# Transport Action Group (TAG) Complete Developer Brief

## React Frontend + Moodle LMS Integration Project

Project Overview: Deploy a professional e-learning platform combining a custom React frontend with Moodle LMS backend for Transport Action Group's certification programs.

Timeline: 4-6 weeks

Budget: $3,500-7,000

Technology Stack: React.js, Moodle LMS, PostgreSQL, Apache, SSL

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Table of Contents

1. [Project Overview](#project-overview)
2. [Architecture & Technology Stack](#architecture--technology-stack)
3. [Frontend Deployment](#frontend-deployment)
4. [Moodle LMS Setup](#moodle-lms-setup)
5. [Integration Requirements](#integration-requirements)
6. [Payment Gateway Configuration](#payment-gateway-configuration)
7. [Content Management](#content-management)
8. [Security Implementation](#security-implementation)
9. [Testing & Quality Assurance](#testing--quality-assurance)
10. [Go-Live Checklist](#go-live-checklist)
11. [Maintenance & Support](#maintenance--support)
12. [Troubleshooting Guide](#troubleshooting-guide)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 1. Project Overview

### 1.1 Business Objectives

Transport Action Group (TAG) requires a comprehensive e-learning platform to deliver professional certification programs in sustainable road freight operations. The platform must:

* Generate Revenue: Two certification programs ($99 and $199)
* Professional Credibility: Industry-leading design and functionality
* Scalability: Handle growth from 0 to 1000+ students
* Integration: Seamless user experience between marketing and learning
* Compliance: Professional certification tracking and registry

### 1.2 Target Audience

* Primary: Professional truck drivers seeking certification
* Secondary: Fleet managers and executives
* Geographic: South Africa (with global expansion potential)
* Technical Level: Basic to intermediate computer skills

### 1.3 Success Metrics

* Technical: 99.9% uptime, <3 second page load times
* Business: 100+ enrollments in first 3 months
* User Experience: <5% bounce rate on course pages
* Conversion: 15%+ conversion from visitor to enrolled student

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 2. Architecture & Technology Stack

### 2.1 System Architecture

┌─────────────────────────────────────────────────────────────┐

│ USER EXPERIENCE │

├─────────────────────────────────────────────────────────────┤

│ Marketing & Discovery → Enrollment → Learning → Certification │

└─────────────────────────────────────────────────────────────┘

┌─────────────────────┐ ┌─────────────────────┐

│ React Frontend │ │ Moodle Backend │

│ www.transportaction │◄──►│ learning.transport │

│ group.com │ │ actiongroup.com │

│ │ │ │

│ • Marketing Site │ │ • Course Delivery │

│ • User Registration │ │ • Assessments │

│ • Professional │ │ • Certificates │

│ Registry │ │ • Progress Tracking │

│ • Books Section │ │ • User Management │

│ • Contact Forms │ │ • Payment Processing│

└─────────────────────┘ └─────────────────────┘

│ │

└─────────┬─────────────────┘

│

┌─────────▼─────────┐

│ SSO Integration │

│ • Single Sign-On │

│ • User Sync │

│ • Session Mgmt │

└───────────────────┘

### 2.2 Technology Stack

#### Frontend (React Website)

* Framework: React 18.x with Vite
* Styling: Tailwind CSS + Custom Components
* Icons: Lucide React
* State Management: React Hooks (useState, useEffect)
* Authentication: JWT tokens + localStorage
* HTTP Client: Fetch API
* Build Tool: Vite
* Deployment: Static files on Apache/Nginx

#### Backend (Moodle LMS)

* LMS Platform: Moodle 4.3 (Latest Stable)
* Database: PostgreSQL 14+
* Web Server: Apache 2.4 with mod\_php
* PHP Version: PHP 8.1
* Authentication: Custom SSO plugin
* Payment: Stripe integration
* SSL: Let's Encrypt certificates
* Caching: Redis (optional for performance)

#### Infrastructure

* Server: Ubuntu 22.04 LTS (minimum 4GB RAM, 2 CPU cores)
* Storage: 50GB+ SSD storage
* Backup: Daily automated backups
* Monitoring: Server monitoring and uptime alerts
* CDN: CloudFlare (recommended for global performance)

### 2.3 Domain Structure

* Main Website: www.transportactiongroup.com
* Learning Portal: learning.transportactiongroup.com
* API Endpoint: api.transportactiongroup.com (optional)
* Admin Panel: learning.transportactiongroup.com/admin

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 3. Frontend Deployment

### 3.1 Pre-Deployment Setup

#### 3.1.1 Server Requirements

* Operating System: Ubuntu 22.04 LTS
* Web Server: Apache 2.4 or Nginx 1.18+
* SSL Certificate: Let's Encrypt or commercial SSL
* Node.js: 18.x or higher (for build process)
* Storage: 10GB minimum for frontend

#### 3.1.2 Domain Configuration

1. DNS Setup: Point www.transportactiongroup.com to server IP
2. SSL Certificate: Install SSL certificate for HTTPS
3. Subdomain: Configure learning.transportactiongroup.com for Moodle

### 3.2 Build Process

#### 3.2.1 Environment Configuration

Create .env file in frontend directory:

# Production Environment Variables

REACT\_APP\_MOODLE\_URL=https://learning.transportactiongroup.com

REACT\_APP\_API\_BASE=https://api.transportactiongroup.com

REACT\_APP\_STRIPE\_PUBLIC\_KEY=pk\_live\_your\_stripe\_public\_key

REACT\_APP\_SITE\_URL=https://www.transportactiongroup.com

REACT\_APP\_SUPPORT\_EMAIL=support@transportactiongroup.com

NODE\_ENV=production

#### 3.2.2 Build Commands

# Navigate to frontend directory

cd frontend/

# Install dependencies

npm install

# Build for production

npm run build

# Verify build output

ls -la dist/

### 3.3 Deployment Process

#### 3.3.1 Automated Deployment

Use the provided deployment script:

# Make script executable

chmod +x scripts/deploy-frontend.sh

# Run deployment

./scripts/deploy-frontend.sh

#### 3.3.2 Manual Deployment Steps

If automated script fails, follow these manual steps:

1. Backup Existing Site:

sudo cp -r /var/www/html /var/backups/website-backup-$(date +%Y%m%d)

1. Deploy New Files:

sudo rm -rf /var/www/html/\*

sudo cp -r frontend/dist/\* /var/www/html/

1. Set Permissions:

sudo chown -R www-data:www-data /var/www/html

sudo find /var/www/html -type d -exec chmod 755 {} \;

sudo find /var/www/html -type f -exec chmod 644 {} \;

1. Configure Apache:

sudo nano /etc/apache2/sites-available/transportactiongroup.conf

Add this configuration:

<VirtualHost \*:80>

ServerName www.transportactiongroup.com

DocumentRoot /var/www/html

Redirect permanent / https://www.transportactiongroup.com/

</VirtualHost>

<VirtualHost \*:443>

ServerName www.transportactiongroup.com

DocumentRoot /var/www/html

SSLEngine on

SSLCertificateFile /etc/letsencrypt/live/www.transportactiongroup.com/fullchain.pem

SSLCertificateKeyFile /etc/letsencrypt/live/www.transportactiongroup.com/privkey.pem

<Directory /var/www/html>

Options -Indexes

AllowOverride All

Require all granted

</Directory>

# React Router support

RewriteEngine On

RewriteCond %{REQUEST\_FILENAME} !-f

RewriteCond %{REQUEST\_FILENAME} !-d

RewriteRule . /index.html [L]

# Security headers

Header always set X-Content-Type-Options nosniff

Header always set X-Frame-Options DENY

Header always set X-XSS-Protection "1; mode=block"

Header always set Strict-Transport-Security "max-age=63072000; includeSubDomains; preload"

</VirtualHost>

1. Enable Site and Restart Apache:

sudo a2ensite transportactiongroup.conf

sudo a2enmod rewrite ssl headers

sudo systemctl restart apache2

### 3.4 Frontend Features Verification

After deployment, verify these features work correctly:

#### 3.4.1 Core Functionality

* [ ] Homepage loads correctly with TAG logo and branding
* [ ] Navigation works - all menu items functional
* [ ] Responsive design - works on mobile and desktop
* [ ] Service sections display properly
* [ ] Knowledge Hub shows both training programs
* [ ] Books section displays both book covers correctly
* [ ] Professional registry search functionality works
* [ ] Contact forms submit successfully

#### 3.4.2 Integration Points

* [ ] Login/Register buttons redirect to appropriate pages
* [ ] Course enrollment buttons trigger authentication check
* [ ] SSO integration preparation (will be tested after Moodle setup)
* [ ] Payment integration preparation (Stripe buttons configured)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 4. Moodle LMS Setup

### 4.1 Pre-Installation Requirements

#### 4.1.1 Server Specifications

* RAM: 4GB minimum, 8GB recommended
* CPU: 2 cores minimum, 4 cores recommended
* Storage: 50GB minimum, 100GB recommended
* Network: 100Mbps connection minimum

#### 4.1.2 Software Dependencies

# Update system

sudo apt update && sudo apt upgrade -y

# Install required packages

sudo apt install -y apache2 postgresql postgresql-contrib php8.1 php8.1-fpm \

php8.1-cli php8.1-pgsql php8.1-xml php8.1-mbstring php8.1-curl php8.1-zip \

php8.1-gd php8.1-intl php8.1-soap php8.1-xmlrpc php8.1-ldap unzip wget curl git

### 4.2 Database Setup

#### 4.2.1 PostgreSQL Configuration

# Switch to postgres user

sudo -u postgres psql

# Create database and user

CREATE USER moodleuser WITH PASSWORD 'secure\_password\_here';

CREATE DATABASE moodle WITH OWNER moodleuser;

GRANT ALL PRIVILEGES ON DATABASE moodle TO moodleuser;

\q

#### 4.2.2 Database Optimization

Edit PostgreSQL configuration for Moodle:

sudo nano /etc/postgresql/14/main/postgresql.conf

Add these optimizations:

shared\_buffers = 256MB

effective\_cache\_size = 1GB

maintenance\_work\_mem = 64MB

checkpoint\_completion\_target = 0.9

wal\_buffers = 16MB

default\_statistics\_target = 100

random\_page\_cost = 1.1

effective\_io\_concurrency = 200

### 4.3 Moodle Installation

#### 4.3.1 Automated Installation

Use the provided script for complete setup:

# Make script executable

chmod +x scripts/deploy-moodle.sh

# Run as root

sudo ./scripts/deploy-moodle.sh

#### 4.3.2 Manual Installation Process

If automated script fails, follow these detailed steps:

1. Download Moodle:

cd /tmp

wget https://download.moodle.org/download.php/direct/stable403/moodle-4.3.tgz

tar -xzf moodle-4.3.tgz

sudo mv moodle /var/www/moodle

1. Create Data Directory:

sudo mkdir /var/moodledata

sudo chown www-data:www-data /var/moodledata

sudo chmod 755 /var/moodledata

1. Set Permissions:

sudo chown -R www-data:www-data /var/www/moodle

sudo chmod -R 755 /var/www/moodle

1. Create Configuration File:

sudo nano /var/www/moodle/config.php

Use the provided moodle/config.php template and update with your database credentials.

1. Configure Apache Virtual Host:

sudo nano /etc/apache2/sites-available/learning.transportactiongroup.com.conf

Add the virtual host configuration provided in the deployment script.

1. Run Installation:

cd /var/www/moodle

sudo -u www-data php admin/cli/install.php \

--lang=en \

--wwwroot=https://learning.transportactiongroup.com \

--dataroot=/var/moodledata \

--dbtype=pgsql \

--dbhost=localhost \

--dbname=moodle \

--dbuser=moodleuser \

--dbpass=your\_password \

--prefix=mdl\_ \

--fullname="Transport Action Group Learning Management System" \

--shortname="TAG LMS" \

--adminuser=admin \

--adminpass=secure\_admin\_password \

--adminemail=transportactiongroup@gmail.com \

--non-interactive \

--agree-license

### 4.4 TAG-Specific Configuration

#### 4.4.1 Course Structure Setup

After installation, configure the course structure:

1. Login to Moodle Admin: https://learning.transportactiongroup.com/login/
2. Navigate to: Site Administration → Courses → Add a category
3. Create Categories:

* Professional Driver Training
* Green Freight Management

1. Create Courses:

Course 1: Professional Truck Driving Program

* Full name: Professional Truck Driving Program
* Short name: PTDP
* Category: Professional Driver Training
* Course format: Topics format
* Number of sections: 20
* Course summary: "Comprehensive 20-module certification program for professional truck drivers based on 'The Professional Truck Driver's Handbook'"

Course 2: Green Freight Professional Program

* Full name: Green Freight Professional Program
* Short name: GFPP
* Category: Green Freight Management
* Course format: Topics format
* Number of sections: 20
* Course summary: "Advanced 20-module certification for sustainable freight management professionals"

#### 4.4.2 Assessment Configuration

For each course, configure assessments:

1. Quiz Settings:

* 5 questions per module
* Pass grade: 80% (4 out of 5 correct)
* Attempts allowed: 3
* Time limit: 10 minutes per quiz
* Question behavior: Immediate feedback

1. Question Bank Setup:

* Create question categories for each module
* Import questions (200 total - 100 per course)
* Set question types: Multiple choice, single answer

#### 4.4.3 Certificate Configuration

1. Install Certificate Plugin:

cd /var/www/moodle

sudo -u www-data git clone https://github.com/markn86/moodle-mod\_customcert.git mod/customcert

1. Configure Certificate Templates:

* Create professional certificate design
* Include TAG logo and branding
* Add unique certificate numbers (PTD-2024-XXX, GFP-2024-XXX)
* Include completion date and student name

### 4.5 Payment Gateway Integration

#### 4.5.1 Stripe Configuration

1. Install Stripe Payment Plugin:

cd /var/www/moodle

sudo -u www-data git clone https://github.com/catalyst/moodle-paygw\_stripe.git payment/gateway/stripe

1. Configure Stripe Settings:

* Navigate to: Site Administration → Plugins → Payment gateways → Stripe
* Add Stripe publishable key: pk\_live\_your\_key\_here
* Add Stripe secret key: sk\_live\_your\_key\_here
* Set webhook endpoint: https://learning.transportactiongroup.com/payment/gateway/stripe/webhook.php

1. Course Pricing Setup:

* Professional Truck Driving Program: $99 USD
* Green Freight Professional Program: $199 USD
* Payment method: One-time fee
* Currency: USD (or ZAR if preferred)

#### 4.5.2 Payment Workflow

1. User Journey:

* User clicks "Enroll Now" on React website
* Redirected to Moodle course page
* Payment form displayed with Stripe integration
* Payment processed securely
* User automatically enrolled upon successful payment
* Confirmation email sent

1. Webhook Configuration:

* Set up Stripe webhook to handle payment confirmations
* Configure automatic enrollment upon payment success
* Set up refund handling (if needed)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 5. Integration Requirements

### 5.1 Single Sign-On (SSO) Implementation

#### 5.1.1 SSO Architecture

The SSO system allows users to login once on the React website and automatically access Moodle courses without re-authentication.

User Login Flow:

1. User enters credentials on React website
2. React website validates credentials with API
3. API returns JWT token and user data
4. Token stored in localStorage
5. When accessing Moodle, token passed as parameter
6. Moodle SSO plugin validates token with API
7. User automatically logged into Moodle

#### 5.1.2 SSO Plugin Installation

The custom SSO plugin is provided in the package:

1. Copy Plugin Files:

sudo cp -r moodle/auth/sso /var/www/moodle/auth/

sudo chown -R www-data:www-data /var/www/moodle/auth/sso

1. Install Plugin:

* Login to Moodle as admin
* Navigate to: Site Administration → Notifications
* Install the SSO authentication plugin

1. Configure Plugin:

* Navigate to: Site Administration → Plugins → Authentication → SSO
* Set TAG API Base URL: https://api.transportactiongroup.com
* Set TAG Website URL: https://www.transportactiongroup.com
* Enable the plugin

1. Enable Authentication Method:

* Navigate to: Site Administration → Plugins → Authentication → Manage authentication
* Enable "SSO" authentication
* Set order: Manual, SSO

#### 5.1.3 API Endpoints Required

You'll need to create these API endpoints (can be simple PHP scripts):

1. Token Validation Endpoint (/auth/validate-token)

<?php

// Simple token validation endpoint

header('Content-Type: application/json');

$input = json\_decode(file\_get\_contents('php://input'), true);

$username = $input['username'] ?? '';

$token = $input['token'] ?? '';

// Validate token (implement your logic here)

$valid = validate\_jwt\_token($token, $username);

echo json\_encode(['valid' => $valid]);

?>

2. User Data Endpoint (/users/{username})

<?php

// User data retrieval endpoint

header('Content-Type: application/json');

$username = $\_GET['username'] ?? '';

// Get user data from your database

$user\_data = get\_user\_data($username);

echo json\_encode($user\_data);

?>

### 5.2 User Data Synchronization

#### 5.2.1 User Registration Flow

1. React Website Registration:

* User fills registration form on React website
* Data sent to API for validation and storage
* User account created in main database
* JWT token generated and returned

1. Moodle Account Creation:

* When user first accesses Moodle via SSO
* SSO plugin checks if Moodle account exists
* If not, creates Moodle account with synced data
* Links accounts via username/email

#### 5.2.2 Data Sync Fields

Synchronize these fields between systems:

* Username (primary key)
* Email address
* First name
* Last name
* Phone number
* Location/City
* Registration date
* Last login date

### 5.3 Professional Registry Integration

#### 5.3.1 Registry Data Flow

1. Course Completion:

* User completes course in Moodle
* Certificate generated automatically
* Completion data sent to main database via API
* Professional registry updated

1. Registry Display:

* React website queries registry database
* Displays certified professionals with:
* Name and location
* Certification numbers
* Completion dates
* Prior education
* Professional ratings

#### 5.3.2 API Integration Points

Course Completion Webhook:

// Moodle sends completion data to this endpoint

POST /api/certifications/complete

{

"username": "john.doe",

"course\_id": "PTDP",

"course\_name": "Professional Truck Driving Program",

"completion\_date": "2024-03-15",

"certificate\_number": "PTD-2024-012",

"grade": "85%"

}

Registry Query Endpoint:

// React website queries this for registry data

GET /api/registry/professionals

Response: [

{

"name": "John Doe",

"location": "Cape Town",

"certifications": [

{

"name": "Professional Truck Driver Certification",

"number": "PTD-2024-012",

"date": "2024-03-15"

}

],

"prior\_education": "Grade 12, NQF 4 Transport Operations"

}

]

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 6. Payment Gateway Configuration

### 6.1 Stripe Integration Setup

#### 6.1.1 Stripe Account Configuration

1. Create Stripe Account:

* Sign up at https://stripe.com
* Complete business verification
* Add bank account for payouts
* Configure tax settings for South Africa

1. Get API Keys:

* Navigate to Developers → API keys
* Copy Publishable key (starts with pk\_)
* Copy Secret key (starts with sk\_)
* Use test keys for development, live keys for production

1. Configure Webhooks:

* Navigate to Developers → Webhooks
* Add endpoint: https://learning.transportactiongroup.com/payment/gateway/stripe/webhook.php
* Select events: payment\_intent.succeeded, payment\_intent.payment\_failed

#### 6.1.2 Moodle Payment Configuration

1. Enable Payment Subsystem:

* Navigate to: Site Administration → Advanced features
* Enable "Enable payment subsystem"

1. Configure Payment Account:

* Navigate to: Site Administration → Payment → Payment accounts
* Add new account: "TAG Course Payments"
* Enable Stripe gateway
* Add Stripe API keys

1. Course Payment Setup:

* Edit each course
* Navigate to: Course → Enrolment methods
* Add "Payment" enrolment method
* Set cost: $99 for PTDP, $199 for GFPP
* Select payment account: "TAG Course Payments"

#### 6.1.3 Payment Flow Testing

1. Test Mode Setup:

* Use Stripe test keys
* Test card: 4242 4242 4242 4242
* Any future expiry date
* Any 3-digit CVC

1. Test Scenarios:

* [ ] Successful payment and enrollment
* [ ] Failed payment handling
* [ ] Webhook processing
* [ ] Email confirmations
* [ ] Refund processing (if applicable)

### 6.2 Alternative Payment Methods

#### 6.2.1 PayPal Integration (Optional)

If Stripe is not available in South Africa:

1. Install PayPal Plugin:

cd /var/www/moodle

sudo -u www-data git clone https://github.com/catalyst/moodle-paygw\_paypal.git payment/gateway/paypal

1. Configure PayPal:

* Create PayPal Business account
* Get Client ID and Secret
* Configure in Moodle payment settings

#### 6.2.2 Local Payment Methods

Consider adding South African payment methods:

* Ozow: Popular South African payment gateway
* PayFast: Local payment processor
* Bank transfers: Manual verification system

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 7. Content Management

### 7.1 Video Content Upload

#### 7.1.1 Video Hosting Options

Option 1: Vimeo Pro (Recommended)

* Cost: $20/month
* Benefits: Professional player, privacy controls, no ads
* Integration: Embed codes in Moodle

Option 2: YouTube Private/Unlisted

* Cost: Free
* Benefits: Reliable hosting, good performance
* Drawbacks: Less control, potential ads

Option 3: Self-hosted

* Cost: Server storage and bandwidth
* Benefits: Complete control
* Drawbacks: Higher server requirements

#### 7.1.2 Video Upload Process

1. Prepare Videos:

* Format: MP4, H.264 codec
* Resolution: 1080p (1920x1080)
* Duration: 5 minutes (drivers), 15 minutes (managers)
* File size: <500MB per video

1. Upload to Hosting Platform:

* Create organized folders/playlists
* Set privacy to "Private" or "Unlisted"
* Generate embed codes

1. Add to Moodle Courses:

* Navigate to course → Add activity → Page or URL
* Embed video using iframe or URL
* Add description and learning objectives

#### 7.1.3 Content Structure

Professional Truck Driving Program (20 modules × 5 minutes):

1. Introduction to Professional Driving
2. Vehicle Inspection and Maintenance
3. Safe Driving Techniques
4. Traffic Laws and Regulations
5. Cargo Handling and Securing
6. Route Planning and Navigation
7. Fuel Efficiency and Eco-Driving
8. Emergency Procedures
9. Customer Service Excellence
10. Documentation and Compliance
11. Health and Safety
12. Vehicle Technology
13. Weather and Road Conditions
14. Defensive Driving
15. Professional Communication
16. Time Management
17. Stress Management
18. Environmental Awareness
19. Continuous Improvement
20. Career Development

Green Freight Professional Program (20 modules × 15 minutes):

1. Introduction to Green Freight
2. Carbon Footprint Assessment
3. Fleet Optimization Strategies
4. Alternative Fuel Technologies
5. Route Optimization
6. Driver Training for Efficiency
7. Vehicle Maintenance for Emissions
8. Supply Chain Sustainability
9. Regulatory Compliance
10. Technology Integration
11. Performance Monitoring
12. Cost-Benefit Analysis
13. Stakeholder Engagement
14. Reporting and Documentation
15. Continuous Improvement
16. Industry Best Practices
17. Future Trends
18. Case Studies
19. Implementation Planning
20. Leadership in Sustainability

### 7.2 Assessment Question Creation

#### 7.2.1 Question Development Guidelines

Question Format: Multiple choice, single correct answer

Questions per Module: 5

Total Questions: 200 (100 per course)

Pass Requirement: 4 out of 5 correct (80%)

#### 7.2.2 Question Quality Standards

1. Clear and Concise: Questions should be easy to understand
2. Relevant: Directly related to module content
3. Practical: Focus on real-world applications
4. Varied Difficulty: Mix of basic, intermediate, and advanced
5. No Tricks: Avoid deliberately confusing questions

#### 7.2.3 Sample Questions

Professional Truck Driving Program - Module 1 Sample:

1. What is the most important aspect of professional driving?

a) Speed of delivery

b) Safety of all road users ✓

c) Fuel efficiency

d) Customer satisfaction

1. How often should a professional driver conduct a vehicle inspection?

a) Weekly

b) Monthly

c) Before each trip ✓

d) Only when problems occur

1. Which document is required for all commercial vehicle operations?

a) Driver's license only

b) Vehicle registration only

c) Professional driving permit (PrDP) ✓

d) Insurance certificate only

1. What is the maximum continuous driving time without a break?

a) 3 hours

b) 4 hours ✓

c) 5 hours

d) 6 hours

1. Professional drivers must maintain a logbook to record:

a) Fuel consumption only

b) Route information only

c) Driving hours and rest periods ✓

d) Customer feedback only

#### 7.2.4 Question Import Process

1. Create Question Bank:

* Navigate to course → Question bank
* Create categories for each module
* Import questions via CSV or manually

1. CSV Import Format:

"Question","Answer A","Answer B","Answer C","Answer D","Correct","Category"

"What is the speed limit in urban areas?","40 km/h","60 km/h","80 km/h","100 km/h","B","Module 1"

1. Quality Assurance:

* Review all questions for accuracy
* Test quiz functionality
* Verify correct answers and feedback

### 7.3 Certificate Templates

#### 7.3.1 Certificate Design Requirements

1. Professional Appearance:

* TAG logo prominently displayed
* Professional color scheme (blue/green)
* High-quality typography
* Official appearance

1. Required Information:

* Student name
* Course name
* Completion date
* Unique certificate number
* Authorized signature
* TAG contact information

1. Security Features:

* Unique certificate numbers
* Verification URL
* Digital signature
* Tamper-evident design

#### 7.3.2 Certificate Configuration

1. Install Certificate Module:

* Use Custom Certificate plugin
* Configure templates for each course

1. Template Setup:

* Upload TAG logo
* Set certificate dimensions (A4 landscape)
* Configure text fields and positioning
* Set up automatic numbering

1. Verification System:

* Create certificate verification page
* Allow public verification via certificate number
* Link to professional registry

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 8. Security Implementation

### 8.1 SSL/TLS Configuration

#### 8.1.1 Certificate Installation

1. Let's Encrypt (Free):

sudo apt install certbot python3-certbot-apache

sudo certbot --apache -d www.transportactiongroup.com -d learning.transportactiongroup.com

1. Commercial Certificate:

* Purchase from trusted CA (Comodo, DigiCert, etc.)
* Install following CA instructions
* Configure Apache virtual hosts

#### 8.1.2 Security Headers

Configure these security headers in Apache:

# Security Headers

Header always set X-Content-Type-Options nosniff

Header always set X-Frame-Options DENY

Header always set X-XSS-Protection "1; mode=block"

Header always set Strict-Transport-Security "max-age=63072000; includeSubDomains; preload"

Header always set Referrer-Policy "strict-origin-when-cross-origin"

Header always set Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline' https://js.stripe.com; style-src 'self' 'unsafe-inline'; img-src 'self' data: https:; connect-src 'self' https://api.stripe.com"

### 8.2 Database Security

#### 8.2.1 PostgreSQL Hardening

1. User Permissions:

-- Create limited user for Moodle

CREATE USER moodleuser WITH PASSWORD 'strong\_random\_password';

GRANT CONNECT ON DATABASE moodle TO moodleuser;

GRANT USAGE ON SCHEMA public TO moodleuser;

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO moodleuser;

1. Connection Security:

# Edit pg\_hba.conf

sudo nano /etc/postgresql/14/main/pg\_hba.conf

# Add these lines (replace with actual IP)

host moodle moodleuser 127.0.0.1/32 md5

host moodle moodleuser ::1/128 md5

1. Regular Backups:

# Create backup script

#!/bin/bash

pg\_dump -U moodleuser -h localhost moodle > /var/backups/moodle-$(date +%Y%m%d).sql

# Add to crontab

0 2 \* /usr/local/bin/backup-moodle.sh

### 8.3 Application Security

#### 8.3.1 Moodle Security Settings

1. Admin Security:

* Change default admin username
* Use strong admin password (16+ characters)
* Enable two-factor authentication
* Limit admin IP addresses

1. User Security:

* Enforce strong password policy
* Enable account lockout after failed attempts
* Set session timeout (2 hours)
* Enable email verification

1. File Security:

* Restrict file uploads by type
* Set maximum file sizes
* Enable virus scanning (if available)
* Secure file storage permissions

#### 8.3.2 React Frontend Security

1. Environment Variables:

* Never expose secret keys in frontend
* Use environment variables for configuration
* Validate all user inputs

1. Authentication:

* Use secure JWT tokens
* Implement token expiration
* Store tokens securely (httpOnly cookies preferred)
* Validate tokens on server side

### 8.4 Server Security

#### 8.4.1 System Hardening

1. Firewall Configuration:

# Configure UFW firewall

sudo ufw default deny incoming

sudo ufw default allow outgoing

sudo ufw allow ssh

sudo ufw allow 'Apache Full'

sudo ufw enable

1. Automatic Updates:

# Install unattended upgrades

sudo apt install unattended-upgrades

sudo dpkg-reconfigure -plow unattended-upgrades

1. Fail2Ban:

# Install and configure Fail2Ban

sudo apt install fail2ban

sudo cp /etc/fail2ban/jail.conf /etc/fail2ban/jail.local

sudo systemctl enable fail2ban

sudo systemctl start fail2ban

#### 8.4.2 Monitoring and Logging

1. Log Monitoring:

* Monitor Apache access/error logs
* Monitor Moodle logs
* Set up log rotation
* Configure alerts for suspicious activity

1. Uptime Monitoring:

* Use external monitoring service (UptimeRobot, Pingdom)
* Monitor both websites
* Set up email/SMS alerts
* Monitor SSL certificate expiration

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 9. Testing & Quality Assurance

### 9.1 Pre-Launch Testing Checklist

#### 9.1.1 Frontend Testing

Functionality Tests:

* [ ] All navigation links work correctly
* [ ] Responsive design on mobile/tablet/desktop
* [ ] Forms submit successfully
* [ ] Search functionality works
* [ ] Professional registry displays correctly
* [ ] Books section shows correct covers
* [ ] Contact information is accurate

Performance Tests:

* [ ] Page load times <3 seconds
* [ ] Images optimized and load quickly
* [ ] No JavaScript errors in console
* [ ] SEO meta tags present
* [ ] Google PageSpeed score >90

Browser Compatibility:

* [ ] Chrome (latest)
* [ ] Firefox (latest)
* [ ] Safari (latest)
* [ ] Edge (latest)
* [ ] Mobile browsers (iOS Safari, Chrome Mobile)

#### 9.1.2 Moodle Testing

Core Functionality:

* [ ] User registration works
* [ ] Login/logout functions correctly
* [ ] Course enrollment process
* [ ] Video playback in courses
* [ ] Quiz functionality (5 questions, 4/5 to pass)
* [ ] Certificate generation
* [ ] Progress tracking
* [ ] Email notifications

Payment Testing:

* [ ] Stripe test payments work
* [ ] Payment confirmation emails
* [ ] Automatic enrollment after payment
* [ ] Failed payment handling
* [ ] Refund processing (if applicable)

Integration Testing:

* [ ] SSO between React and Moodle
* [ ] User data synchronization
* [ ] Professional registry updates
* [ ] Certificate numbers in registry

#### 9.1.3 Security Testing

SSL/HTTPS:

* [ ] SSL certificates valid
* [ ] HTTPS redirects work
* [ ] No mixed content warnings
* [ ] Security headers present

Authentication:

* [ ] Password strength requirements
* [ ] Account lockout after failed attempts
* [ ] Session timeout works
* [ ] JWT token validation

Data Protection:

* [ ] User data encrypted in transit
* [ ] Database access restricted
* [ ] File upload restrictions
* [ ] No sensitive data in logs

### 9.2 User Acceptance Testing

#### 9.2.1 Test User Scenarios

Scenario 1: New Driver Registration

1. Visit www.transportactiongroup.com
2. Browse services and courses
3. Click "Register" and create account
4. Enroll in Professional Truck Driving Program
5. Complete payment process
6. Access course content
7. Complete first module and quiz
8. Verify progress tracking

Scenario 2: Manager Course Enrollment

1. Register as fleet manager
2. Enroll in Green Freight Professional Program
3. Complete payment ($199)
4. Access advanced course content
5. Complete multiple modules
6. Take final assessment
7. Receive certificate
8. Verify registry listing

Scenario 3: Professional Registry Search

1. Visit website as anonymous user
2. Navigate to professional registry
3. Search for certified professionals
4. View professional profiles
5. Download CV (if available)
6. Verify certification details

#### 9.2.2 Performance Testing

Load Testing:

* Test with 50 concurrent users
* Monitor server response times
* Check database performance
* Verify no errors under load

Stress Testing:

* Test payment system under load
* Verify video streaming performance
* Check quiz submission handling
* Monitor system resources

### 9.3 Bug Tracking and Resolution

#### 9.3.1 Bug Reporting Process

1. Bug Documentation:

* Clear description of issue
* Steps to reproduce
* Expected vs actual behavior
* Screenshots/videos if applicable
* Browser/device information

1. Priority Levels:

* Critical: Site down, payment failures
* High: Core functionality broken
* Medium: Minor functionality issues
* Low: Cosmetic issues

1. Resolution Timeline:

* Critical: 2-4 hours
* High: 24 hours
* Medium: 3-5 days
* Low: Next maintenance window

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 10. Go-Live Checklist

### 10.1 Pre-Launch Verification

#### 10.1.1 Technical Checklist

Infrastructure:

* [ ] Server resources adequate (CPU, RAM, storage)
* [ ] SSL certificates installed and valid
* [ ] DNS records pointing to correct servers
* [ ] Backup systems configured and tested
* [ ] Monitoring systems active

Applications:

* [ ] React website deployed and functional
* [ ] Moodle installation complete and configured
* [ ] Database optimized and secured
* [ ] Payment gateway configured and tested
* [ ] Email system working (SMTP configured)

Security:

* [ ] Firewall rules configured
* [ ] Security headers implemented
* [ ] User permissions properly set
* [ ] Admin accounts secured
* [ ] Fail2Ban configured

#### 10.1.2 Content Verification

Website Content:

* [ ] All text content reviewed and approved
* [ ] Images optimized and displaying correctly
* [ ] Contact information accurate
* [ ] Legal pages present (Privacy Policy, Terms)
* [ ] SEO meta tags configured

Course Content:

* [ ] All 40 videos uploaded and accessible
* [ ] 200 assessment questions imported
* [ ] Certificate templates configured
* [ ] Course pricing set correctly
* [ ] Professional registry populated

#### 10.1.3 Integration Testing

SSO Integration:

* [ ] Login from React website to Moodle works
* [ ] User data synchronizes correctly
* [ ] Session management functions properly
* [ ] Logout works from both systems

Payment Integration:

* [ ] Stripe live keys configured
* [ ] Payment webhooks working
* [ ] Enrollment automation functional
* [ ] Email confirmations sending

### 10.2 Launch Day Procedures

#### 10.2.1 Launch Sequence

1. Final Backup (T-2 hours):

# Create complete system backup

sudo rsync -av /var/www/ /backup/pre-launch-www/

sudo pg\_dump moodle > /backup/pre-launch-db.sql

1. Switch to Production (T-1 hour):

* Update environment variables to production
* Switch Stripe to live keys
* Enable production monitoring
* Verify all systems green

1. Go Live (T-0):

* Update DNS if needed
* Announce launch
* Monitor systems closely
* Be ready for immediate fixes

1. Post-Launch Monitoring (T+4 hours):

* Monitor server performance
* Check error logs
* Verify user registrations working
* Test payment processing

#### 10.2.2 Rollback Plan

If critical issues occur:

1. Immediate Actions:

* Put maintenance page up
* Stop payment processing
* Notify stakeholders

1. Rollback Steps:

# Restore previous version

sudo rm -rf /var/www/html/\*

sudo cp -r /backup/pre-launch-www/\* /var/www/html/

sudo systemctl restart apache2

1. Communication:

* Update status page
* Email registered users
* Social media announcement

### 10.3 Post-Launch Activities

#### 10.3.1 Immediate Tasks (First 24 hours)

* [ ] Monitor system performance and errors
* [ ] Verify first user registrations
* [ ] Test payment processing with real transactions
* [ ] Check email delivery
* [ ] Monitor SSL certificate status
* [ ] Verify backup systems running

#### 10.3.2 First Week Tasks

* [ ] Analyze user behavior and feedback
* [ ] Monitor conversion rates
* [ ] Check SEO performance
* [ ] Review server logs for issues
* [ ] Optimize performance based on real usage
* [ ] Plan first content updates

#### 10.3.3 First Month Tasks

* [ ] Comprehensive performance review
* [ ] User feedback analysis
* [ ] Security audit
* [ ] Backup and disaster recovery testing
* [ ] Plan feature enhancements
* [ ] Marketing campaign analysis

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 11. Maintenance & Support

### 11.1 Regular Maintenance Tasks

#### 11.1.1 Daily Tasks

Automated:

* Database backups
* Log rotation
* Security updates (if configured)
* Uptime monitoring
* SSL certificate monitoring

Manual:

* Check error logs
* Monitor system resources
* Verify backup completion
* Review user registrations

#### 11.1.2 Weekly Tasks

* [ ] Review system performance metrics
* [ ] Check for Moodle updates
* [ ] Analyze user activity reports
* [ ] Review payment transactions
* [ ] Update course content if needed
* [ ] Check professional registry updates

#### 11.1.3 Monthly Tasks

* [ ] Full security audit
* [ ] Performance optimization review
* [ ] Backup restoration testing
* [ ] User feedback analysis
* [ ] Content review and updates
* [ ] Financial reporting
* [ ] SEO performance review

### 11.2 Update Procedures

#### 11.2.1 Moodle Updates

1. Preparation:

* Create full backup
* Test update on staging environment
* Schedule maintenance window
* Notify users of downtime

1. Update Process:

# Put site in maintenance mode

sudo -u www-data php admin/cli/maintenance.php --enable

# Backup current installation

sudo cp -r /var/www/moodle /backup/moodle-pre-update

# Download and install update

cd /tmp

wget https://download.moodle.org/download.php/direct/stable404/moodle-4.4.tgz

tar -xzf moodle-4.4.tgz

sudo rsync -av moodle/ /var/www/moodle/

# Run upgrade

sudo -u www-data php admin/cli/upgrade.php --non-interactive

# Disable maintenance mode

sudo -u www-data php admin/cli/maintenance.php --disable

1. Post-Update Testing:

* Verify core functionality
* Test payment processing
* Check SSO integration
* Verify course content

#### 11.2.2 React Frontend Updates

1. Development Process:

* Make changes in development environment
* Test thoroughly
* Build production version
* Deploy to staging for final testing

1. Deployment:

# Build new version

npm run build

# Backup current version

sudo cp -r /var/www/html /backup/frontend-$(date +%Y%m%d)

# Deploy new version

sudo rm -rf /var/www/html/\*

sudo cp -r dist/\* /var/www/html/

sudo systemctl restart apache2

### 11.3 Backup and Recovery

#### 11.3.1 Backup Strategy

Daily Backups:

* Database dump
* Moodle data directory
* Website files
* Configuration files

Weekly Backups:

* Full system backup
* Off-site backup copy
* Backup verification test

Monthly Backups:

* Archive old backups
* Disaster recovery test
* Backup retention cleanup

#### 11.3.2 Backup Scripts

Database Backup:

#!/bin/bash

# /usr/local/bin/backup-database.sh

DATE=$(date +%Y%m%d\_%H%M%S)

BACKUP\_DIR="/var/backups/database"

mkdir -p $BACKUP\_DIR

# Backup Moodle database

pg\_dump -U moodleuser -h localhost moodle > $BACKUP\_DIR/moodle\_$DATE.sql

# Compress backup

gzip $BACKUP\_DIR/moodle\_$DATE.sql

# Remove backups older than 30 days

find $BACKUP\_DIR -name "\*.sql.gz" -mtime +30 -delete

echo "Database backup completed: moodle\_$DATE.sql.gz"

File Backup:

#!/bin/bash

# /usr/local/bin/backup-files.sh

DATE=$(date +%Y%m%d\_%H%M%S)

BACKUP\_DIR="/var/backups/files"

mkdir -p $BACKUP\_DIR

# Backup website files

tar -czf $BACKUP\_DIR/website\_$DATE.tar.gz -C /var/www html

# Backup Moodle files

tar -czf $BACKUP\_DIR/moodle\_$DATE.tar.gz -C /var/www moodle

# Backup Moodle data

tar -czf $BACKUP\_DIR/moodledata\_$DATE.tar.gz -C /var moodledata

# Remove backups older than 7 days

find $BACKUP\_DIR -name "\*.tar.gz" -mtime +7 -delete

echo "File backup completed: $DATE"

#### 11.3.3 Recovery Procedures

Database Recovery:

# Stop web server

sudo systemctl stop apache2

# Drop and recreate database

sudo -u postgres dropdb moodle

sudo -u postgres createdb -O moodleuser moodle

# Restore from backup

gunzip -c /var/backups/database/moodle\_YYYYMMDD\_HHMMSS.sql.gz | sudo -u postgres psql moodle

# Start web server

sudo systemctl start apache2

File Recovery:

# Stop web server

sudo systemctl stop apache2

# Restore website files

sudo rm -rf /var/www/html/\*

sudo tar -xzf /var/backups/files/website\_YYYYMMDD\_HHMMSS.tar.gz -C /var/www/

# Restore Moodle files

sudo rm -rf /var/www/moodle/\*

sudo tar -xzf /var/backups/files/moodle\_YYYYMMDD\_HHMMSS.tar.gz -C /var/www/

# Set permissions

sudo chown -R www-data:www-data /var/www/

# Start web server

sudo systemctl start apache2

### 11.4 Support Procedures

#### 11.4.1 User Support

Common Issues:

1. Login Problems:

* Password reset procedure
* Account activation issues
* SSO troubleshooting

1. Course Access:

* Payment verification
* Enrollment confirmation
* Technical requirements

1. Certificate Issues:

* Certificate generation problems
* Download issues
* Verification requests

Support Channels:

* Email: support@transportactiongroup.com
* Help desk system (if implemented)
* FAQ section on website
* Video tutorials

#### 11.4.2 Technical Support

Monitoring Alerts:

* Server down alerts
* High CPU/memory usage
* SSL certificate expiration
* Payment processing failures
* Database connection issues

Response Times:

* Critical issues: 2 hours
* High priority: 24 hours
* Medium priority: 3 days
* Low priority: 1 week

Escalation Procedures:

1. Level 1: Basic troubleshooting
2. Level 2: Advanced technical issues
3. Level 3: Developer intervention
4. Level 4: Vendor support (if needed)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 12. Troubleshooting Guide

### 12.1 Common Issues and Solutions

#### 12.1.1 Website Issues

Issue: Website not loading

# Check Apache status

sudo systemctl status apache2

# Check error logs

sudo tail -f /var/log/apache2/error.log

# Restart Apache

sudo systemctl restart apache2

# Check disk space

df -h

# Check memory usage

free -m

Issue: SSL certificate problems

# Check certificate status

sudo certbot certificates

# Renew certificate

sudo certbot renew

# Test SSL configuration

openssl s\_client -connect www.transportactiongroup.com:443

Issue: React routing not working

* Verify .htaccess file exists
* Check Apache mod\_rewrite enabled
* Verify virtual host configuration

#### 12.1.2 Moodle Issues

Issue: Moodle site not accessible

# Check Moodle error logs

sudo tail -f /var/www/moodle/moodledata/error.log

# Check database connection

sudo -u postgres psql -c "\l" | grep moodle

# Check file permissions

sudo ls -la /var/www/moodle/

sudo ls -la /var/moodledata/

# Clear Moodle cache

sudo -u www-data php /var/www/moodle/admin/cli/purge\_caches.php

Issue: Payment processing failures

* Check Stripe webhook logs
* Verify API keys are correct
* Test with Stripe test mode
* Check Moodle payment logs

Issue: SSO not working

* Verify SSO plugin installed
* Check API endpoints responding
* Verify JWT token format
* Check user synchronization

#### 12.1.3 Database Issues

Issue: Database connection errors

# Check PostgreSQL status

sudo systemctl status postgresql

# Check database connections

sudo -u postgres psql -c "SELECT \* FROM pg\_stat\_activity;"

# Check database size

sudo -u postgres psql -c "SELECT pg\_size\_pretty(pg\_database\_size('moodle'));"

# Restart PostgreSQL

sudo systemctl restart postgresql

Issue: Slow database performance

# Check slow queries

sudo -u postgres psql moodle -c "SELECT query, mean\_time, calls FROM pg\_stat\_statements ORDER BY mean\_time DESC LIMIT 10;"

# Analyze database

sudo -u postgres psql moodle -c "ANALYZE;"

# Vacuum database

sudo -u postgres psql moodle -c "VACUUM ANALYZE;"

### 12.2 Performance Optimization

#### 12.2.1 Frontend Optimization

Image Optimization:

# Install image optimization tools

sudo apt install imagemagick optipng jpegoptim

# Optimize images

find /var/www/html -name "\*.jpg" -exec jpegoptim --max=85 {} \;

find /var/www/html -name "\*.png" -exec optipng -o2 {} \;

Caching Configuration:

# Add to .htaccess

<IfModule mod\_expires.c>

ExpiresActive On

ExpiresByType text/css "access plus 1 year"

ExpiresByType application/javascript "access plus 1 year"

ExpiresByType image/png "access plus 1 year"

ExpiresByType image/jpg "access plus 1 year"

ExpiresByType image/jpeg "access plus 1 year"

</IfModule>

<IfModule mod\_deflate.c>

AddOutputFilterByType DEFLATE text/plain

AddOutputFilterByType DEFLATE text/html

AddOutputFilterByType DEFLATE text/xml

AddOutputFilterByType DEFLATE text/css

AddOutputFilterByType DEFLATE application/xml

AddOutputFilterByType DEFLATE application/xhtml+xml

AddOutputFilterByType DEFLATE application/rss+xml

AddOutputFilterByType DEFLATE application/javascript

AddOutputFilterByType DEFLATE application/x-javascript

</IfModule>

#### 12.2.2 Moodle Optimization

PHP Configuration:

# /etc/php/8.1/apache2/php.ini

memory\_limit = 512M

max\_execution\_time = 300

upload\_max\_filesize = 100M

post\_max\_size = 100M

max\_input\_vars = 5000

Moodle Caching:

# Enable Moodle caching

sudo -u www-data php /var/www/moodle/admin/cli/cfg.php --name=cachejs --set=1

sudo -u www-data php /var/www/moodle/admin/cli/cfg.php --name=cachetemplates --set=1

sudo -u www-data php /var/www/moodle/admin/cli/cfg.php --name=langstringcache --set=1

Database Optimization:

-- PostgreSQL optimization

ALTER SYSTEM SET shared\_buffers = '256MB';

ALTER SYSTEM SET effective\_cache\_size = '1GB';

ALTER SYSTEM SET maintenance\_work\_mem = '64MB';

ALTER SYSTEM SET checkpoint\_completion\_target = 0.9;

ALTER SYSTEM SET wal\_buffers = '16MB';

ALTER SYSTEM SET default\_statistics\_target = 100;

ALTER SYSTEM SET random\_page\_cost = 1.1;

ALTER SYSTEM SET effective\_io\_concurrency = 200;

-- Reload configuration

SELECT pg\_reload\_conf();

### 12.3 Security Incident Response

#### 12.3.1 Incident Detection

Signs of Security Issues:

* Unusual traffic patterns
* Failed login attempts
* Unexpected file changes
* Database performance issues
* SSL certificate warnings
* Payment processing anomalies

Monitoring Commands:

# Check failed login attempts

sudo grep "authentication failure" /var/log/auth.log | tail -20

# Check Apache access logs for suspicious activity

sudo tail -f /var/log/apache2/access.log | grep -E "(POST|PUT|DELETE)"

# Check for file changes

sudo find /var/www -type f -mtime -1 -ls

# Monitor system resources

top

iotop

netstat -tulpn

#### 12.3.2 Incident Response Procedures

Immediate Actions:

1. Isolate the system - Block suspicious IP addresses
2. Preserve evidence - Create system snapshots
3. Assess damage - Check what was compromised
4. Notify stakeholders - Inform management and users
5. Document everything - Keep detailed incident log

Recovery Steps:

1. Stop the attack - Block malicious traffic
2. Patch vulnerabilities - Apply security updates
3. Restore from backup - If data was compromised
4. Change passwords - All admin and user accounts
5. Monitor closely - Watch for continued attacks

Post-Incident:

1. Conduct review - What went wrong and why
2. Update procedures - Improve security measures
3. Train staff - Share lessons learned
4. Test improvements - Verify fixes work

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 13. Contact Information and Support

### 13.1 Project Contacts

Client: Transport Action Group

Email: transportactiongroup@gmail.com

Location: Durban, South Africa

Developer: [Your Name/Company]

Email: [Your Email]

Phone: [Your Phone]

Support Hours: [Your Hours]

### 13.2 Emergency Contacts

Critical Issues (Site down, payment failures):

* Primary: [Your Emergency Phone]
* Secondary: [Backup Contact]
* Response Time: 2-4 hours

High Priority Issues (Functionality problems):

* Email: [Your Support Email]
* Response Time: 24 hours

### 13.3 Vendor Support

Moodle Community: https://moodle.org/support/

Stripe Support: https://support.stripe.com/

Let's Encrypt: https://letsencrypt.org/docs/

### 13.4 Documentation and Resources

Project Documentation: This document and included files

Moodle Documentation: https://docs.moodle.org/

React Documentation: https://react.dev/

Stripe Documentation: https://stripe.com/docs

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## 14. Appendices

### Appendix A: File Structure

TAG\_Complete\_Developer\_Package/

├── frontend/ # React website files

│ ├── src/

│ │ ├── App.jsx # Main application component

│ │ ├── App-Moodle-Integrated.jsx # Moodle-integrated version

│ │ ├── App.css # Styles

│ │ └── assets/ # Images and assets

│ ├── package.json # Dependencies

│ ├── .env.example # Environment variables template

│ └── dist/ # Built files (after npm run build)

├── moodle/ # Moodle configuration files

│ ├── config.php # Moodle configuration template

│ └── auth/sso/ # SSO authentication plugin

│ ├── auth.php # SSO plugin code

│ ├── version.php # Plugin version info

│ └── lang/en/ # Language files

├── scripts/ # Deployment scripts

│ ├── deploy-frontend.sh # Frontend deployment script

│ └── deploy-moodle.sh # Moodle deployment script

├── documentation/ # Project documentation

│ └── TAG\_Complete\_Developer\_Brief.md # This document

└── assets/ # Project assets

├── TAGhiresblack.png # TAG logo files

├── TAGlogohires.jpg

├── road\_freight\_sustainability\_cover.png

└── truck\_driver\_handbook\_cover.png

### Appendix B: Environment Variables

# React Frontend Environment Variables

REACT\_APP\_MOODLE\_URL=https://learning.transportactiongroup.com

REACT\_APP\_API\_BASE=https://api.transportactiongroup.com

REACT\_APP\_STRIPE\_PUBLIC\_KEY=pk\_live\_your\_stripe\_public\_key

REACT\_APP\_SITE\_URL=https://www.transportactiongroup.com

REACT\_APP\_SUPPORT\_EMAIL=support@transportactiongroup.com

NODE\_ENV=production

# Moodle Configuration Variables

DB\_HOST=localhost

DB\_NAME=moodle

DB\_USER=moodleuser

DB\_PASS=secure\_password\_here

MOODLE\_URL=https://learning.transportactiongroup.com

ADMIN\_EMAIL=transportactiongroup@gmail.com

### Appendix C: Port and Service Configuration

Services and Ports:

* Apache HTTP: Port 80 (redirects to HTTPS)
* Apache HTTPS: Port 443
* PostgreSQL: Port 5432 (localhost only)
* SSH: Port 22 (firewall restricted)

Domain Configuration:

* www.transportactiongroup.com → React Frontend
* learning.transportactiongroup.com → Moodle LMS
* api.transportactiongroup.com → API Endpoints (optional)

### Appendix D: Database Schema Overview

-- Key Moodle Tables for TAG Integration

mdl\_user -- User accounts

mdl\_course -- Course information

mdl\_enrol -- Enrollment methods

mdl\_grade\_grades -- Student grades

mdl\_certificate -- Generated certificates

mdl\_course\_completions -- Course completion tracking

-- Custom Tables (if needed)

tag\_professional\_registry -- Professional registry data

tag\_certifications -- Certification tracking

tag\_sso\_tokens -- SSO token management

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Document Version: 1.0

Last Updated: September 2024

Total Pages: 47

This comprehensive developer brief provides everything needed to successfully deploy and maintain the Transport Action Group e-learning platform. Follow the procedures carefully and don't hesitate to contact support if you encounter any issues.

Good luck with the deployment! 🚀