a destination on a 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 driver or the system (your choice) chooses the successful bid. Each user has an account. Administrators can create, modify and delete all entries. Please refer to www.sharetransport.sg or similar sites for examples and data.

In order to meet the minimum requirements, the application and interface should allow 1) browsing and searching of entries (basic and advanced search features) and ancillary data, and 2) creation, deletion, and modification of entries and ancillary data. These features should be implemented in SQL rather than in the application code whenever possible and without an ORM or in the conditions indicated if an ORM is used. Simple login and sessions may be required by the topic, but advanced access control or shopping basket is not required. Extra credit will be gained by appropriate usage and demonstration of advanced SQL and DBMS features (advanced SQL queries such as aggregates, nested queries, views, non-standard integrity constraints, triggers, and stored procedures).

The system must implement an assertion of your choice. For example, the assertion could prevent a project to be funded beyond the amount sought or several task sourcing bids whose date and time overlap to be successfully awarded to the same user. The assertion must be implemented using views, functions and triggers, as needed. It must not be implemented in the application code (PHP). The application code (PHP) should be restricted to displaying the error message if the assertion is violated. You may choose to either reject the updates causing the violation (this is the expected feature) or to compensate them. The report must clearly present the assertion, its expected behaviour and its implementation.

TECHNOLOGY

The architecture of your application is the typical three tiers (clients, application server, database server) architecture of a Web-database application. Because it is a Web application, the clients are Web browsers and the applications server is a Web server with a server page language extension.

Figure 1 is an overview of the architecture of your application. The architecture consists of a Web server, a server page language for server side programming and, of course, a relational database management system. The application is accessible from Web browsers.

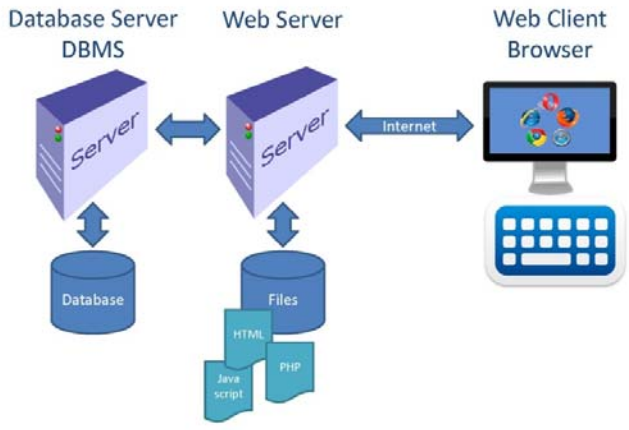


Figure 1: General Architecture

The project uses Bitnami stacks. Bitnami stacks, WAPP (for Windows), LAPP (for Linux) and MAPP (for Mac OS X), provide a ready to run Web-database development environment. It contains PostgreSQL, phpPgAdmin, Apache and PHP for the development and deployment of the project's Web-database application. It is easy to download, install and start. We provide you with a sample PHP page to get started. The project effort can be focused on the design of the database, SQL and the application. It is recommended that complex queries be implemented as views.

*You may also use other stacks or frameworks but the teaching assistants may not be able to provide technical support for your option. We suggest that you only opt for such an option if a member of your team has experience with these technologies. If you use an ORM, you MUST implement all updates as stored procedures or functions and your queries as views, parameterized queries, stored procedures or functions. The ORM is not allowed to build other than simple updates and lookup queries (SELECT fields -no aggregate or calculated fields- FROM one table WHERE simple conjunctive condition).*

Download the stack for your operating system, WAPP, MAPP or LAPP, at https://bitnami.com/tag/postgresql and install it.

You may (optional) also download and use PgAdmin III or IV. It is PostgreSQL's interactive administrative and SQL programming interface. They may be useful for development purposes.

DELIVERABLES & DEADLINES

Project groups are formed of 4 members.

In the event of contentious issues within the group (e.g. member not participating), contact lecturers as soon as possible (in any case before the end of Week 9) to resolve the problem.

- Check IVLE Lesson plan for the following deadlines:
- Project start. As soon as your group has been assigned a topic.
- Pre-Alpha Demonstration.
- Design Report Submission Deadline.
- Alpha Demonstration
- Project Report and Code Submission Deadline

The project deliverables are:

1. A pre-alpha demonstration (showing that you can access and interact with the Web server and the database from a Web browser using server side programs). Each group is given 5 minutes to demonstrate that they are able to connect their Web-application code from a Web browser and that the server-side code can connect the database for queries and updates.
2. An initial design report that presents the Entity-relationship diagram for your project and its translation into a relational schema (in SQL DDL code with integrity constraints). This initial design can be modified and improved for the final report and demonstration.
3. An alpha demonstration (an early draft of your application with sufficiently enough example data: e.g. 20 user profiles, 100 items with relevant data and images). Each group is given 5 minutes to demonstrate a draft of their application and to show the demonstration data that they have prepared (as a rule of thumb you should have created at least 20 to 30 realistic user profiles and 100 realistic items. For instance Topic E has created 10 pet owners, 10 pets and 10 pets and 100 pet care requests).
4. A brief report (with all group members' names and matric numbers on the front page) containing:
    a. Indication of the web server, server page language and database management system used (in particular if you use another environment than the one prescribed)
    b. The Entity-relationship diagram for the application
    c. The translation of the Entity-relationship diagram into relational schema (in SQL DDL code with integrity constraints), chosen sample and representative SQL code (indicate the SQL code and the function of the service it helps to implement)
    d. The presentation of the implemented assertion
    e. 2 or 3 representative screen-shots of the Web interface.
5. The application code in a Zip file.
6. A demonstration of your application. Each group is given 15 minutes to demonstrate the main features of its system and to answer questions. Your group will need to register to one of the available time slots (how to register will be announced later). It is not necessary that the entire group be present for the demonstration.

Put a soft copy in PDF of the design report in the IVLE Workbin – Design Report folder. The folder opens approximately one week before the submission deadline. The file should be called DesignGroupXX.pdf, where XX is the group number. If you miss the deadline, put a soft copy in PDF of the report in the IVLE Workbin - Design Report Late Submission folder. There is a late submission penalty (1 mark per day started after the deadline).

Put a soft copy in PDF of the project report in the IVLE Workbin –Project Report folder. The folder opens approximately one week before the submission deadline. The file should be called GroupXX.pdf, where XX is the group number. If you miss the deadline, put a soft copy in PDF of the report in the IVLE Workbin - Project Report Late Submission folder. There is a late submission penalty (1 mark per day started after the deadline).

Put a soft copy in a ZIP file containing the source (PHP and SQL) code in the IVLE Workbin - Project Code folder. The folder opens approximately one week before the submission deadline. The file should be called GroupXX.zip, where XX is the group number. If you miss the deadline, put a soft copy in a ZIP file of the code in the IVLE Workbin - Project Code Late Submission folder. There is a late submission penalty (1 mark per day started after the deadline).

MARKING SCHEME

The project marking scheme is as follows (21 marks)

- Pre-alpha demonstration: 1 marks (show that you can connect the web browser, the web server and server-side code and the database).
- Design report: 3 marks (presents the initial; Entity-relationship and logical designs).
- Alpha demonstration: 2 marks (demonstrate the running draft of your application following the initial design).
- Report: 6 marks
    - database design ER and DDL (2 marks)
    - SQL samples (1 mark)
    - assertion (2 marks)
    - and screen shots (1 mark).
- Demonstration (8 marks)
    - appropriate use of SQL (2 marks)
    - assertion (2 marks)
    - other advanced SQL (2 marks),
    - application and interface (2 marks).
- 1 marks for the top 10 best projects