**TEACHING PRACTICE MANAGEMENT SYSTEM (TPMS)**  
*Project Proposal and Development Plan*

**1. INTRODUCTION**

Teaching Practice (TP) is a mandatory requirement for all education students in Kenya, typically undertaken in their 3rd or 4th year of university. The TP period involves students finding a placement in a school, being grouped by zones, supervised by lecturers, and graded at the end of the session. Managing this process manually leads to inefficiencies, miscommunication, and poor tracking of progress.

This proposal outlines the design and development of a **Teaching Practice Management System (TPMS)** built using **PHP Laravel**, **MySQL** with **ORM**, and **Tailwind CSS** for frontend styling.

The system will be developed in **phases**, starting with a fully functional **Phase 1** (independent), while subsequent phases will build upon the features of earlier ones. This avoids confusion or going back and forth during development.

**2. OBJECTIVES**

* Streamline the TP process for students, lecturers, and the university.
* Track each student's lesson plans, supervision visits, and final grading.
* Improve communication through built-in alerts and reporting systems.
* Enforce role-based access, permissions, and system accessibility based on TP phase.

**3. TOOLS AND TECHNOLOGIES**

| **Tool** | **Purpose** |
| --- | --- |
| PHP (Laravel) | Backend framework |
| MySQL | Database |
| Laravel ORM | DB interaction layer |
| Tailwind CSS | Frontend styling (via CDN) |
| HTML + Blade | Templating engine for views |

**4. SYSTEM USERS AND ROLES**

* **Admin**: Controls TP period, assigns zones, manages access, views reports.
* **Lecturer**: Supervises students, inputs grades, requests zones.
* **Student**: Uploads documents, requests supervision, submits reports.

Permissions will be implemented early to ensure:

* Access to features is gated based on user roles.
* TP period enforces system access windows.
* Students can be reassigned or scheduled based on TP activity.

**5. CORE MODULES**

**5.1. Authentication and Role Access**

* Unified login system using university credentials.
* Enum-based roles: admin, lecturer, student.
* Access restrictions enforced early in development.

**5.2. TP Period Setup**

* Admin defines TP window.
* System enforces time-based access control.

**5.3. School Application & Approval**

* Student uploads stamped placement form.
* Admin reviews and approves placement.

**5.4. Zone and Lecturer Assignment**

* Admin creates zones and maps lecturers to each zone.
* Students are assigned zones based on approved school.

**5.5. Daily Lesson Planning**

* Students upload lesson plans per day.
* Consistent formatting via predefined templates.

**5.6. Supervision Scheduling**

* Students/Lecturers may request early supervision.
* Includes approval workflow.
* Max 3 supervision visits enforced.

**5.7. Alerts & Reporting**

* Students can send alerts to lecturers.
* Confidential reports (e.g., misconduct) go directly to Admin.

**5.8. Final Grading**

* Lecturers input grades and recommendations.
* Once graded, students can only view records.

**6. DATABASE DESIGN (PHASE ONE)**

**Core Tables:**

* users (id, name, email, password, role enum, status)
* students (id, user\_id, reg\_no, year, semester, tp\_status, graded)
* lecturers (id, user\_id, department)
* admins (id, user\_id)
* tp\_periods (id, year, start\_date, end\_date, is\_active)
* zones (id, name, county)
* schools (id, name, county, zone\_id)
* tp\_applications (id, student\_id, school\_id, stamped\_form, status)
* lecturer\_zone\_requests (id, lecturer\_id, zone\_id, status)
* lecturer\_zone\_assignments (id, lecturer\_id, zone\_id)
* supervision\_schedule (id, student\_id, lecturer\_id, date, time, requested\_by, status)
* lesson\_plans (id, student\_id, date, file\_path, topic)
* supervision\_visits (id, lecturer\_id, student\_id, date, feedback, score)
* alerts (id, student\_id, lecturer\_id, message, type, status)
* reports (id, type, student\_id, description, reported\_id, target\_type)
* final\_docs (id, student\_id, file\_path, type)
* grades (id, student\_id, lecturer\_id, score, recommendation, date)

**7. PROJECT PHASES**

| **Phase** | **Title** | **Description** |
| --- | --- | --- |
| **1** | Database Design | Design database schema, implement Laravel migrations. |
| **2** | Authentication & Roles | Role-based login with permissions and access control. |
| **3** | TP Period Module | Admin defines TP window; system access is phase-gated. |
| **4** | Student Onboarding | Placement upload, lecturer assignment, approval checks. |
| **5** | Zone & Lecturer Mapping | Assign zones, map lecturers and students. |
| **6** | Supervision Scheduling | Early request logic, visit limits, lecturer sync. |
| **7** | Document Management | Upload lesson plans, final docs, daily progress. |
| **8** | Alerts & Reporting | Emergency alerts, misconduct reporting, admin feedback. |
| **9** | Dashboards & Views | Dynamic views per user role using Blade templates. |
| **10** | Testing & Deployment | Full QA, performance testing, deployment rollout. |

⚠️ Each phase is **independent in execution**, but later phases **depend** on the completion of earlier ones. No phase will revisit prior implementation—**no back and forth** loops.

**8. COMPONENT DESIGN APPROACH**

Laravel Blade will be used to:

* Build reusable UI elements like navbars, cards, alerts.
* Use @yield, @include, @extends for dynamic views.
* Maintain separate layouts for student, lecturer, and admin.

**9. CONCLUSION**

This project is built on the philosophy of **good planning, modular development**, and **clear role separation**. Early enforcement of system access rules and permission logic makes future expansion easier. With Laravel's robust backend, Tailwind styling, and a database-first design, TPMS will evolve phase by phase into a comprehensive, user-friendly platform.

**Prepared by**: Rombo  
**Role**: Software Engineering Student  
**Tech Stack**: PHP (Laravel), MySQL, Tailwind CSS