Software Requirements Specification

SOFTUNI BLOG

Team FAFNIR :

**Gergana Guncheva**

**Gergana Bozova**

**Georgi Lukov**

(Project Title)

(Team Name and Number)

(Team Members)

**Software Requirements Specification**

**Document**

**Version: (n)** **Date: (mm/dd/yyyy)**

**Table of Contents**

|  |  |  |  |
| --- | --- | --- | --- |
| [1. Introduction ...............................................................................................................................................](#page3) | | | [1](#page3) |
| [1.1](#page3) | [Purpose ................................................................................................................................................](#page3) | | [1](#page3) |
| [1.2](#page3) | [Scope ...................................................................................................................................................](#page3) | | [1](#page3) |
| [1.3](#page3) | [Definitions, acronyms, and abbreviations ...........................................................................................](#page3) | | [1](#page3) |
| [1.4](#page4) | [References ...........................................................................................................................................](#page4) | | [2](#page4) |
| [2. Overall description ....................................................................................................................................](#page6) | | | [4](#page6) |
| [2.1](#page6) | [Product perspective .............................................................................................................................](#page6) | | [4](#page6) |
| [2.2](#page6) | [Product functions ................................................................................................................................](#page6) | | [4](#page6) |
| [2.3](#page7) | [User characteristics .............................................................................................................................](#page7) | | [5](#page7) |
| [2.4](#page7) | [Constraints ..........................................................................................................................................](#page7) | | [5](#page7) |
| [2.5](#page7) | [Assumptions and dependencies ..........................................................................................................](#page7) | | [5](#page7) |
| [2.6](#page8) | [Apportioning of requirements .............................................................................................................](#page8) | | [6](#page8) |
| [3. Specific requirements ................................................................................................................................](#page9) | | | [7](#page9) |
| [3.1.1](#page9) | | [User interfaces .............................................................................................................................](#page9) | [7](#page9) |
| [3.1.2](#page10) | | [Hardware interfaces .....................................................................................................................](#page10) | [8](#page10) |
| [3.1.3](#page10) | | [Software interfaces .......................................................................................................................](#page10) | [8](#page10) |
| [3.1.4](#page11) | | [Communications interfaces ..........................................................................................................](#page11) | [9](#page11) |
| [3.2](#page11) | [Functional requirements ......................................................................................................................](#page11) | | [9](#page11) |
| [3.2.1](#page11) | | [User Class 1 - The User ...............................................................................................................](#page11) | [9](#page11) |
| [3.2.2](#page16) | | [User Class 2 - Restaurant Owner ...............................................................................................](#page16) | [14](#page16) |
| [3.2.3](#page20) | | [User Class 3 - Administrator ......................................................................................................](#page20) | [18](#page20) |
| [3.3](#page23) | [Performance requirements ................................................................................................................](#page23) | | [21](#page23) |
| [3.4](#page25) | [Design constraints .............................................................................................................................](#page25) | | [23](#page25) |
| [3.5](#page25) | [Software system attributes ................................................................................................................](#page25) | | [23](#page25) |
| [4. Prioritization and Release Plan ...............................................................................................................](#page29) | | | [27](#page29) |
| [4.1](#page29) | [Choice of prioritization method ........................................................................................................](#page29) | | [27](#page29) |
| [Appendix I: Selection for Cost-Value Approach ........................................................................................](#page31) | | | [29](#page31) |
| [Appendix II: Prioritization Result of 10 selected Requirements Using Cost-Value Approach ..................](#page34) | | | [32](#page34) |
| [Appendix III: Five-Way Priority Scheme ...................................................................................................](#page38) | | | [36](#page38) |
| [Appendix IV: Release Plan .........................................................................................................................](#page49) | | | [47](#page49) |
| [Appendix V: I-star ......................................................................................................................................](#page57) | | | [55](#page57) |

# Introduction

*~~The following subsections of the Software Requirements Specifications (SRS) document should provide an overview of the entire SRS. The thing to keep in mind as you write this document is that you are telling what the system must do – so that designers can ultimately build it. Do not use this document for design!!!~~*

~~This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.~~

## Purpose

*~~Identify the purpose of this SRS and its intended audience. In this subsection, describe the purpose of the particular SRS and specify the intended audience for the SRS.~~*

The purpose of this document is to give a detailed description of the requirements for the “SoftUni Blog” system. ~~, basic blog. It will illustrate the purpose and complete declaration for the development of system, for the moment. It will also explain system constraints and interface. This document is primarily intended …~~ It will explain the purpose and features of the system, the interfaces of the system and what the system will do. This document is intended for both the developers of the system and the quality assurance team.

## Scope

*~~In this subsection:~~*

1. *~~Identify the software product(s) to be produced by name~~*
2. *~~Explain what the software product(s) will, and, if necessary, will not do~~*
3. *~~Describe the application of the software being specified, including relevant benefits, objectives, and goals~~*
4. *~~Be consistent with similar statements in higher-level specifications if they exist~~*

*~~This should be an executive-level summary. Do not enumerate the whole requirements list here.~~*

~~The “SoftUni Blog”~~ The system is a basic online blog, ~~It’s an online journal or informational website displaying information in the reverse chronological order, with latest posts appearing first. It is a platform where a person or could share their views on an individual subject. The blog should be free for visitors, who are interested in it.~~ which provides functionality, where the users can post their own thoughts, their visions on exact problems or things that are in their interests. The system also lets the users to edit or delete their posts.

~~Bloggers can fill in new information and change the old one .~~

~~An administrator also uses the blog platform in order to administer the system and keep the information accurate. For example, an administrator may block a blog user if he / she deems it necessary to violate the rules of the platform.~~

The users have different permissions as ordinary users and as administrator. The registered users can make posts and edit or delete their posts and the administrator monitors for the correctness of the posts in the blog.

~~The software needs only of Internet connection.~~ The system works on different browsers and needs internet connection.

## 1.3 Definitions, acronyms, and abbreviations

*~~Provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendices in the SRS or by reference to documents. This information may be provided by reference to an Appendix.~~*

|  |  |
| --- | --- |
|  | **Table 1 - Definitions** |
|  |  |
| **Term** | **Definition** |
| Blog | ~~An online journal or informational website displaying information~~  .  A regularly updated web-based system, which is displaying the posts of individuals or small group with same interests. |
| Admin/Administrator | ~~System administrator~~ A person who is given specific permission for managing and |
|  | controlling the system |
| User | ~~Someone who use the blog platform~~ A person who uses the functionalities, provided by the system and have some limited permissions |







Stakeholder



Any person who has interaction with the system who is not a developer.



DESC



Description



RAT



Rational



DEP



Dependency



TAG



A unique, persistent identifier contained in a PLanguage statement [2]



GIST



A short, simple description of the concept contained in a PLanguage statement [2]



SCALE



The scale of measure used by the requirement contained in a PLanguage statement [2]



METER



The process or device used to establish location on a SCALE contained in a PLanguage statement [2]



MUST



The minimum level required to avoid failure contained in a PLanguage statement [2]



PLAN



The level at which good success can be claimed contained in a PLanguage statement [2]



WISH



A desirable level of achievement that may not be attainable through available means contained in a PLanguage statement [2]



DEFINED



The official definition of a term contained in a PLanguage statement [2]



## 1.4.References

*~~In this subsection:~~*

*~~(1) Provide a complete list of all documents referenced elsewhere in the SRS~~*

*~~(2) Identify each document by title, report number (if applicable), date, and publishing organization~~*

1. *~~Specify the sources from which the references can be obtained.~~*

*~~This information can be provided by reference to an appendix or to another document. If your application uses specific protocols or RFC’s, then reference them here so designers know where to find them.~~*

1. ~~https://firstsiteguide.com/what-is-blog/~~
2. ~~[http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs\_example\_2010\_group2.pdf](http://www.cse.chalmers.se/~feldt/courses/reqeng/examples/srs_example_2010_group2.pdf_x0005_)~~

<http://www.softwaretestingstandard.org/> International standards for software testing

## 1.5.Overview

*~~In this subsection:~~*

1. *~~Describe what the rest of the SRS contains~~*
2. *~~Explain how the SRS is organized~~*

*~~Don’t rehash the table of contents here. Point people to the parts of the document they are most concerned with. Customers/potential users care about section 2, developers care about section 3.~~*

~~The remainder of this document includes three chapters and appendixes. The second one provides an overview of the system functionality.~~

~~The third chapter provides the requirements specification in detailed terms and a description of the different system interfaces.~~

~~The fourth chapter deals with the prioritization of the requirements. It includes a motivation for the chosen prioritization methods.~~

~~The Appendixes in the end of the document include the all results of the requirement prioritization and a release plan based on them.~~

# ~~2.Overall description~~

*~~Describe the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in section 3, and makes them easier to understand~~*~~.~~ *~~In a sense, this section tells the requirements in plain English for the consumption of the customer. Section3 will contain a specification written for the developers.~~*

~~This section will give an overview of the whole system. The system will be explained in its context to show introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.~~

~~They can also share videos (with some funny commentary in the background), share mysteries which other people can solve, post any question. Through the platform people coulld meet eachother and share the common interests and passion, chat with them and have some fun.~~

## ~~2.1.Product perspective~~

*~~Put the product into perspective with other related products. If the product is independent and totally self-contained, it should be so stated here. If the SRS defines a product that is a component of a larger system, as frequently occurs, then this subsection relates the requirements of the larger system to functionality of the software and identifies interfaces between that system and the software. If you are building a real system,compare its similarity and differences to other systems in the marketplace. If you are doing a research-oriented project, what related research compares to the system you are planning to build.~~*

*~~A block diagram showing the major components of the larger system, interconnections, and external interfaces can be helpful. This is not a design or architecture picture. It is more to provide context, especially if your system will interact with external actors. The system you are building should be shown as a black box. Let the design document present the internals.~~*

*~~The following subsections describe how the software operates inside various constraints~~*~~.~~

~~Blog structure~~

~~The appearance of blogs changed over time, and nowadays blogs include different items. But, most blogs include some standard features and structure. Here are common features that a typical blog will include:~~

* ~~Header with the menu or navigation bar~~
* ~~Main content area with highlighted or latest blog posts~~
* ~~Sidebar with social profiles, favorite content, or call-to-action~~
* ~~Footer with relevant links like a disclaimer, privacy policy, contact page, etc.~~

~~Blog portal will communicate with the database. All of the database communication will go over the Internet~~

### 2.1.1 System Interfaces

*~~List each system interface and identify the functionality of the software to accomplish the system requirement and the interface description to match the system. These are external systems that you have to interact with. For instance, if you are building a business application that interfaces with the existing employee payroll system, what is the API to that system that designer’s will need to use?~~*

### 2.1.2 Interfaces

*~~Specify:~~*

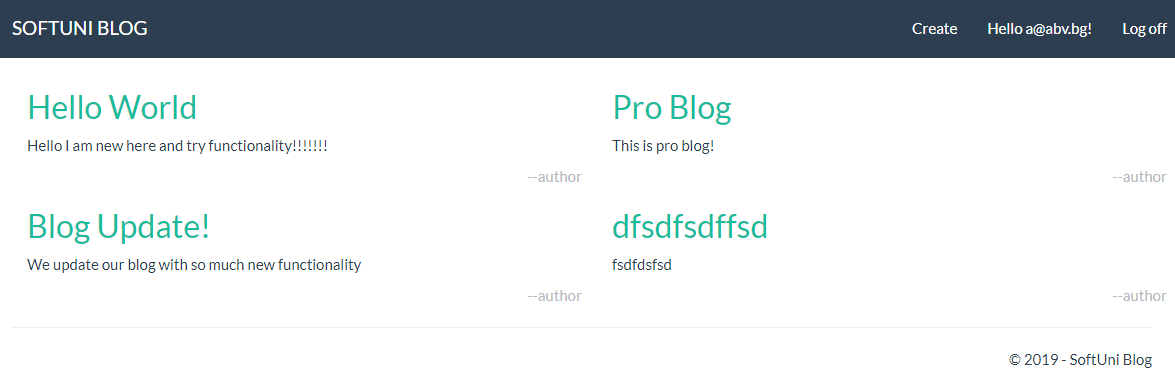
1. *~~The logical characteristics of each interface between the software product and its users.~~*
2. *~~All the aspects of optimizing the interface with the person who must use the system~~*

*~~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?~~*

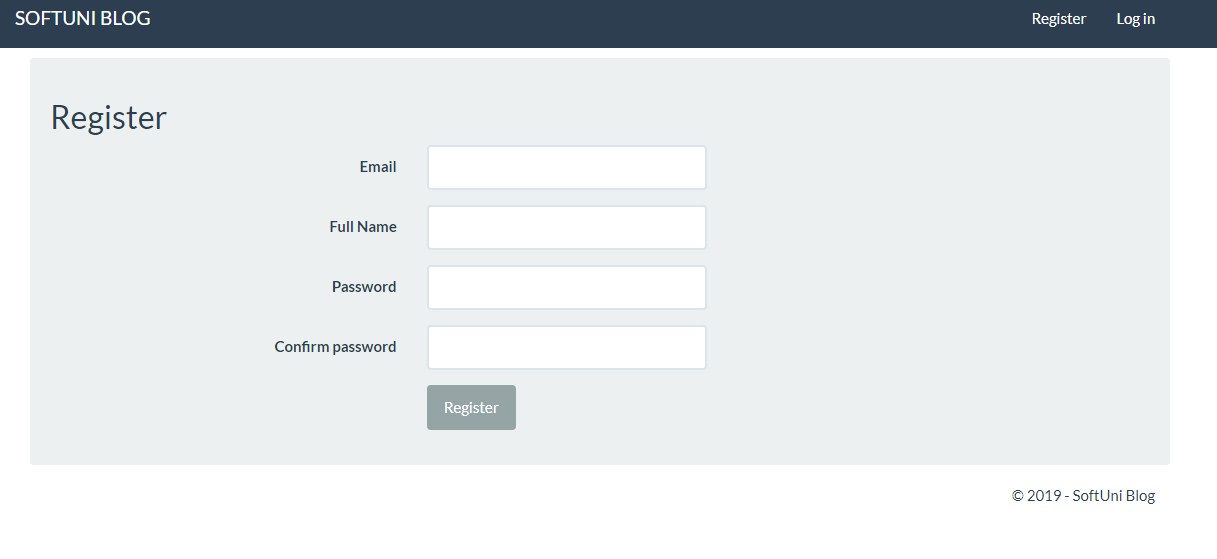
## 2.1.2.1User interfaces

~~A first-time user of the blog should see the log-in page when he/she opens the application, see Figure. If the user has not registered, he/she should be able to do that on the log-in page.~~

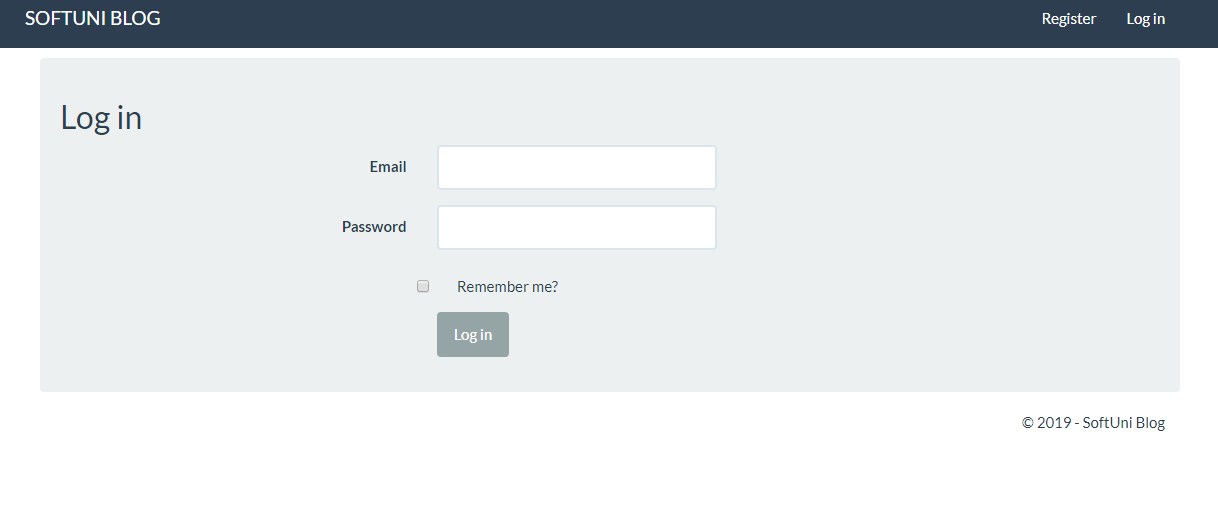
. When an user loads the internet address of the system, he/she sees the home page of the web site:



If the user is not registered, he must register his own profile in the system from the Register menu:

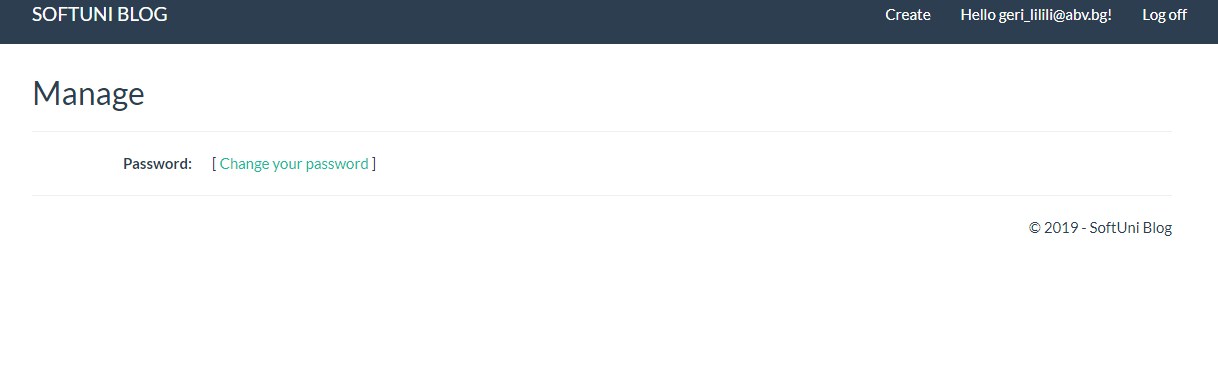


If the user is ~~not a first-time user, he/she should be able to see the blog page for log in ,see Figure :~~ already registered, he must enter his username and password form the Log in menu:



~~Every user should have a profile page where they can change password, see Figure:~~

Registered user can manage his profile from special menu, with functionality for changing the password:

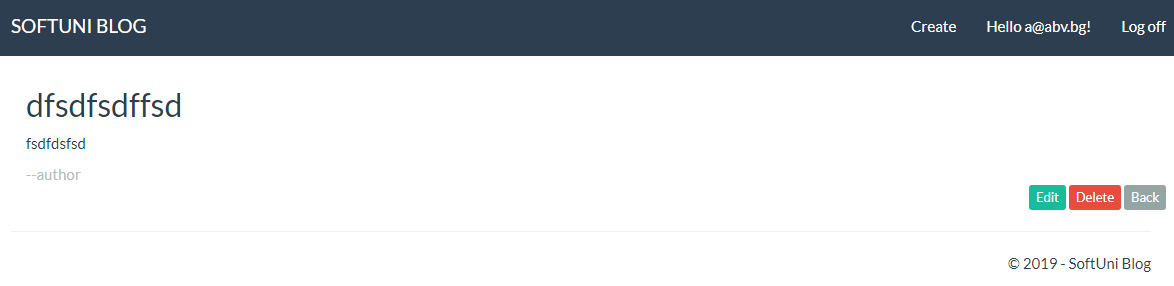


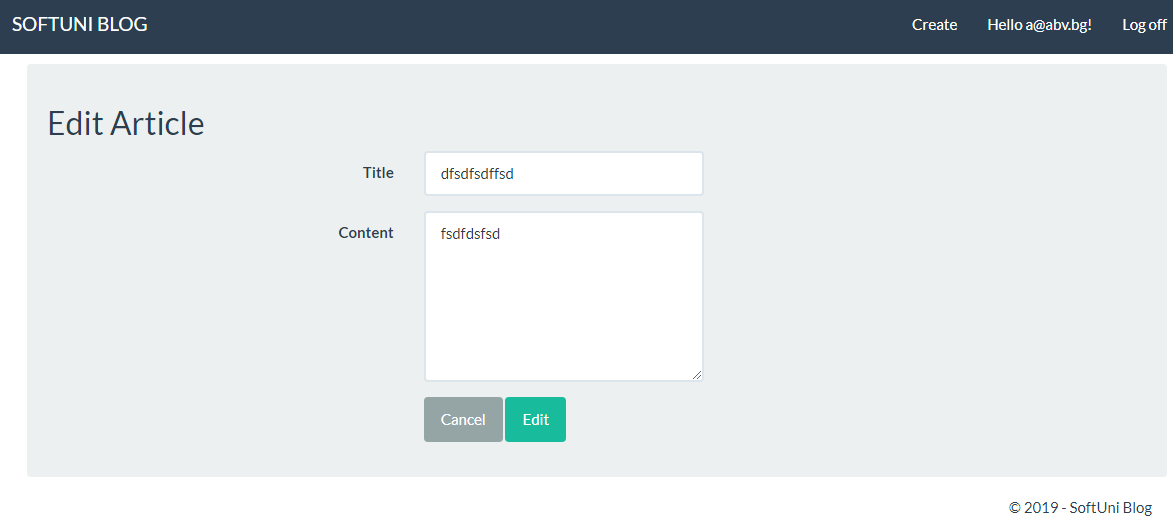
~~After registration and log in the blog, there is button “Create “ in the header .By this button each blogger/user will be able to create new article with “Title ” and “Content” .~~

Registered and logged user can create his own article and publish it to the main page of the blog. He must provide Title and Content for his post.

### Screenshot_4

The user can edit or delete his post by clicking on it from the main page:





### ~~2.1.2.2 Hardware Interfaces~~

*~~Specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes configuration characteristics. It also covers such matters as what devices are to be supported, how they are to be supported and protocols. This is not a description of hardware requirements in the sense that “This program must run on a Mac with 64M of RAM”. This section is for detailing the actual hardware devices your application will interact with and control. For instance, if you are controlling X10 type home devices, what is the interface to those devices? Designers should be able to look at this and know what hardware they need to worry about in the design. Many business type applications will have no hardware interfaces. If none, just state “The system has no hardware interface requirements” If you just delete sections that are not applicable, then readers do not know if: a. this does not apply or b. you forgot to include the section in the first place.~~*

~~For the blog it is not necessary to have some special memory, we need only of internet connection and installed internet browser. It can’t be used like web or mobile application~~

### ~~2.1.2.3 Software Interfaces~~

*~~Specify the use of other required software products and interfaces with other application systems. For each required software product, include:~~*

1. *~~Name~~*
2. *~~Mnemonic~~*
3. *~~Specification number~~*
4. *~~Version number~~*
5. *~~Source~~*

*~~For each interface, provide:~~*

1. *~~Discussion of the purpose of the interfacing software as related to this software product~~*
2. *~~Definition of the interface in terms of message content and format~~*

*~~Here we document the APIs, versions of software that we do not have to write, but that our system has to use. For instance if your customer uses SQL Server 7 and you are required to use that, then you need to specify i.e.~~*

*~~2.1.4.1 Microsoft SQL Server 7. The system must use SQL Server as its database component. Communication with the DB is through ODBC connections. The system must provide SQL data table definintions to be provided to the company DBA for setup.~~*

*~~A key point to remember is that you do NOT want to specify software here that you think would be good to use. This is only for~~* ***~~customer-specified systems~~*** *~~that you~~* ***~~have~~*** *~~to interact with. Choosing SQL Server 7 as a DB without a customer requirement is a Design choice, not a requirement. This is a subtle but important point to writing good requirements and not over-constraining the design.~~*

### ~~2.1.2.4 Communications Interfaces~~

*~~Specify the various interfaces to communications such as local network protocols, etc. These are protocols you will need to directly interact with. If you happen to use web services transparently to your application then do not list it here. If you are using a custom protocol to communicate between systems, then document that protocol here so designers know what to design. If it is a standard protocol, you can reference an existing document or RFC.~~*

~~The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems.~~

### ~~2.1.6 Memory Constraints~~

*~~Specify any applicable characteristics and limits on primary and secondary memory~~*~~.~~ *~~Don’t just make up something here. If all the customer’s machines have only 128K of RAM, then your target design has got to come in under 128K so there is an actual requirement. You could also cite market research here for shrink-wrap type applications “Focus groups have determined that our target market has between 256-512M of RAM, therefore the design footprint should not exceed 256M.” If there are no memory constraints, so state.~~*

### ~~2.1.7 Operations~~

*~~Specify the normal and special operations required by the user such as:~~*

1. *~~The various modes of operations in the user organization~~*
2. *~~Periods of interactive operations and periods of unattended operations~~*
3. *~~Data processing support functions~~*
4. *~~Backup and recovery operations~~*

*~~(Note: This is sometimes specified as part of the User Interfaces section.) If you separate this from the UI stuff earlier, then cover business process type stuff that would impact the design. For instance, if the company brings all their systems down at midnight for data backup that might impact the design. These are all the work tasks that impact the design of an application, but which might not be located in software.~~*

### ~~2.1.8 Site Adaptation Requirements~~

*~~In this section:~~*

1. *~~Define the requirements for any data or initialization sequences that are specific to a given site, mission, or operational mode~~*
2. *~~Specify the site or mission-related features that should be modified to adapt the software to a particular installation~~*

*~~If any modifications to the customer’s work area would be required by your system, then document that here. For instance, “A 100Kw backup generator and 10000 BTU air conditioning system must be installed at the user site prior to software installation”.~~*

*~~This could also be software-specific like, “New data tables created for this system must be installed on the company’s existing DB server and populated prior to system activation.” Any equipment the customer would need to buy or any software setup that needs to be done so that your system will install and operate correctly should be documented here.~~*

## ~~2.2.Product functions~~

*~~Provide a summary of the major functions that the software will perform. Sometimes the function summary that is necessary for this part can be taken directly from the section of the higher-level specification (if one exists) that allocates particular functions to the software product.~~*

*~~For clarity:~~*

1. *~~The functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the first time.~~*
2. *~~Textual or graphic methods can be used to show the different functions and their relationships. Such a diagram is not intended to show a design of a product but simply shows the logical relationships among variables.~~*

*~~AH, Finally the real meat of section 2. This describes the functionality of the system in the language of the customer. What specifically does the system that will be designed have to do? Drawings are good, but remember this is a description of what the system needs to do, not how you are going to build it. (That comes in the design document).~~*

~~Nowadays blogging is very popular.~~

~~The blog :~~

~~- a mainstream source of information~~

~~- keeps clients and customers up to date~~

~~- visitors can comment and interact each other~~

~~- a way to earn money~~

* ~~Users should be able to register through their already existing accounts.~~
* ~~They should be able to share snaps/videos/snaps.~~
* ~~People should be able to like and comment on any post. One person can follow another person who share common interests and likings which would enable them to find mates apart from their usual friend circle.~~
* ~~Each user can have his/her profile picture, status- в нашия слуай не може~~
* ~~People can post mysteries and other people can solve the mysteries.~~
* ~~Users will get points for the popularity of their posts/the number of mysteries they solve.- не може в нашия случай~~
* *~~Add own funny commentary on any video~~*
* *~~Post any questions regarding their interests and people can answer.~~*

## 2.3.User characteristics

*~~Describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise. Do not state specific requirements but rather provide the reasons why certain specific requirements are later specified in section 3.~~*

*~~What is it about your potential user base that will impact the design? Their experience and comfort with technology will drive UI design. Other characteristics might actually influence internal design of the system~~*

There are two types of users that interact with the system: users of the blog and administrators. Each of these two types of users has different use of the system so each of them has their own requirements.

The administrators also only interact with the blog portal. They are managing the overall system so there is no incorrect information within it.

## ~~2.4.Constraints~~

*~~Provide a general description of any other items that will limit the developer's options. These can include:~~*

*~~(1) Regulatory policies~~*

*~~(2) Hardware limitations (for example, signal timing requirements)~~*

*~~(3) Interface to other applications~~*

*~~(4) Parallel operation~~*

*~~(5) Audit functions~~*

*~~(6) Control functions~~*

*~~(7) Higher-order language requirements~~*

1. *~~Signal handshake protocols (for example, XON-XOFF, ACK-NACK)~~*
2. *~~Reliability requirements~~*

*~~(10) Criticality of the application~~*

*~~(11) Safety and security considerations~~*

*~~This section captures non-functional requirements in the customers language. A more formal presentation of these will occur in section 3~~*

~~The Internet connection is a constraint for the blog. Since the blog fetches data from the database over the Internet, it is crucial that there is an Internet connection for the blog to function.~~

~~When the blog portal is getting update from the administrator it might be not working for a while.~~

~~There is no special requirement for kind of use language. Only there is a limit for use of word in unpropriate language.~~

## ~~2.5. Assumptions and dependencies~~

*~~List each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system would be available on the hardware designated for the software product. If, in fact, the operating system were not available, the SRS would then have to change accordingly.~~*

*~~This section is catch-all for everything else that might influence the design of the system and that did not fit in any of the categories above.~~*

## ~~2.6.Apportioning of requirements~~

*~~Identify requirements that may be delayed until future versions of the system. After you look at the project plan and hours available, you may realize that you just cannot get everything done. This section divides the requirements into different sections for development and delivery. Remember to check with the customer – they should prioritize the requirements and decide what does and does not get done. This can also be useful if you are using an iterative life cycle model to specify which requirements will map to which interation.~~*

~~In the case that the project is delayed, there are some requirements that could be transferred to the next version of the blog.~~

# ~~3.Specific requirements~~

*~~This section contains all the software requirements at a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should include at a minimum a description of every input (stimulus) into the system, every output (response) from the system and all functions performed by the system in response to an input or in support of an output. The following principles apply:~~*

1. *~~Specific requirements should be stated with all the characteristics of a good SRS~~*
2. *~~correct~~*
3. *~~unambiguous~~*
4. *~~complete~~*
5. *~~consistent~~*
6. *~~ranked for importance and/or stability~~*
7. *~~verifiable~~*
8. *~~modifiable~~*
9. *~~traceable~~*
10. *~~Specific requirements should be cross-referenced to earlier documents that relate~~*
11. *~~All requirements should be uniquely identifiable (usually via numbering like 3.1.2.3)~~*
12. *~~Careful attention should be given to organizing the requirements to maximize readability (Several alternative organizations are given at end of document)~~*

*~~Before examining specific ways of organizing the requirements it is helpful to understand the various items that comprise requirements as described in the following subclasses. This section reiterates section 2, but is for developers not the customer. The customer buys in with section 2, the designers use section 3 to design and build the actual application.~~*

*~~Remember this is not design. Do not require specific software packages, etc unless the customer specifically requires them. Avoid over-constraining your design. Use proper terminology:~~*

*~~The system shall… A required, must have feature~~*

*~~The system should… A desired feature, but may be deferred til later~~*

*~~The system may… An optional, nice-to-have feature that may never make it to implementation.~~*

*~~Each requirement should be uniquely identified for traceability. Usually, they are numbered 3.1, 3.1.1, 3.1.2.1 etc. Each requirement should also be testable. Avoid imprecise statements like, “The system shall be easy to use” Well no kidding, what does that mean? Avoid “motherhood and apple pie” type statements, “The system shall be developed using good software engineering practice”~~*

*~~Avoid examples, This is a specification, a designer should be able to read this spec and build the system without bothering the customer again. Don’t say things like, “The system shall accept configuration information such as name and address.” The designer doesn’t know if that is the only two data elements or if there are 200. List every piece of information that is required so the designers can build the right UI and data tables.~~*

~~This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features. Each requirement is uniquely identified for traceability. Usually, they are numbered.~~

~~Specific requirementsare stated with all the characteristics of a good SRS~~

1. ~~correct~~
2. ~~unambiguous~~
3. ~~complete~~
4. ~~consistent~~
5. ~~ranked for importance and/or stability~~
6. ~~verifiable~~
7. ~~modifiable~~
8. ~~traceable~~

## ~~3.1.External interface Requirements~~

*~~This contains a detailed description of all inputs into and outputs from the software system. It complements the interface descriptions in section 2 but does not repeat information there. Remember section 2 presents information oriented to the customer/user while section 3 is oriented to the developer.~~*

*~~It contains both content and format as follows:~~*

1. *~~Name of item~~*
2. *~~Description of purpose~~*
3. *~~Source of input or destination of output~~*
4. *~~Valid range, accuracy and/or tolerance~~*
5. *~~Units of measure~~*
6. *~~Timing~~*
7. *~~Relationships to other inputs/outputs~~*
8. *~~Screen formats/organization~~*
9. *~~Window formats/organization~~*
10. *~~Data formats~~*
11. *~~Command formats~~*
12. *~~End messages~~*

~~This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.~~

~~There are information about Name of item, Description of items, Screen formats/organization, Window formats/organization, Data formats,Comma,End messages~~

## 4.Functional requirements

~~This section includes the requirements that specify all the fundamental actions of the software system. Functional requirements define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs.~~

1. ~~Validity checks on the inputs~~
2. ~~Exact sequence of operations~~
3. ~~Responses to abnormal situation, including~~
4. ~~Overflow~~
5. ~~Communication facilities~~
6. ~~Error handling and recovery~~
7. ~~Effect of parameters~~
8. ~~Relationship of outputs to inputs, including~~
9. ~~Input/Output sequences~~
10. ~~Formulas for input to output conversion~~

### 4.1. Use~~r Class 1 - The User~~ Cases

#### 4.1.1 ~~Functional requirement~~ Use Case 1

**ID: ~~FR1~~ UC1**

TITLE: User registration ~~– blog URL~~

DESC: ~~Given that a user has enter in the HTTP address, then the user should be able to register through the blog page.~~When the user loads the internet address of the system he should be able to register his own profile from the menu for registration. The user must provide e-mail, full name, password and ~~have to confirm the created password in different field.~~password confirmation.

Guest user:

1. Loads the internet address of the system;
2. Goes to the Registration menu;
3. Enters e-mail, full name, password, password confirmation;
4. Submits the form

System:

1. Verifies the information for e-mail. If the e-mail format is not standard the system provides error message;
2. If the password and password confirmation are not the same the system provides error message;
3. Verifies general information and displays an error message if all mandatory fields are not filled;
4. Sends registered details to the database and gives acknowledgement

RAT: In order for a user to register ~~on the blog~~.to the system.

DEP: Guest user

#### 4.1.2 ~~Functional requirement~~ Use Case 2

**ID: ~~FR2~~ UC2**

TITLE: User Log in ~~- Blog~~

DESC: ~~Given that a~~ When the user ~~has~~ is registered, ~~then the user~~ he/she should be able to log in to the ~~blog~~ system from the Log in menu. ~~The log-in information will be stored on the web browser and in the future the user should be logged in automatically.~~

The user must be redirected to the main page of the system.

Registered user:

1. Loads the internet address of the system;
2. Goes to the Log in menu;
3. Enters valid e-mail and password;
4. Submits the Log in form

System:

1. Verifies login information and displays an error message if user name and/or password are incorrect;
2. Redirects the user to the main page.

RAT: In order for a user to register on the blog.

DEP: Registered user

#### 4.1.3 ~~Functional requirement~~ Use Case 3

**ID: ~~FR3~~ UC3**

TITLE: Retrieve password

DESC: ~~Given that a user has registered, then the user should be able to retrieve his/her password by e-mail.~~ When registered user try to log in, but don`t remember his password, he must have the option to retrieve his password by providing the valid e-mail address.

Registered user:

1. Loads the internet address of the system;
2. Goes to the Log in menu;
3. Goes to the Forgot password menu;
4. Provides valid e-mail address
5. Submits the information

System:

1. Redirects the user to the Forgot password menu;
2. Provides e-mail address submission option;
3. Submits the e-mail id and provides the reset password link through mail.

RAT: In order for a user to retrieve his/her password.

DEP: ~~FR1~~ The user must be registered and not logged in

#### 4.1.4 Use Case 4

**ID: UC4**

TITLE: Profile page

DESC: When registered user wants to change the information for his profile, he can do this from the profile menu.

Logged user:

1. Goes to the profile menu;
2. Changes the personal information

System:

1. Redirects the user to the profile page
2. Actualize the database with new added personal information

RAT: In order for a user to change his/her profile information.

DEP: The user must be logged in

#### 4.1.5 Use Case 5

**ID: UC5**

TITLE: Profile page

DESC: When registered user wants to change the password, he can do this from the profile menu.

Logged user:

1. Loads the internet address of the system;
2. Goes to the Log in menu;
3. Provides correct username and password;
4. Submits the information;
5. Goes to the profile menu;
6. Add the old password;
7. Add new password
8. Confirm new password
9. Save changes

System:

1. Verifies the username and password;
2. Redirects the user to the main page;
3. Redirects the user to the profile page;
4. Verifies that the new password and the password confirmation are equal;
5. Actualize the database with new password;

RAT: In order for a user to change his/her password.

DEP: The user must be logged in

#### 4.1.6 ~~Functional requirement~~ Use Case 1.6

**ID: ~~FR4~~ UC6**

TITLE: ~~Blog –~~ Create Post

DESC: ~~Given that a user is logged in to the blog, then the first page that is shown should be the create. The user should be able to write posts.~~ When the user is logged, from the main page he can choose Create menu, where he can create his post to the system.

Logged user:

1. Goes to the Create menu;
2. Add the title for the post;
3. Add the content of the post;
4. Click Create to publish the post;

System:

1. Redirects the user to the creation of post page;
2. Actualize the database with added title;
3. Actualize the database with the added content;
4. Visualize the post on the main page.

RAT: ~~In order for a user to search for a restaurant.~~ In order for a user to create post

DEP: The user must be logged in

#### 4.1.7 ~~Functional requirement~~ Use Case 1.7

**ID: ~~FR5~~ UC7**

TITLE: ~~Blog - Profile page~~ Edit/Delete Post

DESC: ~~On the blog, a user should have a profile page. On the profile page a user can edit his/her information, which includes the password.~~ When the user click on the text of his/her published post, the system redirects him/her to page, where the user can edit or delete his/her post.

Logged user:

1. Click on the text of the post;
2. Changes the title of the post;
3. Changes the content of the post;
4. Click on the button Edit/Click on the button Delete.

System:

1. Redirects the user to the creation of post page;
2. Actualize the database with new added title;
3. Actualize the database with the new added content;
4. Visualize the post on the main page.

RAT: In order for a user to ~~have a profile page on the blog.~~ edit or delete his/her post

DEP: The user must be logged in

#### 4.1.8 Use Case 1.8

**ID: UC8**

TITLE: Review blog posts

DESC: The user can review the post in the blog made by another user, but he/she must not have the permissions to edit or delete reviewed post.

Logged user:

1. Click on the text of the post;
2. Click on the button Edit/Delete.

System:

1. Redirects the user to the creation of post page;
2. The system returns error message with information,that the user don`t have permissions for this action.

RAT: In order for a user to review the others posts.

DEP: The user must be logged in

#### 4.1.9 Use Case 1.9

**ID: UC9**

TITLE: Log out

DESC: The user can log out from the system by clicking the Log off button.

Logged user:

1. Click on the Logoff button;

System:

1. Terminates the session of the user;
2. Redirects the user to the main page.

RAT: In order for a user to log out from the system

DEP: The user must be logged in

4.2 Non functional requirements

4.2.1 Usability Requirements   
  
•    Colors what we use in this web system design must be attractive.  
•    Easy navigations are preferable to do any task.  
•    Multiple flows (ways) are preferable to do any task.  
•    Home page should be centralized system (Screen/Window) to go to any feature and to get any result.  
•    The facility to return to Home page from any page should available.  
  
4.2.2    Performance Requirements   
  
•    It has to load, with in the industry standard time.  
•    It has to support up to 1 000 concurrent users.  
•    Response time should be 4 seconds

4.2.3    Compatibility Requirements

•    It should support all types of hardware versions, operating systems and browsers  
  
4.2.4    Operating Systems   
  
•    Operating systems -Wndows98 and upper Versions (EX: Win98, windows 2000 prof, XP Vista and win NT Server,windows 2000 server, 2003 server and windows 2008 server)  
   
4.2.5    Browsers   
  
•    IE 6.0 and higher versions , Latest versions of Mozilla Firefox, Opera and Google Chrome  
  
4.2.6    Security Requirements  
  
•    Data should be encrypted during Login and while performing publishing.  
•    Server on which database resides must have its own security to prevent unauthorized access. It should be accessed only to the privileged users  
•    Server on which application resides must have its own security. Authorization and Authentications checks should be made.

#### ~~4.2. Functional requirement~~

#### ~~4.2.1 Functional requirement 2.1~~

**~~ID: FR2.1~~**

**~~Feature: Create an account~~**

~~In order to create an account~~

~~A blogger~~

~~Should register on the blog register page~~

**~~Scenario: Required information for registration~~**

~~When the blogger registers on the blog by providing~~

~~Full Name~~

~~And e-mail address~~

~~And password~~

~~Then the blogger should be able to apply for verification~~

**~~Scenario: Full information for registration~~**

~~Given the blogger to create an account And the blogger does not have an account~~

~~When the blogger registers on the blog by providing~~

~~full name~~

~~And password~~

~~And e-mail address~~

~~Then the blogger should be able to apply for verification~~

**~~Scenario: Confirmed registration~~**

~~Given the blogger has applied for verification~~

~~And has not succssed to log in after registration~~

~~When the blogger succssed to log in after registration~~

~~Then the blogger should be able to log in~~

#### ~~4.2.2 Functional requirement 2.2~~

**~~ID: FR2.2~~**

**~~Feature: Blogger log-in~~**

~~In order to use the system~~

~~A blogger~~

~~should be logged in to the blog~~

**~~Scenario: Successful log-in~~**

~~Given the restaurant owner wants to log in~~

~~When the blogger logs in with his/her account~~

~~Then the blogger should be logged in as a user\blogger.~~

**~~Scenario: Retrieve password---it t is bug, how can user received new password~~**

~~Given the blogger wants to log in~~

~~And has lost the password~~

~~When the blogger enters his/her email address in the “Retrieve password” form And submits the form (it’s missing similar form)~~

~~Then the blogger should receive an email containing the password!!!!!!~~

#### ~~4.2.3 Functional requirement 2.3~~

**~~ID: FR2.3~~**

**~~Feature: Manage information~~**

~~In order to manage information~~

~~A user/blogger~~

~~Should be logged in to the blog -site~~

**~~Scenario: Create information~~**

~~Given the user/blogger is logged in~~

~~And information could be writen (title+content)~~

~~New informatio should exist in the blog, after saveing it~~

**~~Scenario: Deleting information~~**

~~Given the user/blogger is logged in~~

~~And information exists (title+content)~~

~~When the user/blogger deletes information~~

~~Then the information should be deleted~~

**~~Scenario: Editing information~~**

~~Given the user/blogger is logged in~~

~~And information exists~~

~~When the user/blogger edits information~~

~~Then the information should be edited~~

### ~~4.3 User Class 3 - Administrator~~

#### ~~4.3.1 Functional requirement 3.1~~

**~~ID: FR3.1~~**

**~~Feature: Administrator log in~~**

~~In order to administer the system~~

~~An administrator~~

~~Should be logged in to the blog portal~~

**~~Scenario: Successful log-in~~**

~~Given the administrator wants to log in~~

~~When the administrator logs in with an administrator account Then the administrator should be logged in as an administrator~~

#### ~~4.3.2 Functional requirement 3.2~~

**~~ID: FR3.2~~**

**~~Feature: Verify user/blogger~~**

~~In order to allow a user/blogger to use the system~~

~~An administrator~~

~~Should be able to verify the user/blogger~~

**~~Scenario: Verify a use/blogger~~**

~~Given the administrator is logged in~~

~~When the administrator verifies a user/blogger~~

~~Then the user/blogger should be able to log in~~

**~~Scenario: Reject a user/blogger~~**

~~Given the administrator is logged in~~

~~When the administrator rejects a user/blogger~~

~~Then the user/blogger should not be able to log in~~

~~And information about a blog exists~~

~~When the administrator deletes the information~~

~~Then the information about the blog should be deleted~~

**~~Scenario: Edit blog information~~**

~~Given the administrator is logged in~~

~~And information about a blog exists~~

~~When the administrator edits the information~~

~~Then the information about the blog should be edited~~

#### ~~4.3.3 Functional requirement 3.3~~

**~~ID: FR3.3~~**

**~~Feature: Manage users~~**

~~In order to keep track of the users~~

~~An administrator~~

~~Should be able to manage the users~~

**~~Scenario: Edit an existing user’s information~~**

~~Given the administrator is logged in~~

~~When the administrator edits an existing user~~

~~Then the user information should be updated~~

**~~Scenario: Delete/Inactivate an existing user~~**

~~Given the administrator is logged in~~

~~When the administrator deletes an existing user~~

~~Then the user should be deleted~~

#### ~~Functional requirement 3.4~~

## 4.3.Performance requirements

*~~This subsection specifies both the static and the dynamic numerical requirements placed on the software or on human interaction with the software, as a whole. Static numerical requirements may include:~~*

*~~(a) The number of terminals to be supported~~*

*~~(b) The number of simultaneous users to be supported~~*

*~~(c) Amount and type of information to be handled~~*

*~~Static numerical requirements are sometimes identified under a separate section entitled capacity.~~*

*~~Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions.~~*

*~~All of these requirements should be stated in measurable terms.~~*

*~~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.~~*

*~~(Note: Numerical limits applied to one specific function are normally specified as part of the processing subparagraph description of that function.)~~*

~~The requirements in this section provide a detailed specification of the user interaction with the software and measurements placed on the system performance.~~

~~This subsection specifies both the static and the dynamic numerical requirements placed on the software or on human interaction with the software, as a whole. Static numerical requirements may include:~~

~~(a) The number of terminals to be supported~~

~~(b) The number of simultaneous users to be supported~~

~~(c) Amount and type of information to be handled~~

### ~~4.3.1 Usage of the result in the list view~~

**~~ID: QR3.1~~**

~~TITLE: Usage of the result in the list view-every created items in front page of blog~~

~~DESC: The results displayed in the list view should be user easy to understand. Selecting an~~

~~element in the result list should only take one click.~~

~~RAT: In order to for a user to use the list view easily.~~

~~DEP: none~~

### ~~4.3.2 System dependability~~

**~~ID: QR3.2~~**

~~TAG: System Dependability~~

~~GIST: The fault tolerance of the system.~~

~~SCALE: If the system loses the connection to the Internet or the system gets some strange input, the user should be informed.~~

~~METER: Measurements obtained from 1000 hours of usage during testing.~~

~~MUST: 100% of the time.~~

## ~~4.4 Design constraints~~

~~This section includes the design constraints on the software caused by the hardware.~~

## ~~Logical Database Requirements~~

*~~This section specifies the logical requirements for any information that is to be placed into a database. This may include:~~*

1. *~~Types of information used by various functions~~*
2. *~~Frequency of use~~*
3. *~~Accessing capabilities~~*
4. *~~Data entities and their relationships~~*
5. *~~Integrity constraints~~*
6. *~~Data retention requirements~~*

*~~If the customer provided you with data models, those can be presented here. ER diagrams (or static class diagrams) can be useful here to show complex data relationships. Remember a diagram is worth a thousand words of confusing text.~~*

### ~~4.4.1 Hard drive space-нашият блог не използва други приложения , а само интернет~~

**~~ID: QR4.1~~**

~~TAG: HardDriveSpace~~

~~GIST: Hard drive space.~~

~~SCALE: The application’s need of hard drive space.~~

~~METER: MB.~~

~~MUST: No more than 20 MB.~~

~~PLAN: No more than 15 MB.~~

~~WISH: No more than 10 MB.~~

~~MB: DEFINED: Megabyte~~

# ~~4.5. Design Constraints~~

*~~Specify design constraints that can be imposed by other standards, hardware limitations, etc.~~*

### ~~4.5.1 Standards Compliance~~

*~~Specify the requirements derived from existing standards or regulations. They might include:~~*

*~~(1) Report format~~*

*~~(2) Data naming~~*

*~~(3) Accounting procedures~~*

*~~(4) 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.~~*

## ~~4.6. Software system attributes~~

~~The requirements in this section specify the required reliability, availability, security and maintainability of the software system. These are characteristics the system must possess, but that pervade (or cross-cut) the design. These requirements have to be testable just like the functional requirements.~~

*~~There are a number of attributes of software that can serve as requirements. It is important that required attributes by specified so that their achievement can be objectively verified. The following items provide a partial list of examples. These are also known as non-functional requirements or quality attributes.~~*

### ~~4.6.1 Reliability~~

*~~Specify the factors required to establish the required reliability of the software system at time of delivery. If you have MTBF requirements, express them here. This doesn’t refer to just having a program that does not crash. This has a specific engineering meaning.~~*

**~~ID: QR6.1~~**

~~TITLE: Internet Connection~~

~~DESC: The blog should be connected to the Internet.~~

~~RAT: In order for the blog-portal to communicate with the database.~~

~~DEP: none~~

### ~~4.6.2 Availability~~

*~~Specify the factors required to guarantee a defined availability level for the entire system such as checkpoint, recovery, and restart. This is somewhat related to reliability. Some systems run only infrequently on-demand (like MS Word). Some systems have to run 24/7 (like an e-commerce web site). The required availability will greatly impact the design.~~*

*~~What are the requirements for system recovery from a failure? “The system shall allow users to restart the application after failure with the loss of at most 12 characters of input”.~~*

### ~~4.6.3 Security~~

*~~Specify the factors that would protect the software/blog from accidental or malicious access, use, modification, destruction, or disclosure. Specific requirements in this area could include the need to:~~*

1. *~~Utilize certain cryptographic techniques~~*
2. *~~Keep specific log or history data sets~~*
3. *~~Assign certain functions to different modules~~*
4. *~~Restrict communications between some areas of the program~~*
5. *~~Check data integrity for critical variables~~*

**~~ID: QR6.3.1~~**

~~TAG: CommunicationSecurity~~

~~GIST: Security of the communication between the system and server.~~

~~SCALE: The messages should be encrypted for log-in communications, so others cannot get user-name and password from those messages.~~

~~METER: Attempts to get user-name and password through obtained messages on 1000 log-in session during testing.~~

~~MUST: 100% of the Communication Messages in the communication of a log-in session should be encrypted.~~

~~Communication Messages: Defined: Every exchanged of information between client and server.~~

**~~ID: QR6.3.2~~**

~~TAG: AdminLoginAccountSecurity~~

~~GIST: Security of accounts.~~

~~SCALE: If an admin tries to log in to the web portal with a non-existing account then the admin should not be logged in. The admin should be notified about log-in failure.~~

~~METER: 1000 attempts to log-in with a non-existing user account during testing.~~

~~MUST: 100% of the time.~~

**~~ID: QR6.3.3~~**

~~TAG: AdminAccountSecurity~~

~~GIST: Security of admin accounts.~~

~~SCALE: An admin and IP address should not be able to log-in to the web portal for a certain time period after three times of failed log-in attempts.~~

~~METER: 1000 attempts to log-in during the lock period after user account has been locked because of failed log-in attempts of three times.~~

~~MUST: The locking period should be half an hour, and during that period the log-in function is disabled.~~

**~~ID: QR6.3.4~~**

~~TAG: UserCreateAccountSecurity~~

~~GIST: The security of creating account for users of the system.~~

~~SCALE: If a user wants to create an account and the desired user name is occupied, the user should be asked to choose a different user name.~~

~~METER: Measurements obtained on 1000 hours of usage during testing.~~

~~MUST: 100% of the time.~~

### ~~4.6.4 Maintainability~~

*~~Specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices. If someone else will maintain the system~~*

**~~ID: QR19~~**

~~TITLE: Application extendibility~~

~~DESC: The blog should be easy to extend. The code should be written in a way that it favors implementation of new functions.~~

~~RAT: In order for future functions to be implemented easily to the blog.~~

~~DEP: none~~

**~~ID: QR21~~**

~~TITLE: Blog testability~~

~~DESC: Test environments should be built for the blog to allow testing of the blog different functions.~~

~~RAT: In order to test the blog.~~

~~DEP: none~~

### ~~4.6.5 Portability~~

### ~~The Blog is a web site, where people can write down and read news, conversations, to seeother users opinion to some questions.~~

### ~~Bloggers is necessary to have internet connection and browser. Blog could be chech in every machine. It is not using special portable language or particular operating system.~~

### ~~Blog should have special quality characteristics like:~~

### ~~Correctness as to perpose of the blog~~

### ~~Efficiency- necessary quality and quantity of code~~

### ~~Reliability - program should be performs with high level of precision~~

### ~~Testability – during the test blog is working correctly~~

*~~Specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems. This may include:~~*

1. *~~Percentage of components with host-dependent code~~*
2. *~~Percentage of code that is host dependent~~*
3. *~~Use of a proven portable language~~*
4. *~~Use of a particular compiler or language subset~~*
5. *~~Use of a particular operating system~~*

*~~Once the relevant characteristics are selected, a subsection should be written for each, explaining the rationale for including this characteristic and how it will be tested and measured. A chart like this might be used to identify the key characteristics (rating them High or Medium), then identifying which are preferred when trading off design or implementation decisions (with the ID of the preferred one indicated in the chart to the right).~~*

*~~Definitions of the quality characteristics not defined in the paragraphs above follow.~~*

*~~• Correctness - extent to which program satisfies specifications, fulfills user’s mission objectives~~*

*~~• Efficiency - amount of computing resources and code required to perform function~~*

*~~• Flexibility - effort needed to modify operational program~~*

*~~• Interoperability - effort needed to couple one system with another~~*

*~~• Reliability - extent to which program performs with required precision~~*

*~~• Reusability - extent to which it can be reused in another application~~*

*~~• Testability - effort needed to test to ensure performs as intended~~*

*~~• Usability - effort required to learn, operate, prepare input, and interpret output~~*

## ~~4.7. Organizing the Specific Requirements~~

*~~For anything but trivial systems the detailed requirements tend to be extensive. For this reason, it is recommended that careful consideration be given to organizing these in a manner optimal for understanding. There is no one optimal organization for all systems.~~*

### ~~4.7.1 System Mode~~

*~~Some systems behave quite differently depending on the mode of operation. When organizing by mode there are two possible outlines. The choice depends on whether interfaces and performance are dependent on mode.~~*

~~При нас не зависи от режима.~~

### ~~4.7.2 User Class~~

*~~Some systems provide different sets of functions to different classes of users.~~*

### ~~4.7.3 Objects~~

*~~Objects are real-world entities that have a counterpart within the system. Associated with each object is a set of attributes and functions. 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.~~*

### ~~4.7.4 Feature~~

*~~A feature is an externally desired service by the system that may require a sequence of inputs to effect the desired result. Each feature is generally described in as sequence eof stimulus-response pairs.~~*

### ~~4.7.5 Stimulus~~

*~~Some systems can be best organized by describing their functions in terms of stimuli.~~*

### ~~4.7.6 Response~~

*~~Some systems can be best organized by describing their functions in support of the generation of a response.~~*

### ~~4.7.7 Functional Hierarchy~~

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

## ~~4.8 Additional Comments~~

*~~Whenever a new SRS is contemplated, more than one of the organizational techniques given in 3.7 may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.~~*

*~~Three 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; when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.~~*

*~~In any of the outlines below, those sections called “Functional Requirement i” may be described in native language, in pseudocode, in a system definition language, or in four subsections titled: Introduction, Inputs, Processing, Outputs.~~*

# ~~5.Change Management Process~~

*~~Identify the change management process to be used to identify, log, evaluate, and update the SRS to reflect changes in project scope and requirements. How are you going to control changes to the requirements. Can the customer just call up and ask for something new? Does your team have to reach consensus? How do changes to requirements get submitted to the team? Formally in writing, email or phone call?~~*

# ~~Document Approvals~~

*~~Identify the approvers of the SRS document. Approver name, signature, and date should be used.~~*

# ~~Supporting Information~~

*~~The supporting information makes the SRS easier to use. It includes:~~*

1. *~~Table of Contents~~*
2. *~~Index~~*
3. *~~Appendices~~*

*~~The Appendices are not always considered part of the actual requirements specification and are not always necessary. They may include:~~*

*~~(a) Sample I/O formats, descriptions of cost analysis studies, results of user surveys~~*

*~~(b) Supporting or background information that can help the readers of the SRS~~*

*~~(c) A description of the problems to be solved by the software~~*

*~~(d) Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements~~*

*~~When Appendices are included, the SRS should explicitly state whether or not the Appendices are to be considered part of the requirements.~~*

# ~~6. Prioritization and Release Plan-необходима ли е тази точка???~~

~~In order to get a view of how to divide the requirements into different releases and what requirements should be included in which release, a prioritization of the requirements is needed. This section discusses the choice of prioritization methods and gives a suggestion of how the release plan for these requirements could look like.~~

## ~~6.1 Choice of prioritization method~~

~~When prioritizing the requirements the ten most important ones were picked out first. This was done with a simple “1 to 10” ranking method, with one being “not important” and ten “very important”. Based on the elicitation meetings, and the perceived ideas of what was important to the different stakeholders, a number was set for each requirement. The numbers were then summed up for each requirement and the ten with the highest score were chosen to be prioritized with the cost value approach. The results, which are red-marked, can be seen in Appendix I and as shown, it turned out to be five functional requirements and five quality requirements. These requirements were then prioritized according to the cost value approach and the results can be viewed under Appendix II.~~

~~The remaining requirements were prioritized according to the “Five-Way Priority Scheme” as shown in Appendix III. This method was chosen since it gives the different stakeholders the same importance and has an enough wide range for determining which requirement is more important than the other [3].~~

~~However, in this prioritization process, the development team was not included as a stakeholder since the different features were not considered to be as important to them as for the other stakeholders.~~

~~Other methods for prioritization, such as the hundred-dollar test and the yes-no vote, were also considered. The hundred-dollar test is quite similar to the five-way priority scheme, since it also gives a wide range for ranking the requirements. However, it is more easily misused since someone could save all their money and put them on a requirement that they think is very important [3]. Others might not agree that this requirement is important but it might still get the most votes since one person cared about it [3].~~

~~The yes-no vote method might be fairly simple to carry out, however the range is too narrow. For instance, if two requirements are not very important it would be hard to determine which of those requirements that is more important than the other [3].~~

~~In conclusion, weighing the disadvantages and advantages of these methods against each other lead us to choose the five-way priority scheme.~~

## ~~6.2 Release Plan~~

~~The requirements were divided into three releases based on the prioritization and their dependencies. The three different releases were assembled so that each would work as a fully functional блог.~~

~~In the first release the requirements that build up the foundation of the application were included, together with the most highly prioritized requirements and their dependencies.~~

~~The second release also includes important requirements. However, these requirements are not vital for a functional blog. They are more suited to act as additional features that can contribute to making the software product more attractive.~~

~~The third release includes the requirements that can be afforded to discard if the project gets delayed or overruns the budget.~~

~~For further details about the release plan, see Appendix IV.~~