Department of Information Systems Systems Design & Implementation (INF2011S)

Project Guidelines

Objectives of the project

The objectives of the project are to:

- Complete the systems development life cycle by designing, constructing, testing and implementing a segment of the Phumla Kamnandi Hotels
- Develop a detailed systems specification for a single iteration of the project
- Build, populate and maintain a relational database
- Understand how C#.net 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#.Net system
   - Test cases should be included in the documentation AND handed in at the demonstration

   For control purposes you are required to hand in your source code library on the due date and the markers will request you to use this file when demonstrating your project. Please submit your systems specification document AND your source code on Amathuba.

2. ***Functionality to appear in the system:***

   The required use cases must be developed in C#.Net by groups of 4 students (i.e., your existing theory workshop group) and should contain all the functionality detailed in this document.

3. *Due date*

Due date for hand-in of the system is **4 October 2024 at 12 noon.** In the interests of a rapid marking process as well as limiting the impact on exams, this deadline is not negotiable. The usual 10% per day penalty will be deducted from the marks of deliverables that are late, limited to one day (24 hours) after the due date. **Submissions submitted later than this will not be marked**. Note that you will only be able to hand in once!

4. *Getting help*

Tutor support will be implemented to assist you with coding problems. This will run from the 9 September to 3 October.

5. *The project marking process*

You will be required to prepare a demonstration of your project to your markers in a half-hour appointment (including question time). The objective of this demonstration is to allow you to present your system in the best possible light and ensure the markers see the results of your efforts. Make sure when you demonstrate the system, you have sufficient sample data in the databases to demonstrate that the system works properly. In your presentation, each member should 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 section of code works.

6. *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.

7. *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	40%	How many of the specified functions are present and working. Are they simplistic or will they solve the real-world problem elegantly?
Interface and system 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)	10%	Must contribute to system (make good business sense)
Code Walkthrough	5%	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.

8. 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.
- To obtain a pass mark for the system, the system needs to allow the user to do everything specified in a reasonably efficient and businesslike 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 markers 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

The first phase of the Phumla Kamnandi Hotels computerisation project will be completed in a number of iterations. Your **project requires you to complete a single iteration** which is to construct and implement the **Guest Booking** function. This will include the following use cases:

- **Included in Scope**

***Make a Telephone Booking**:*
Guests will phone the hotel to make a booking. The following extract from the "Make a Booking" use case details the required functionality:

Actor	System
1. Customer contacts hotel and asks to make a reservation	
2. Receptionist enters reservation details into the system	*3. Checks room availability and room price and displays details on screen*
4. If accommodation not available, ask for new dates and return to 2 above.	
5. Customer confirms reservation to be made	
6. Receptionist enters new or existing guest details	*7. Verifies existing guest or* **adds new guest**
8. Receptionist indicates reservation should be created	*9. System creates reservation and guest account. System calculates deposit amount and generates reservation reference number.*
10. Receptionist enters guest credit card details	*11. System verifies payment with credit card company. System records payment and adjusts guest account to record deposit.*
12. Receptionist confirms that reservation is complete and advises guest of reservation reference number.	*13. System generates confirmation letter to be emailed, faxed or posted to guest.*

Change a Guest Booking
The system should allow the receptionist to change an existing booking and adjust hotel occupancy details accordingly.

Cancel a Guest Booking
The system should allow the receptionist to cancel a current booking and adjust hotel occupancy details.

Make a Guest Booking Enquiry
The system should allow the receptionist to enquire on a guest booking to review its details and determine its status of unconfirmed or confirmed (deposit paid).

Reports
Report 1:
The system should generate a dynamic electronic report specifying occupancy levels. The receptionist should be allowed to chose the dates for which the occupancy level report will be generated.
Report 2:
Any other relevant report which could bring business value to Phumla Kamnandi Hotel group.

- **Out of Scope**

*The system is **not** required to authorise deposits paid at the time of booking (assume all payments are authorised)*

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 (Please see the systems specification template for more details):

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 an Interface Flow Diagram 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 two complex use cases interaction in detail using sequence diagrams (including possible boundary and controller classes and detailed messages). One of them should include the “Make a Booking” use case.
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 in a data dictionary table. Include this detail for only the tables required for this iteration of the project.
6. ***Report Design***
   This Section will be where you describe the reports that the system will generate. Ensure to add valuable relevant reports only, and to outline complete Requirements Definitions for each Report discussed. You should have 2 reports outlined in this section one of which should be a dynamic electronic report specifying occupancy levels. The receptionist should be allowed to chose the dates for which the occupancy level report will be generated.
7. ***Outputs and Controls***
   Output: All system output should be displayed on the screen. You should be able to display the confirmation letter on the screen if requested, 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#.Net 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 (see use cases in scope)
- developed as a single user system
- developed in C#.Net
- 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 your SQL 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 database 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 Booking and Guest)
  - 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 (at the final presentation and in your documentation) to demonstrate that your project meets the specified functionality requirements. 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:

- function to be tested
- system state before executing the test
- test data
- expected outcome of the test

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

End of Project Specification Document