Software Requirements Specification (SRS)

Student Registration & Payment Tracking System

Document Information

- **Project Name**: Student Registration & Payment Tracking System
- **Version**: 1.0
- **Date**: August 2025
- **Prepared by**: Development Team
- **Approved by**: Mr. Thilina / Mr. Sandaruwan

---

1. INTRODUCTION

1.1 Purpose

This Software Requirements Specification (SRS) document describes the functional and non-functional requirements for the Student Registration & Payment Tracking System. This web-based application will enable educational institutions to manage student registrations, track payments, and generate reports efficiently.

1.2 Scope

The system will provide:

- Student registration with QR code generation
- Payment tracking and recording
- Automated notification system for overdue payments
- Comprehensive reporting for administrators
- Responsive web interface for desktop and mobile devices

1.3 Definitions and Abbreviations

- **SRS**: Software Requirements Specification
- **QR**: Quick Response (code)
- **SMS**: Short Message Service
- **API**: Application Programming Interface
- **UI**: User Interface

- **Admin**: Administrator user with full system access
- **Receptionist**: User with registration and payment recording privileges

1.4 References

- IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications
- Project requirements document
- User feedback and stakeholder interviews

---

2. OVERALL DESCRIPTION

2.1 Product Perspective

The Student Registration & Payment Tracking System is a standalone web application that will replace manual student registration and payment tracking processes. The system consists of:

- Web-based frontend (React.js)
- Backend API server (Laravel)
- Database management system (MySQL/PostgreSQL)
- External integration with SMS/Email/WhatsApp services

2.2 Product Functions

- **F1**: Student Registration Management
- **F2**: QR Code Generation and Distribution
- **F3**: Payment Recording and Tracking
- **F4**: Automated Notification System
- **F5**: Reporting and Analytics
- **F6**: User Authentication and Authorization
- **F7**: Mobile-Responsive Interface

2.3 User Characteristics

2.3.1 Administrator (Admin)

- **Experience Level**: Intermediate to Advanced computer user
- **Responsibilities**: System configuration, user management, report generation
- **Access Level**: Full system access

2.3.2 Receptionist

- **Experience Level**: Basic to Intermediate computer user
- **Responsibilities**: Student registration, payment recording, QR code scanning
- **Access Level**: Limited to registration and payment functions

2.3.3 Student/Parent

- **Experience Level**: Basic computer/mobile user
- **Responsibilities**: Receive QR codes and payment notifications
- **Access Level**: Read-only (receive notifications)

2.4 Constraints

- **Technical Constraints**: Must work on modern web browsers, mobile-responsive
- **Security Constraints**: User authentication required, data encryption
- **Performance Constraints**: System must handle concurrent users efficiently
- **Business Constraints**: Must integrate with existing SMS/Email services

2.5 Assumptions and Dependencies

- Reliable internet connectivity available
- SMS/Email/WhatsApp gateway services will be available
- Users have access to modern web browsers
- Mobile devices have camera capability for QR scanning

---

3. SPECIFIC REQUIREMENTS

3.1 FUNCTIONAL REQUIREMENTS

3.1.1 Student Registration (Epic 1)

FR-1.1: Add New Student Details

- **Description**: System shall allow receptionists to register new students with complete details
- **Priority**: High
- **Inputs**:
  - First Name (Text, Required, Max 100 characters)
  - Last Name (Text, Required, Max 100 characters)

  - Email Address (Email, Required, Unique)
  - Student Phone (Text, Required, Valid phone format)
  - Parent Phone (Text, Required, Valid phone format)
  - Course Selection (Dropdown, Required)
  - Date of Birth (Date, Required)
  - Address (Text, Optional, Max 500 characters)
- **Processing**:
  - Validate all required fields
  - Check email uniqueness
  - Generate unique registration number
  - Store in database with timestamp
- **Outputs**:
  - Success/Error message
  - Student registration confirmation
  - Generated registration number

FR-1.2: Generate Unique QR Code

- **Description**: System shall automatically generate unique QR code upon successful student registration
- **Priority**: High
- **Inputs**: Student registration number
- **Processing**:
  - Create QR code containing student registration number
  - Generate high-quality PNG image (200x200 pixels)
  - Store QR code reference in student record
- **Outputs**: Base64 encoded QR code image

FR-1.3: Send QR Code to Student/Parent

- **Description**: System shall send generated QR code via SMS/Email/WhatsApp
- **Priority**: Medium
- **Inputs**:
  - Student contact information

  - Generated QR code
  - Communication method preference
- **Processing**:
  - Format message with QR code attachment
  - Send via selected communication channel
  - Log delivery status
- **Outputs**:
  - Delivery confirmation
  - Notification record

3.1.2 Payment Tracking (Epic 2)

FR-2.1: Scan Student QR Code

- **Description**: System shall allow receptionists to scan student QR codes using device camera
- **Priority**: High
- **Inputs**: QR code scan from camera
- **Processing**:
  - Decode QR code data
  - Validate registration number
  - Retrieve student information from database
- **Outputs**:
  - Student details display
  - Current payment status
  - Payment history

FR-2.2: Record Payment

- **Description**: System shall allow receptionists to record student payments
- **Priority**: High
- **Inputs**:
  - Student ID (from QR scan)
  - Payment Amount (Decimal, Required, > 0)
  - Payment Date (Date, Required, Default: Today)
  - Payment Method (Dropdown: Cash/Card/Bank Transfer)

  - Receipt Number (Text, Optional)
  - Notes (Text, Optional, Max 200 characters)
- **Processing**:
  - Validate payment amount
  - Calculate outstanding balance
  - Update payment schedule
  - Generate receipt
- **Outputs**:
  - Payment confirmation
  - Updated balance information
  - Digital receipt

FR-2.3: View Payment History

- **Description**: System shall display complete payment history for each student
- **Priority**: Medium
- **Inputs**: Student ID
- **Processing**:
  - Retrieve all payment records for student
  - Calculate totals and outstanding amounts
  - Format for display
- **Outputs**:
  - Chronological payment list
  - Payment summary statistics
  - Outstanding balance

3.1.3 Notification System (Epic 3)

FR-3.1: Detect Overdue Payments

- **Description**: System shall automatically identify students with payments overdue by 2 weeks
- **Priority**: High
- **Inputs**: Current date, Payment schedules
- **Processing**:
  - Compare due dates with current date

  - Identify payments overdue by 14+ days
  - Flag students requiring notification
- **Outputs**: List of overdue payment cases

FR-3.2: Send Overdue Notifications

- **Description**: System shall automatically send notifications to students with overdue payments
- **Priority**: High
- **Inputs**:
  - Student contact information
  - Overdue amount and duration
  - Preferred communication method
- **Processing**:
  - Generate personalized notification message
  - Send via SMS/Email/WhatsApp
  - Log notification delivery
  - Schedule follow-up if needed
- **Outputs**:
  - Notification delivery status
  - Communication log entry

FR-3.3: Configure Notification Settings

- **Description**: System shall allow administrators to configure notification parameters
- **Priority**: Low
- **Inputs**:
  - Overdue threshold (days)
  - Notification frequency
  - Message templates
  - Communication method preferences
- **Processing**: Store configuration in system settings
- **Outputs**: Updated notification configuration

3.1.4 Reporting (Epic 4)

FR-4.1: Generate Course-wise Reports

- **Description**: System shall generate comprehensive reports organized by course
- **Priority**: High
- **Inputs**:
  - Date range (optional)
  - Course selection (optional)
  - Report type selection
- **Processing**:
  - Query database for relevant records
  - Calculate statistics and summaries
  - Format data for presentation
- **Outputs**:
  - Student registration summary by course
  - Payment status breakdown
  - Visual charts and graphs

FR-4.2: Payment Status Reports

- **Description**: System shall generate detailed payment status reports
- **Priority**: High
- **Inputs**: Filter criteria (course, date range, status)
- **Processing**:
  - Categorize payments (Paid/Pending/Overdue)
  - Calculate financial summaries
  - Identify trends and patterns
- **Outputs**:
  - Payment status dashboard
  - Financial summary reports
  - Exportable data (PDF/Excel)

FR-4.3: Overdue Cases Report

- **Description**: System shall generate specific reports for overdue payment cases
- **Priority**: Medium

- **Inputs**: Overdue threshold, Course filter
- **Processing**:
  - Identify all overdue cases
  - Calculate overdue amounts and durations
  - Rank by severity/amount
- **Outputs**:
  - Prioritized overdue cases list
  - Recovery action recommendations
  - Contact information for follow-up

3.1.5 User Management

FR-5.1: User Authentication

- **Description**: System shall authenticate users before granting access
- **Priority**: High
- **Inputs**: Username/Email and Password
- **Processing**:
  - Validate credentials against database
  - Check user status (active/inactive)
  - Generate session token
- **Outputs**:
  - Authentication success/failure
  - User session establishment
  - Role-based access permissions

FR-5.2: Role-Based Access Control

- **Description**: System shall enforce different access levels based on user roles
- **Priority**: High
- **Processing**:
  - Admin: Full system access
  - Receptionist: Registration and payment functions only
  - Restrict sensitive operations by role
- **Outputs**: Customized interface based on user role

3.2 NON-FUNCTIONAL REQUIREMENTS

3.2.1 Performance Requirements

- **NFR-1**: System shall respond to user requests within 3 seconds under normal load
- **NFR-2**: System shall support minimum 50 concurrent users
- **NFR-3**: Database queries shall execute within 1 second for standard operations
- **NFR-4**: QR code generation shall complete within 2 seconds

3.2.2 Security Requirements

- **NFR-5**: All user passwords shall be encrypted using industry-standard hashing
- **NFR-6**: System shall implement HTTPS for all communications
- **NFR-7**: Session timeout shall occur after 30 minutes of inactivity
- **NFR-8**: All financial transactions shall be logged for audit purposes

3.2.3 Usability Requirements

- **NFR-9**: System shall be fully responsive on devices with screen sizes from 320px to 1920px
- **NFR-10**: Interface shall be intuitive for users with basic computer skills
- **NFR-11**: System shall provide clear error messages and user guidance
- **NFR-12**: QR code scanning shall work reliably in various lighting conditions

3.2.4 Reliability Requirements

- **NFR-13**: System uptime shall be minimum 99% during business hours
- **NFR-14**: Data backup shall be performed automatically daily
- **NFR-15**: System shall recover from failures within 15 minutes
- **NFR-16**: No data loss shall occur during normal system operations

3.2.5 Compatibility Requirements

- **NFR-17**: System shall work on Chrome, Firefox, Safari, and Edge browsers
- **NFR-18**: Mobile interface shall function on iOS and Android devices
- **NFR-19**: System shall integrate with popular SMS/Email gateway services
- **NFR-20**: Database shall be compatible with MySQL 8.0+ or PostgreSQL 12+

---

4. SYSTEM ARCHITECTURE

4.1 Technology Stack

- **Frontend**: React.js 18+ with responsive design
- **Backend**: Laravel 10+ (PHP 8.1+)
- **Database**: MySQL 8.0 or PostgreSQL 12+
- **Authentication**: Laravel Sanctum
- **QR Code**: SimpleSoftwareIO/simple-qrcode
- **Communication**: Third-party SMS/Email/WhatsApp APIs

4.2 System Components

- **User Interface Layer**: React components, responsive design
- **API Layer**: RESTful APIs built with Laravel
- **Business Logic Layer**: Laravel controllers and services
- **Data Access Layer**: Eloquent ORM with database
- **External Integration Layer**: SMS/Email/WhatsApp services

---

5. USER INTERFACE REQUIREMENTS

5.1 General UI Requirements

- Clean, modern design following current web standards
- Consistent color scheme and typography
- Mobile-first responsive design approach
- Intuitive navigation with minimal clicks to complete tasks
- Clear visual feedback for user actions

5.2 Specific Interface Requirements

- **Login Page**: Simple form with username/password and role selection
- **Dashboard**: Quick access to common functions, recent activity summary
- **Registration Form**: Step-by-step student registration with validation
- **QR Scanner**: Full-screen camera interface with scanning guidance
- **Payment Recording**: Quick payment entry with student information display

- **Reports**: Interactive charts and tables with export options

---

6. ACCEPTANCE CRITERIA

6.1 Sprint 1 Acceptance Criteria

□ Receptionist can successfully register a new student

□ System generates unique registration number automatically

□ QR code is created and displayed upon registration

□ All required student information is captured and stored

□ Basic user authentication is functional

6.2 Sprint 2 Acceptance Criteria

□ QR code scanning works reliably on mobile and desktop cameras

□ Student information is displayed immediately after QR scan

□ Payment recording updates the database correctly

□ Payment history is accessible and accurate

□ Outstanding balance calculations are correct

6.3 Sprint 3 Acceptance Criteria

□ System automatically identifies overdue payments (14+ days)

□ Notifications are sent successfully via at least one channel (SMS/Email)

□ Notification delivery status is tracked and logged

□ Admin can configure notification settings

□ Notification content is personalized and clear

6.4 Sprint 4 Acceptance Criteria

□ Course-wise reports generate accurate data

□ Payment status reports show correct categorization

□ Reports can be exported to PDF/Excel format

□ Admin dashboard provides useful system overview

□ All user interface elements are responsive and functional

---

7. TESTING REQUIREMENTS

7.1 Testing Types Required

- **Unit Testing**: Individual component testing

- **Integration Testing**: API and database integration
- **User Acceptance Testing**: End-user scenario testing
- **Performance Testing**: Load and stress testing
- **Security Testing**: Authentication and data protection
- **Mobile Testing**: Cross-device compatibility testing

7.2 Test Coverage Requirements

- Minimum 80% code coverage for critical functions
- All user stories must have corresponding test cases
- All API endpoints must be tested for valid/invalid inputs
- Cross-browser testing on major browsers
- Mobile responsiveness testing on various screen sizes

---

8. DEPLOYMENT REQUIREMENTS

8.1 Production Environment

- Web server with PHP 8.1+ support
- MySQL/PostgreSQL database server
- SSL certificate for HTTPS
- Backup and recovery systems
- Monitoring and logging systems

8.2 Development Environment

- Local development server setup (XAMPP/WAMP)
- Version control system (Git)
- Code deployment pipeline
- Testing environment separate from production

---

9. MAINTENANCE AND SUPPORT

9.1 Maintenance Requirements

- Regular security updates and patches

- Database optimization and cleanup
- Performance monitoring and tuning
- User training and documentation updates

9.2 Support Requirements

- User documentation and help guides
- Technical support contact information
- Bug reporting and tracking system
- Feature request management process

---

10. GLOSSARY

Term	Definition
QR Code	Quick Response code containing encoded student information
Registration Number	Unique identifier assigned to each student
Payment Schedule	Timeline of expected payments for a course
Overdue Payment	Payment that is 14+ days past due date
Course-wise Report	Report organized by academic course
Receptionist	Staff member responsible for student registration
Admin	Administrator with full system access
API	Application Programming Interface for system communication

---

Document Version History:

- v1.0 - Initial SRS document creation
- Date: August 2025
- Status: Draft for Review