Table of Contents

Introduction

This document provides a comprehensive overview of the Quiz Management System, detailing the implementation of various features such as role-based access control, email notifications, quiz assignment logic, authentication mechanisms, and more. The system is built using the Laravel framework and leverages several packages and services to enhance functionality.

Installation

git clone https://github.com/hadiya8149/learning-management-with-laravel.git

composer install

Run the following commands in sequence.

- set database name to leaning_management_system
- php artisan migrate
- php artisan key:generate
- php artisan jwt:secret
- php artisan db:seed

- Php artisan db:seed –class RolesAndPermissionsSeeder
- Php artisan queue:work
- Php artisan schedule:run
-

Project Description

This project aims to create a learning management backend for students. In this system students can take assigned quizzes and can see real time results. The students can retake their quiz and improve their score.

The admin can review student quiz recording. The admin, manager and supervisor can assign quiz to user.

The admin can approve or reject student registration application. The admin can add managers via back office.

Requirements

Admin and Manager Roles Setup:

1. Create a Seeder for Admin.
2. Implement role-based user management for Admin, Manager, and Student roles.
3. Allow Admin to add users (Manager) via back office.
4. Set up email notifications:
5. When Admin adds a Manager, send an email to set up a password (valid for 24 hours).
6. After Admin approves a Student, send an email to set up a password (valid for 24 hours).
7. Handle the logic for resending the password set email if the link expires.
8. Email rejection notice for rejected student requests.
9. Develop logic for assigning quizzes to Students:
10. Quiz scheduling (activate after 2 days).
11. Quiz expiration logic (if not attempted within the defined time).
12. Manage video recording during quiz attempts.
13. Store videos for Admin review.
14. Setup permission system:
15. Admin can add users, view/accept/reject student requests, assign quizzes, and view students.
16. Manager can assign quizzes and view students.
17. Student can only view their assigned quizzes and results.
18. Track quiz attempts and provide real-time result calculation.

JWT-Based Login System:

1. Implement JWT authentication for Admin, Manager, and Student login.
2. After setting a password, users (Manager, Student) will log in using JWT to access their respective dashboards.

Student Public Form Submission:

1. Develop a public route for student submissions.
2. Handle file uploads (CVs in doc, pdf, csv, docx formats).
3. Send confirmation emails to students upon form submission.
4. Admin interface for accepting/rejecting student requests.

Quiz Management:

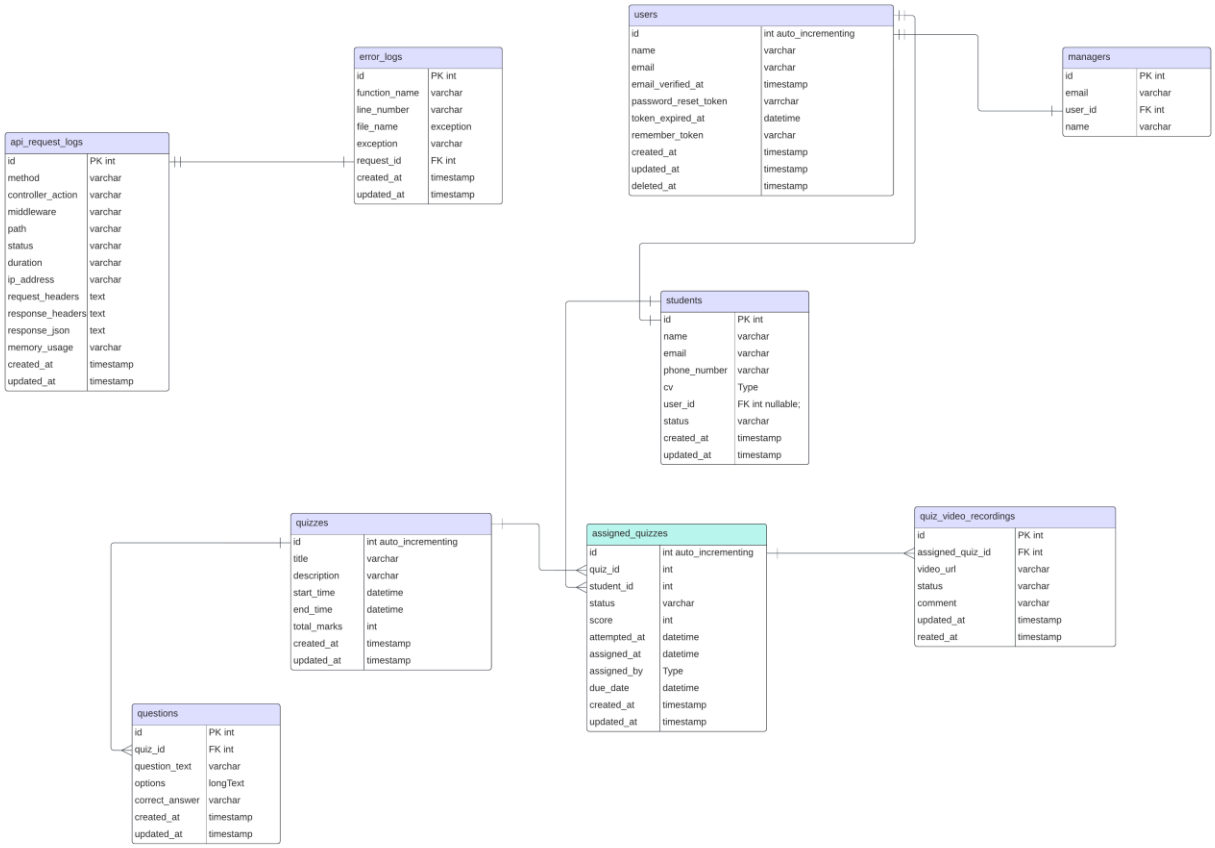
1. Set up a quiz assignment system.
2. Enable quizzes based on the defined schedule (2-day activation).
3. Store quiz attempt results and manage quiz expiration.

Filter Users:

1. Implement filtering functionality in the Admin and Manager dashboards to filter students based on their acceptance or rejection status.

Database Design

Database : learning_management_system



API routes

POST /login

{“email”:admin_office@lms.edu.pk”

“password”:”admin123@

}

POST /forgot-password

{“email”:}

POST /set-password?email=testemail@gmail.com&name=testname

{“email”:,
“token”:,

“password”:,

“password_confirmation”:}

All other routes require Authorization token in request headers

{"headers”:{”Authorization”:”Bearer token”}}

POST /logout

Admin apis

GET /quizzes

GET /quizzes/quiz?id=1

GET /student/{student_id}/assigned_quizzes

POST /assign-quiz

{“student_id”:,
“quiz_Id”:

}

GET /assigned-quizzes

POST /add-manager

{name:,email:,role:}

POST /add-student

{email:}

POST /reject-student

{email:}

GET /students

GET /managers

DELETE /delete-manager

{id:}

Manager or supervisor apis

GET /students

GET /assigned-quizzes

POST /assign-quiz

{“student_id”:,
“quiz_Id”:

}

Student apis

POST /register/student

{“email”:

“phone_number”:

“cv”:

“name”:,

}

GET /my-assigned-quizzes

GET /take-quiz?id=1

POST /submit-quiz-attempt

{“id”:,

“student_id”:

“answers”: {

“answers”:

[

{“question_id”:1,”answer”:”A”}

]

“video”:,
}

Packages

- tymon/jwt-auth
- `spatie/laravel-permission`

Services

1. UserService.php
2. ManagerService.php
3. StudentService.php
4. QuizService.php
5. ForgotPasswordController.php

Controllers

1. UserController.php
2. ManagerController.php
3. StudentController.php
4. QuizController.php

Middleware

1. JWTMiddleware.php
2. APIRequestLogs.php

Events

1. QuizAssigned.php

Listeners

1. ActivateQuizCreation.php

Jobs

1. QueuedPasswordResetJob.php
2. SendRegistrationEmaiUob
3. ExpireQuizJob
4. SendNotificationEmaiUob

Helper Functions

1. SendSuccessResponse()
2. SendFailureResponse()
3. addUserAndSendSetPasswordMail()
    a. This helper function adds a user in users_table and send emailnotification job using queue

Request Validation

All requests are validated through request validation classes.

Database Migration and Seeder

1. Roles And Permissions Seeder
    a. Total 14+ permissions are seeded for different roles
    b. **Roles:**
        i. Super Admin
        ii. Manager
        iii. Supervisor
        iv. Student

Admin is seeded through seeder. Students can register through register/students form and the signup request is the approved through admin. If admin approves the request the student is added to user table, otherwise the student is notified about rejected application.