Software Requirements Specification

<Your Project Name Here>

<Course, Semester, Year>

Team Name: <your team name here>

Team Members:

- <person1>
- <person2>
- ..

Contents of this Document

Section 1: Introduction

Software to be Produced:

- <Include one paragraph to identify the software product(s) to be produced. Refer the reader to the reference documents for more information.>

Applicable Standards

- <You do not have to repeat the standards included in the project plan. Instead, cite any standards that are specific to the system requirements.>

Definitions, Acronyms, and Abbreviations

- <Include any that are needed to read this document or "none" if document is self-explanatory and no acronyms or abbreviations will be used>

Section 2: Product Overview

Assumptions:

- <List all the assumptions the developers are making. For example: assumptions about other systems this product will interface with; assumptions about the technological environment in which the product will operate (how much memory, what type of processor, ...); assumptions about availability and capability of COTS, GOTS, or other re-used products, ...>

Stakeholders:

- <A stakeholder is anyone who has an interest in the software to be developed. For example, the customer, the various classes of users, applicable regulatory agencies, ... List each category of stakeholder and give a phrase or a sentence to describe their interest or concerns>

Event Table:

- <An event table identifies all the external events to which the software must respond. This is a first step in determining the required overall system functionality. The event list should be consistent with the context diagram and the interest of each stakeholder. Make sure that exceptions are considered.>
- <Use the following table format:>

Event Name	External Stimuli	External Responses	Internal data and state

Use Case Diagram

- <Include a use case diagram here. It should be consistent with all the above work. >

Use Case Descriptions:

- <Briefly describe each use case included in the above diagram. >

Section 3: Specific Requirements

<Use the following template for each requirement. >

No: <unique requirement number>
Statement: <the "shall" statement of the requirement>
Source: <source of the requirement>
Dependency: <list each other requirement on which satisfaction of this requirement depends. (May be "None")>
Conflicts: <list each other requirements with which this requirement conflicts. (May be "None")>
Supporting Materials: <list any supporting diagrams, lists, memos, etc.>
Evaluation Method: <How can you tell if the completed system satisfies this requirement? >
Revision History: <who, when, what>

3.1 Functional Requirements

- < Describe the fundamental actions that the software must perform. Functional requirements can be partitioned into subfunctions or subprocesses. Note: the software design partition does not have to correspond with the functional requirements partition. Functional requirements include:
  - validity checks on the inputs,
  - exact sequence of operations,
  - responses to abnormal situations
  - relationship of outputs to inputs
    - input/output sequences, formulas for input to output conversion, etc.
  - ...>

3.2 Interface Requirements

- < Describe the interactions of the software with other entities. Interface requirements include a precise description of the protocol for each interface:
  - what data items are input
  - what data items are output
  - what is the data type, the format, and the possible range of values for each data item? (i.e. what is the "domain" of this data item?)
  - how accurate must each data item be?
  - how often will each data item be received or sent?
  - timing issues (synchronous/asynchronous)>
  - how many will be received or sent in a particular time period?
  - how accurate must the data be?
  - ...>

3.3 Physical Environment Requirements

- < Describe the environment in which the software must run. Physical environment requirements include:
  - type of equipment on which the software must run
  - location of the equipment
  - environmental considerations: temperature, humidity, ...
  - ...>

3.4 User and Human Factors Requirements

- <Describe the users and their constraints:
  - What different types of users must the system support?
  - What is the skill level of each type of user? What type of training and documentation must be provided for each user?
  - Do any users require special accommodations (large font size, ...)
  - Must the system detect and prevent misuse? If so, what types of potential misuse must the system detect and prevent?
  - ...>

3.5 Documentation Requirements

- <Describe what documentation is required:
  - on-line, printed, or both?
  - what is the assumed skill level of the audience of each component of documentation?
  - ...>

3.6 Data Requirements

- <Describe any data calculations: what formula will be used? to what degree of precision must the calculations be made? >
- <Describe any retained data requirements: exactly what must be retained?
- ...>

3.7 Resource Requirements

- <Describe the system resources:
    - skilled personnel required to build, use, and maintain the system?
    - physical space, power, heating, air conditioning, ...?
    - schedule?
    - funding?
    - hardware/software/tools?
    - ...>

3.8 Security Requirements

- <Describe any security requirements:
    - must access to the system or information be controlled?
    - must one user's data be isolated from others?
    - how will user programs be isolated from other programs and from the operating system?
    - how often will the system be backed up?
    - must the backup copies be stored at a different location?
    - should precautions be taken against fire, water damage, theft, ...?
    - what are the recovery requirements?
    - ...>

3.9 Quality Assurance Requirements (AKA nonfunctional, Quality Requirements)

- <Describe quality attributes:
    - What are the requirements for reliability, availability, maintainability, security, portability ...?
    - How must these quality attributes be demonstrated?
    - Must the system detect and isolate faults? If so, what types of faults?
    - Is there a prescribed mean time between failures?
    - Is there a prescribed time the system must be available?
    - Is there a maximum time allowed for restarting the system after a failure?
    - What are the requirements for resource usage and response times?
    - ...>

Section 4: Supporting Material

- <Here is where you put all your analysis work from which you derived the above requirements. It may include UML or other diagrams, notes, memos, etc.)