Software Requirements Specification

for

Hostel Management System

Version 1.0 approved

Prepared by
Nilita Anil Kumar
Nandini AV
<organization>

17th january 2019

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

Name	Date	Reason For Changes	Version
nandini	17th jan 2019		1
nandini	5th feb		1.1

1.Introduction

1.1Purpose

The purpose of this document is to present a detailed description of the Web Publishing System. It 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 stimuli. This document is intended for both the stakeholders and the developers of the system and will be proposed to the Regional Historical Society for its approval.

1.2 Document Conventions

This document uses the following conventions.
DB Database
DDB Distributed Database
ER Entity Relationship
JS javascript

1.3 Intended Audience and Reading Suggestions

This project is a prototype for the hostel management system and it is restricted within the college premises. This has been implemented under the guidance of college professors. This project is useful for the hostel management and as well as to the students

1.4 Product Scope

The software product “Hostel Management System” will be an application that will be used for maintaining records in an organised manner and to replace old paper work system. This project aims at automating the hostel management for smooth working of the hostel by automating almost all the activities. Updations and modifications will be easily achievable and all the calculations and accounting work would be more accurate.

1.5References

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements

Specifications. IEEE Computer Society, 1998.

2.Overall Description

2.1Product Perspective

5.1.1.System Interface
The HMS is a complete web enabled system which can be accessed through web browser.
5.1.2.User Interface
The user interface is as follows:
Screen Name Description
Login Login into system as student or admin
Student Module: Profile Student can view and update personal details.
Student Module: Apply Room Student applies room by selecting the preferred room, semester.
Student Module: Status of Application Student can check their application status.
Student Module: View history Students can view history of past applications
Student Module: Change Password Student can change his/her password.
Student Module: Logout After the student is done using the system, he/she logs out.
Administrator Module: Add user Admin can add a new user by assigning new user name and passwords respectively.
Administrator Module: View Applicants Admin can view the applicants by selecting the year and semester.
Administrator Module: View Students Admin can view the
student's
details
Administrator Module: Change Password Admin can change his/her password.

2.2Product Functions

5.2.1.
User Functions
The administrator of HMS shall add new users to the system who is basically the student. After entering the information about the user, the system gives a unique username and password to the user.

The administrator shall view applicants and students, and change password.
5.2.2.
Student Functions
The student shall view and update their profiles.
The student shall apply a room.
The student shall view the status of application.
The student shall view their history and change their password.

2.3User Classes and Characteristics

The Administrator
This user has to have at least Window 7/Linux OS and Internet browsing skills for administrating HMS user profiles.
The Student
This user has to have at least Window 7/Linux OS and Internet browsing skills to use the system.

2.4Operating Environment

<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>

2.5Design and Implementation Constraints

<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer's organization will be responsible for maintaining the delivered software).>

2.6User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

2.7Assumptions and Dependencies

<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless

they are already documented elsewhere (for example, in the vision and scope document or the project plan).>

3.External Interface Requirements

3.1User Interfaces

The user interface is as follows:
Login Login into system as student or admin
Student Module:

- Profile Student can view and update personal details.
- Apply Room Student applies room by selecting the preferred room, semester.
- Status of Application Student can check their application status.
- View history Students can view history of past applications
- Change Password Student can change his/her password.
- Logout After the student is done using the system, he/she logs out.

Administrator Module:

- Add user Admin can add a new user by assigning new user name and passwords respectively.
- View Applicants Admin can view the applicants by selecting the year and semester.
- View Students Admin can view the Student's Details
- Change Password Admin can change his/her password.
- Add user Admin can add a new user by assigning new user name and passwords respectively.
- View Applicants Admin can view the applicants by selecting the year and semester.
- View Students Admin can view the student's Details.
- Change Password Admin can change his/her password.
- Add user Admin can add a new user by assigning new user name and passwords respectively.
- View Applicants Admin can view the applicants by selecting the year and semester.
- View Students Admin can view the student's Details.
- Change Password Admin can change his/her password.

3.2Hardware Interfaces

Client Side
Any Personal computer, which can support any 7-window or Windows environment with a mouse support, is acceptable.
Server Side
HMS will be run on a web server, which is installed into the school server.The school servers have requirements to operate PHP scripts (Apache Web server 1.3.2 with PHP 4.0
modules).

3.3Software Interfaces

Server Side

Apache Web server is installed and will enable HMS to interact with its users. PHP is a server-side scripting language, which will be used to code the HMS.

Client Side

On the client side the required software product is Internet Explorer/Google Chrome/Mozilla Firefox supporting at least HTML version 3.2, java enabled, and any operating system that can run the browsers.

3.4Communications Interfaces

The default communication protocol for data transmission between server and the client is Transmission Control Protocol/ Internet Protocol (TCP/IP). At the upper level Hyper Text Transfer Protocol (HTTP) will be used for communication between the web server and client.

4.System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:
REQ-2:

4.2System Feature 2 (and so on)

5.Other Nonfunctional Requirements

5.1Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

5.3Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.4Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

5.5Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

6.Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>