**Software Requirements Specification**

**for**

**Moodle Learning Management System**

**Version 1.0 approved**

**Prepared by**

**P S Lathashree**

**Prajna**

**B N Yashaswini**

**17-01-2019**

**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** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to present a detailed description of the Learning Management System. It will explain the purpose and features of the system, the interfaces and what the system can do. It also explains about the constraints under which it must operate and how the system would react to the external factors.

## Document Conventions

1. Admin - Refers to the superuser who owns the website.

2. Student - Student who enrolls himself in a course.

3. Parent - Parent of the respective student.

4. Mentor - Employed by admin to teach students

5. LMS - Learning Management System

## Intended Audience and Reading Suggestions

Developer - The developers would use this document to implement the functionalities of the system and to ensure the traceability of the software.

Tester - The testers would use this document to know the interface design and functionalities which would help them in testing the software accordingly.

User - The user would use this document to verify if the requirements specified in the document satisfies their needs.

## Product Scope

The project aims at improving the efficiency in the online education and reduces the complexities

involved to the maximum possible extent.

Home-schooled students, part-time workers, college/school going students, people who want to restart their career, if they want to do some courses outside their daily routine or curriculum,they need flexible hours to study. Online education is receiving a lot of attention these days due to flexibility of learning hours and the huge content of information available. This website can be used by parents, students and mentors. It aims at making online education flexible. So this system uses several programming and database techniques to elucidate the work involved in this process. As this is a matter of Security, the system has been carefully verified and validated in order to satisfy it.

The System provides an online interface to the parent and teacher where they can view Information about student’s performance. It provides a communication platform between the teacher, parent, student and the administrator.

## References

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

# Overall Description

## Product Perspective

*<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>*

## Product Functions

*<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>*

## User Classes and Characteristics

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

## Operating Environment

|  |  |
| --- | --- |
| Processor | INTEL CORE processor or any other of better performance |
| Operating System | WINDOWS, UBUNTU |
| Memory | 1GB RAM or more |
| Hard Disk space | Minimum 3GB for database usage in future |
| Database | Mysql |

## Design 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).>*

## User 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.>*

## Assumptions 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).>*

# External Interface Requirements

## User Interfaces

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

## Hardware Interfaces

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>*

## Software Interfaces

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

## Communications Interfaces

*<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>*

# 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.>*

## System 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:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

This software should perform the same way irrespective to its Operating System environments. Time taken for importing files and publishing the multimedia presentation should be minimum.

The video quality should be clear and good .The audio could be heard well. The video and audio of lectures should be synchronised well.

## Safety Requirements

This requirement does not apply for our software as this is can’t pose a threat in no way.

## Security Requirements

As all the operations are to be done within a single system security is not an issue for this software. HTTPS enables access to web application to secure access of confidential data (student information). Database Access There will be no external access to the database, except through the XML protocol. Administrators of the system will have full database administration rights and tutors/faculties may have access to a copy of parts of the VCS database, for editing purposes..

## Software Quality Attributes

Quality has a number of attributes some of the important attributes for this software are Portability: As this software is to work on multiple platforms portability is an essential attribute

User Training: We assume that the users already have some previous experience in working with similar software’s. So the users will not need any specific training for using this software. Testability: As a basic characteristic the software needs to be testable to ensure correctness.

## Business Rules

The LMS operations management team is responsible for ensuring that the LMS operates reliably, is managed in conformance with standards, and meets the needs of the organization. The operations management team works closely with the working groups and steering team, helping to surface issues that need attention.

LMS administrators are responsible for the accuracy and thoroughness of content configuration in the LMS. They must provide timely response to requests from content owners. LMS administrators must consistently implement LMS standards, conventions, policies, and processes.

# 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.>*