*Department of Information Systems*

***Systems Design & Development***

**

**Systems Specification for** *[Project Name]*

***Team Members***

|  |  |
| --- | --- |
| ***Student Number:*** | ***Student Name:*** |
| ***Student Number:*** | ***Student Name :*** |
| ***Student Number:*** | ***Student Name:*** |
| ***Student Number:*** | ***Student Name :*** |

**Plagiarism Declaration**

1. We know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. This Systems Specification is our own work.
3. We have not allowed, and will not allow, anyone to copy our work with the intention of passing it off as their own work.

Full Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signed:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_/\_\_\_\_/

Full Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Signed:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_\_/\_\_\_\_/

**Table Of Contents**

[1. Introduction 1](#_Toc428175304)

[1.1. Overview of Specification 1](#_Toc428175305)

[1.2. Context & Scope of System Specification 1](#_Toc428175306)

[1.3. Design Assumptions & Constraints 1](#_Toc428175307)

[2. User Interface & Dialogue Design 1](#_Toc428175308)

[2.1. Interface Flow Diagrams 2](#_Toc428175309)

[2.2. Screen Standards 2](#_Toc428175310)

[2.3. Detailed Screen Layout 2](#_Toc428175311)

[3. Design Sequence Diagrams 2](#_Toc428175312)

[3.1. Design Sequence Diagram 1 2](#_Toc428175313)

[3.2. Design Sequence Diagram 2 3](#_Toc428175314)

[4. Design Class Diagrams 3](#_Toc428175315)

[5. Entity Relationship Diagram 3](#_Toc428175316)

[6. Report Design 4](#_Toc428175317)

[6.1. Report 1 5](#_Toc428175318)

[6.1.1. Detailed Output Requirements 5](#_Toc428175319)

[6.1.2. Report Layout 5](#_Toc428175320)

[6.2. Report 1 5](#_Toc428175321)

[6.2.1. Detailed Output Requirements 5](#_Toc428175322)

[6.2.2. Report Layout 5](#_Toc428175323)

[7. Input-Output Standards & Controls 6](#_Toc428175324)

[7.1. Formalised Outputs: 6](#_Toc428175325)

[7.2. Built-In Validation to Ensure Requirements are Met 6](#_Toc428175326)

[7.3. Input Integrity Controls 6](#_Toc428175327)

[7.4. Output Integrity Controls 6](#_Toc428175328)

[8. Implementation Plan 6](#_Toc428175329)

[9. Test Plan 6](#_Toc428175330)

[9.1. Test Environment 7](#_Toc428175331)

[9.2. Test Items 7](#_Toc428175332)

[9.3. Test Approaches 7](#_Toc428175333)

[9.4. Problem Tracking (Test Cases) 8](#_Toc428175334)

[9.5. Test Schedule 8](#_Toc428175335)

***Important Notice***

**The comments and suggestions provided in this template must NOT be present in your final submission! Try to customise, apply and re-work this template as much as possible to ensure that your Project’s Value Proposition is maximized** (i.e. Try to avoid just completing this document, heading by heading, and then just updating the contents page). **If nothing else, please just ensure to remove the initial content guidelines provided in this template in each section** (that includes this paragraph!). **The template is only here to assist you in completing your Systems Specification Document, you must try and find ways of showing us that you have actually thought about what each section means, how it creates value, and how it applies to your specific understanding of the problem, and your solution.**

***General Hints/Tips***

1. **The contents page makes use of Microsoft Word’s built in “Table of Contents” feature (found on the far left under the References tab). Try and make use of these helpful built in features** (e.g. once you have completed your document, get Word to automatically ensure that the contents page is accurate and has the correct page numbers by first right-clicking on the table of contents and selecting “Update field” which then updates any altered headings and inserts the correct page numbers)
2. **You should be able to generate an Initial/Draft ERD directly from your SQLExpress database. Doing this is a great way to reduce the amount of work you have to do, while also ensuring completeness. Take the generated model and modify it / comment on it / restructure it for the purposes of your project.**

# Introduction

## Overview of Specification

*Write a short overview for this document (providing a brief overview of the documents structure. Also try to remind your readers of where the project came from (i.e. relevant previous documents and/or work), where the project is currently (i.e. the current completeness of work/documentation, and any resultant changes), and finally what is on the horizon (i.e. what comes next, what work is left to be done, what might require revision, etc)*

## Context & Scope of System Specification

*Write a brief description of the overall project, providing some context and background to the problem, solution, etc. This should quickly remind readers of the project’s relevance and significance, while also reminding them of the objectives and requirements that have been outlined previously, and which are crucial for design.*

*Provide a package diagram to show what particular part of the system is being worked-on/re-designed/developed, ensuring to explain/justify (either explicitly or implicitly) why and how that particular scope/approach was chosen (e.g. dependencies and addressing project critical systems first). Ensure also to provide a valuable scope statement, that will be useful to reaffirm the project’s focus and avoid scope creep.*

## Design Assumptions & Constraints

*This section describes any constraints in the system design (reference any trade-off analyses conducted such, as resource use versus productivity, or conflicts with other systems) and includes any assumptions made by the project team in developing the system design.*

# User Interface & Dialogue Design

*It is a good habit to always write a very short introduction to each new section (i.e. never have heading directly after each other without some explanation. Provide the layout of all input data screens or graphical user interfaces (GUIs) (for example, windows). Provide a graphic representation of each interface. Define all data elements associated with each screen or GUI, or reference the data dictionary. This section should contain edit criteria for the data elements, including specific values, range of values, mandatory/optional, alphanumeric values, and length. Also address data entry controls to prevent edit bypassing. Discuss the miscellaneous messages associated with operator inputs, including the following:*

* *Copies of form(s) if the input data are keyed or scanned for data entry from printed forms*
* *Description of any access restrictions or security considerations*
* *Each transaction name, code, and definition, if the system is a transaction-based processing system*

## Interface Flow Diagrams

* *Dialog design can be modelled using a wireframe diagram or Interface Flow Diagrams and the resulting implementation classes can be included in your sequence diagrams.*
* *Ensure to map out and explain the core architecture of the page/form/etc and why you have decided to structure in that particular way.*

## Screen Standards

* *Once you have identified the various forms and their purpose, define your overall screen standards (layout, colour and overall appearance).*

## Detailed Screen Layout

* *Design each screen layout in detail.*
* *This can be done by doing “realistic mockups” or by simply adding realistic detail to your wireframe. Use* <https://ninjamock.com/> *to design your screens*.

**

# Design Sequence Diagrams

*You are now in a position to model any complex use case interaction in detail using Design Sequence Diagrams, remember the difference between previous sequence diagrams (as utilized during the analysis phase) and the Design Sequence Diagram (which includes, amongst others, possible boundary and controller classes, patterns and detailed messages). Please provide* ***2 design sequence diagrams for two different use cases, one of which should be the “Make a Booking” use case.***

## Design Sequence Diagram 1

* *Ensure to provide some explanation, context and clarification of the model (either in text, or using comments on the model)*

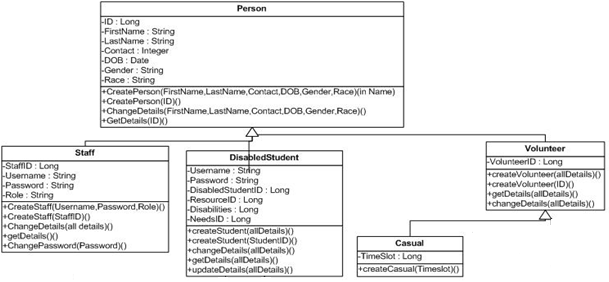
## Design Sequence Diagram 2

* *Ensure to provide some explanation, context and clarification of the model (either in text, or using comments on the model)*

# **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 classes, their attributes and methods required by the use cases described earlier. Use the standard UML notation for interface classes to indicate where forms are used. (Examples of an expanded class diagram can be seen below)*

*Ensure to provide some explanation, context and clarification of the model (either in text, or using comments on the model)*

**

# **Entity Relationship Diagram**

*As in most commercial environments today, your system will be implemented using a relational database. Model and Describe your Entity Relationship Diagram (or Database Diagram-DD) using Crows-foot Notation, ensuring all Entities and Associations have correct Cardinality and Multiplicity. Also ensure that your ERD goes beyond simple domain entities, and is fully Normalised to 3rd Normal Form (3NF). 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.*

**

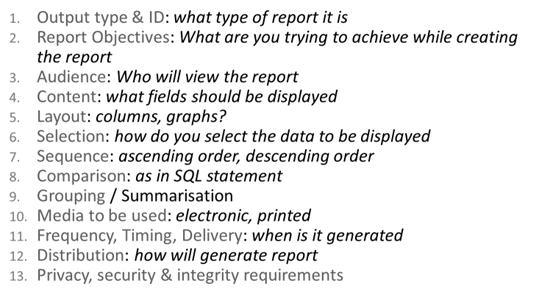
# **Report Design**

*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.*

## Report 1

*Briefly describe the report, why it is valuable, the scenario in which it will be generated, who will generate it, why, etc..*

### Detailed Output Requirements

******

### Report Layout



## Report 1

### Detailed Output Requirements

### Report Layout

# **Input-Output Standards & Controls**

*This section provides the detailed design of the system and subsystem inputs and outputs relative to the user/operator. Any additional information may be added to this section and may be organized according to whatever structure best presents the operator input and output designs. Depending on the particular nature of the project, it may be appropriate to repeat these sections at both the subsystem and design module levels. Additional information may be added to the subsections if the suggested lists are inadequate to describe the project inputs and outputs*

*Developers of sensitive State systems are required to develop specifications for the following minimum levels of control:*

* *security to restrict access of critical data items to only those access types required by users*
* *Application audit trails to dynamically audit retrieval access to designated critical data*
* *Standard Tables to be used or requested for validating data fields*
* *Verification processes for additions, deletions, or updates of critical data*

## Formalised Outputs:

*All key system outputs 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. What were the formalized forms of output included in your system?*

## Built-In Validation to Ensure Requirements are Met

*What were the validation controls built into the system to optimize the performance of the application and to minimize the possibility of user error which may hinder system performance?*

## Input Integrity Controls

*Describe any input integrity control included in the system*

## Output Integrity Controls

*Describe any output integrity control included in the system*

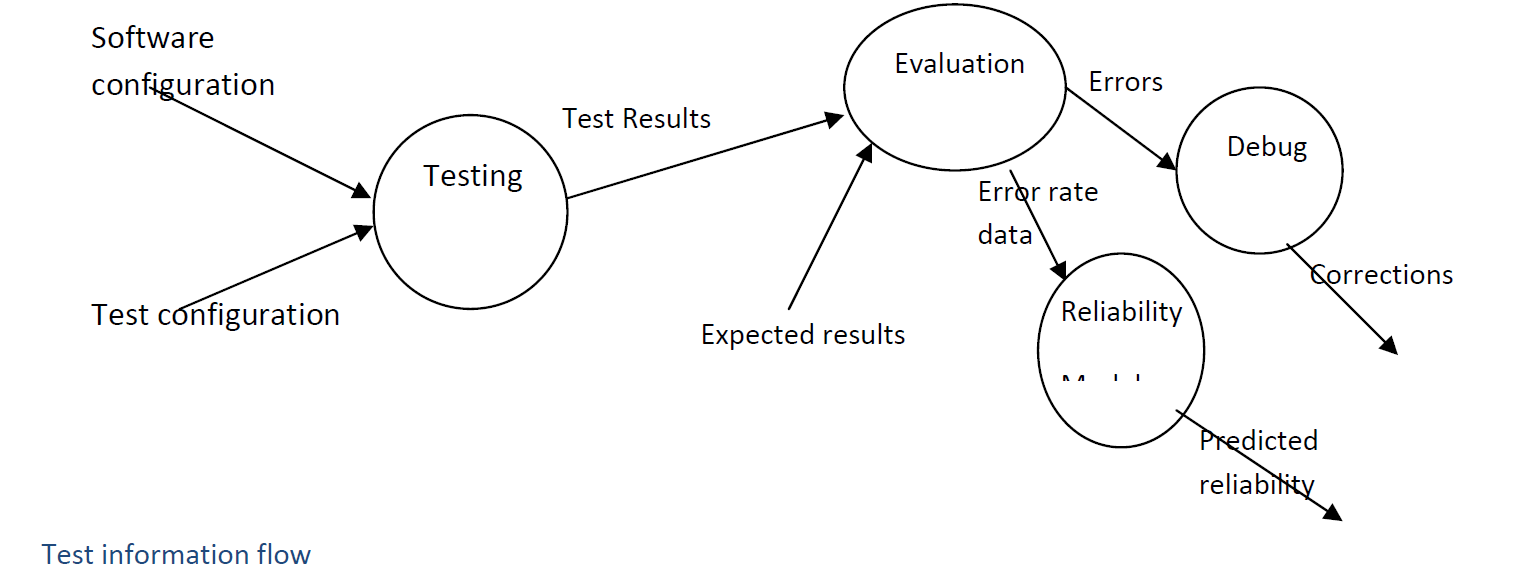
# **Implementation Plan**

*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.*

# **Test Plan**

*During testing the software engineering produces a series of test cases that are used to “rip apart” the software they have produced. Testing is the one step in the software process that can be seen by the developer as destructive instead of constructive. Software engineers are typically constructive people and testing requires them to overcome preconceived concepts of correctness and deal with conflicts when errors are identified. A number of rules that act as testing objectives are:*

* *Testing is a process of executing a program with the aim of finding errors.*
* *A good test case will have a good chance of finding an undiscovered error.*
* *A successful test case uncovers a new error.*



## Test Environment

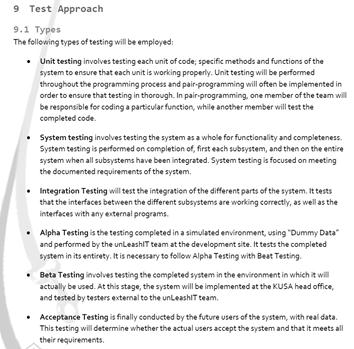
*Describe the minimum hardware & software requirements*

## Test Items

*Provide a description of all the features to be tested*

## Test Approaches

*Describe the types of tests which will be performed*

**

## Problem Tracking (Test Cases)

*Describe the process followed for tracking and resolving errors*

## Test Schedule

*Describe how the test schedule was co-ordinated with the development schedule* 