PROGRESS OUTCOME 6

Selling cellphone cases

Context

A Young Enterprise group called Caseman has designed and made iPhone cases with a 3D printer. They have approached Hemi to help them create a system for producing invoices.

In consultation with Caseman's CEO (John Smith), Hemi has identified the information each invoice should show and that a database will be required to meet end-user requirements.

Insight 1: Design decisions based on client needs

During discussion with John, I identified that the database would require a number of tables (e.g., customers, items, and invoices) to allow the group to keep track of each invoice. I had to consider how to enter the data, the information the database needed to produce, and who would be using this information.

I explained to John that using validation and drop-down lists based on already available information would minimise data-entry mistakes. John wanted to allow other members of the team to access information, but to avoid the duplication of orders (a problem that had already occurred), he requested that the database be set up so that only he could add new invoices. He agreed to the use of permissions and passwords to make this possible.

John also wanted to be able to create emails to offer new products or special offers to previous customers and to offer the specials to customers who had paid in full. I explained that this could be done using advanced queries and sending the results to his email program (Apple Mail).

Insight 2: Developing the outcome

I started by identifying the information I needed for the tables. My solution could not be in the form of a flat file because a customer might want to order several different cases for family and friends. I decided to set up a relational database.

The main table for the database became the invoices table, where a record of each invoice is stored. It includes a primary key field (pk_InvoiceID) that is unique and increases by one for each new invoice. I also included an items table, in which each item appears once with a specific price attached to it. If John needs to update pricing in his invoices, he only needs to change it in that table.

Because each invoice can have more than one item and any item can appear on many invoices, I needed to create a many-to-many relationship. I did this by creating a simple join table.

I linked the 'Invoice_Item' field to the 'Item_Name' field in items so that each line in the invoice would only be able to select an item available in the items list. 'Item_Cost' is retrieved from the items list and populated with a simple script. Although not strictly necessary for creating invoices, I added a customer table to make it easy to create invoices for previous customers and to support the email function that that John had requested.

Insight 3: Testing using data

Once I had designed the main invoice table layout, I tried adding some data to the system. I immediately realised that as well as looking up previous customers, John would need to add new ones. I included that option in the layout and tested it by adding several different records.

My display for the line items worked well, but I needed to add a delete button and a simple script to give the correct value for the price for multiple orders of the same item.

I also realised the need for a total, which I created with a summary field. I tested the invoicing system by selecting a name from the existing customer list, which updates every time a new customer is added.

Insight 4: Data security measures

I then considered data security and privacy in the design.

To allow other members of the team to look at invoices but not create new ones, I set up two password-protected accounts. The one for 'staff' allows read-only access. The other account is specifically for John. It allows him to create and delete invoices but not to change layouts or scripts. Only I can do that as the designer, which protects the intellectual property that I have asserted over the design of this database.

Because both of these accounts use passwords, all the customer information and database details are kept private and secure.

Insight 5: Setting up the auto-email

John was very pleased with the final design of the database and the ability to keep track of invoices and find customers quickly. I started work on the option to create a simple email that would send a promotional message to customers who meet certain criteria. This was much more complicated than I'd expected because the same customer can have more than one invoice.

I set up a field to count the total number of invoices a customer has in the system. I then created a new 'Paid' field for each invoice, which John selects when an invoice is paid. To complete the query, I created a calculation that compares the number of invoices a customer has against the number of 'Paid' entries. If they match, the field status is marked as 'True' and the customer is eligible to receive an email.

To generate the email that John wanted, I created a simple data-entry form for adding a message title and message content. I then created a script that would carry out the query and produce an email that merged customer information with the message.

ISBN: 978-1-77669-255-2

MINISTRY OF EDUCATION
TE TĀHUHU O TE MĀTAURANGA