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. As a team, you will:

- Analyse a business requirements document, to understand data and operations.
- Design a conceptual information model of the business, using the UML class diagram.
- Design a normalised data model, based on the above, using the UML database profile.
- Design a UML state machine diagram for the user interface, showing how different user roles access different sets of operations.
- Implement a database in MySQL, following your normal database table designs.
- Implement Java middleware to perform the business operations, which access or update the database.
- Implement a user interface in Java Swing, following the state machine design for the user interface, which will trigger business operations.
- Write a short 10-12 page report about your project, including the three UML models, two pages of representative screenshots, and security considerations. Describe how you allocated the workload and managed the project.
- Individually, you will also complete a test questionnaire about another team's system.

The main thing is to ensure that your design and build stages clearly follow on from one another, for example, the information model should faithfully capture the data needs of the application; the database model should clearly be derived by normalising the information model; the state machine model should clearly capture screens of the user interface and support user roles that access these screens; and the implementation should follow directly from these designs (not independently built). The system should be able to perform the requested functions and will be robust to obvious cyber-attacks, such as SQL injection.

Background Information

Your customer is *Trains of Sheffield*, a retailer of model railways (train sets, locomotives, rolling stock, and track). It wishes to provide a new in-store computer system used by both the shop staff and customers. The system will capture details about products, customers, and orders/sales.

Stakeholder Information

The different kinds of stakeholder include customer, staff and manager roles. Customers browse the products on sale and make up orders, which they can later purchase, after supplying bank details. Staff may also be customers, but also have rights to view and update stock data in the system, process orders and view sales, but not view any other customer's bank details. The manager is a staff member, who also has the authority to appoint other users as staff. Any individual user may therefore have up to three different roles.

Business Domain Information

Trains of Sheffield sells model railways, either in boxed sets, or as individual models and parts. There are different kinds of boxed sets, where a set is a convenient assembly of its model parts, which may also be sold separately. A *train set* includes one or more locomotives, some rolling stock (several carriages, or wagons) and one or more *track packs*, and a *controller*. A *track pack* is a starter oval of track, or an extension pack containing a point-switch and additional track pieces.

Every product in the shop has a *brand name* (manufacturer), a *product name*, a *product code* identifying the type of product and a *retail price* (in money). Boxed sets are sold under their own product codes and contain parts which have product codes, according to type. Sets may contain quantities of each type of product, for example the *2nd-radius starter oval pack* contains 8 x *2nd-radius double curve* and 2 x *single straight*. Locomotives, rolling stock and track pieces are optionally part of a boxed set, or are sold independently.

Examples of UK-sold brands include: *Hornby, Bachmann, Graham Farish, Peco,* and *Dapol*. Product names are described in more detail below (see business data examples). Product codes identify the type of product (not the individual instances) and consist of a single letter followed by a three to five-digit serial number. Track codes start with R, controller codes with C, locomotives with L, rolling stock with S, train sets with M, and track packs with P.

All products come in several modelling *gauges*, which relate to the *scale* of the model. Trains of Sheffield only deal in the following UK gauges: *OO Gauge* (1/76th scale), *TT Gauge* (1/120th scale) and *N gauge* (1/148th scale). The *scale* can be derived from the gauge name.

All locomotives and rolling stock come from a particular UK historical era, which is important when modelling a consistent historical period. These range from *Era 1* (the pioneering days of railways) to

Era 11 (the current period)[1]. Locomotives, rolling stock, and train sets are labelled with an *Era code*, a string like: "Era 4-5" to indicate that the vehicle was used during those periods.

Each locomotive is labelled with a *DCC code*, to indicate whether it is ready to be used with *Digital Command and Control* (DCC), a microchip-powered system that allows you to drive more than one train on the same track section. Ranging from cheapest to most expensive, these codes are: *Analogue, DCC-Ready, DCC-Fitted and DCC-Sound*. *DCC-Ready* models have a socket ready for the chip; *DCC-Fitted* have the control chip fitted; and *DCC-Sound* also have miniature speakers that play realistic locomotive and rail noises. *Analogue* models cannot be digitally controlled.

Track is supplied in each of the gauges. Examples of track pieces have names like: *Single Straight, Double Straight, Single Curve, Double Curve, Left-Hand point, Right-Hand Point, Left-Hand Crossover* and *Right-Hand Crossover*. Furthermore, all curved track comes in three different radii, so the names of curved pieces are prefixed with *1st radius, 2nd radius* or *3rd radius* (don't model this as a distinct attribute). Points turn out with 2nd radius curves only.

Controllers for model trains are sold as parts of a train set, or individually. They have names like *Standard Controller, DCC Controller, DCC Elite Controller*. Controllers indicate whether or not they are *digital*. A train set has one controller (we assume any power transformers are built into the controller, and are not sold as a separate item).

Business Data Examples

This section contains representative examples of business data, mostly the product names, to give you an idea. You can use these, or find suitable names by browsing manufacturer websites (e.g. Hornby, Bachmann, Dapol) or model railway retailers (e.g. Rails, Hattons).

The names of locomotives consist of a BR class number and name, or sometimes the individual name of the locomotive. Some examples include: *Class A3 "Flying Scotsman", Class A4 "Mallard", Class 5MT Black Five, Class 07 Austerity, Class 7P Castle, Class 7P Britannia, Class 8P Coronation, Class 8P Merchant Navy, Class 8F Stanier, Class 9F "Evening Star", Class 08 Shunter, Class 20 Shunter, Class 35 Hymek Diesel, Class 43 InterCity 125 HST (power car), Class 55 Deltic Diesel, Class 91 InterCity 225 Electra (power car), Class 110 Diesel Multiple Unit (power car), Class 150 Sprinter DMU (power car),* or *Class 800 Azuma (power car)*.

The names of carriages include the historic company name or later British Rail (BR) standard mark, followed by a description of the kind of carriage. Company marks include: *LMS, LNER, GWR and SR*[2]. BR standard marks include *Mark 1* (maroon), *Mark 2* (blue/grey), *Mark 3* (InterCity 125) and *Mark 4* (InterCity 225). Kinds of carriage include: *Corridor First, Open First, Corridor Second, Open Second, Sleeper Car, Restaurant Car, Buffet Car, Composite Coach* (mixed 1st and 2nd class), *General Utility Van* (parcels), *Post Office sorting Van* (for mail), *Brake Van* (with guard and parcels areas), *Brake Second* (mixed brake and 2nd) or *Composite Brake Van*. Special VIP edition coaches were called *Pullman* (after the designer). Historically, there was also a 3rd class, which was merged with 2nd class; this is nowadays called "standard class".

Examples of wagons have names like: *Cattle Wagon, Horse Box Wagon, Parcels Van, 16t Mineral Wagon, 6-Plank Coal Wagon, 8-Plank Coal Wagon, 20t Hopper Wagon, 21t Clam Ballast Wagon,*

[1] If you are curious about what periods each Era covers, look this up e.g. on the Hatton's models website.
[2] The "big four" UK railway companies: *London Midland and Scottish, London North East Region, Great Western Railway,* and *Southern Railway*.

102t Bogie Hopper Wagon, Coalfish Open Wagon, Urchin Bogie Open Wagon, Bogie Bolster Wagon, 20t Brake Van, GWR Toad Guards Van.

Track packs include: *2nd Radius Starter Oval* (including: 8 x *2nd Radius Double Curve, 2 x Single Straight*), *3rd Radius Starter Oval* (including: 8 x *3rd Radius Double Curve, 2 x Single Straight*), *Track Pack A* (including: *Single Straight, Double Straight, 2 x 2nd Radius Single Curve, Left-Hand Point, Buffer Stop*), *Track Pack B* (including: 2 x *Single Straight*, *2nd Radius Single Curve*, 4 x *2nd Radius Double Curve*, *Right-Hand Point, Buffer Stop*).

All the following train set examples happen to come with a *3rd Radius Starter Oval* and *Track Pack A*. Train sets have names such as: *Eurostar Train Set* (including a *Class 373 Eurostar EMU* power car, unpowered trailer car, and two *Passenger Saloons* and an analogue controller), *Mallard Record Breaker Train Set* (including *Class A4 "Mallard",* 2 x *BR Gresley Composite* coaches, 1 x *BR Gresley Brake* coach in cream/maroon livery and an analogue controller), *Flying Scotsman Train Set* (including *Class A3 "Flying Scotsman"*, 2 x *LNER Gresley Composite* coaches, 1 x LNER *Gresley Brake Van* in teak livery and an analogue controller), *Mixed Freight DCC Train Set* (including a *Class 08 Shunter (DCC), Class J83 Locomotive (DCC), 12t Vent Van, Tanker Wagon, 21t Iron Ore Tippler, 7-Plank Wagon* and a digital controller).

Business Operation Information

The shopping system is installed on computers all around the shop. All users (staff and customers) must self-register on the system. They must supply an *email* address and a *password*, but the system actually stores this information under a surrogate *userID* (to allow for changes to an email address).

When a user registers, the system will also ask for their personal details and address information. A personal record consists of a *forename* and *surname*. The address consists of a *house number, road name, city name* and *postcode*. The house number and postcode uniquely identify an address. Every user has one personal record. One or more people may live at one address.

Someone with manager authority must promote a registered user to become a staff member. We assume there is at least one person with the manager-role (set up initially in the database) and this person can appoint other users as staff members or remove this privilege.

Registered customers will browse the items for sale and choose which they wish to purchase. While browsing, they will add items individually to an order. Each selected item creates an order line (part of the order). Customers can add single items (e.g. a locomotive), or multiple copies of an item (e.g. a quantity of a desired track piece) on an order line. They can also remove order lines, up until they confirm the order. If an item is known to be out of stock, it cannot be added.

Many orders may be created by a customer, and each order relates to one customer. An order has a unique *order number*, the current day's *date*, and a *total cost*, which is derived by summing the individual line costs of each order line. An order also has a *status* field, one of {*pending, confirmed, fulfilled*}. It is *pending* until the customer confirms the order, *confirmed* until the shop has made up the order, and *fulfilled* after this, when payment is taken.

An order consists of one or more order lines which are materially part of the order and do not exist independently. Each order line has a *number* 1..n which is only unique for that order. Each order line refers to one of the shop's products and will specify a *quantity* of this product. The order line will derive a *line cost* from the quantity and the related product's retail price. Each order line refers to exactly one product, and a product may be referred to by many order lines.

On screen, the system will display the order number, the date, the related customer's personal details, email and address (but not the secret userID). Under this header information, the system will display the order lines in order, showing: the related product code, the brand and product name, the quantity, and the derived line-cost. An example order line will display: {*R607, Hornby, 2nd Radius Double Curve, 8, £32*}, assuming that the R607 double curve costs £4 each. The system will also display a total cost and the status of the order.

The first time a customer confirms an order, the system requests their bank details. This consists of: a bank card name (e.g. *Visa, Mastercard*), a card holder name (a single string), a uniquely identifying bank card number (sixteen digits – four groups of four digits), a card expiry date (*mm/yy* format on the card) and a three-digit security code. This confidential information is only viewed by the owning customer. The system only stores one set of bank details for a purchasing user, and some users never need to give bank details (e.g. staff who never purchase).

When a customer confirms their order, this is sent to a queue of orders to be handled by the shop counter staff. If the bank details are absent or invalid, the system will not allow order confirmation – and will feed this back to the customer. A staff member will see the confirmed order, which displays the order number, the date, the customer's personal information, email and postal address, the order line contents and the cost of the order, the order status, and the fact that a valid bank card exists. A staff member will then go to the shelves or warehouse and fetch the items purchased. When these have been presented to the customer at the counter, the staff member will complete the order, at which point payment is taken and the order is marked as fulfilled, and is removed from the queue and archived.

The shop's inventory database records what kinds of product, and what quantities of each, are in stock. From time to time, the shop will receive new deliveries. When this happens, a staff member checks the product items in, and updates the *quantity* of each item. When a staff member fulfils an order to a customer, the corresponding quantity of each item is decreased, so that the inventory always has a correct record of the items in stock. Orders are processed serially, according to the queue, such that if stock runs out, the next customer's order requesting this item cannot be fulfilled. A blocked order may be declined by the customer and deleted by the staff.

After logging in to the default page, a customer may choose to view their recent orders, or to change their personal, email and address details, or to change their bank details. When browsing their recent orders, they will see a list of order numbers and the date of that order. Clicking on an order will take them to the display of that order (similar to the view seen by a staff member). Orders cannot be altered once they have been confirmed. A staff member can browse a list of all fulfilled orders, by order number and date, viewing these in the same way.

User interface considerations

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

- A customer can create and edit personal, email and address details and bank details, can browse products, create and edit orders and view old orders; but cannot alter any stock records or view another customer's details, or their order history;
- A staff member can also create, edit and view product records, update stock levels, and can view orders waiting in the queue, with full customer details, and can then fulfil or delete the order; but cannot view any customer's bank details.
- A manager is a staff member who can also appoint users to be staff members or dismiss users from this role. The manager role is fixed in the database and is not changed.

The system will be designed to face customers primarily, with a discreet button to enter the staff area (invisible to non-staff users), which has a discreet button to enter the manager area (invisible to non-managers). The default view will display all the categories of products (train sets, track packs, locomotives, rolling stock, track, controllers). After selecting a category, customers may add items from this category, update the quantity, or remove items, or go back to the default view. From the default view, another button will take the customer to their order screen, where they can edit order lines to alter quantities, or delete the line, or confirm the order, or go back to the default view.

In the staff view, the default display will also display all the categories of products. Within each category, staff may add, delete or edit individual products and update their quantity (typically only increasing, since decreasing is automatic upon sale). Another button will take them to the pending order queue, ranked by date/time of submission. They can access the top order, which can be fulfilled or deleted. When fulfilled, the order disappears from this queue and it is archived.

The manager view displays a screen showing existing staff (displaying email, forename, surname) with options to remove a staff member, and a box to enter the email of a non-staff user (with only customer privileges) to promote to the staff role.

Security considerations

Users can only edit their own personal, email, address and bank details. Customers can browse all products, but not change quantities. Customers can create and edit orders. Staff can browse and edit products and quantities, but not see any customer bank details (other than the validity check). Staff can browse orders and see customer details attached to a confirmed order. Staff cannot change orders but can only fulfil or decline (delete) them. Staff may also be customers. Managers are also staff.

All users are authenticated through an email username and password. The passwords known to the system, and a customer's bank details, 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 Operations

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

- Allow three users to self-register with email, personal details and address.
- Allow the manager to promote two users to the staff role, and dismiss one later.
- Allow a staff member to add products to the inventory, in different quantities.
- Allow a customer to browse products and select items in different quantities.
- Allow a customer to view and modify their pending order and delete an order line.
- Allow a customer to confirm their first order, then have to submit bank details.
- Allow a customer to confirm a second order against their existing bank details.
- Allow an order to be refused, for various reasons (stock levels, bank card).
- Allow staff to fulfil two orders for a customer, and see the depleted stock levels.
- Allow a customer to view their past orders, but not modify them.
- Allow a customer to edit and change their email, address and bank details.
- Allow a customer to login under a changed email, and still see all past orders.
- Allow staff to view a history of all orders and select an order to view.

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 that is not required by the business scenario. 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, generalisations, aggregations, compositions, associations and association classes (for relationships with attributes). All associations should specify multiplicities (end-role names are optional). The class diagram should factor out common information in a suitable class hierarchy, using abstract superclasses. Take care to model any derived properties as operations. Indicate any identifying attributes (see lecture on *Information Modelling*). You may assume the pre-existence of simple datatypes *Date* and *Money*, and may also assume the pre-existence of UML enum types for enumerations (such as *Status*). Other codes are often *Strings*.
- a UML class diagram of the **normalised database model** (using the **UML database profile**), developed by normalising all the relationships in the initial information model and identifying primary and foreign keys. Use the rules for normalisation given in the lecture on *Database Design*, to convert generalisation, aggregation, composition and association relationships into simple directed associations, according to the required direction of data dependency (table linkage); these should all have a many-to-one multiplicity. Take care to use the correct primary keys, whether natural, surrogate, or composite (e.g. for a weak entity). Remember to merge one-to-one, and split many-to-many relationships, creating any necessary linker tables. When normalising generalisations, preserve a separate table for the *Product* superclass, but flatten other generalisations according to the disjoint subclass rule.
- a UML **state machine diagram**, showing the different screens of the system, with transitions describing what actions can be performed in these states. The diagram should capture the ability to login/logout of the system. The diagram should show how the system enters the customer view by default, and may navigate to and from the staff and manager views for users with the required authorisation. We suggest you use hierarchical state machines, with one machine for the activities of each role (see the lecture on *State-Based Design*).
- 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

We will attempt to have two anonymous **buddy check** polls for your team after the design stage, and after the implementation stage. This depends on you registering your team in Blackboard, and taking the polls at the indicated point (we will send announcements).

Your report must finish with a **factual account** of what each person did, showing a table of contributions of each member. These can include which UML diagrams you contributed to, or which pieces of software you wrote. This measure is distinct from the effort indicated in the buddy check polls. This last section **must be signed off by all team-members** to be valid. If there is a team

disagreement over contributions, this should be made clear in the report. Assessors reserve the right to arbitrate and alter individual weighted grades.

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 the *Project Management* lecture), or even in a drawing package such as Visio (or 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 5, 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 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 2023