BSCS FINAL PROJECT

<Requirements Specification>

<Project Title>



Project Advisor

**<Advisor Name>**

Presented by:

**Group ID: xxxxxxx**

Student Reg# Student Name

**Faculty of Information Technology**

**University of Central Punjab**

Software Requirements Specification

Version <Version #>

<Project Name>

Advisor: <Advisor Name>

Group: <Group ID>

|  |  |
| --- | --- |
| Member Name | Primary Responsibility |
|  |  |
|  |  |
|  |  |
|  |  |

Table of Contents

Table of Contents i

Revision History ii

Abstract iii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Project Scope 1

1.5 Objective(s)/Aim(s)/Target(s) 1

1.6 Challenges 1

1.7 Learning Outcomes 1

1.8 Nature of End Product 1

1.9 Completeness Criteria 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Features 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentations 2

2.7 Assumptions and Dependencies 3

3. Product Features / Functional Requirements 4

3.1 Name of Use-Case 1 4

3.2 Name of Use-Case 2 (and so on) 4

3.3 Requirement 3 (and so on) 4

3.4 Analysis and Modeling of Requirements 5

4. External Interface Requirements 5

4.1 User Interfaces 5

4.2 Hardware Interfaces 5

4.3 Software Interfaces 5

4.4 Communication Interfaces 5

5. Other Nonfunctional Requirements 6

5.1 Performance Requirements 6

5.2 Safety Requirements 6

5.3 Security Requirements 6

5.4 Additional Software Quality Attributes 6

6. Other Requirements 6

7. Revised Project Plan 6

8. References 7

Appendix A: Glossary 8

Appendix B: IV & V Report 9

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Abstract

Abstract gives the summary of your project. You should focus on the problem description, significance of the problem, knowledge areas to be used and the results to be acquired. Make sure it does not turn out to be an introduction to the introduction / background but summarizes the whole project.

# Introduction

## Purpose

<Define software requirements for the end product.>

## 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 italicized nouns represent external systems.>

## 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.>

>

## Project Scope

<Provide a short description of the software that would run the product, 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.>

## Objective(s)/Aim(s)/Target(s)

<List all the objectives to be achieved after the completion of this project.>

## Challenges

## Learning Outcomes

## Nature of End Product

## Completeness Criteria

*<State the criteria that if fulfilled will let evaluators consider the project as complete >*

# Overall Description

## 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. Illustrate the importance of software’s place in the project.>*

## Product Features

<Summarize the major features the product contains or the significant functions that it performs or lets the user perform. Details will be provided in Section 3, so only a high level summary is needed here. Organize the functions to make them understandable to any reader of the SRS.

How underlying software in the product will control all aspects of the project. Software requirements for the product>

## 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 favored user classes from those who are less important to satisfy.>

## 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.>

## 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).>

## User Documentations

<<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. >

## 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).>

# Product Features / Functional Requirements

< This section of the document would list key features expressed as use-cases. Fill out the following template for each use-case. Don’t really say “Use-Case 1.” State the use-case name in just a few words e.g. “Withdraw Cash from ATM”. A use-case may have multiple alternate courses of action.>

<Provide a Use Case Diagram before describing the use cases.>

## Name of Use-Case 1

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-1 | |
| **Purpose** | | … | |
| **Priority** | | <Choose one from {High, Medium, Low}> | |
| **Pre-conditions** | | … | |
| **Post-conditions** | | … | |
| **Typical Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** |  | |  |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |
| **Alternate Course of Action** | | | |
| **S#** | **Actor Action** | | **System Response** |
| **1** |  | |  |
| **2** |  | |  |
| **3** |  | |  |
| **…** |  | |  |

Table 1: UC-1

## Name of Use-Case 2 (and so on)

< If there are features that cannot be expressed as use-cases, provide them one by one such as Section 3.3. below.>

## Requirement 3 (and so on)

*<For example a requirement can be written as follows:*

The proposed system should be able to perform tasks….>

## Analysis and Modeling of Requirements

<Include the following analysis models: use-case diagram (provide before Section 3.1), entity-relationship diagram, abstract class diagram, sequence diagram (to model interactions between system and external world). Additional diagrams may be added for example state diagram, data flow diagram (to model interactions between system and external world), decision table, flow chart, circuit diagram, event table etc.>

# External Interface Requirements

## 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.>

## 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.>

## 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.>

## Communication 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.>

# Other Nonfunctional Requirements

## 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.>

## Safety 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.>

## Security 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.>

## Additional 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.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. These might include database requirements, external (hardware, software, or communication) interface requirements, internationalization requirements, legal requirements, and reuse objectives for the project.>

# Revised Project Plan

<Provide current status of the project in accordance with the plan provided in project proposal. Gantt chart should be used in this regard. Use Microsoft Office to develop the Gantt chart. Also provide an updated project plan.>

# References

*<List all books, conference papers, journal articles, websites, etc. used in preparing the content of this SRS. Provide enough information so that the reader could access a copy of each reference, including title, author, volume/edition number, page number(s), and publication year. Mention complete URLs for websites.>*

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, 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 SRS.>

Appendix B: IV & V Report

**(Independent verification & validation)**

**IV & V Resource**

Name Signature

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S#** | **Defect Description** | **Origin Stage** | **Status** | **Fix Time** | |
| **Hours** | **Minutes** |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| … |  |  |  |  |  |

**Table 3: List of non-trivial defects**