WeBlog Requirements Specification

Version 1.0

April 19, 2022

Table of Contents

CONFIDENTIAL

1.   Executive Summary

1.1   Project Overview

WeBlog is a social publishing platform, where anyone can share their ideas or give their insights on different published topics. The goal is to provide registered members with a web-app, that can help them launch their own blog, the contents of which, they can share with the users of this platform.

1.2   Purpose and Scope of this Specification

WeBlog is intended to be a platform that enables the management of all client bloggers, who will purchase a blog, as well as specifically designing and managing an individual blog as a product. Thus, this documentation intends to regulate the behavior amongst the administrators, bloggers interacting with the platform. This specification will contain the description of every use case and all the respective features, explained in details, avoiding so the ambiguity during development, as well as during usage of the platform.

2.   Product/Service Description

A blog is a web page, the contents of which are regularly updated by its owner. It is used as a personal webpage or for business marketing. A blog post enables your business to rank on the search engine for a variety of keywords. Additionally, a blog is a tool that allows you to engage more with an audience, either by analyzing how many readers share your blog posts on social, or by allowing readers to comment on your individual posts. Bloggers should not have to go through the hassle of hiring a web developer, come up with a theme template and wait for the development process. Blogging should be as easy as clicking a button. As creators of WeBlog, we aim to provide bloggers with a fast way to launch their blog and start sharing.

2.1   Product Context

This product is independent and self-contained, in that is does not rely or communicate with other APIs.

2.2   User Characteristics

This system will be used by three types of users: Administrator, Blogger-Client, Blog-Visitor.

1. Administrator
    - Login / Logout
    - Manage Bloggers (CRUD)
    - Manage Blog Visitors (CRUD)
    - Approve and activate blog
2. Blogger
    - Register
    - Request for creation of blog
    - Login / Logout
    - Manage Blog (CRUD)
    - Manage Posts (CRUD)
    - Manage Tags (CRUD)
    - Manage Comments (R-D)

3. Blog Visitor
    - Register
    - Login / Logout
    - Manage Comment (CRUD)
    - View Posts
    - Like / Unlike Comments

2.3 Assumptions

It is assumed that the selection of technologies (Spring Boot, JS, MySQL) will be optimal for the developments process of this platform.

It is assumed that users of this platform will be working on a computer that is connected to the Internet, in order to access the webpage.

2.4 Constraints

Describe any items that will constrain the design options, including

- 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

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_01	Administrator as a user of the system should be provided with a specific User Interface appropriate for his role.	When administrator logs in, he should be able to see a dashboard and pick from all the entities he manages, in order to view the tables of records.	1		
BR_02	Blogger as a user of the system should be provided with a specific User Interface appropriate for his role.	When a blogger first enters, he should be able to select a theme in order to request the creation of his blog, the proceed with the registration. When he gets his blog, the interface should make it possible to manage posts and tags.	1		
BR_03	Blog Visitor as a user of the system should be provided with a specific User Interface appropriate for his role.	When a blog visitor visits the homepage, he should be able to see a list of different blogs. To be an active part of a specific blog he should register and login. Then he can view posts, comment and like other visitors' comments.	1		
BR_04	The system should provide various templates for blog creation.	While the blogger is creating his blog page, he should be able to chose from a selection of different themes.	3		
BR_05	The platform should provide blog visitors with a like and unlike button.	A like button improved the interaction between blog visitors.	2		
BR_06	The system should enable visitors to tell that a comment has been edited.	An edited label over a comment provides readers with more information on the history of comments.	3		
BR_07	The system will provide administrators with tables to view and manage users.	Records of users should be displayed in a table, with options of editing and deleting.	1		

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
BR_08	Administrator will have access to records of requests for creation of blogs. There will be displayed an option to approve the request and activate blog.	In a table of requests by blogger-clients, there will be options to activate blogs as selected by admin.	1		

3.2   Non-Functional Requirements

–   **Product Requirements**

3.2.1.1 **User Interface Requirements**

Initially the blogger-client will see the homepage of the platform. Amongst other parts, a button will be display that will redirect the user to the page where the blog creation will happen. In order for the client to request for a blog to be created, he will choose a template and then register as a client. The registration form will include the necessary information to be completed by the client. On login the correct credentials should be provided.

As for the administrator, he will be entering the system and work with a dashboard. This part of the system will include the tables that will hold records of blogger-clients, blog visitors to specific blogs and options to manage these users. For each record of a blog creation request, the administrator will be provided with the option of editing the record to activated.

Blog visitors will be displayed a list of different blogs, from which they can select based on their interest. To help them pick, the blogs will be described by different category tags. After selecting a specific blog, the visitor must register in order to be able to interact. Upon registration, he will have to provide a few details to complete the form. If logged in, the visitor will be able to like, unlike comments or leave their own comments to posts.

3.2.1.2 **Usability**

•   The system should be user friendly for all types of users.

•   The blog creation should be as easy as clicking the button create.

•   The design of the user interface should provide a great and easy user experience.

3.2.1.3 **Efficiency**

3.2.1.3.1   Performance Requirements
--------------------------To do------------------------------

3.2.1.3.2   Space Requirements
-------------------------To do----------------------------

3.2.1.4 **Dependability**

Availability

- The system is available 24 hours a day, seven days a week, and can be accessed at any time.

- The system can be used regardless of the location.

- Users can use the system if they have an internet connection.

Monitoring

The activity on the system will be traced by logs which will be written to files.

Maintenance
Maintenance and the deployment of new features will be scheduled during the night, in order not to interrupt the activity of users, during the hours while the system is being used the most.

Integrity

3.2.1.5 **Security**

- Sensitive information such as passwords will be encrypted.
- Blog creation will be approved by administrators, after contact with client.
- Access to different functionalities of the system is provided based on user roles.

3.2.2   Organizational Requirements

3.2.2.1 **Environmental Requirements**

3.2.2.2 **Operational Requirements**

3.2.2.3 **Development Requirements**

3.2.3   External Requirements
–   Requirements which arise from factors which are external to the system and its development process e.g. interoperability requirements, legislative requirements, etc.

3.2.3.1 **Regulatory Requirements**

3.2.3.2 **Ethical Requirements**

3.2.3.3 **Legislative Requirements**
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

3.2.3.3.1   Accounting Requirements

3.2.3.3.2   Security Requirements

3.3   Domain Requirements

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

A.1.1.1.

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