Font style: Time New Roman,**Font size:** 18 Caps, Bold & Center aligned

PROJECT NAME

Student Name 1

Student Name 2

Student Name 3

Student Name 4

Student Name 5

Font style: Time New Roman**, Font size**: 16 Bold&Center aligned

Do not use Ms/Mr/Sir , can use Dr./Prof./Engr.

Font style: Time New Roman, **Font size**: 14 Italic, &Center aligned

Supervised By

Supervisor Name

Submitted for the partial fulfillment of Bachelor of Computer Science degree.

JUN, 2024

Font style: Time New Roman, **Font size**: 14 Bold, Center aligned, **Date**must be at last line of the page. It should be month of submission of your report.

ABSTRACT

Font style: Time New Roman, **Font size:** 18 Bold, Caps &, Center aligned

The abstract should consist of 2-3 paragraphs. First brief paragraph should give an overview of the existing system/topic. 2nd detailed paragraph should deal with project methodology explaining what has been done and how it has been done. In the last brief paragraph (only in Report 2) regarding testing and results, tools used for development, validation and achievements should be discussed.

The body is written in 12pt Time New Roman font, single line spacing

Font style: Time New Roman/Arial,**Font size**: 18 Bold, Caps &, Center aligned

DECLARATION

We hereby declare that our dissertation is entirely our work and genuine / original. We understand that in case of discovery of any PLAGIARISM at any stage, our group will be assigned an F (FAIL) grade and it may result in withdrawal of our Bachelor's degree.

Group members:

Name	**Signature**
Student name 1	_____
Student name 2	_____
Student name 3	_____
Student name 4	_____
Student name 5	_____

Font style: Time New Roman,**Font size:** 18 Bold, Caps and , Center aligned.

ACKNOWLEDGMENT

(Optional)

Students may acknowledge the persons who supported them in the project work but should be very brief and precise.

Font style: Time New Romanl,**Font size:** 18 Bold, Caps&, Center aligned.

TABLE OF CONTENTS

Font style: Time New Roman,**Font size**: 18 Bold, Caps &, Center aligned.

LIST OF FIGURES

Captions should be exactly same as in text

Screen shots and photographs should be avoided

Font style: Time New Roman,**Font size**: 18 Bold, Caps &, Center aligned.

LIST OF TABLES

Captions should be exactly same as in text

Font style: Time New Roman,**Font size:** 18 Bold, Caps and , Center aligned.

CHAPTER 1

INTRODUCTION

Font style: Time New Roman,**Font size**: 12 Justified aligned. Single line spapced

Font style: Time New Roman,**Font size**: 18 Bold, Caps and , Center aligned.

In this chapter an introduction about the project should be provided. Problem statement, Objectives, details of resources required for the completion, development methodology and organization of the project report (chapter wise) should be given.

1.0 Introduction

Font style: Time New Roman,**Font size**: 16 Bold, Justified aligned.

1.1 Problem domain

1.2 Problem statement

1.3 Proposed system

1.3.1 Aims and Objectives

1.3.2 Proposed system features

Font style: Time New Roman,**Font size**: 14 Bold, Justified aligned.

1.4 Project Methodology

1.5 Resource Requirement

1.6 Report Layout

CHAPTER 2

BACKGROUND/EXISTING WORK

(Literature Review, Existing Work)

In chapter 2 literature review of the area and the existing work that has been done in the area **duly supported by references** must be discussed. Too basic things should be avoided. Material to be read/ understood by the students be included.

2.0 Introduction

2.1 Overview of existing projects

2.2 Limitations of existing projects

2.3 Innovations of our project

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 Introduction

3.1.1 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 SRS, particularly if this SRS describes only part of the system or a single subsystem.>

3.1.2 Document Conventions

<Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.>

3.1.3 Intended Audience and Reading Suggestions

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS 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.>

3.1.4 Product Scope

<Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.>

3.1.5 References

<List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could

access a copy of each reference, including title, author, version number, date, and source or location.>

3.2 Overall Description

3.2.1 Product Perspective

<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>

3.2.2 Product Functions

Put your uses cases here with their descriptions.
<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>

3.2.3 User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

3.2.4 Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

3.2.5 Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security

considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

3.2.6 User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

3.2.7 Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

3.2.8 BPMN diagram

Provide the system BPMN if applicable and describe it.

3.3 External Interface Requirements

3.3.1 User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

3.3.2 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, and communication protocols to be used.>

3.3.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial

components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

3.3.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

3.4 System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

3.4.1 System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

a Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

b Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

c Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and

necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:
REQ-2:

3.4.2 System Feature 2 (and so on)

3.5 Other Non-functional Requirements

3.6Performance 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.>

3.7Safety 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.>

3.8Security Requirements

<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. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

3.9Software 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.>

3.10 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

3.11 Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

CHAPTER 4

SOFTWARE DESIGN

1 Design Overview

1.1 Introduction

This section should highlight the design approach that you adopt (e.g. object-oriented design or structured design), the proposed architecture of the system (e.g. client-server) and the relevant techniques and tools used (e.g., Microsoft Visio, Dia, ArgoUML, etc.).

1.2 Environment Overview

This section should describe the environment in which the system will be run. This should include diagrams and descriptions of the environment. For example, if we are describing a web app, this will include a network diagram or topology. If we are developing a library, we should describe what might be calling us. In order to do this effectively, you should have previously stated your system's scope.

Be sure to address where this application will reside and how it will be executed.

1.3 System Architecture

Provide a high-level description of the system architecture. Use a few block diagrams to show the major components and their interaction. Remember to describe the conventions and notations used in your diagrams.

1.4 Constraints and Assumptions

Mention the major design constraints here. These may have been imposed by the customer, which can be found in the requirements document. Explain how your design accommodates these constraints.

There may also be constraints imposed because of your system interacting with other external systems or being dependent on some external systems to provide part of the functionality. In such cases clearly mention the type of software your system interacts with (e.g. XYZ database software, ABC email software) and the constraints this imposes (e.g. only text based email messages are allowed).

2 Interfaces and Data Stores

This section describes the interfaces into and out of the system as well as the data stores you will be including in your system.

2.1 System Interfaces

The various interfaces provided to users and/or other external systems should be defined here. If you had included user interface descriptions in your requirements document you may refer to them here. If you provide interfaces to other systems, say export and import data to a different software, you should mention them here.

Do not include JavaDocs here. You should describe the interfaces using prose and graphics if your interface includes a GUI.

2.2 Data Stores

Describe the data stores you will create AS PART OF YOUR SYSTEM. This should NOT include databases or other data stores that are external to your system. This information should be included in a previous section.

3 Structural Design

3.1.1 Class Diagram

Provide the class diagram for the system. If the class diagram is too big, partition the diagram using some reasonable criteria. For example, you may provide the client-side and the server-side object models as separate diagrams.

What should go into a class diagram?

All the classes should find a place here. All associations between classes should be identified and associations should be decorated with the right cardinality. Aggregation and inheritance relationships should be identified. ***A brief explanation should accompany each diagram including your justification for your design.*** *Make sure you include figure numbers to help you reference figures in the document.*

Remember: You may have to iterate several times before you agree on the right class diagram for your system.

3.1.2 Class Descriptions

3.2.1 Sequence diagram

3.2.2 Sequences diagrams descriptions

4 ER diagram and relation schema

CHAPTER 5

(Required only for project 2)

SYSTEM IMPLEMENTATION & VALIDATION

Validation and testing for determining the specifications and achievements of the project should be discussed here. All the results, graphs/charts, performance comparisons, testing methodology and setup should be given.

CHAPTER 6

CONCLUSION AND FUTUURE WORK

It should be the last chapter of the project report. Overview of the project with features and limitations will be discussed. Enhancements that were beyond the scope of the project but could be done in future will be suggested. The future work recommended should not be generalized statements but should be the outcome/ conclusions of one year hard work. Must be specific and to the point.

APPENDICES

Appendices should be inserted as Appendix – I, Appendix – II, and so on. These should include extra information (user manual, conversions tables, Source codes, long tables, proofs, definitions of terms, or any material that would help in understanding contents of the report/thesis), software installation guide and user manual of the system etc.

REFRENCES

The list of books, articles and other sources should be listed at the end of report. All references must be used/cited in the text. The general format is as follows (use IEEE format):

Book

1. A.A. Beaker. Systems *Security* . Belmont, CA: Wadsworth, 2000, pp. 100-110.

Book Chapters

2. S.B. Jonson. "Information Systems," *in Systems*, 2nd ed., vol 3. J.Peters, Ed. New York: McGraw-Hill, 1964,pp. 15-67.

Article in a Journal

3. G. Pevere. "Infrared Nation." *The International Journal of Infrared Design*, vol. 33, pp. 56-99, Jan. 1979.

Articles from Conference Proceedings (Published)

4. D.B. Payne and H.G. Gunhold. "Digital Sundials ang broadband technology," in*Proc.* IOOC-ECOC, 1986,PP. 557-998.

Papers Presented at Conferences (Published)

5. B. Brandli and M. Dick. "Engineering names and concepts," presented at the 2nd Int.

Conf. Engineering Education, Frankfurt, Germany, 1999.