

# **[YourProject] Requirements Specification**

**Version 1.0**

**April 19, 2021**

Use this Requirements Specification template to document the requirements for your product or service, including priority and approval. Tailor the specification to suit your project, organizing the applicable sections in a way that works best, and use the checklist to record the decisions about what is applicable and what isn't.

The format of the requirements depends on what works best for your project.

This document contains instructions and examples which are for the benefit of the person writing the document and should be removed before the document is finalized.

To regenerate the TOC, select all (CTL-A) and press F9.

# Table of Contents

<b>1. EXECUTIVE SUMMARY .....</b>	<b>3</b>
1.1 PROJECT OVERVIEW .....	3
1.2 PURPOSE AND SCOPE OF THIS SPECIFICATION .....	3
<b>2. PRODUCT/SERVICE DESCRIPTION.....</b>	<b>3</b>
2.1 PRODUCT CONTEXT .....	3
2.2 USER CHARACTERISTICS .....	3
2.3 ASSUMPTIONS.....	3
2.4 CONSTRAINTS.....	3
2.5 DEPENDENCIES .....	4
<b>3. REQUIREMENTS .....</b>	<b>4</b>
3.1 FUNCTIONAL REQUIREMENTS .....	5
3.2 NON-FUNCTIONAL REQUIREMENTS .....	5
3.2.1 <i>User Interface Requirements</i> .....	6
3.2.2 <i>Usability</i> .....	6
3.2.3 <i>Performance</i> .....	6
3.2.4 <i>Manageability/Maintainability</i> .....	6
3.2.5 <i>System Interface/Integration</i> .....	7
3.2.6 <i>Security</i> .....	8
3.2.7 <i>Data Management</i> .....	8
3.2.8 <i>Standards Compliance</i> .....	8
3.2.9 <i>Portability</i> .....	9
3.2.10 <i>Other Non-Functional Requirements</i> .....	9
3.3 DOMAIN REQUIREMENTS .....	9
<b>4. USER SCENARIOS/USE CASES .....</b>	<b>9</b>
<b>APPENDIX .....</b>	<b>10</b>
APPENDIX A. DEFINITIONS, ACRONYMS, AND ABBREVIATIONS .....	10
APPENDIX B. REFERENCES.....	10
APPENDIX C. REQUIREMENTS TRACEABILITY MATRIX .....	10
APPENDIX D. ORGANIZING THE REQUIREMENTS .....	12

# 1. Executive Summary

## 1.1 Project Overview

In the last 10 years farms in Albania had a noticeable development after years of forced collectivization of agricultural cooperatives. According to official statistics, today in Albania, more than 90% of farms are mainly based on family farms that have an economical and social dimension. Our goal is to provide the local farmers with an efficient solution based on their needs. Our project is centered on providing a management system for farmers in order to ease the management of their daily activity.

The Farming Management System will help to plan, monitor and analyze all activities on the farm easily. Tillage, planting, crop protection, fertilization, irrigation, harvesting and all other activities are managed with a few clicks. Plus, you can track input usage quantities, costs and work hours for every activity. The system will be also centered around livestock management: that include buying livestock, selling, and observing or calculating profit.

## 1.2 Purpose and Scope of this Specification

The purpose of this project is to implement technology in order to improve and facilitate the everyday operations of farms.. This document will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external actions.

This document is intended for both the stakeholders and the developers of the system.

This software system will be a Farming Management System for local farm owners. This system will be designed to support production management and to comply with actual farming standards. By maximizing the farmer's work efficiency and production, the system will meet the farmer's needs while remaining easy to understand and use.

# 2. Product/Service Description

In this section, describe the general factors that affect the product and its requirements. This section should contain background information, not state specific requirements (provide the reasons why certain specific requirements are later specified).

## 2.1 Product Context

How does this product relate to other products? Is it independent and self-contained? Does it interface with a variety of related systems? Describe these relationships or use a diagram to show the major components of the larger system, interconnections, and external interfaces.

## 2.2 User Characteristics

Create general customer profiles for each type of user who will be using the product. Profiles should include:

- Student/faculty/staff/other
- experience
- technical expertise
- other general characteristics that may influence the product

## 2.3 Assumptions

List any assumptions that affect the requirements, for example, equipment availability, user expertise, etc. For example, a specific operating system is assumed to be available; if the operating system is not available, the Requirements Specification would then have to change accordingly.

## 2.4 Constraints

Describe any items that will constrain the design options, including

## **[YourProject] Requirements Specification**

- parallel operation with an old system
- audit functions (audit trail, log files, etc.)
- access, management and security
- criticality of the application
- system resource constraints (e.g., limits on disk space or other hardware limitations)
- other design constraints (e.g., design or other standards, such as programming language or framework)

### **2.5 Dependencies**

List dependencies that affect the requirements. Examples:

- This new product will require a daily download of data from X,
- Module X needs to be completed before this module can be built.

## **3. Requirements**

- Describe all system requirements in enough detail for designers to design a system satisfying the requirements and testers to verify that the system satisfies requirements.
- Organize these requirements in a way that works best for your project. See [Appendix D, Organizing the Requirements](#) for different ways to organize these requirements.
- Describe every input into the system, every output from the system, and every function performed by the system in response to an input or in support of an output. (Specify what functions are to be performed on what data to produce what results at what location for whom.)
- Each requirement should be numbered (or uniquely identifiable) and prioritized. See the sample requirements in Functional Requirements, and System Interface/Integration, as well as these example priority definitions:

### **Priority Definitions**

The following definitions are intended as a guideline to prioritize requirements.

- Priority 1 – The requirement is a “must have” as outlined by policy/law
- Priority 2 – The requirement is needed for improved processing, and the fulfillment of the requirement will create immediate benefits
- Priority 3 – The requirement is a “nice to have” which may include new functionality

It may be helpful to phrase the requirement in terms of its priority, e.g., "The value of the employee status sent to DIS **must be** either A or I" or "It **would be nice** if the application warned the user that the expiration date was 3 business days away". Another approach would be to group requirements by priority category.

- A good requirement is:
  - Correct
  - Unambiguous (all statements have exactly one interpretation)
  - Complete (where TBDs are absolutely necessary, document why the information is unknown, who is responsible for resolution, and the deadline)
  - Consistent
  - Ranked for importance and/or stability
  - Verifiable (avoid soft descriptions like “works well”, “is user friendly”; use concrete terms and specify measurable quantities)
  - Modifiable (evolve the Requirements Specification only via a formal change process, preserving a complete audit trail of changes)
  - Does not specify any particular design
  - Traceable (cross-reference with source documents and spawned documents).

### 3.1 Functional Requirements

In the example below, the requirement numbering has a scheme - BR\_LR\_0## (BR for Business Requirement, LR for Labor Relations). For small projects simply BR-## would suffice. Keep in mind that if no prefix is used, the traceability matrix may be difficult to create (e.g., no differentiation between '02' as a business requirement vs. a test case)

The following table is an example format for requirements. Choose whatever format works best for your project.

For Example:

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_LR_05	The system should associate a supervisor indicator with each job class.	Business Process = "Maintenance"	3	7/13/04	Bob Dylan, Mick Jagger
BR_LR_08	The system should handle any number of fees (existing and new) associated with unions.	Business Process = "Changing Dues in the System" An example of a new fee is an initiation fee.	2	7/13/04	Bob Dylan, Mick Jagger
BR_LR_10	The system should capture and maintain job class status (i.e., active or inactive)	Business Process = "Maintenance" Some job classes are old and are no longer used. However, they still need to be maintained for legal, contract and historical purposes.	2	7/13/04	Bob Dylan, Mick Jagger
BR_LR_16	The system should assign the Supervisor Code based on the value in the Job Class table and additional criteria as specified by the clients.	April 2005 – New requirement. It is one of three new requirements from BR_LR_03.	2		
BR_LR_18	The system should provide the Labor Relations office with the ability to override the system-derived Bargaining Unit code and the Union Code for to-be-determined employee types, including hourly appointments.	April 2005 – New requirement. It is one of three new requirements from BR_LR_04. 5/11/2005 – Priority changed from 2 to 3.	2 3		

### 3.2 Non-Functional Requirements

In here try to use the Structure given at slide 13 in Requirements Engineering Lecture Slides, with main categories of:

**Product Requirements**

**Organizational Requirements**

**External Requirements**

And their sub categories, try to arrange them in a proper visible way.

### **3.2.1 User Interface Requirements**

In addition to functions required, describe the characteristics of each interface between the product and its users (e.g., required screen formats/organization, report layouts, menu structures, error and other messages, or function keys).

### **3.2.2 Usability**

Include any specific usability requirements, for example,

#### **Learnability**

- The user documentation and help should be complete
- The help should be context sensitive and explain how to achieve common tasks
- The system should be easy to learn

(See <http://www.usabilitynet.org/>)

### **3.2.3 Performance**

Specify static and dynamic numerical requirements placed on the system or on human interaction with the system:

- Static numerical requirements may include the number of terminals to be supported, the number of simultaneous users to be supported, and the amount and type of information to be handled.
- Dynamic numerical requirements may include the number of transactions and tasks and the amount of data to be processed within certain time period for both normal and peak workload conditions.

All of these requirements should be stated in measurable form. For example, "95% of the transactions shall be processed in less than 1 second" rather than "an operator shall not have to wait for the transaction to complete".

#### **3.2.3.1 Capacity**

Include measurable capacity requirements (e.g., the number of simultaneous users to be supported, the maximum simultaneous user load, per-user memory requirements, expected application throughput)

#### **3.2.3.2 Availability**

Include specific and measurable requirements for:

- Hours of operation
- Level of availability required
- Coverage for geographic areas
- Impact of downtime on users and business operations
- Impact of scheduled and unscheduled maintenance on uptime and maintenance communications procedures
- reliability (e.g., acceptable mean time between failures (MTBF), or the maximum permitted number of failures per hour).

#### **3.2.3.3 Latency**

Include explicit latency requirements, e.g., the maximum acceptable time (or average time) for a service request.

### **3.2.4 Manageability/Maintainability**

#### **3.2.4.1 Monitoring**

Include any requirements for product or service health monitoring, failure conditions, error detection, logging, and correction.

## ***[YourProject] Requirements Specification***

### **3.2.4.2 Maintenance**

Specify attributes of the system that relate to ease of maintenance. These requirements may relate to modularity, complexity, or interface design. Requirements should not be placed here simply because they are thought to be good design practices.

### **3.2.4.3 Operations**

Specify any normal and special operations required by the user, including:

- periods of interactive operations and periods of unattended operations
- data processing support functions
- backup and recovery operations
- safety considerations and requirements
- disaster recovery and business resumption

### **3.2.5 System Interface/Integration**

Specify the use of other required products (e.g., a database or operating system), and interfaces with other systems (e.g., UWHires package interfaces with PubCookie and ODS, HEPPS system interfaces with Budget system). For each interface, define the interface in terms of message format and content. For well-documented interfaces, simply provide a reference to the documentation.

Outline each interface between the product and the hardware or network components of the system. This includes configuration characteristics (e.g., number of ports, instruction sets), what devices are to be supported, and protocols (e.g., signal handshake protocols).

#### **3.2.5.1 Network and Hardware Interfaces**

Specify the logical characteristics of each interface between the product and the hardware or network components of the system. This includes configuration characteristics (e.g., number of ports, instruction sets), what devices are to be supported, and protocols (e.g., signal handshake protocols).

#### **3.2.5.2 Systems Interfaces**

Example systems interface requirements:

##### ***A. System1-to-System2 Interface***

The <external party> will create and send a fixed length text file as an email attachment to [System2mail@u.washington.edu](mailto:System2mail@u.washington.edu) to be imported into the System2 system for payroll calculation. This file must be received on EDIT day by 4:00 PM in order to be processed in the EDIT night run. The requirements below document the file specifications, data transfer process, and specific schedule. This file is referred to as "FileName" in this document.

##### ***File Structure and Format***

- A1. The FileName file is a fixed length text file.
- A2. The FileName file is an unformatted ASCII file (text-only).
- A3. The FileName file contains a batch totals record and several detail records.

##### ***File Description: Batch Totals Record***

- A4. The batch totals record can be placed at the beginning, in the middle, or at the end of the file.
- A5. The batch totals record contains the following:

- Record Type (value: XA)
- Process Type (value: A)
- Batch Number (3 digit number assigned by Payroll Dept)
- Origin Code (AIG)
- Total number of detail records
- Total deduction amount

## **[YourProject] Requirements Specification**

### **File Description: Detail Records**

A6. The FileName file contains a row for each record meeting xxx criteria.

A7. Each row in the FileName file contains the following fields, comma-delimited and encased in double-quotes where the data includes commas or spaces:

- Employee Id
- Record Type
- Process Date (MMDDYY)
- XYG Number
- Element Code
- Amount
- Amount Sign
- Year Flag
- Total Amount
- Total Amt Sign

### **3.2.6 Security**

#### **3.2.6.1 Protection**

Specify the factors that will protect the system from malicious or accidental access, modification, disclosure, destruction, or misuse. For example:

- encryption
- activity logging, historical data sets
- restrictions on intermodule communications
- data integrity checks

#### **3.2.6.2 Authorization and Authentication**

Specify the Authorization and Authentication factors. Consider using standard tools such as PubCookie.

### **3.2.7 Data Management**

Specify the requirements for any information that is to be placed into a database, including

- types of information used by various functions
- frequency of use
- data access rules
- data entities and relationships
- integrity constraints
- data retention
- valid range, accuracy, and/or tolerance
- units of measure
- data formats
- default or initial values

### **3.2.8 Standards Compliance**

Specify the requirements derived from existing standards, policies, regulations, or laws (e.g., report format, data naming, accounting procedures, audit tracing). For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database must be recorded in a trace file with before and after values.



### **3.2.9 Portability**

If portability is a requirement, specify attributes of the system that relate to the ease of porting the system to other host machines and/or operating systems. For example,

- Percentage of components with host-dependent code;
- Percentage of code that is host dependent;
- Use of a proven portable language;
- Use of a particular compiler or language subset;
- Use of a particular operating system;
- The need for environment-independence - the product must operate the same regardless of operating systems, networks, development or production environments.

### **3.2.10 Other Non-Functional Requirements**

Please provide all necessary non-functional requirements, similar to the requirements explained in the lesson slides or in the textbook.

## **3.3 Domain Requirements**

Everything related to the domain that might be needed in the project shall be mentioned in here. Sometimes the domain Requirements might be thought as part of either functional or non-functional requirements.

## **4. User Scenarios/Use Cases**

Provide a summary of the major functions that the product will perform. Organize the functions to be understandable to the customer or a first time reader. Include use cases and business scenarios, or provide a link to a separate document (or documents). A business scenario:

- Describes a significant business need
- Identifies, documents, and ranks the problem that is driving the scenario
- Describes the business and technical environment that will resolve the problem
- States the desired objectives
- Shows the “Actors” and where they fit in the business model
- Is specific, and measurable, and uses clear metrics for success

## APPENDIX

The appendixes are not always considered part of the actual Requirements Specification and are not always necessary. They may include

- Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
- Supporting or background information that can help the readers of the Requirements Specification;
- A description of the problems to be solved by the system;
- Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the Requirements Specification should explicitly state whether or not the appendixes are to be considered part of the requirements.

### Appendix A. Definitions, Acronyms, and Abbreviations

Define all terms, acronyms, and abbreviations used in this document.

### Appendix B. References

List all the documents and other materials referenced in this document.

### Appendix C. Requirements Traceability Matrix

The following trace matrix examples show one possible use of naming standards for deliverables (FunctionalArea-DocType-NN). The number has no other meaning than to keep the documents unique. For example, the Bargaining Unit Assignment Process Flow would be BUA-PF-01.

For example (1):

Business Requirement	Area	Deliverables	Status
BR_LR_01 The system should validate the relationship between Bargaining Unit/Location and Job Class.---Comments: Business Process = "Assigning a Bargaining Unit to an Appointment" (Priority 1)	BUA	BUA-CD-01 Assign BU Conceptual Design	Accepted
		BUA-PF-01 Derive Bargaining Unit-Process Flow Diagram	Accepted
		BUA-PF-01 Derive Bargaining Unit-Process Flow Diagram	Accepted
BR_LR_09 The system should provide the capability for the Labor Relations Office to maintain the job class/union relationship.---Comments: Business Process = "Maintenance" (Priority 1)	BUA	BUA-CD-01 Assign BU Conceptual Design	Accepted
		BUA-PF-02 BU Assignment Rules Maint Process Flow Diagram	ReadyForReview

For example (2):

BizReqID	Pri	Major Area	DevTstItems DelivID	Deliv Name	Status
BR_LR_01	1	BUA	BUA-CD-01	Assign BU Conceptual Design	Accepted
BR_LR_01	1	BUA	BUA-DS-02	Bargaining Unit Assignment DB Modification Description	Accepted
BR_LR_01	1	BUA	BUA-PF-01	Derive Bargaining Unit-Process Flow Diagram	Accepted

**[YourProject] Requirements Specification**

BizReqID	Pri	Major Area	DevTstItems DelivID	Deliv Name	Status
BR_LR_01	1	BUA	BUA-UCD-01	BU Assign LR UseCase Diagram	ReadyForReview
BR_LR_01	1	BUA	BUA-UCT-001	BU Assignment by PC UseCase - Add Appointment and Derive UBU	Reviewed
BR_LR_01	1	BUA	BUA-UCT-002	BU Assignment by PC UseCase - Add Appointment (UBU Not Found)	Reviewed
BR_LR_01	1	BUA	BUA-UCT-006	BU Assignment by PC UseCase - Modify Appointment (Removed UBU)	Reviewed
BR_LR_09	1	BUA	BUA-CD-01	Assign BU Conceptual Design	Accepted
BR_LR_09	1	BUA	BUA-DS-02	Bargaining Unit Assignment DB Modification Description	Accepted
BR_LR_09	1	BUA	BUA-PF-02	BU Assignment Rules Maint Process Flow Diagram	Accepted
BR_LR_09	1	BUA	BUA-UCD-03	BU Assign Rules Maint UseCase Diagram	Reviewed
BR_LR_09	1	BUA	BUA-UCT-045	BU Assignment Rules Maint: Successfully Add New Assignment Rule	Reviewed
BR_LR_09	1	BUA	BUA-UCT-051	BU Assignment Rules MaintUseCase: Modify Rule	Reviewed
BR_LR_09	1	BUA	BUA-UCT-053	BU Assignment Rules MaintUseCase - Review Assignment Rules	Reviewed
BR_LR_09	1	BUA	BUA-UCT-057	BU Assignment Rules MaintUseCase: Inactivate Last Rule for a BU	Reviewed
BR_LR_09	1	BUA	BUA-UI-02	BU AssignRules Maint UI Mockups	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-021	BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Success	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-027	BU Assignment Rules Maint TestCase: Modify Rule - Success	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-035	BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Error Condition	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-049	BU Assignment Rules Maint TestCase: Modify Rule - Error Condition	ReadyForReview

For example (3):

BizReqID	CD01	CD02	CD03	CD04	UI01	UI02	UCT01	UCT02	UCT03	TC01	TC02	TC03	TC04
BR_LR_01			X		X		X			X		X	
BR_LR_09	X			X		X			X		X		X
BR_LR_10	X			X					X		X		
BR_LR_11		X											

## **Appendix D. Organizing the Requirements**

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

### **By System Mode**

Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

### **By User Class**

Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

### **By Objects**

Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

### **By Feature**

A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

### **By Stimulus**

Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

### **By Response**

Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

### **By Functional Hierarchy**

When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

### **Additional Comments**

Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.