Software Requirements Specification (SRS)

Edumeup (Specialize in Olevel)

---

1. Introduction

1.1 Purpose

This document defines the functional and non-functional requirements for a web-based Learning Management Platform that integrates with Moodle LMS using REST APIs while providing a custom frontend and backend system. The platform will allow students, parents, schools, and administrators to manage courses, enrollments, payments, progress tracking, and reporting.

1.2 Scope

The system will provide:

- Course marketplace
- Student learning dashboard
- Parent monitoring dashboard
- School bulk purchase and licensing
- Payment processing via Pakistani gateways
- Single Sign-On with Moodle
- Reporting and analytics
- Forms management
- Notifications
- User management
- Secure authentication

Moodle will act as the learning engine while the custom system manages business logic.

1.3 Definitions

- LMS: Learning Management System
- SSO: Single Sign-On
- API: Application Programming Interface
- RBAC: Role Based Access Control

---

2. Overall Description

2.1 Product Perspective

The platform is a headless LMS consisting of:

- React frontend
- Node backend
- PostgreSQL database
- Moodle LMS
- AWS cloud infrastructure

2.2 Product Functions

The system shall:

- Allow users to register and authenticate
- Display course catalog
- Process payments
- Enroll users into courses via Moodle
- Track learning progress
- Provide dashboards for different roles
- Generate reports
- Manage licenses and bulk enrollments
- Support forms submission
- Send notifications

2.3 User Classes

- Student: Access courses, track progress, purchase courses
- Parent: Monitor child performance and reports
- School / Organization: Bulk purchase and manage students
- Admin: Full system control
- Guest: Browse public content

---

3. System Architecture

3.1 Frontend

- React / Next.js

- Responsive UI
- Role-based dashboards
- API integration

3.2 Backend

- Node.js (NestJS recommended)
- REST API
- Authentication service
- Payment service
- Moodle integration service

3.3 Database

PostgreSQL storing:

- Users
- Orders
- Roles
- Licenses
- Forms
- Notifications

3.4 External Systems

- Moodle LMS
- PayFast payment gateway
- AWS S3 storage
- Email service

---

4. Functional Requirements

4.1 Authentication

The system shall allow:

- User registration
- Login/logout
- Password reset

- Change password
- Session management
- JWT authentication
- Email verification

4.2 User Profile Management

Users shall be able to:

- Update personal details
- Upload profile image
- Manage contact information
- View account activity

4.3 Role Management

The system shall support:

- Student
- Parent
- School
- Admin

RBAC permissions shall control access.

4.4 Course Catalog

The system shall:

- Display courses fetched from Moodle
- Allow search and filters
- Show course details
- Display pricing
- Allow add to cart

4.5 Course Enrollment

The system shall:

- Enroll users after successful payment
- Sync enrollment with Moodle
- Display enrolled courses

- Allow course access via SSO

4.6 Student Dashboard

The system shall display:

- Enrolled courses
- Course progress
- Grades
- Completion status
- Certificates
- Activity timeline
- Notifications

4.7 Orders and Payments

The system shall allow:

- Course purchase
- Order history
- Invoice download
- Payment tracking
- Refund handling
- Payment status updates

4.8 Payment Processing

The system shall:

- Integrate with PayFast Pakistan
- Handle payment redirection
- Process webhooks
- Verify transactions
- Store payment logs
- Support partial payments
- Retry failed payments when permitted
- Handle chargebacks and disputes

4.9 Cart Management

The system shall:

- Add/remove items
- Apply coupons
- Calculate totals
- Display order summary

4.10 Parent Dashboard

The system shall allow parents to:

- View children profiles
- Monitor progress
- View grades
- Receive alerts
- Download reports

4.11 School / Organization

The system shall:

- Purchase bulk licenses
- Assign seats
- Upload students via CSV
- Track usage
- Generate reports

4.12 Admin Management

Admin shall be able to:

- Manage users
- Suspend accounts
- Reset passwords
- Assign roles
- View activity logs

4.13 Course Management (Admin)

Admin shall:

- Sync courses from Moodle

- Set pricing
- Manage categories
- Control visibility

4.14 Analytics and Reporting

The system shall generate:

- Revenue reports
- Enrollment reports
- Progress analytics
- Usage metrics

4.15 Forms Management

The system shall support:

- Contact forms
- Teacher application forms
- Custom forms
- File uploads
- Submission storage
- Export CSV

4.16 Notifications

The system shall send:

- Email notifications
- In-app notifications
- Payment confirmations
- Course updates

4.17 Support System

Users shall be able to:

- Submit support tickets
- Track status
- Receive responses

4.18 Certificates

The system shall:

- Display certificates from Moodle
- Allow download
- Track completion

4.19 SSO Integration

The system shall:

- Automatically authenticate users into Moodle
- Generate secure tokens
- Provide seamless course access

4.20 Error Handling

The system shall:

- Handle graceful API failures
- Display user-friendly error messages
- Use retry mechanisms for transient failures
- Send failed background jobs to a dead letter queue

4.21 Background Processing

The system shall support:

- Email queue
- Notification queue
- Payment verification jobs
- Moodle sync jobs

Recommended: Redis + BullMQ or AWS SQS.

4.22 Search

The system shall:

- Provide full-text search
- Support filters
- Support sorting
- Provide pagination

Optional: Elasticsearch.

5. Non-Functional Requirements

5.1 Performance

- Support thousands of concurrent users
- Response time under 3 seconds
- Efficient caching

5.2 Security

- HTTPS encryption
- Password hashing
- RBAC access control
- Webhook verification
- Input validation
- Audit logging

5.3 Scalability

The system shall support horizontal scaling via cloud infrastructure.

5.4 Reliability

- System uptime 99.9%
- Automatic backups
- Failover support

5.5 Usability

- Responsive UI
- Accessible interface
- Simple navigation

5.6 Caching Strategy

- Redis caching layer
- Course catalog cache
- Session cache
- CDN caching

6. Database Requirements

Tables include:

- users
- roles
- profiles
- courses_cache
- orders
- payments
- licenses
- license_users
- notifications
- forms
- form_fields
- form_submissions
- activity_logs

---

7. Integration Requirements

Moodle APIs

- Course retrieval
- Enrollment
- Completion tracking
- Grades
- Sync frequency definition
- Failure handling strategy
- Data mapping strategy
- Moodle API rate limits

Payment Gateway

- PayFast API
- Webhooks

- Transaction verification

Cloud Storage

- AWS S3 for media

---

8. Workflow Requirements

Course Purchase Workflow

1. User selects course
2. Adds to cart
3. Completes payment
4. Payment verified
5. Enrollment via Moodle
6. Course available

User Registration Workflow

1. User signs up
2. Account created
3. Moodle user created
4. Email verification

---

9. Security Requirements

- Token-based authentication
- Rate limiting
- Data encryption
- Secure cookies
- Session timeout

---

10. Logging and Monitoring

The system shall log:

- Login attempts
- Payment transactions

- User activity
- Errors

---

11. CI/CD

- Automated testing pipeline
- Build automation
- Deployment automation
- Environment separation (dev/staging/prod)

---

12. Deployment Requirements

Infrastructure shall include:

- Cloud hosting
- Managed database
- CDN
- Backup system

---

13. Future Extensibility

System architecture shall support:

- Mobile applications
- Microservices
- Additional payment gateways
- Additional roles
- Live classes

---

14. Assumptions

- Moodle instance is available
- Payment gateway account approved
- Cloud hosting provisioned

---

15. Constraints

- Dependence on Moodle API availability
- Payment gateway limitations
- Network connectivity

---

16. Acceptance Criteria

The system shall be accepted when:

- Users can register and login
- Payments process successfully
- Course enrollments work
- Dashboards display correct data
- Reports generate correctly