Systems Design & Security: Team Project

Overview

Coursework will form 50% of the assessment for COM2008/COM3008. The team project will be completed in teams of four. The project brief will be released **Monday week 4**. Teams will hand in their reports and software in Friday week 10 (one report and software bundle per team). This will be worth 40% of the module grade. There will be a further testing stage, completed individually in week 11. This will be worth 10% of the module grade.

Submission details

There will be two submission links under Assessment in Blackboard. One is for your team report, and the other is for your team software bundle.

- 1 team report (typically 10-12pp) to be submitted as PDF to Blackboard (only one report per team), **by 15:00 Friday week 10.**
- 1 zipped bundle of your software system to be submitted to Blackboard (only one bundle per team), **by 15:00 Friday week 10.**

Individual testing

Students will be allocated individually to test another team's system in week 11. You will arrange to meet virtually with a representative of the team you are testing, who will share with you their software bundle. You will run a series of prescribed tests (to be released later) of their system.

- Submit your online testing web-form any time between Monday-Friday in week 11, but **no later than 17:00 Friday week 11.**

Late submissions

Standard lateness penalties will apply (deducting 5% of your actual score per working day).

Objective

The objective of this project is to create a software system to meet the needs of an organisation described below. The system will be implemented in Java and will include a database in MySQL and a user interface in Java Swing. You will develop a number of UML models during your analysis, which will form part of your report. The design of the database and the user interface should follow directly from these models. The system should be able to perform the requested functions. The system will support different user roles having different access rights, and will be robust to obvious cyber-attacks.

Background Information

Your customer is Build-a-Bike Ltd., a wholesale warehouse and retail business that sells custom-built bicycles. It has an in-store computer system used by both the shop staff and shoppers. The system will capture details about bicycles, customers and orders.

Stakeholder roles

The different kinds of stakeholder include shopper, customer and staff roles. Shoppers visit the warehouse and browse the products available. Customers have one or more orders placed in the system. Staff have complete rights to view and update data in the system.

Business information

Build-a-Bike Ltd. sells custom-built bicycles. To make things slightly easier for shoppers, bicycles can be assembled from combinations of pre-specified frame-sets, handlebars and (two) wheels. A frame-set includes the frame and front forks as standard (do not model the forks separately) and comes in a certain size (cm). It also may or may not offer shocks (shock absorbers, built into the seat pillar and forks). Each frame-set comes with pre-specified gears to suit the frame (so not a separate user-selectable option). A handlebar comes in one of several styles, either straight, high or dropped. Wheels come in different diameters (cm) and offer different styles of tyre, either road, mountain or hybrid. Wheels may also offer either rim or disk brake systems (including the brake levers).

Every kind of product in the shop, including the components and the assembled bicycle, has a product name, a unit cost (in money), a brand name and a serial number. While the serial number will be unique to all products offered by one brand, different brands may sometimes use overlapping serial numbers. While all components have predetermined brands and serial numbers, we create a brand for the assembled bicycle from the frame-set brand followed by one of {road, mountain, hybrid}, and give it a unique serial number and a custom name chosen by the customer. The unit cost for a bicycle is the £10 additional charge for assembly.

When a customer places an order, the order will have a unique order number and will specify the date of the order. The order will consist of a number of lines which are materially part of the order. Each line is simply numbered 1..n per order. Each order line will specify a quantity of one of the shop's products (typically 1, but 2 for wheels), and will specify a line cost for this quantity. A typical order will therefore have four lines for the bicycle, frame-set, handlebar and (2) wheels. Many orders may refer to the same products. The order will have a total cost, calculated from the line costs. The order has a secret status field, denoting one of: pending, confirmed, fulfilled.

Every order is for a given customer and is (eventually) processed by a member of staff. When placing an order, a customer must also specify their personal details. They give their forename and surname and their address. An address records the house number, road name, city name and postcode. The house number and postcode uniquely identify an address. A customer may place many orders; and more than one customer may reside at the same address. Different customers may have overlapping names.

System operations

The sales system is used as follows. It is installed on various computers around the warehouse. A shopper can browse what is available. They can start assembling a virtual bicycle, adding a frame-set, handlebar and wheels. They can change their minds and replace components. When ready to place an order, they become customers. They enter the required name and address information and submit their current bicycle configuration to be ordered. This creates an order with an order number and order lines referring to the correct quantities of each product. They can cancel or save the order.

A customer can review an order that has been saved by entering its order number. If they forget this, they can enter their forename, surname, house number and postcode and be presented with a (hopefully very short) list of recent orders from which to select. They can still delete the order at this stage and start again. Once happy with their order(s), they can go to the checkout area where a staff member will ask for the same order information and will process the order, adding their staff username to the order. Once the order has been paid for by the customer, it cannot be changed again. The order is moved to a queue where other staff in the shop can access it and assemble the bicycle described from components in the warehouse.

A staff member can login to the system using a unique username and password. A Mandatory Access Control policy is required, where one IT staff member sets up the encrypted table of users and passwords offline. This is read at system start-up, and users do not have permission to self-register or change their login details. Once logged in, a staff member can view any of the data, including customers, addresses, orders and products of each kind. The main staff activities involve adding new product items that arrive in stock, advising customers and accepting payment from customers, and assembling a bicycle from components, which depletes these items from the stock. Orders are moved from a pending queue (while the customer can delete it) to a confirmed queue (paid for, but awaiting fulfilment) and a fulfilled queue (when the bicycle has been assembled and the order is archived). Payment is handled separately by the usual visa system. Customers can wait for their bicycle, or collect later (with their order number).

User interface considerations

The system is a single system used by all three kinds of stakeholder.

- A shopper can only browse various products and assemble a virtual bicycle. They cannot view orders or other customer data, nor modify any product information;
- A customer is created once a first order is placed in the system, and their customer details remain in the system. They can view and change their own name and address details and view any orders relating to them, including archived orders.
- A staff member can act on behalf of a customer they are helping; and can also access the catalogue to enter new products; and fulfil orders, which also depletes products.

Security considerations

The system will be designed to face shoppers initially, with discreet buttons for staff and customer access. A shopper need not be authenticated. A customer will be authenticated through their forename, surname, house number and address. When an order is placed, the system will ask for these details to see if this matches an existing customer. If not, the new customer is asked for their full details. Once in the system, a customer can update their name and address, without losing their order history. They can view all orders and delete pending orders, but not otherwise alter the state of an order. Staff members are authenticated through username and password. The passwords known to the system must be stored securely, such that a hacker could not download and use them. The system will support authorisation (of users to perform specific tasks). The system will be resistant to privilege escalation (obtaining higher authorisation). The system will be resistant to SQL-injection (triggering malicious database updates).

Software System

Your software system should be able to perform the following test-tasks, as evidence that its functionality works as desired:

- Allow a staff member to add several products of the different component kinds
- Allow a staff member to add products with the same serial but different brand
- Allow a shopper to create a custom bicycle (generating a brand and serial)
- Allow a shopper to dis-assemble and re-assemble with different components
- Allow a shopper to place a first order and enter new customer personal details
- Allow a customer to place a second order against their existing personal details
- Allow a customer to view their orders, delete a pending order, but not a fulfilled one
- Allow a customer to change their personal details, without losing previous orders

- Allow a staff member to progress an order upon receiving payment
- Allow a staff member to fulfil an order, depleting items from the stock

Final Team Report

The main purpose of the team report is to show your design process, leading to the implemented system, which will be handed in separately and tested. The data capture and data normalisation stage are especially important and should be done accurately, reflecting the background information exactly, and not contain extraneous material. The report should contain the following:

- a short introduction, clarifying any interpretations you made of the requirements;
- a UML class diagram of the **initial information model**, developed by analysing the given background information, showing classes, attributes, associations and association classes. All associations should have end roles and multiplicities;
- a UML class diagram of the **normalised database model** (using the **UML database profile**), which should have normalised all the relationships in the initial information model and identified primary and foreign keys. All remaining associations should be directed, according to table-linkage;
- a UML **state machine diagram**, showing the different authenticated states of the system, with transitions describing what actions can be performed in these states.
- some screenshots (max 2 sides) showing off what you think are the best aspects of what your system can do (screenshots before/after critical events are best). Don't cram in so much that it is unreadable;
- a short discussion of the security features your system implemented.

Individual Weighting

Your report must finish with **two separate measures** of the contribution and the effort put in by individuals in the team:

- a table describing what actual tasks were carried out by each individual;
- divide up 100 points among the team members, according to effort invested;

The first of these is a **factual account** of what each person did; and the second is an **agreed sharing** of credit for effort invested, especially if this was disproportionate. These measures are distinct; a student may have invested significant effort, but fewer of their contributions may eventually have been included.

This last section **must be signed off by all team-members** to be valid, otherwise equal effort and contributions are assumed by default. If there is serious disagreement, this should be reported, and different parties should report their different views of contribution/effort. Assessors reserve the right to arbitrate and alter effort scores in this case.

Marking Scheme

The marking scheme will be as follows:

- 30% for accurate UML Diagrams
- 60% for correctly operating software
- 10% for team working strategy

Individual marks will be weighted *partly* according to the whole team score, and *partly* according to the share of contributions/effort, following a **non-linear scheme** that encourages collaborative working, rather than heroic individual efforts. No individual mark will be boosted by more than one degree-class. Low-contributors may have their mark reduced below the pass threshold. Non-contributors will receive a mark of zero.

Hints on Team Working

This team project is large enough that you must tackle it by a divide-and-conquer strategy. This means you must learn how to work effectively as a team. Choose a team leader and delegate tasks to different team members. Check up regularly that assigned tasks have been completed. When work has been completed, have someone else in the team review it to point out any possible mistakes or things that need to be improved. You can book break-out rooms in the Diamond for your team meetings. Meet often and do a little each week.

You may find you have a different spread of skills among the team. Use this: some may be good at UML design, others at database normalisation, others at Java Swing coding, etc. All of these aspects are important (not just the coding). The marking scheme is designed to reward collaborative working. Individual weighting will not excessively reward heroic efforts. It is your collective responsibility to make the team work as a group.

Tools and Technology

The UML diagrams can either be developed in any of the suggested Open Source UML tools (see end of Project Management lecture), or even in a drawing package such as Visio (or even PowerPoint). Please use a UML 2.x compliant notation. **Do not use non-UML database diagrams**.

The software should be implemented in Java, using Swing to build the user interface. You may use any GUI designer tool that generates your Swing look-and-feel, so long as you know how to link the generated code to the events that your system must process. You may also build your Swing GUI by hand, from the ground up, if you understand that better.

The MySQL database accounts are created by the DCS IT support team and will be distributed by your lecturer (when he gets them). These group accounts are on the Computer Science internal network, not available to other students or outside the department. You will need to use the Connector/J driver for MySQL (instructions to follow in lectures).

Further Help

From week 6, there will be several Online Helpdesk surgeries, operated at different times by senior student Demonstrators who have signed up for this. There will be additional Java/MySQL labs, operated by staff or University Teachers, to help build up your coding skills. Please see the times advertised in Blackboard, under the module overview.

We will use the Blackboard discussion forums for this course to answer further technical questions. Please follow the etiquette of using the named threads to ask questions, or share answers, on similar topics; and only post a new thread if there is not one already on this topic. Module instructors will respond to questions posted there on a regular basis, so that answers to common questions can be seen by all. Email will not be used for technical questions.

AJHS, August 2022