# **1. Introduction**

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.

## 1.1 Purpose

The purpose of this document is to present a detailed description of the “Syllabus Manager And Report Tracking” (SMART) software. 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 a customer for its approval.

## 1.2 Scope of Project

This “Syllabus Manager And Report Tracking” will be a local Windows Operating System based software which will maximize the instructor’s productivity by providing tools to assist in semi-automating and digitizing the syllabus planning and progress tracking process. By maximizing the instructor’s work efficiency and production the system will meet the instructor’s needs while remaining easy to understand and use.

More specifically, this system is designed to allow an instructor, manage and track his/her course syllabus completion plan for each of their class. The instructor first creates a local personal account with details like the classes and courses they take. Then the instructor is required to select working days for each of the classes and the syllabus to each of the courses. The software will then facilitate generation of a tentative syllabus completion plan using the inputs from the instructor and provide a mechanism to track their progress in each course for each class.

## 1.3 Glossary

### Table 1 - Definitions

|  |  |
| --- | --- |
|  |  |

## 1.4 References

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

## 1.5 Overview of Document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.

Both sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

# 

# 

# **2. Overall Description**

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and 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.

## 2.1 Product perspective

This system will consist of a single software.

The users first create their individual accounts storing all their course and class information. For each of their class and course the software maintains topics and subtopics retrieved from the input. Once that is done, the software facilitate in making tentative schedules that they desire to follow.

The users get progress reports of how they are doing in a particular course or class by giving syllabus realization details to the software.

The software generates informative graphs on both stages which can be used to further optimize & make changes to their syllabus completion planning process and report keeping.

## 2.2 Product functions

1. The software allows the users to make individual password protected accounts with information about their classes and courses.
2. There are three cases:

* users may teach multiple courses to same class.
* users may teach same course to multiple classes.
* users may teach multiple courses to multiple classes.

all of which can be easily accommodated and accounted for.

1. For confidentiality no one user can view or change the progress of other user.
2. Topic and subtopics names are shared among users undertaking the same courses. This will liberate them from re-entering those details; saving time and effort.
3. Progress tracking and report generation is possible at all stages.
4. Both soft-copies and hard-copies of the syllabus completion plan can be generated.

## 2.3 User characteristics

The user is expected to be a Windows OS Environment literate and be able to use selectors, buttons, input fields, menus, and similar tools.

## 2.4 Constraints

The main component of the project is the software application, which will be limited to the Windows operating system (specifically Windows 7 and above). The application is not resource or graphics intensive, so there are no practical hardware constraints. However the software will rely on several functionalities built into the Windows OS. Besides that, the application will also need other programming languages and libraries, namely - Python, wxPython & MySQL preinstalled and configured.

## 2.5 Assumptions and dependencies

The primary design constraint is the input mechanism. Since the application depends heavily on the data provided by the user such as the workdays and the topic/subtopic names it can become a daunting task to do and requires patience. Also if the inputs aren’t correct or change later on, extra steps need to be performed in order to incorporate those changes in the system.

## 2.6 Apportioning of requirements

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

# **3. Specific Requirements**

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

### 3.1.2 Hardware Interfaces

### 3.1.3 Software Interfaces

### 3.1.4 Communications Interfaces

## 3.2 Functional Requirements

### 3.2.1 <Functional Requirement or Feature #1>

3.2.1.1 Introduction

3.2.1.2 Inputs

3.2.1.3 Processing

3.2.1.4 Outputs

3.2.1.5 Error Handling

### 3.2.2 <Functional Requirement or Feature #2>

…

## 3.3 Use Cases

### 3.3.1 Use Case #1

### 3.3.2 Use Case #2

…

## 3.4 Classes / Objects

### 3.4.1 <Class / Object #1>

3.4.1.1 Attributes

3.4.1.2 Functions

<Reference to functional requirements and/or use cases>

### 3.4.2 <Class / Object #2>

…

## 3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

### 3.5.1 Performance

### 3.5.2 Reliability

### 3.5.3 Availability

### 3.5.4 Security

### 3.5.5 Maintainability

### 3.5.6 Portability

## 3.6 Inverse Requirements

State any \*useful\* inverse requirements.

## 3.7 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

## 3.8 Logical Database Requirements

Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc.

## 3.9 Other Requirements

Catchall section for any additional requirements.

# **4. Analysis Models**

List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.

## 4.1 Sequence Diagrams

## 4.3 Data Flow Diagrams (DFD)

## 4.2 State-Transition Diagrams (STD)

# **5. Change Management Process**

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.

# **A. Appendices**

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## A.1 Appendix 1

## A.2 Appendix 2