Department of Information Systems
Systems Design & Development (INF2011S)
Project Guidelines

Objectives of the project

In the first half of the year students learnt how to develop a user requirements specification for a specific business problem (Poppel Order Processing System) using an object oriented approach and the UML. The objectives of the project in the second semester are to:

- Complete the systems development life cycle by designing, constructing, testing and implementing a segment of the Poppel Order Processing System
- Develop a detailed systems specification for a single iteration of the project
- Build, populate and maintain a relational database
- Understand how C# programs are structured and coded in an object oriented environment
- Provide students with the skills to develop larger and more complex systems in their third year projects.

How the Project Works

In order to ensure consistency across the whole class, these guidelines will help you define what will be acceptable as a project.

1. Overall Deliverables

There are two major deliverables making up the project. Both deliverables must be handed in on the due date of the project. They are:

- Systems Specification document as detailed below
- Working C# system (test cases can be handed in at the demonstration & included in systems specification document)

For control purposes you are required to hand in your source code library on the due date and the marker will request you to use this file when demonstrating your project. More details of the hand-in procedure will be published on the course notice board closer to the hand-in date.

2. Functionality to appear in the system:

The required use cases must be developed in C# by groups of 2 students and should contain all the functionality detailed in this document.

Due date for hand-in of the entire system is **16 November 2020 at 4pm.** In the interests of a rapid marking process as well as limiting the impact on exams, this deadline is not negotiable. The usual 5% per day penalty will be deducted from the marks of deliverables that are late.

Note that you will only be able to hand in once! If you wish to re-hand in, you will have to request the lecturer to delete the first one you handed in. After you have handed in **DO NOT MAKE CHANGES TO YOUR SYSTEM, IT'S DATABASE STRUCTURES OR CONTENTS**.

3. Getting help

A Hot Seat will be implemented to assist you with coding problems during the second term. The Hot Seat will be online, conducted in the same manner as the workshops. Detailed dates and times will be posted on Vula closer to the time.

4. The project marking process

You will be required to demonstrate your project to your markers in a half-hour (max 45 mins) appointment. The objectives of this demonstration are to:

- Allow you to present your system in the best possible light and ensure the markers see the results of your efforts
- Allow the markers to ask the student more technical questions about the project, if the marker believes that the project is not the student's own work.

Information about the procedure of demonstration will be provided closer to the time. Students should prepare to demonstrate their system and should have sufficient sample data in the databases to demonstrate that their system works properly.

Further, the marker may, at his or her discretion, require either member of the group to demonstrate sufficient technical understanding of the system to be able to state how certain changes might be made to the system, or to explain how a particular piece of script works.

5. Dishonesty and uneven effort in the group

Marks will be awarded jointly to the group, even where there may have been differing skills levels on the part of each member. If however, in the opinion of the markers, the ***workload and effort*** in the group has been unequally distributed, or the system has suffered significantly because of non-participation or poor effort by a partner, students may be required to demonstrate the project separately and individually for re-evaluation. In such circumstances, the total marks for the project are likely to be redistributed on a basis which reflects the student effort.

6. Project mark allocation

The following mark breakdown gives a rough indication of how many marks will be allocated to aspects of the project.

Presentation	5%	Punctuality, professionalism and demonstration flow.
Test Cases	10%	Full coverage of functionality and system capabilities. Test cases compatible with reality. Suitability of data (input and test database).
System functionality	35%	How many of the specified functions are present and working. Are they simplistic or will they solve

		the real-world problem elegantly?
Interface design	15%	GUI features implemented, ease of use, thoughtful features, dropdowns, no unnecessary keystrokes
Integrity, quality and fault tolerance	20%	No bugs, databases updated correctly, all data correctly presented on screens, referential integrity preserved, system robustness, no system crashes
General	5%	General quality, professionalism, evidence of effort, attention to detail
Additional Features (Bonus)	5%	Must contribute to system (make good business sense)
Code Walkthrough	10%	One member of the group will be selected to walk the markers through the source code (on the computer) to demonstrate his/her understanding of the code and to explain how particular aspects of the system have been developed.

7. *Hints when developing the system:*

- Remember the golden rule. For each activity in the systems development process, ask yourself why are you doing it and what purpose does it serve. Do not follow the process blindly and only complete work that is important and appropriate. You will gain very little by "painting by numbers".
- Read this document carefully. It is important that you develop the system as requested. Ensure you understand the functionality contained in the specified use cases and the roles of the various classes.
- Complete the systems specification first. This should include all the sections detailed in this document. You should not contemplate building parts of the system until the detailed "blue print" has been completed. This part should be completed and submitted by **25th September 2020 at 4pm**. You will then receive feedback to assist you in your implementation.
- BE AWARE THE LABS WILL BE VERY CROWDED TOWARDS THE END OF THE PROJECT. Response times will be slower, help resources will be stretched, and everything you try to do will take twice as long. Get smart and put in the hours you need as soon as you have completed the necessary lectures and workshops.
- To obtain a pass mark for the system, the system needs to allow the user to do everything specified in a reasonably efficient and business-like way. Bugs, crashes or errors will mean that the user can't use that function, and you will lose marks.
- Ensure that you have a fair amount of proper data in the system! This indicates to the marker that you have, in fact, been able to use the system yourself, that you have tested it properly, and it also makes your report and enquiries much more interesting!
- Higher marks will be awarded for more elegant designs and solutions, non-hard coded aspects, lookups, validation, better business solutions, more effort on the interface etc.

Project Definition Statement

Your solution needs to provide the following functionality.

1. Create a customer
2. Create a customer order for at least 3 products (including checking if customer is black listed & checking of inventory)
    a. Reserve Inventory for the items
3. Cancel an item that is not invoiced as yet
4. Generate picking list to initiate delivery
5. Print report to identify all expired products in inventory

You will also need customers and products in the database but your application does not need to have functionality to create Product.

Systems Specification

In order to build the required functionality you must first complete the design phase for the required use cases. This will involve the following deliverables bound into a systems specification document:

1. ***Introduction***
   Short introduction providing context for your document. Brief description of the overall project, package diagram to show what part of the system is being developed and scope statement.
2. ***Dialog Design***
   Where appropriate, model the dialog between the user and the system. Dialog design can be modelled using a wireframe diagram and storyboard and the resulting implementation classes can be included in your sequence diagrams. Once you have identified the various forms and their purpose, define your overall screen standards (layout, colour and overall appearance) and design each screen layout in detail.
3. ***Design Sequence Diagrams***
   You are now in a position to model any complex use case interaction in detail using sequence diagrams (including possible boundary and controller classes and detailed messages)
4. ***Design Class Diagrams***
   This is an important set of diagrams building onto your class diagram developed in the analysis stage to describe design components within the classes. You must provide detailed documentation of all domain and controller classes, their attributes and methods required by the use cases described earlier. You may want to show one diagram depicting the classes and their associations and more detailed diagrams showing attributes and methods.
5. ***Database Design***
   As in most commercial environments today, your system will be implemented using a relational database. Derive your Entity Relationship Diagram from your entity classes, normalised to 3rd normal form. Provide detailed documentation on all attributes, keys (primary and foreign), data types and field sizes. Include this detail for only the tables required for this iteration of the project.
6. ***Reports Design***

   You should describe at least 1report that the system will generate (Generate report to identify all expired products in inventory). Ensure to add valuable relevant reports only, and to outline complete Requirements Definitions for each Report discussed.

7. ***Outputs and Controls***
   Output: All system output should be displayed on the screen. You should be able to display report to identify all expired products in inventory on the screen, but **no** other forms of output (printed report, email and fax) are required.

   Controls: Controls should be designed into every facet of the system (eg: validation checks and business rules).

8. ***Implementation Plan***
   Once the above steps in the design phase are over, you should have a "blue print" of the final system and can now plan the final construction and implementation stage of the project. The system specification therefore needs to include a detailed implementation plan scheduling the tasks your team must complete to move from the design stage to the final delivery of this phase of the system to the users.

9. ***Test Plan***
   Thorough and effective testing requires planning and preparation. Your document should include a plan as to how you will test the system and a detailed set of test cases covering the use cases developed.

Working C# System

Once you have completed your systems design and developed an implementation plan, you can commence construction of the system. The system must meet the following requirements:

- demonstrate the user requirements as specified
- developed as a single user system
- developed in C#
- system functionality should be supported by an underlying relational database environment.
- demonstrate all controls required to ensure integrity and security of the system

Follow the order specified in your implementation plan to develop the various components of the system. Your plan should include at least the following:

- Create database and populate the tables with sample data. Note that certain tables are required by this iteration of the project but the use cases to maintain these tables are not in scope. In this case you will create the tables and enter your sample data directly into Access rather than through the system. A document with specified sample data with which to populate the database tables for the final presentation will be provided. You must also enter additional realistic data to create a proper test database with which to test and demonstrate your system.
- Build the classes that are required by the system. These may include (depending on the design of your classes):
  - Entity Classes (such as Order and Customer)
  - Boundary Classes (screen forms)
  - Data access layer classes
  - Controller Classes (representing the various use cases)
  - Unit test the system as you construct and use your test cases to test each completed use case. Ensure that your test approach and data are rigorous.
  - Provide time to prepare the system and your group for the final demonstration to markers. This session is designed as a walk through of the system, using the test cases provided to show the functionality and integrity of the system.

Test Cases

You are required to hand in a set of test cases as part of your systems specification document to demonstrate that your project meets the specified functionality requirements. Bring a copy of your test cases with you as these test cases will be executed during the project evaluation.

A test case must describe how to test a specific path through a use case. The description must include:

- system state before executing the test (test environment)
- function to be tested (test items)
- test approaches
- Problem tracking (test cases including outcome of the test)
- Test schedule

The test cases must be documented using a test case template that will be provided.

End of Project Specification Document