SRS

Software Requirements Specification (SRS)

Project Name: Student Exam Result Portal

Version: 1.0

Author: Laleet Krishna R

Date: 15 October 2025

---

1. Introduction

1.1 Purpose

The purpose of this document is to define the **software requirements** for the **Student Exam Result Portal**, a secure web-based system that allows students to view their exam results and administrators to manage student records. This system emphasizes **access control, secure session handling, and protection against common web vulnerabilities**. The intended audience includes the development team, project stakeholders, and QA engineers.

1.2 Scope

The Student Exam Result Portal will provide the following functionalities:

- Secure login/logout for students and administrators.
- Role-based access control: Students can view only their results, while admins can manage student data.
- Ability for admins to add, edit, or delete student exam results.
- Implementation of security mechanisms, including HTTPS, secure session management, input validation, and CSRF protection.
- Deployment via **Docker** for containerization and future scalability with Kubernetes.

1.3 Definitions, Acronyms, and Abbreviations

Term	Definition
SRS	Software Requirements Specification
HTTPS	Hypertext Transfer Protocol Secure
CSRF	Cross-Site Request Forgery
SQLi	SQL Injection
PHP	Hypertext Preprocessor

1.4 References

- PHP 8 Official Documentation: https://www.php.net/docs.php
- MySQL 8 Official Documentation: https://dev.mysql.com/doc/
- Docker Official Documentation: https://docs.docker.com/

---

2. Overall Description

2.1 Product Perspective

The Student Exam Result Portal is a **web-based application** that will interact with a MySQL database for storing user credentials and exam results. It will follow a **client-server architecture** where the PHP web server handles business logic and the MySQL database stores persistent data.

High-Level Architecture:

- Client (Browser) → PHP Web Server → MySQL Database

2.2 Product Functions

1. **Student Functions:**
    - Login and logout securely.
    - View only their personal exam results.
2. **Admin Functions:**
    - Login and logout securely.

- Add, edit, and delete student records and exam results.

2.3 User Characteristics

- **Students:** Moderate technical knowledge, require intuitive and responsive interface.
- **Administrators:** Familiar with basic web applications and data management.

2.4 Operating Environment

- Windows 10/11 or Linux server
- PHP 8+, MySQL 8+
- Docker Desktop (Windows) or Docker Engine (Linux)
- Modern web browsers (Chrome, Firefox, Edge)

2.5 Constraints

- Must be developed using **PHP and MySQL**.
- Must be deployable using Docker.
- All code must follow **secure coding standards**.

2.6 Assumptions and Dependencies

- Users have internet access and a compatible web browser.
- Admin credentials are securely managed.
- HTTPS certificates are configured on the deployment server.

---

3. Specific Requirements

3.1 Functional Requirements

ID	Requirement	Description
FR1	Secure Student Login/Logout	Students can log in and log out securely; sessions are managed and regenerated upon login.

ID	Requirement	Description
FR2	Secure Admin Login/Logout	Admins can log in and log out securely; sessions include timeout and secure cookies.
FR3	View Exam Results	Students can view only their results; system ensures proper access control.
FR4	Manage Student Data	Admins can add, edit, or delete student records and exam results.
FR5	Input Validation	All user inputs are validated and sanitized to prevent SQL Injection and XSS.
FR6	Audit Logging	Admin actions are logged with timestamp and user ID for accountability.

3.2 Non-Functional Requirements

ID	Requirement	Description
NFR1	Security	Passwords stored using bcrypt; HTTPS used for all communication.
NFR2	Session Management	Session regeneration, timeout, secure cookies (`HttpOnly`, `Secure`, `SameSite`).
NFR3	Performance	Page load time within 2 seconds for up to 100 concurrent users.
NFR4	Reliability	System should be available 99% of the time; Docker ensures portability.
NFR5	Maintainability	Code must follow modular design; functions separated by responsibility.

3.3 External Interface Requirements

User Interfaces:

- Login page, student dashboard, admin dashboard, results page.
- Responsive design for desktop and tablet.

Hardware Interfaces:

- Standard server hardware capable of running Docker containers.

Software Interfaces:

- PHP 8+ runtime
- MySQL 8+ database
- Docker and Docker Compose

Communication Interfaces:

- HTTPS for all client-server communication.

---

4. Use Cases

Use Case 1: Student Login

- **Actor:** Student
- **Precondition:** Student is registered in the database.
- **Flow:**
  1. Student enters username and password.
  2. System validates credentials.
  3. If valid, system starts secure session.
  4. Redirect to student dashboard.

Use Case 2: Admin Upload Results

- **Actor:** Admin
- **Precondition:** Admin is logged in.
- **Flow:**
  1. Admin selects “Upload Results.”
  2. Admin inputs or uploads results.
  3. System validates input and updates database.
  4. Logs action with timestamp and admin ID.

---

5. System Features

- **Access Control:** Role-based, ensuring students cannot access admin functions.
- **Secure Session Handling:** Sessions regenerated on login, timeout after inactivity.
- **Data Integrity:** Input validation and database constraints prevent unauthorized modification.
- **Audit Logging:** Tracks admin actions for accountability.

---

6. Appendix

- **Sample User Stories:**
  - *As a student, I want to securely log in and view my results so that my privacy is maintained.*
  - *As an admin, I want to manage student data securely so that the portal stays accurate.*
- **Future Enhancements:**
  - Add 2FA for admins.
  - Integrate email notifications for students when results are updated.