**Software Design/Functional Specification (SDS/FS) Template**

**(Delete this page after reading)**

***In this template you will find text bounded by the “< >” symbols. This text appears in italics and is intended to guide you through the template and provide explanations regarding the different sections in this document. There are two types of comments in this document. The comments that are in black are intended specifically for that course. The comments that are in blue are more general. Please, make sure to delete all of the comments before submitting the document.***

*The explanations provided below, do not cover all of the material, but merely, the general nature of the information you would usually find in SDS documents. It is based on the IEEE guidelines and was adapted specifically for the needs of Software Engineering courses. Most of the sections in this template are required sections, i.e. you must include them in your version of the document. Failure to do so will result in marks deductions. Optional sections will be explicitly marked as optional. >*

Anything highlighted with this color is optional.

Please write ‘to the point text’ in this proposal. No lengthy stories!

**DEADLINE: Check Slides!**

**You are required to submit the following two files:**

**Group<Number>\_SDS\_Document.doc**

**Group<Number>\_SDS\_Document.pdf**

**Software Design Specification Document**

**(CS360)**

**<Project Name>**



**Group Number: <*your group number here*>**

|  |
| --- |
| **<name>** |
| **<name>** |
| **<name>** |
| **<name>** |
| **<name>** |

|  |  |
| --- | --- |
| **Course:** Software Engineering CS360  **Instructor:** Suleman Shahid  **University:** Lahore University of Management Sciences (LUMS |  |

**Version: 1.0**

**Date: (dd/03/2020)**

**Number of hours spent on this document:** <Total hours>

|  |  |
| --- | --- |
|  |  |

**Contents**

**Contents iii**

**1** **Change Log v**

1.1 Project Scope v

1.2 Change log v

**2** **Introduction 1**

2.1 Document Purpose 1

2.2 Product Scope 1

2.3 Intended Audience and Document Overview 1

2.4 Definitions, Acronyms and Abbreviations 1

2.5 References and Acknowledgments 1

**3** **Overall Description 2**

3.1 System overview 2

3.2 System constraints 2

3.3 Architectural strategies 2

**4** **System Architecture 3**

4.1 System Architecture 3

4.2 Subsystem Architecture 4

4.3 Data Structure 5

4.4 Database Model 5

4.5 External Interface Requirements 6

**5** **User Interface Design 8**

5.1 Description of the user interface 8

5.2 Information architecture 8

5.3 Screens 8

5.4 User interface design rules 8

**6** **Other Non-functional Requirements 9**

<Refine this section based on the final system design. Highlight the changes or additions> 9

6.1 Performance Requirements 9

6.2 Safety and Security Requirements 9

6.3 Software Quality Attributes 9

**Appendix A - Group Log 10**

**Appendix B – Contribution Statement 11**

# Change Log

*<TO DO: Please use this section to update us on the project scope, if it is different from the one you proposed in the SRS document. Highlight major changes or deviations>*

## Project Scope

*<TO DO: Write 1-2 paragraphs highlighting the project scope.>*

## Change log

# Introduction

*<TO DO: Please provide a brief introduction to your project and a brief overview of what the reader will find in this section.>*

## Document Purpose

*<Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by this SDS, particularly if this SDS describes only part of the system or a single subsystem.*

*TO DO: Write 1-2 paragraphs describing the purpose of this document as explained above.>*

## Product Scope

*<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals.*

*TO DO: You can copy text from your SRS document.*

## Intended Audience and Document Overview

*<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers (In your case it would probably be the “client”, teaching assistants and the instructor). Describe what the rest of this SDS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>*

*TO DO: For the first part, you can copy text from your SRS document.*

## Definitions, Acronyms and Abbreviations

*<Define all the terms necessary to properly interpret the SDS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SDS.*

*TO DO: Please provide a list of all abbreviations and acronyms used in this document sorted in alphabetical order.>*

## References and Acknowledgments

*<List any other documents or Web addresses to which this SDS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. .>*

# Overall Description

## System overview

*< Provide a general description of the software system including its functionality and matters related to the overall system and its design (perhaps including a discussion of the basic design approach or organization). Feel free to split this discussion up into subsections (and subsubsections, etc ...).*

*TO DO: Provide at least one paragraph describing system overview. Provide a general diagram that will illustrate how your product interacts with the environment and in what context it is being used. - Final Context Diagram>*

## System constraints

*<**Describe any global limitations or constraints that have a significant impact on the design of the system's software (and describe the associated impact). Such constraints may be imposed by any of the following (the list is not exhaustive):*

* *Hardware or software environment*
* *End-user environment*
* *Availability or volatility of resources*
* *Standards compliance*
* *Interoperability requirements*
* *Interface/protocol requirements*
* *Data repository and distribution requirements*
* *Security requirements (or other such regulations)*
* *Memory and other capacity limitations*
* *Performance requirements*
* *Network communications*
* *Verification and validation requirements (testing)*
* *Other means of addressing quality goals*
* *Other requirements described in the requirements specification*

*>*

## Architectural strategies

*<**Describe any design decisions and/or strategies that affect the overall organization of the system and its higher-level structures. These strategies should provide insight into the key abstractions and mechanisms used in the system architecture. Describe the reasoning employed for each decision and/or strategy (possibly referring to previously stated design goals and principles) and how any design goals or priorities were balanced or traded-off. Such decisions might concern (but are not limited to) things like the following:*

* *Use of a particular type of product (programming language, database, library, etc. ...)*
* *Reuse of existing software components to implement various parts/features of the system*
* *Future plans for extending or enhancing the software*
* *User interface paradigms (or system input and output models)*
* *Hardware and/or software interface paradigms*
* *Memory management policies*
* *External databases and/or data storage management and persistence*
* *Distributed data or control over a network*
* *Concurrency and synchronization*
* *Communication mechanisms*

*>*

# System Architecture

## System Architecture

*< This section should provide a high-level overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components. Don't go into too much detail about the individual components themselves (there is a subsequent section for detailed component descriptions). The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.*

*At the top-most level, describe the major responsibilities that the software must undertake and the various roles that the system (or portions of the system) must play. Describe how the system was broken down into its components/subsystems (identifying each top-level component/subsystem and the roles/responsibilities assigned to it). Describe how the higher-level components collaborate with each other in order to achieve the required results. Don't forget to provide some sort of rationale for choosing this particular decomposition of the system (perhaps discussing other proposed decompositions and why they were rejected). Feel free to make use of design patterns, either in describing parts of the architecture (in pattern format), or for referring to elements of the architecture that employ them.*

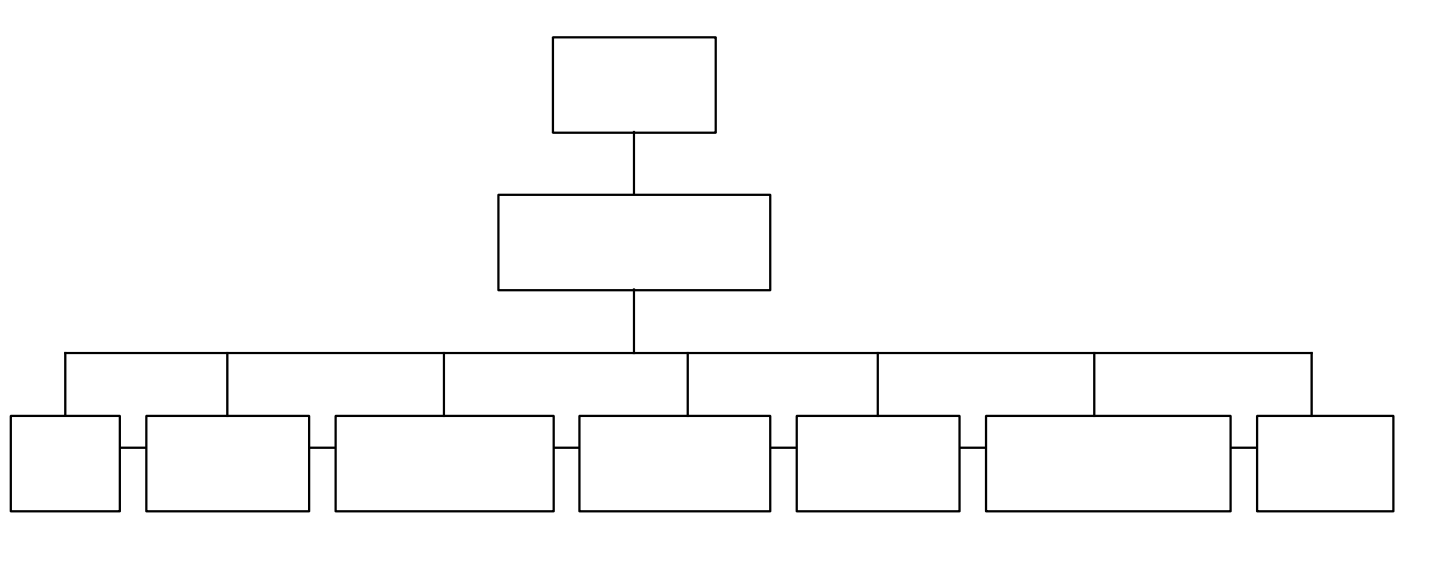
*If there are any diagrams, models, flowcharts, documented scenarios or use-cases of the system behavior and/or structure, they may be included here. Diagrams that describe a particular component or subsystem should be included within the particular subsection that describes that component or subsystem.*

***Note:***

*This section (and its subsections) really applies only to newly developed (or yet-to-be developed) portions of the system. If there are parts of the system that already existed before this development effort began, then you only need to describe the pre-existing parts that the new parts of the system depend upon, and only in enough detail sufficient to describe the relationships and interactions between the old parts and the new parts. Pre-existing parts that are modified or enhanced need to be described only to the extent that is necessary for the reader to gain a sufficient understanding of the nature of the changes that were made.>*

*TO DO: Provide the relevant information as discussed above.*

* + *Develop a component diagram to show relationship between different components (A pictorial representation, using a UML component diagram, of the architecture is presented.)*
  + *Develop activity diagrams to show the internal flow of components. This should only cover the top 3 use cases you mentioned in your SRS document.*

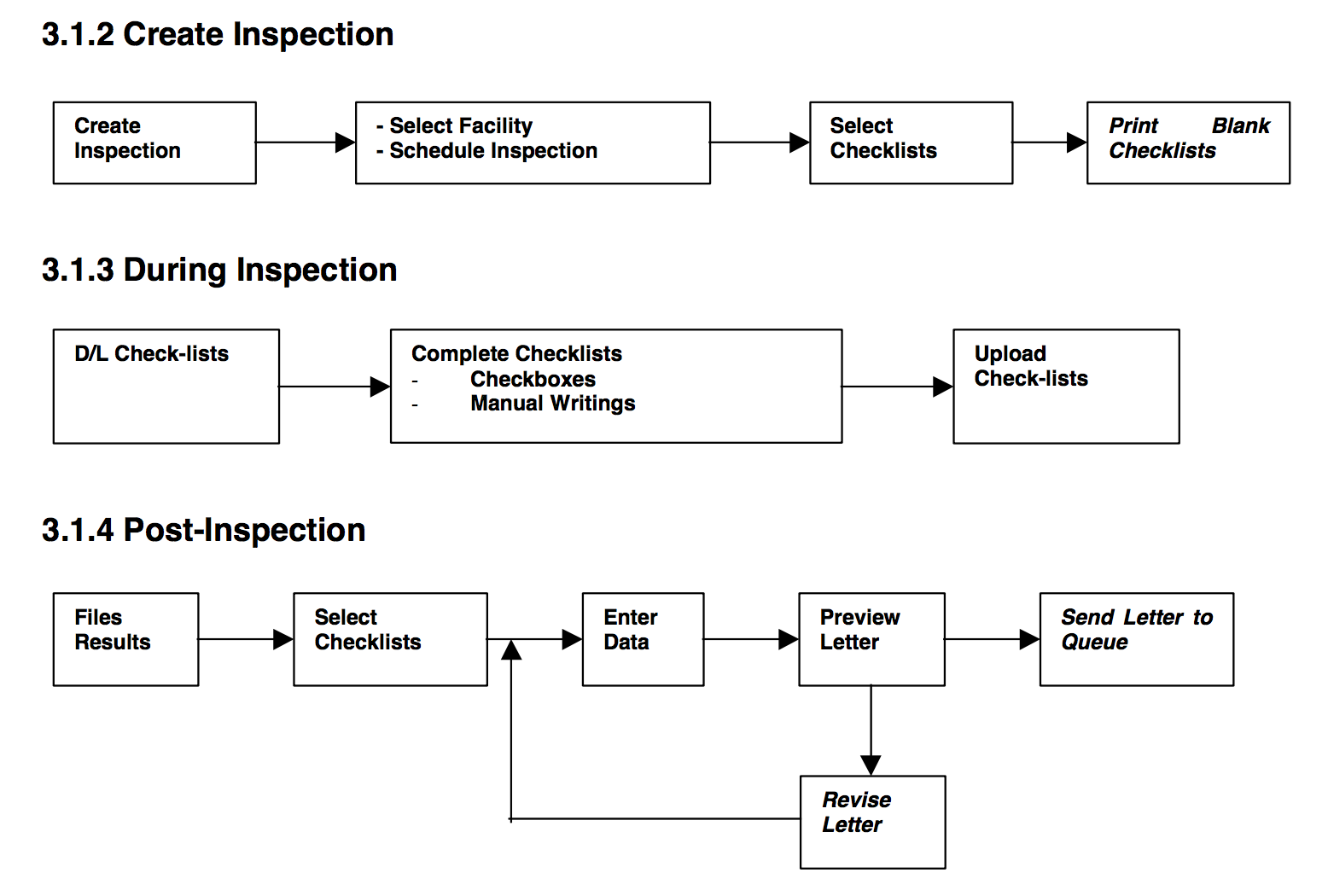
**

## Subsystem Architecture

*< If a particular component is one which merits a more detailed discussion than what was presented in the System Architecture section, provide that more detailed discussion in a subsection of the System Architecture section. If necessary, describe how the component was further divided into subcomponents, and the relationships and interactions between the subcomponents (similar to what was done for top-level components in the System Architecture section).>*

TO DO: Provide the relevant information as discussed above.

* A sequence diagram, for each use case the component realizes, is presented. (Only do this for the top 3 use cases from your SRS document).
* Class diagram to show the relationship between classes/components

**

*Example: Flow of each component*

## Data Structure

<A description of all data structures including internal, global, and temporary data structures.

**4.3.1 Internal software data structure**

Data structures that are passed among components of the software are described.

**4.3.2 Global data structure**

Data structured that are available to major portions of the architecture are described.

**4.3.3 Temporary data structure**

Files created for interim use are described.

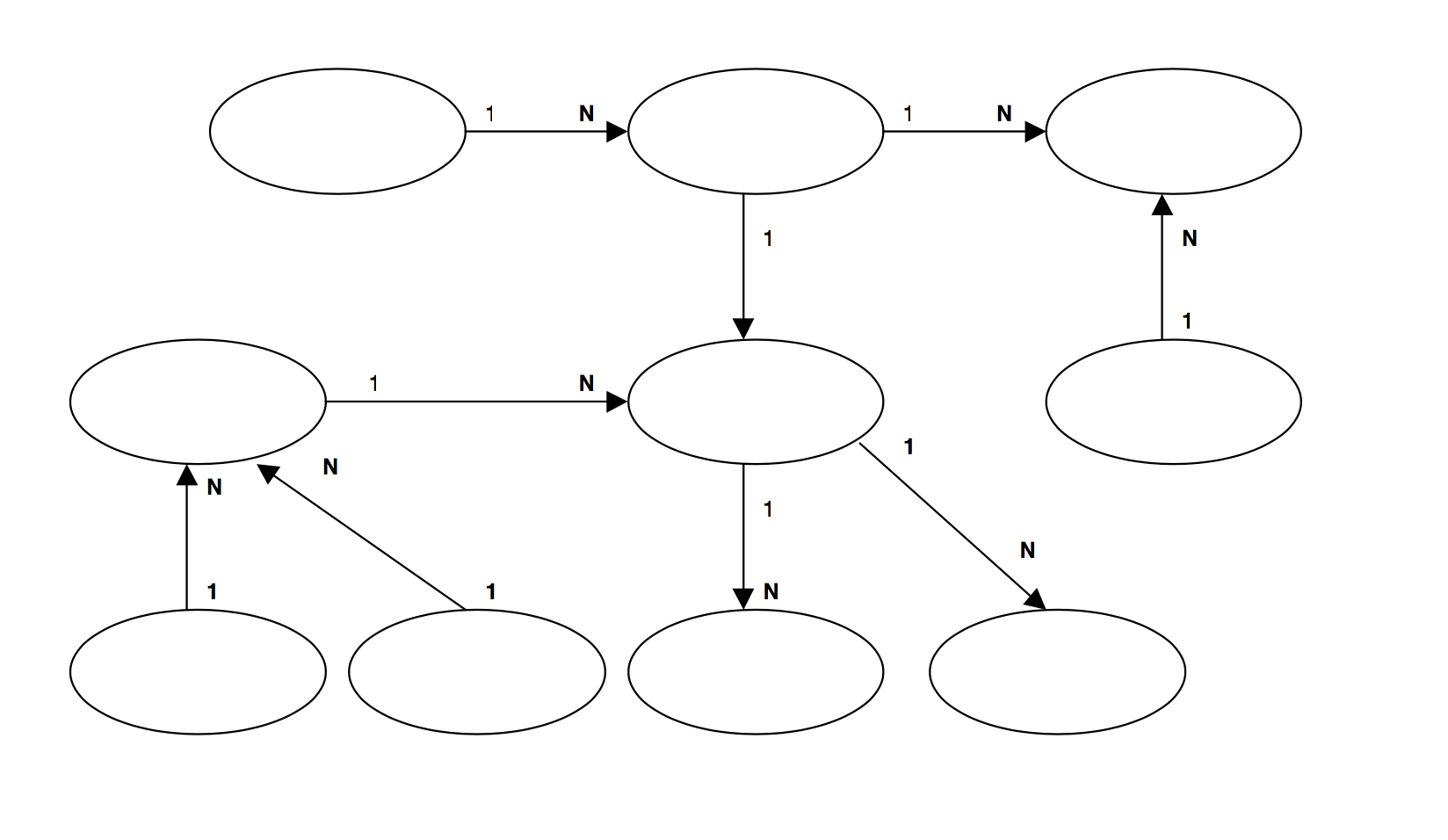
>

## Database Model

<Database(s) created as part of the application is(are) described.>

### Database scheme and detailed description

*<Visual presentation of DB Schema and description of all tables and relationships >*



### Database

*<Choice of the DB – give reasons >*

## External Interface Requirements

### User Interfaces

*<* *Specify:*

1. *The logical characteristics of each interface between the software product and its users.*
2. *All the aspects of optimizing the interface with the person who must use the system*

*This is a description of how the system will interact with its users. Is there a GUI, a command line or some other type of interface? Are there special interface requirements? If you are designing for the general student population for instance, what is the impact of ADA (American with Disabilities Act) on your interface?>*

### Hardware Interfaces

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware. You are not required to specify what protocols you will be using to communicate with the hardware, but it will be usually included in this part as well.*

*TO DO: Please provide a short description of the different hardware interfaces. If you will be using some special libraries to communicate with your software mention them here. In case you have more than one hardware interface divide this section into subsections.>*

# User Interface Design

<A description of the user interface design of the software is presented.>  
**note : *The User interface need to be complete in all functionality***

## Description of the user interface

<A detailed description of user interface.

* Front end tools and technologies. Give rationale of your choice.
* Describe the presentation layer of the MVC architecture, if used.>

## Information architecture

<To show flow between different screen. Google it for more details. I will also discuss this in class.>

TO DO: Design the complete Information architecture and present it visually

* Discuss the relationship between screens

## Screens

To do: Screen images with description – explain each item on the screen e.g. button, text field, etc.

Explain how the screen is mapped on one of the user requirements/use cases

## User interface design rules

<Conventions and standards used for designing/implementing the user interface are stated. >

# Other Non-functional Requirements

## <Refine this section based on the final system design. Highlight the changes or additions>

## Performance Requirements

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.*

*TODO: Provide at least 5 different performance requirements based on the information you collected from the client. For example you can say “1. Any transaction will not take more than 10 seconds, etc…>*

## Safety and Security Requirements

*<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied. Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements.*

*TODO:*

* *Provide at least 3 different safety requirements based on your interview with the client or, on your related research, and again you need to be creative here.*
* *Describe briefly what level of security is expected from this product by your client and provide a bulleted (or numbered) list of the major security requirements.>*

## Software Quality Attributes

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.*

*TODO: Use subsections (e.g., 4.3.1 Reliability, 4.3.2 Portability, etc…) provide requirements related to the different software quality attributes. Base the information you include in these subsections on the material you have learned in the class. Make sure, that you do not just write “This software shall be maintainable…” Indicate how you plan to achieve it, & etc…Do not forget to include such attributes as the design for change. Please note that you need to include at least 2 quality attributes, but it is the mere minimum and it will not receive the full marks.>*

**Appendix A - Group Log**

*<Please include here all the minutes from your group meetings, your group activities, and any other relevant information that will assist* ***ME and the Teaching Assistants*** *to determine the effort put forth to produce this document>*

**Appendix B – Contribution Statement**

|  |  |  |  |
| --- | --- | --- | --- |
| *Name* | *Contributions in this phase* | *Approx. Number of hours* | *Remarks* |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |