Desktop Application Development

Final Project

DigitalX is a newly founded business, with plans to sell music, DVD's, games and various components to customers worldwide. The business has already begun creation of an online store, where customers will have the ability to sign-up and electronically purchase any of the varying items available. You have been contracted by DigitalX to develop an application for their main warehouse, encompassing the aspects required to successfully package and ship purchased goods. A SQL Server database has already been created to work with the proposed desktop application, and has been designed to provide everything necessary to complete the project. DigitalX have provided you with the following description of how they envision the system working, followed by a role hierarchy describing the differing employee roles and the privileges or restrictions for employees regarding the use of the application.

DigitalX Management System

Requirements

Optional (Choose A/B)

As well as the requirements specified below you need to choose one of two optional tasks A or B and apply these tasks while implementing your solution.

A. Complete at least two sections of the application using asynchronous techniques. This may involve multitasking to prevent poor user interface experience for the user.

B. Implement application wide styling and resources instead of styling page by page

Orders

The main objective is to develop a system that interacts with the existing database, providing the necessary functionality to efficiently package and track orders received from the DigitalX website.

This application should provide an interface to add orders or modify existing orders. Orders belong to specific customers so you will need to consider how to tie orders to the customer in user interface design.

Once orders are ready they are packaged in the warehouse before despatch. An option to initiate this process is required.

The required process for order packaging and despatch:

- Each completed detail of an order must record the id of the employee who packaged it
- The units packaged must then be deducted from the units in stock for that particular product

- All finished orders must be assigned a date and employee id to indicate completion **and** have invoices created for them.
- An order must not be marked as complete without an associated invoice.
- The order system must provide the ability to split an order in case certain items ordered are out of stock.
    - The original order should retain all items that can be packaged
    - A second back-order order should be created comprised of all the items that currently can't be packaged.

Employees

An employee management interface must be provided for administrators to view, create and edit employee information including password and role designation. Employees may have more than one role.

Customers

A customer information / management system must be created to provide contact information, and a means of updating customer information.

Products

The ability to view, add or edit product information must be provided, including functionality to increase stock amounts, change prices.

Employee Roles

Following is a list of roles that DigitalX want to be assigned to any employees who uses the application along with the functionality that each role should be able to achieve.

Administrator – 'Admin'

- Unrestricted access to the all aspects of the application.

Order & Shipping – 'Despatch'

- Access to complete or view product information (**not edit**) for an order.
- Access to view (**not edit**) customer information.

Customer Service – 'Service'

- Access to view and edit customer details.
- Unrestricted access to order information.

Stock Control – 'Product'

- Unrestricted access for product management.

Human Resources - 'HR'

- Unrestricted access for Employee management.

General

Provide thorough help functionality throughout the application. (See documentation section following)

Planning and Documentation

You are required to plan and document the development of the application; this should include information about the windows you intend to create and for what purpose; how you plan to complete certain requirements; technologies you will use and the reasons for those choices; a final overview of your finished application detailing changes you have made to your original plan and any particular areas of success or failure you may have encountered.

Suggested Documentation Research

There are many ways of documenting a project like this. Although you haven't been shown any documentation techniques other than commenting code, documentation will form an important part of your role as a software developer.

Firstly you will have requirements documentation. Think of this project outline you are reading now as a summary of requirements or discussions given to you by the client. You will need to summarise and list the requirements and this will vary depending on the option A or B you have chosen.

Secondly once you have summarised your requirements you will need to start to work out the way that you will need to design your solution to meet those requirements. In this case as the database is supplied you will not need to design that. In some cases the database may need to be further developed to support the new functionality, but in this case that is not required. What you will need to do is determine the data you will need to access to accomplish the tasks outlined in the requirements. This will determine the classes you will need to design to access and process that data.

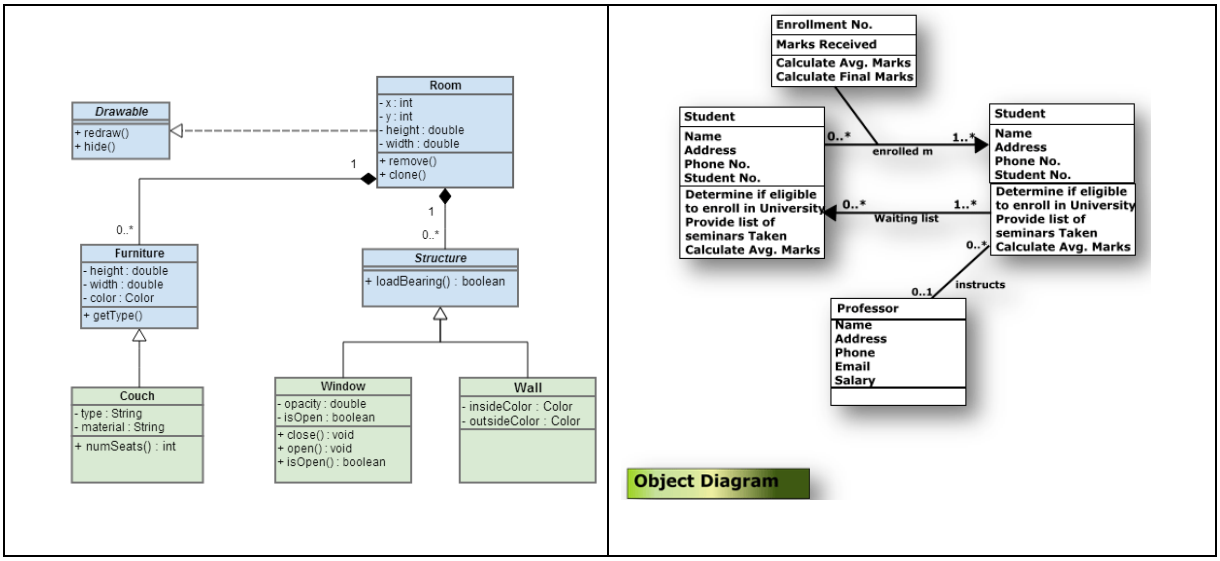
Having done this you will be able to determine the user interfaces required as you will know which data needs to be processed and whether this requires data from single or multiple tables from the database to be displayed on each window.

So firstly you can design and document the classes required for data access and processing. You may need additional helper classes other than just the data classes themselves. Then the classes required for the user interfaces. These can each be documented in different ways and you do not need to document every little item.

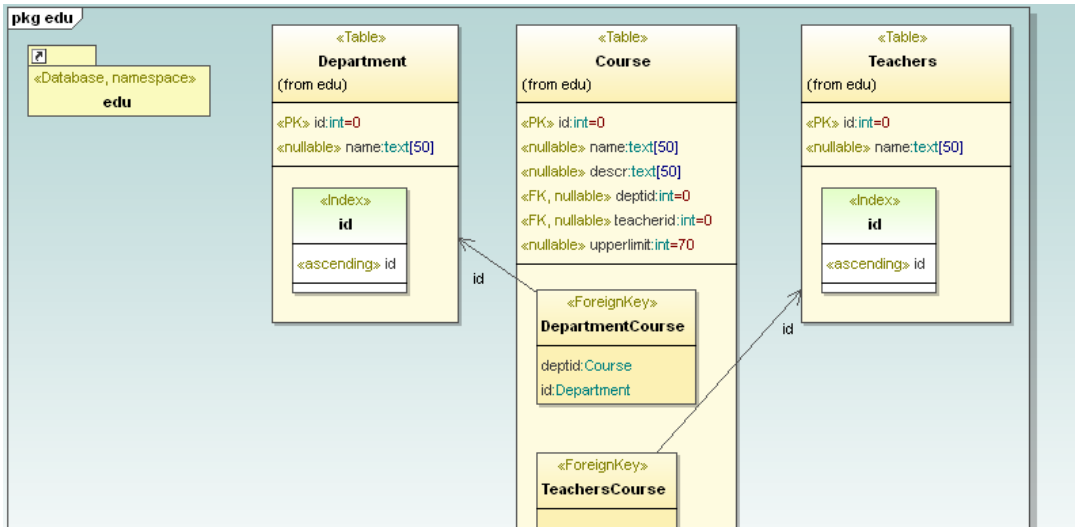
In general when converting requirements to classes names will become classes such as Employee, Order etc. and adjectives will become methods i.e view, add, edit on each of those classes. Where you are looking at different categories of named items these potentially may become a class hierarchy especially if each category has different behaviour for example employee roles. Equally they could be represented by enumeration types used to select behaviour it depends on the complexity required. The more complex the

more they are a candidate for inheritance. Don't forget too that aggregation is possible, inheritance isn't always the answer.

Investigate UML for documenting your classes; this is available in an application named Visio which is provided in your VM. Depending on your version it can also generate code once the class definition is complete. Try not to get overwhelmed by complexity and keep it simple. You can document class inheritance using this and also the relationships between classes or their member. This can be useful for documenting the database. http://www.c-sharpcorner.com/UploadFile/nipuntomar/uml-diagrams-part-1/



This shows a database diagram (some apps do not do nested items in the diagrams)



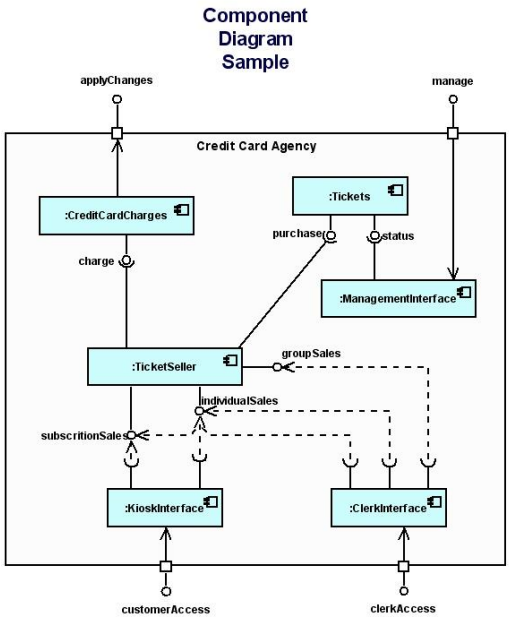
There are over a dozen types of UML diagram types. You will find other types of diagram in Visio also.

Another alternative is simply to use a work processor to outline the members of your classes and describe any relationships between them as you will see in MSDN documentation.

Data flow diagrams provide information about the flow of data through the different parts of you solution including methods and user interfaces along with the way operations transform data flows. These are

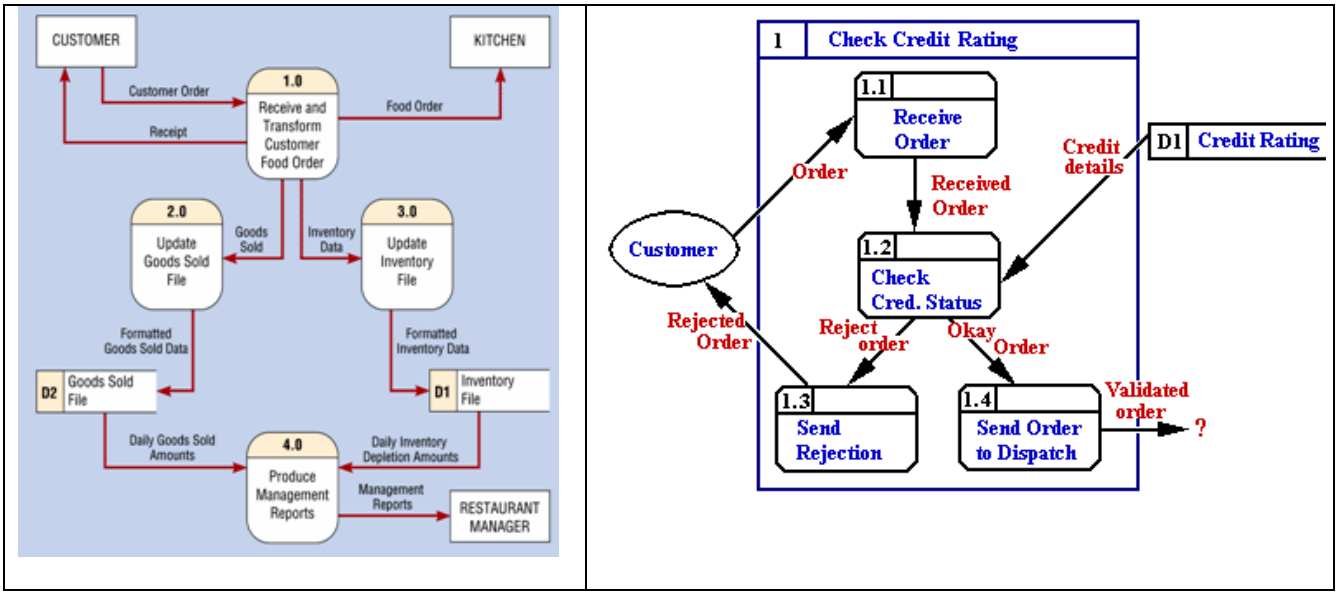
general broken down from high level views showing all modules to low level views showing a single module or sub operations of a module. We would not expect this but you may want to provide a top level diagram showing all of the components of your application.

This is a component diagram. This may be too complex. Do not spend too much time on documentation.

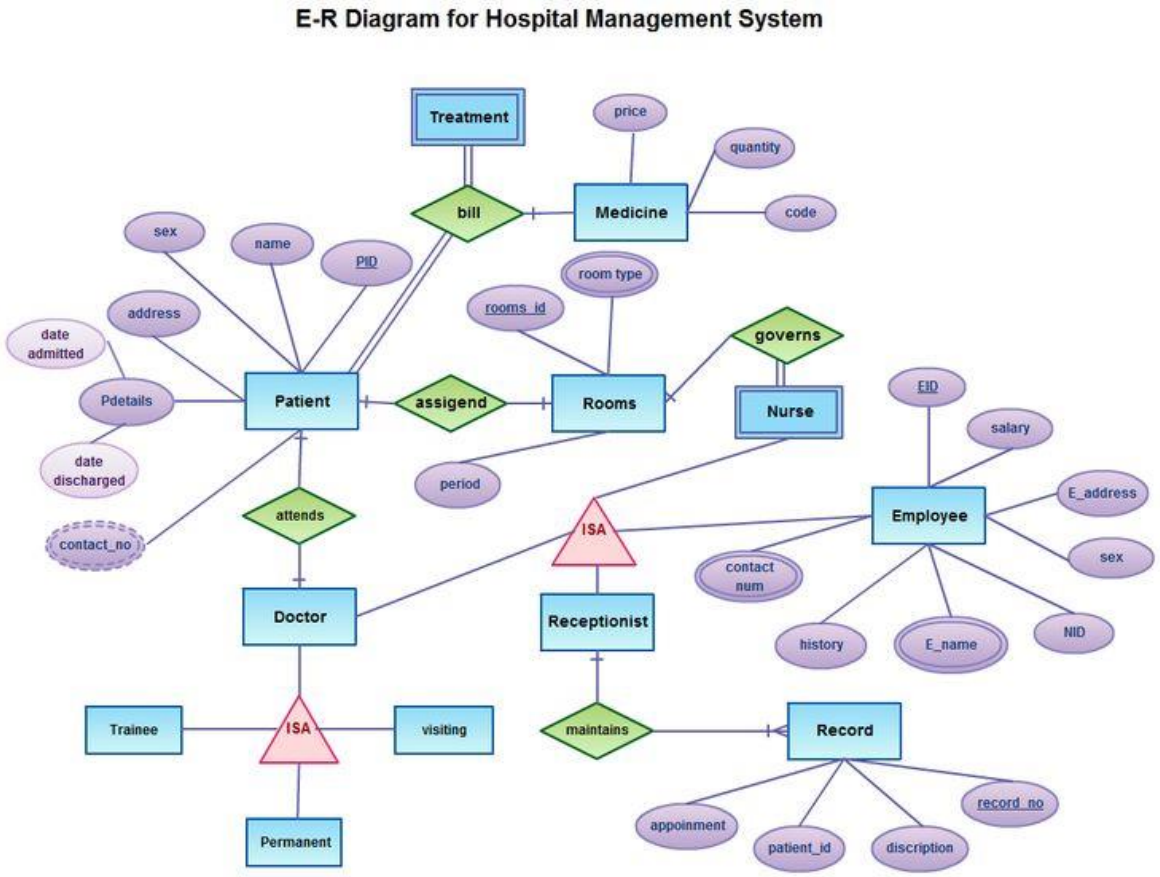


I would recommend drawing or sketching diagrams first and then creating the in the app you are using until you get used to creating them from scratch yourself.

Data flow diagrams

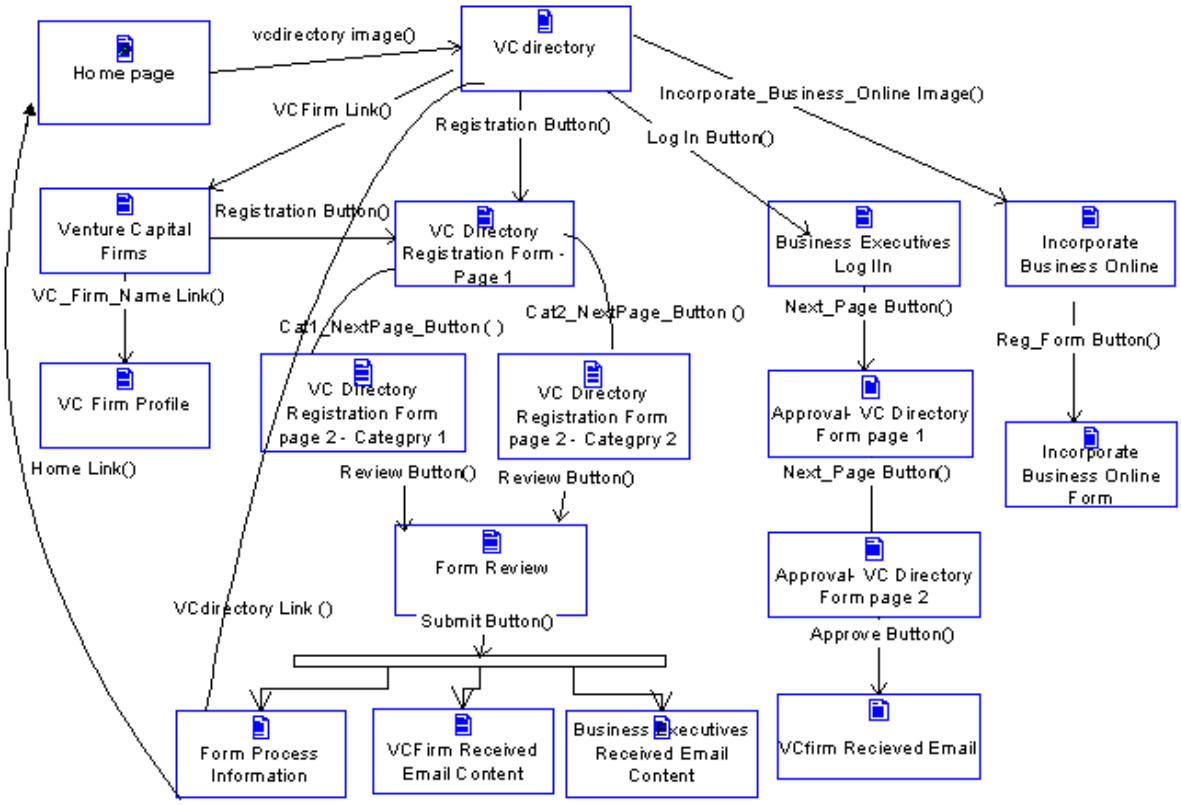


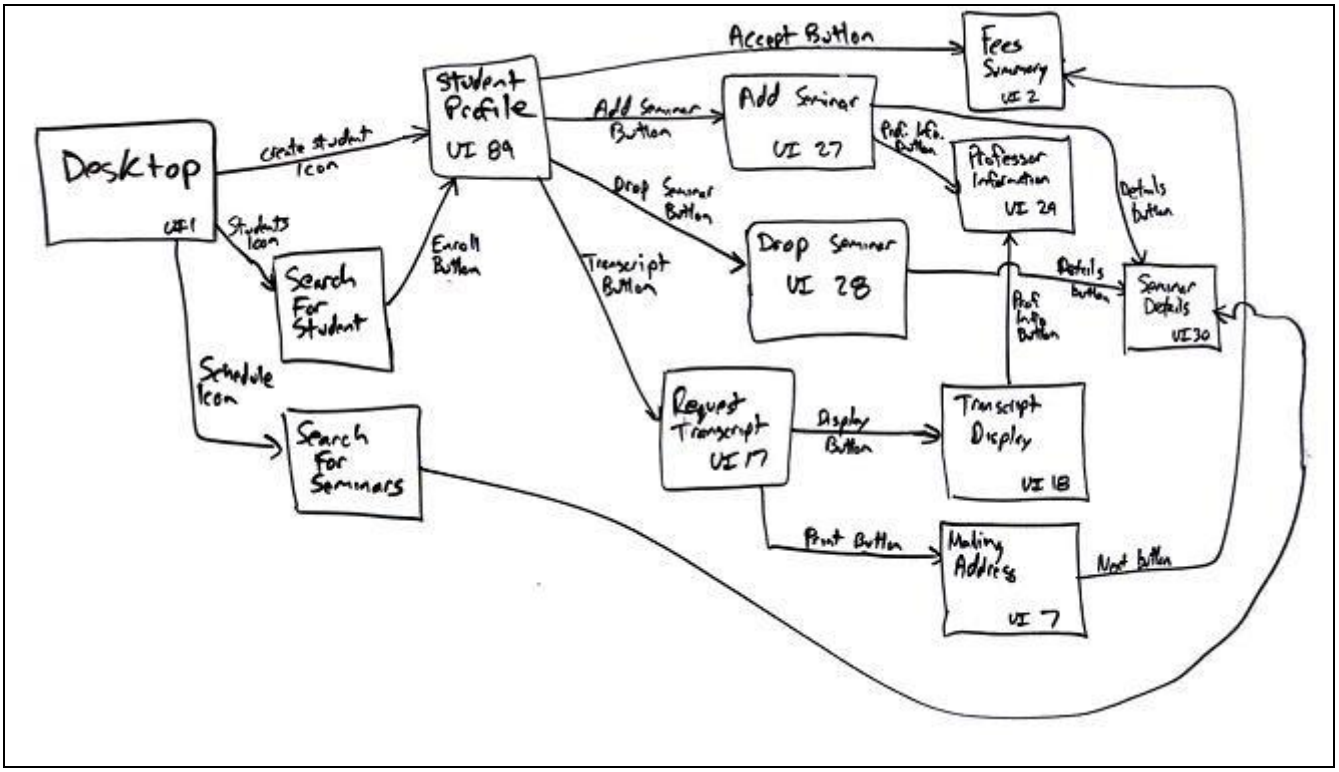
Another type of diagram is an Entity Relationship Diagram (ERD). This documents relationships between entities in your solution. This may be top level or down to member level.



Don’t forget you can split diagrams up, you should be able to find examples of this especially with DFD.

As this is a GUI based application you should provide a diagram or a description of the user interfaces involved. In the case of a diagram this may show the navigation paths between all of the UI windows. You can also provide brief descriptions of the functionality for each page.





Use case is another type of diagram you may consider. It describes the functionally of the app from each users point of view and may accompany formal use case documentation. This can be specialised and like state transition diagrams and swim lanes showing application state we wouldn't expect these. But please read a little on these so you are familiar with them.

We will be looking for proper comments on your code. Each class should have a brief description of what it is for and mention any other classes it interacts with. Just do this with classes you have created. Each method should describe what it does overall. Your code should use meaningful variable and method names. This means you do not need to document each line of code. Keep comments for more complex sections or where a technique may not be clear or you may not remember in a few weeks why you did something. Always comment code based on the fact it may need to be read by others.

Provide at least tooltip help in the application and a brief summary help page for each window.

Your documentation needs to provide a complete description of your solution but do not get carried away.

Summary

- Class documentation
- Database documentation
- Top level application diagram
- User interface documentation
- Commented Code
- Help (Don't get too detailed for this project, tooltips and a help summary page will be fine)

Checklist

Requirement	**Mark**	**Achieved**
Orders can be added or modified Orders can be split or edited as required. Employees can complete individual items and orders. Orders are invoiced before completion.	5 5 5 5	
Employees management Login functionality, password and role designation is provided for employees.	5 5	
Customers management	5	
Products management	5	
Roles specified are used to provide varying access to the application.	5	
Plans and Documentation for the application is complete.	10	
Mandatory tasks: WPF Data Access (ADO.NET Entity Framework) Style/Theme Animation – Possibly progress bar, mouse icon, use your imagination.	 5 15 15 5	
One of the optional task, either A or B	5	
Total Marks:	100	