CCT395, Fall 2011

The Final Project

This assignment consists of two deliverables.

1. The initial design is worth 10% of your course and is due at **1:10 pm on October 26**.
2. The final implementation and report are jointly worth 25% of your grade and are due at **1:10 pm on November 23**.

There will be a penalty for lateness of **10% deducted per day** and work that is not handed in one week (168 hours) after the due date will not be accepted. An assignment will be considered late if not submitted at the stated due time, that is by 1:10 pm on the due date. "Per day" means "per 24 hour period."

Assignments submitted on time should be handed in on paper (with an additional electronic component for the second deliverable). Assignment submitted late should be delivered by email, as PDF files.

The assignment can be done individually or in groups of 2 or 3. You can choose your own group. If you are having trouble finding a group then contact the instructor as soon as possible. If several students are working on an assignment together then accommodations for illness and other valid reasons will only be provided if ***all*** members of the team prove to be incapacitated. (In other words, if you work with a partner and your partner gets sick, be prepared to finish the work on your own.)

The Problem

For this assignment you will build a prototype of an online bookstore. Your store will allow customers to find books, get information about them, read or post reviews of books, and, of course, to order them.

Below are some of the specific features that you should offer, together with a number of out-of-scope items. (Out-of-scope means you specifically do *not* need to do it.) You should ideally implement all of those features. If you cannot, then do what you can. Please remember, however, that this is a course on *database design and implementation*. You will therefore be primarily evaluated on the underlying database, the queries, and your report. A team that designs and builds an excellent database but fails to implement a working website will likely get a higher grade than a team that provides a beautiful web interface to a broken database. Also, your website can be ugly. Your SQL should be beautiful.

Some specific requirements:

- The user should be able to search for books by entering a part of a title or a part of an author's name. (This can either be done with a single search box or with different search boxes: one for title and one for authors' names.) They should then get back a list of books matching the query, up to ten. Ideally, if there are more than ten matches then there will be a link that would show the next ten matches.
- If the user clicks on a book in the returned list of books, they should be able to see the following information:
    - The title of the book.
    - The names of the authors.

    - The average rating of the book by the users.
    - The price of the book.
    - Number of copies of the book available for sale.
    - A list of reviews with individual ratings.
    - Ideally there should also be an image of a cover. (You can use http://openlibrary.org/dev/docs/api/covers as a resource.)
- If the book is available, the user should be able to place an order for it by clicking on "Buy" and entering their name, address and credit card number (any 16 digit number). The order should be somehow recorded in the database and the number of available copies of the book should be reduced. Ideally the user would be able to buy several copies of the book with a single order and would not have to reenter their information if they had already bought books from the store before.
- The user should be able to post a review of a book. The review should include a rating on a 1-5 scale and should be displayed together with the user's username. The new review should be factored into the average rating for the book. Ideally it should be possible to click on the username to see other reviews by the same user.
- The system should offer a "manager's interface," a page that would provide the store's manager with the following information:
    - the list of outstanding orders,
    - the list of books out of stock,
    - the best selling titles.
- Ideally, your system will extend the functionality described above with one additional feature. However, you should only do this once you have completed all items described above, including those described above as things that should "ideally" be supported.

As you design the system, avoid making up features that are not asked for (apart from the one additional feature mentioned above.) When in doubt, ask if a particular functionality is expected. Your design should probably have 5-7 entities.

All web components should be implemented using Python CGI, which will be introduced in class on Week 7. If your team would prefer to use a different approach to web development, *please contact me to discuss this*. Be advised that permission to use a different method will only be granted in cases where *all* members of the team are familiar with it.

Out of scope items (overall)

- **Your web interface does not need to handle authentication.** E.g., when a user leaves a comment they will just enter the username and we'll trust them that this is who they are.
- **Your web interface can be bare bones. You do not need to worry about aesthetics and usability.** For instance, if a user enters wrong information, you can just tell them so and expect them to go back and correct it.
- **Your system only needs to support very simple search** – the kind that can be implemented using "like". In particular:

    - It is not going to handle spelling errors. If the customer who is interested in "The Hitchhiker's guide to the Galaxy" enters "hichhiker," they are out of luck.
    - It will only find items where the search string matches exactly a part of a title or a author's name. So, "guide to the Galaxy" would work, but "guide galaxy" won't.
- **Your system will not track fulfillment of orders.** That is, you just need to make sure that the customers can place orders, that you record their orders, and that you don't accept orders that are for books that have already been sold to other customers. You do not need to be worried about what happens afterwards.

Formatting of the Deliverables

Both of the deliverables described below should strictly adhere to the following format:

- The main text should be typed in a 12 point serif font (such as Times New Roman) and be 1.5-spaced.
- All SQL code should be in a monospace font. (`A monospace font is one in which all letters are the same width.`)
- All pages should have 2 cm margins.
- All pages should be numbered at the bottom.
- There should be no cover page and no artwork other than the ER diagrams.
- The names of the team members should shown on the first page.
- The ER diagrams need to be drawn using a software tool and have labels for all relationships.

Deliverable 1: The Preliminary Design (10% of your course grade)

Your first deliverable is the preliminary design. It should be submitted **on paper** and should include the following components:

1. **The list of team members.** Include the name and email address of each team member.
2. **A functional specification.** Detailed descriptions of how your system is going to behave from the customer's and the manager's points of view. Include specific use cases. (For example. "The customer wants to buy a copy of Harrington's *SQL*. She goes to the website and sees...".) Make sure to explain any non-obvious decisions. (Around 1000 words.) If you are proposing an additional feature, please state so clearly and describe it here.
3. **An ER diagram and notes.** Include an ER diagram and attach explanations of how it is going to support your design. Make sure to explain any non-obvious decisions – if your diagram does not make immediate sense to me, I will be looking to your notes for the explanation. Make sure your diagram matches your functional specification.
4. **The implementation plan.** Describe the schedule for implementing and testing of the system. State clearly who is responsible for each piece and by when they will need to finish it.

Deliverable 2: The Final Report and the Implementation (25% of your course grade.)

Your final deliverable will consist of two parts: the final report that you will submit on paper and the actual

implementation that you will setup on the database server.

The **implementation** should include:

1. **The database.** Your database should contain for for at least 100 books. You can find the necessary data on the web (library catalogs, bibliography databases) and import it using any of the methods discussed in class. (You may need to clean up the data by hand, but you should not have to type it all from scratch.) Your initial database should provide reasonable numbers of how many copies of each book you have in stock. (Between 1 to 10 would be reasonable.) It should also contain at least 10 comments. (Your report should state which books have been commented on.)
2. **A web interface.** The web interface should include a front page which which it should be possible to access all other functionality. In particular, the front page should contain a search box that can be used to search for books and a link to the page showing outstanding orders.

The **final report** should include the following elements, ***in this order***:

1. **The list of team members.** If this list is different from the one in your proposal, attach a note explaining why. (If you expect a change in team composition, please discuss this with me as soon as possible.)
2. **The name of the database.** (I need to be able to locate your database.)
3. **The main URL of the web interface.** This is the URL from which all of the functionality should be accessible. If you did not manage to get the web interface to work, please provide here a detailed explanation of what you tried to do to get it to work, what problems you ran into, and why.
4. **A summary of features.** Explain whether your implementation supports all the required features and whether it matches your functional specification. If not, explain why not. Also, if you implemented an additional feature, explain here what it is and how it works.
5. **The source of book data.** Explain how you obtained and imported the data for the books in your database.
6. **Testing information.** The information that would help verify that your system works as described. This should include search terms that one could enter into the search box to get meaningful results.
7. **The final ER diagram and explanation.** Include an ER diagram and an explanation of how it satisfies the needs of your system. *The diagram should match your implementation.*
8. **CREATE statements.** Include the CREATE TABLE statements that you used to create your database. Those should properly setup the necessary primary and foreign key and should satisfy the conditions for 1NF, 2NF, and 3NF. Provide an explanation for any non-obvious decisions. If you used ALTER TABLE statements, please include those and explain why they were necessary.
9. **Queries.** Include in your report *all* SQL queries used by your system. The latter should include all the SELECT, UPDATE, and INSERT queries, including those used inside Python files. Explain any non-obvious decisions.
10. **Additional comments.** You can optionally provide additional explanations.
11. **Statement of responsibilities.** Describe briefly the contribution of each team member.
12. **Credits.** The ER diagram and the SQL used in this assignment should be written exclusively by your team. For other components (e.g., images, CSS, HTML, Python functions) you can use what's available on the web, if you wish. If you do so, state what you got where, including specific URLs. You can only use components that are publicly available and they cannot be custom-made for your project. (In other words, if you find some CSS you want to use, go for it. But do not *ask* someone to write CSS for you.)