Software requirement specification (SRS) document template

Project name:	school management software
Date:	25/10/2013
Version:	1.0
By:	

Revision history

Version	Author	Verson description	Date completed

Review history

Reviewer	Version reviewed	Signature	Date

Approval history

Approver	Version approved	Signature	Date

Table of contents

1 Introduction

Describe the purpose of the document.

1.1 Product scope

List the benefits, objectives, and goals of the product.

The University Student Portal aims to provide an integrated platform for university students to access and manage their schedules, attendance, documents, and courses. This system will enhance the efficiency and convenience of administrative and academic tasks for both students and university staff.

1.2 Product value

Describe how the audience will find value in the product.

The intended audience, comprising university students, will find value in the product by gaining streamlined access to essential academic information and services. This includes efficient scheduling, tracking attendance, accessing course materials, and managing documents, thereby simplifying their academic journey.

1.3 Intended audience

Write who the product is intended to serve.

The product is intended to serve university students, faculty, and administrative staff. It is designed to meet the specific needs of students while offering administrative capabilities for university staff to manage course-related data.

1.4 Intended use

Describe how will the intended audience use this product.

-View and manage their class schedules.
- Keep track of their attendance in various courses.
- Access and download course materials and documents.

1.5 General description

Give a summary of the functions the software would perform and the features to be included.

- User-friendly interface for students and staff.
- Compatibility with various web browsers and devices.
- Integration with the university's existing database and systems for course information.
- Security measures to protect sensitive student data.

2 Functional requirements

List the design requirements, graphics requirements, operating system requirements, and constraints of the product.

Design Requirements:

-The user interface should follow a clean and intuitive design to ensure a positive user experience.
-Design should be responsive, adapting to different screen sizes and orientations (e.g., mobile, tablet, desktop).
-Ensure consistency in design elements, including color schemes, typography, and navigation menus.

Graphics Requirements:

-Utilize appropriate graphics and icons to enhance the user interface and provide visual cues.

-Ensure all graphics and images used are properly licensed and comply with copyright regulations.

Operating System Requirements:

The system should be accessible and fully functional on major operating systems, including but not limited to:
- **Windows (versions in use at the university)**
- **macOS (versions in use at the university)**
- **Linux (common distributions)**

-The web application should be cross-browser compatible, supporting popular web browsers such as:
Google Chrome
Mozilla Firefox
Apple Safari
Microsoft Edge
Constraints:
-The project should be completed within the allocated budget and timeline.
-The system should not compromise the privacy and security of student data in any way.

The development team should consider the limitations of the university's network and server resources.
These design requirements, graphics requirements, operating system requirements, and constraints will help ensure that the University Student Portal meets the necessary standards and restrictions while providing an efficient and user-friendly experience

3 External interface requirements

3.1 User interface requirements	Describe the logic behind the interactions between the users and the software (screen layouts, style guides, etc).

- The user interface should be intuitive, with easy navigation.
- Accessible on major web browsers.
- Support for mobile and desktop devices.

3.2 Hardware interface requirements	List the supported devices the software is intended to run on, the network requirements, and the communication protocols to be used.

- The system should run on standard hardware configurations.
- It should support hardware commonly used by students (e.g., laptops, smartphones, tablets).

3.3 Software interface requirements	Include the connections between your product and other software components, including frontend/backend framework, libraries, etc.

- Integration with the university's existing databases to access course and student data.
- Compatibility with standard software tools and platforms.

3.4 Communication interface requirements	List any requirements for the communication programs your product will use, like emails or embedded forms.

- The system should support communication through email for notifications.
- It should provide embedded forms for data submission and inquiries.

4 Non-functional requirements

4.1 Security	Include any privacy and data protection regulations that should be adhered to.

- Implement user authentication and authorization mechanisms to protect student data.
- Comply with data protection regulations and ensure data encryption.

4.2 Capacity	Describe the current and future storage needs of your software.

- The system should be able to handle concurrent user loads during peak times.
- Minimum hardware requirements for client devices should be documented.

4.3 Compatibility	List the minimum hardware requirements for your software.

Ensure compatibility with common web browsers (e.g., Chrome, Firefox, Safari).
- Compatibility with different operating systems (Windows, macOS, Linux).

4.4 Reliability	Calculate what the critical failure time of your product would be under normal usage.

- Define expected system uptime and recovery procedures for unexpected failures.

4.5 Scalability	Calculate the highest workloads under which your software will still perform as expected.

4.6 Maintainability	Describe how continuous integration should be used to deploy features and bug fixes quickly.

- Ensure ease of maintenance and updates for the system.
- Develop documentation for system administrators and maintainers.

4.7 Usability	Describe how easy it should be for end-users to use your software.

- The user interface should be user-friendly and intuitive.

4.8 Other	List any additional non-functional requirements.

5 Definitions and acronyms
