

INTEGRATION BLUEPRINT

Custom Strak Payment Gateway to Moodle LMS via React

1. Overview

This document outlines the workflow for replacing WooCommerce and Edwiser Bridge with a custom integration using the **Strak Payment Gateway** and **Moodle Web Services (REST API)**. This setup allows for automated user enrollment in Moodle courses immediately after a successful transaction in a React-based application.

2. Moodle Configuration (Server-Side)

Moodle must be prepared to receive external enrollment commands. You will need high-level administration access to perform these steps:

A. Enable Web Services

- Navigate to **Site Administration > Server > Web Services > Overview**.
- Enable **Web Services** and the **REST Protocol**.

B. Service and User Setup

- **Create a Web Service User:** Create a dedicated user (e.g., "api_manager") with permissions to view courses and manage enrollments.
- **Add Functions:** Create a new service and assign the following required functions:
 - `core_user_get_users_by_field`: To search for existing users.
 - `core_user_create_users`: To register new students automatically.
 - `enrol_manual_enrol_users`: The core function to link a user to a course.

C. Security Token

- Generate a **Web Services Token** for the specific user and service created above.
Note: Keep this token secure; it grants administrative access to your LMS.

3. Strak Payment Gateway Configuration

Ensure your Strak account is configured to communicate with your application:

- **API Credentials:** Note your API_Key and Merchant_ID.
- **Webhook Setup:** Configure a Webhook URL pointing to your backend. This acts as a "Fail-Safe." If the user's internet drops after payment, Strak will still notify your server to process the enrollment.
- **Redirects:** Set a Success URL that points back to your React application's "Success" dashboard.

4. React & Backend Logic (The Bridge)

Since calling Moodle APIs directly from the frontend is insecure, use a backend proxy (Node.js/PHP) to handle the logic.

Step 1: Initiate Transaction

The React app sends the course ID and user details to Strak. The user completes the payment on the Strak hosted page.

Step 2: Verification

Upon return to your site, your server verifies the transaction with Strak using the Transaction ID.

Step 3: Moodle API Execution

Execute a POST request to your Moodle REST endpoint:

```
POST /webservice/rest/server.php
Parameters:
- wstoken: [Your_Moodle_Token]
- wsfunction: enrol_manual_enrol_users
- moodlewsrestformat: json
- enrolments[0][roleid]: 5 (Student)
- enrolments[0][userid]: [User_Moodle_ID]
- enrolments[0][courseid]: [Purchased_Course_ID]
```

Pro-Tip: Before enrolling, always call `core_user_get_users_by_field` using the user's email. If the user doesn't exist, call `core_user_create_users` first, then proceed to enrollment.

5. Comparison & Benefits

- **Independence:** You are no longer reliant on third-party plugins like Edwiser or WooCommerce.
- **Speed:** Direct API calls are faster than processing through a heavy CMS like WordPress.
- **Customization:** You have full control over the checkout UI in your React application.