G-35

MATH-TREK SYSTEM.

IMPLEMENTATION REPORT

NAME	REGISTRATION NUMBER	ROLE
KANGIRE FRANK		Database management
KISEMBO ROGERS BANGIRANA		System Integration/Github
OLOWO OMONDI PHILLY		Java server/ client
NEEMA DAVID PONTIANO		Web portal
SSEMAGANDA BALDWIN		APIs/ Error resolution

Contents

Introduction 4
Objectives 4
Methodology 4
System Architecture 5
Backend 5
Frontend 5
Database 5
API Integration 5
Implementation Details 5
Backend Development 5
Frontend Development 6
Database Design 6
API Integration 6
Screenshots 7
Challenges and Solutions 16
Challenge 1: Vite Integration 16
Challenge 2: Data Synchronization 16
Results 16
Conclusion 16

Figure 1 The welcome page of the web portal 7
Figure 2 The Admin dashboard after registering and signing in 7
Figure 3 The navigation bar 7
Figure 4 School creation by admin in school management view 8
Figure 5 Schools table updated on every creation 8
Figure 6 School view detailed with participant count 8
Figure 7 School detail edit process view 9
Figure 8 Updated school table with new details 9
Figure 9 Question and answer file upload and management view 10
Figure 10 Example wrong file upload 10
Figure 11 Error message 10
Figure 12 Challenge creation view 11
Figure 13 Created challenge view 11
Figure 14 Sent email after school creation to school rep 12
Figure 15 Initial views of server and client 12
Figure 16 Student registration 12
Figure 17 Representative login CLI 13
Figure 18 View and confirm applicant 13

Figure 19 Confirm by username since table has many 13
Figure 20 Student login 14
Figure 21 Successful student login and challenge view 14
Figure 22 Challenge attempt selection 14
Figure 23 Challenge attempt view on server and client sides 15
Figure 24 Analytics view 15

Introduction

This report details the implementation of the MathTrek school system project. It covers the steps taken from conception to deployment. The system is designed to manage schools, participants, challenges, and reports, providing comprehensive analytics and performance metrics.

Objectives

The project was undertaken with the following objectives in mind:

- To develop a web-based system for managing challenges, the participants and the schools to which they belong.
- To implement an administrator's portal for managing schools, their representatives, generating reports and handling corresponding emails.
- To integrate a Java CLI client for challenge participants to interact with and submit data to the database through a Java server.
- To integrate a Java client for user interaction and data submission.
- To generate and provide detailed analytics and visualizations of performance data to respective stakeholders.

Methodology

The project was implemented using a couple of software methodologies as follows:

- Agile Development which involves Iterative development with regular feedback loops undertaken by all the team members where we allowed ourselves to productively improve one another's code and subsequently collaborate effectively.
- Test Driven Development (TDD) which involved writing tests before implementing functionalities and so as a team, we made it a point to test every component by using IDE inbuilt features custom made to do Java code and PHP-laravel code testing throughout the code writing process and module deployment.
- Continuous Integration/Continuous Deployment (CI/CD) that included automating the deployment process to ensure smooth updates through using resources like Github to safely handle sections of the program that had been successfully implemented.

System Architecture

The system architecture has several components, including the backend, frontend, database, and external APIs.

Backend

The backend is developed using Java and Laravel (PHP), providing RESTful APIs for data manipulation and retrieval.

Frontend

The frontend is built using Blade templates, CSS, and JavaScript, with Vite for module bundling.

Database

The system uses MySQL for data storage, with tables for schools, participants, challenges, and reports handling their respective data and all the relations.

API Integration

External APIs such as mailtrap and browsershot are used for email notifications and PDF generation.

Implementation Details

Backend Development

- Framework: Laravel 5.8.3
- Controllers:
    - SchoolController
    - QuestionController
    - ChallengeController
    - ReportController
    - ParticipantController
- Models:
    - School: model of the school detailing how the table will look like and what other models it interacts with.
    - Participant: model of the student and how the student relates with other models such as the challenge one and the school one.
    - Challenge: model on the features of the challenges as picked from the web portal and picked by the server to generate challenges for the clients.
    - Question: model of the question detailing the structure of a question and answer as extracted from excel files.
    - Report: model of the analytics data used by the web portal to generate and display graphs.

- Features:
    - School representative login and management
    - School and participant management
    - Challenge creation and management
    - Report generation
    - Email notifications

Frontend Development

- Tools: Blade, CSS, JavaScript, Vite for visualization and interactivity.
- Libraries: Chart.js for data visualization, Spatie Simple excel for excel imports.
- Features:
    - Responsive design
    - Dynamic data binding
    - Interactive charts and graphs

Database Design

- Tables:
    - schools: handles all school attributes entered as table values including the name, district, registration number, representative name and email.
    - applicants: this includes registered details of pupils including the username, first and last names, date of birth, email and registration number.
    - participants: this includes username and password of confirmed applicants that are used to verify when signing in.
    - challenges: includes the challenge name and descriptions as well as the duration, number of questions, start and end dates.
    - Questions: includes the questions and answer options.
    - reports: this includes subsequent data from other tables that is put together for easier tallying and statistics generation.
- Relationships:
    - One-to-many relationships between schools and participants
    - Many-to-many relationships between challenges and participants

API Integration

- Email Notifications: Integrated using Laravel's built-in mail functionalities tested using Mailtrap.
- PDF Generation: Implemented using spatie/browsershot.

Screenshots

This includes screenshots that illustrate the various parts of the system with detailed captions.

- Dashboard

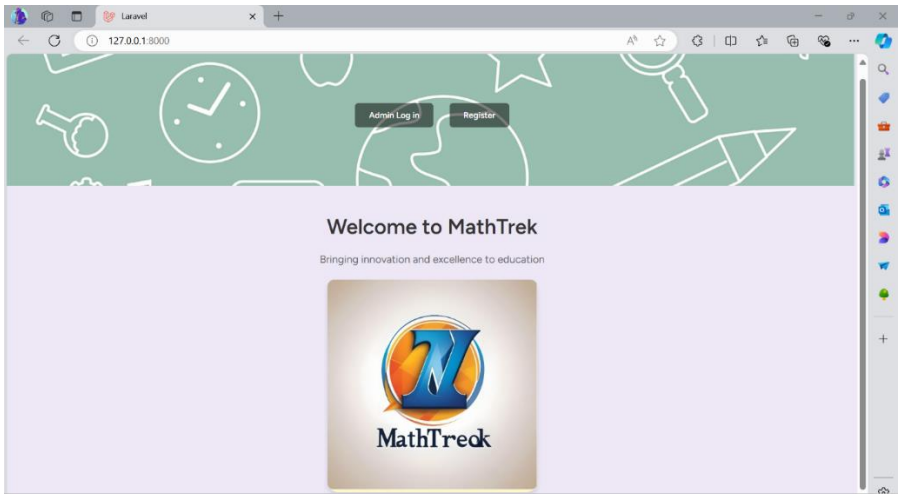

Figure 1 The welcome page of the web portal

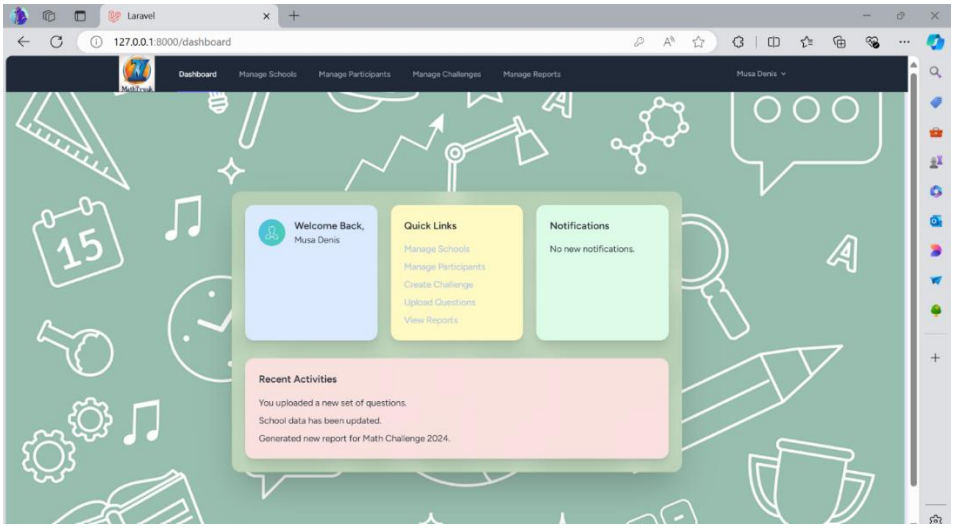

Figure 2 The Admin dashboard after registering and signing in

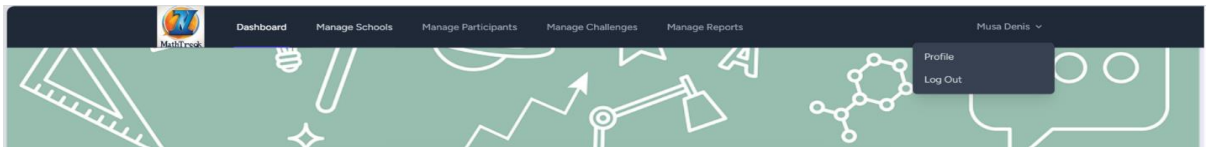

Figure 3 The navigation bar

- School Management

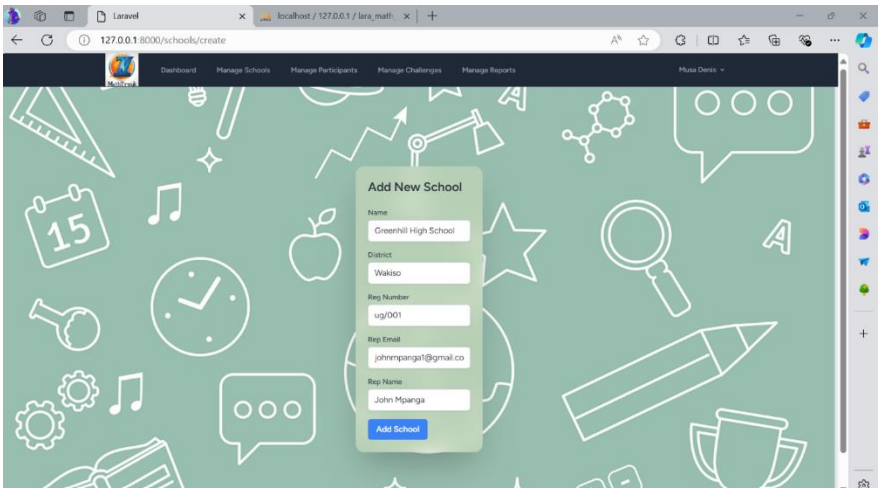

Figure 4 School creation by admin in school management view

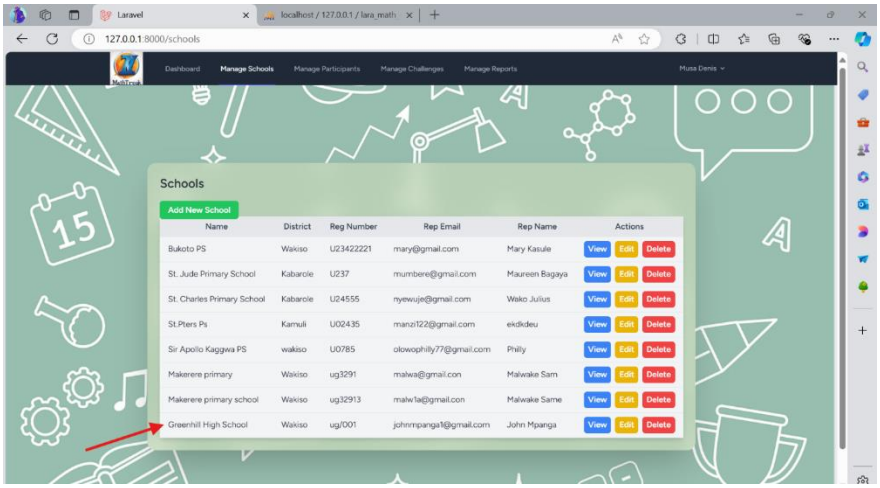

Figure 5 Schools table updated on every creation

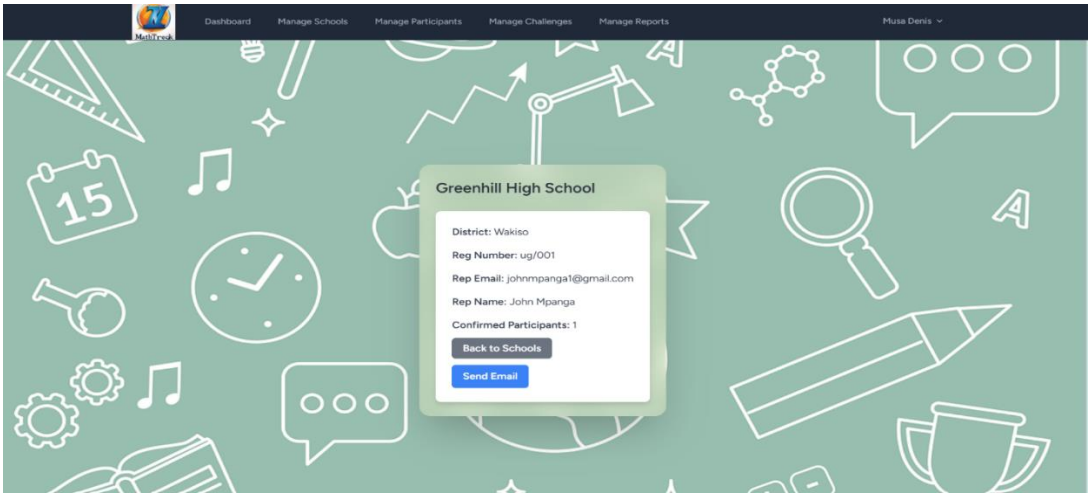

Figure 6 School view detailed with participant count

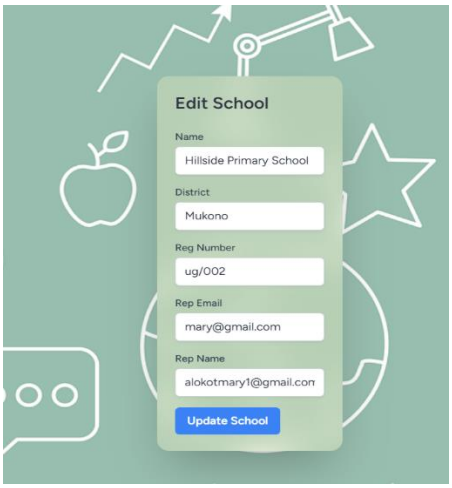

Figure 7 School detail edit process view

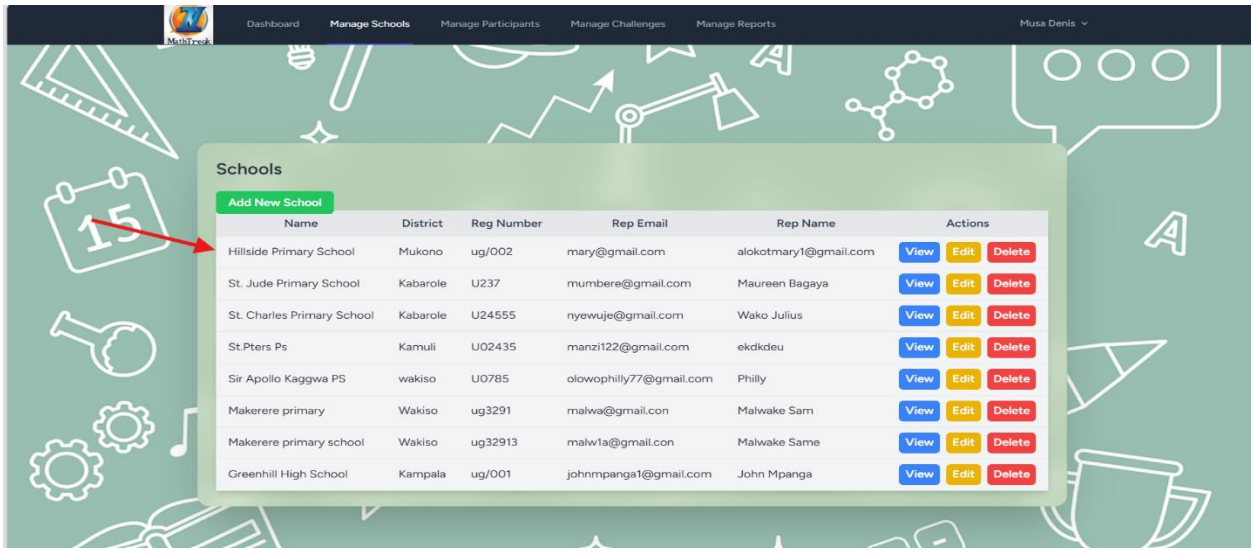

Figure 8 Updated school table with new details

- Challenge Management

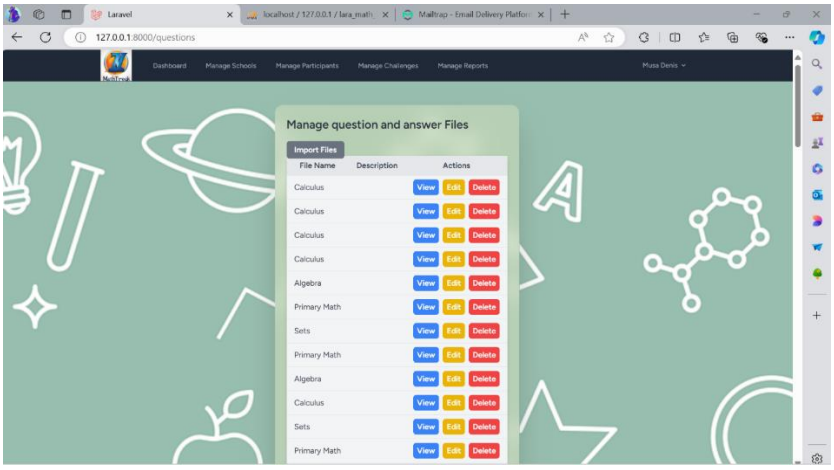

Figure 9 Question and answer file upload and management view

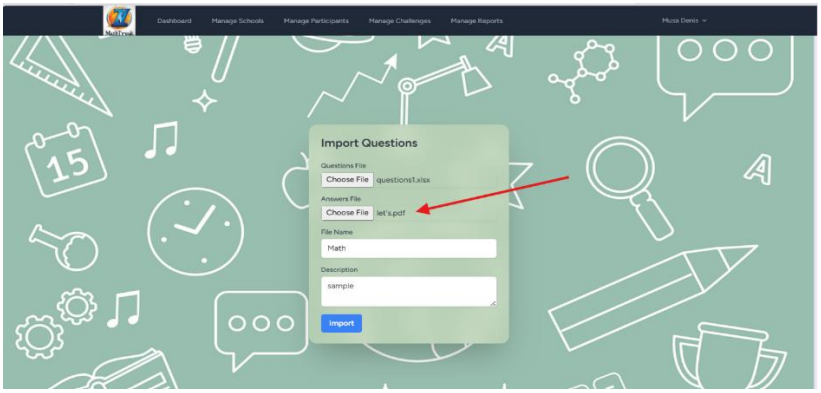

Figure 10 Example wrong file upload

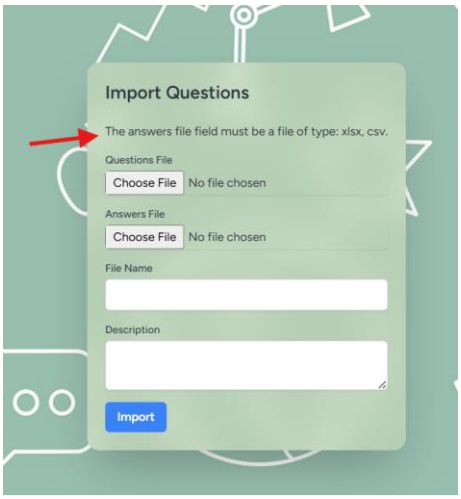

Figure 11 Error message

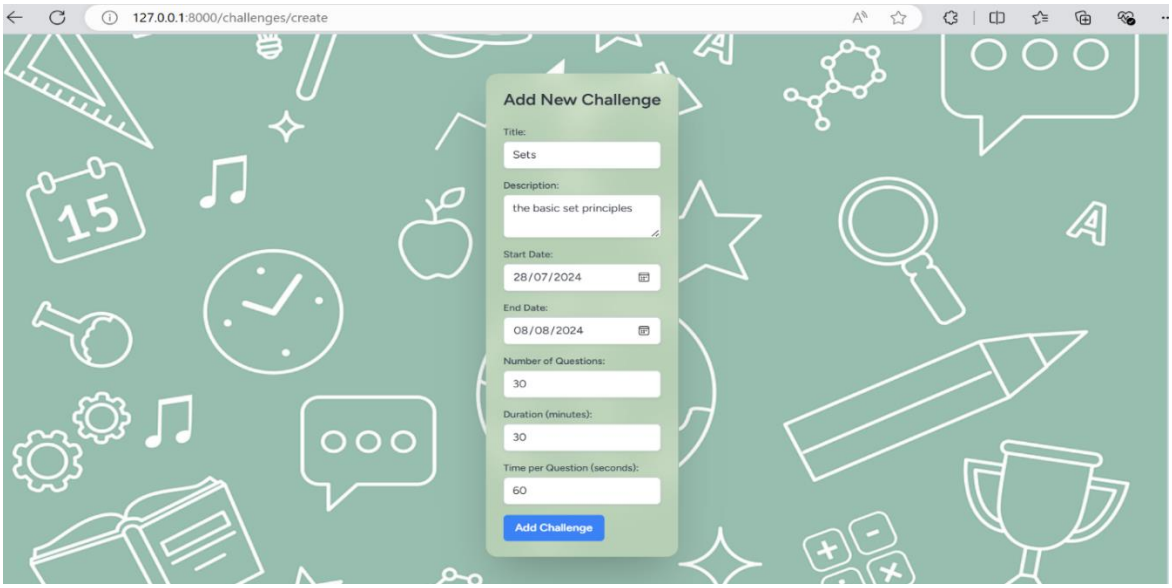

Figure 12 Challenge creation view

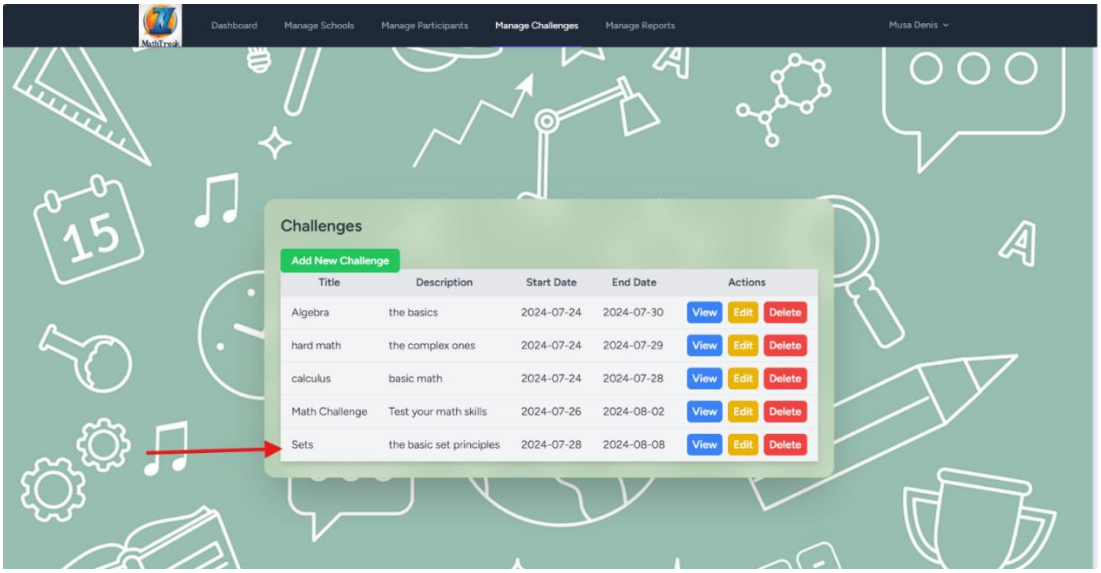

Figure 13 Created challenge view

- School representative management

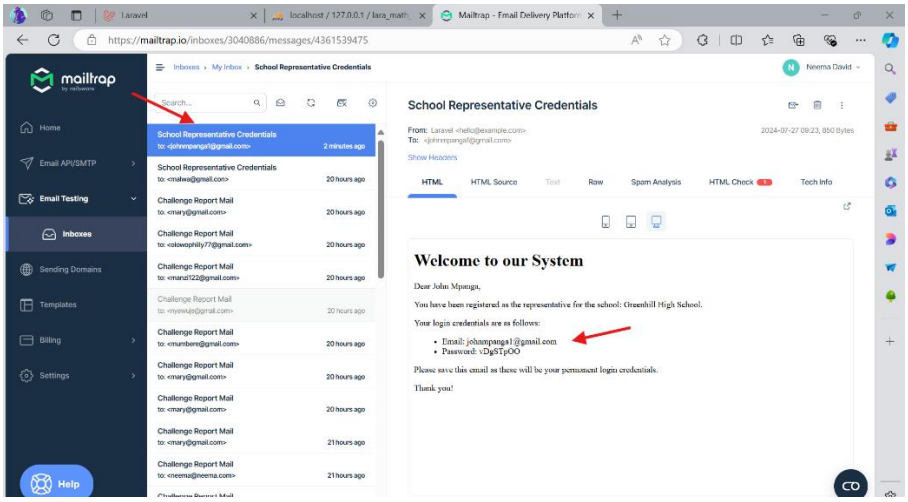

Figure 14 Sent email after school creation to school rep

- CLI

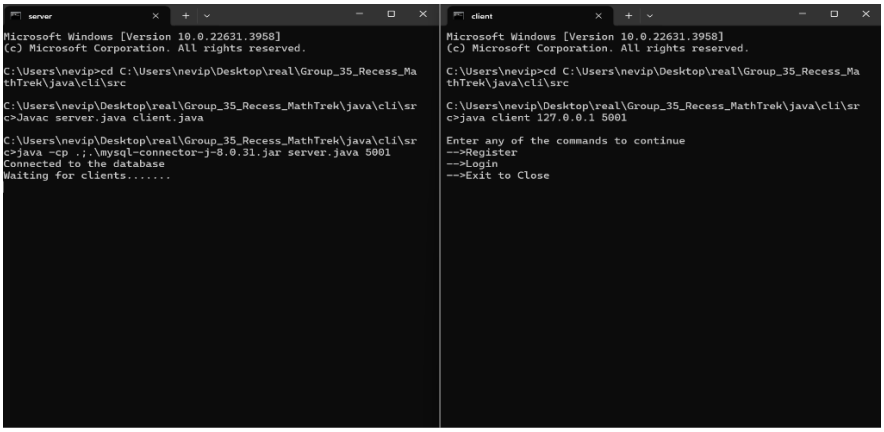

Figure 15 Initial views of server and client

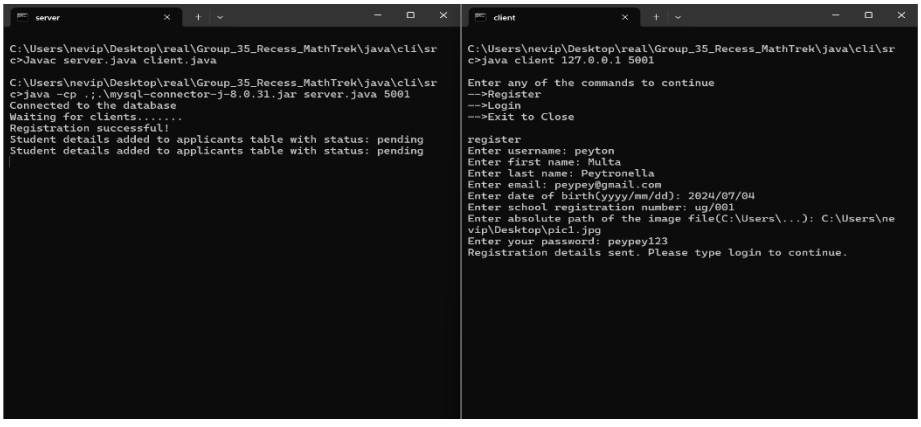

Figure 16 Student registration

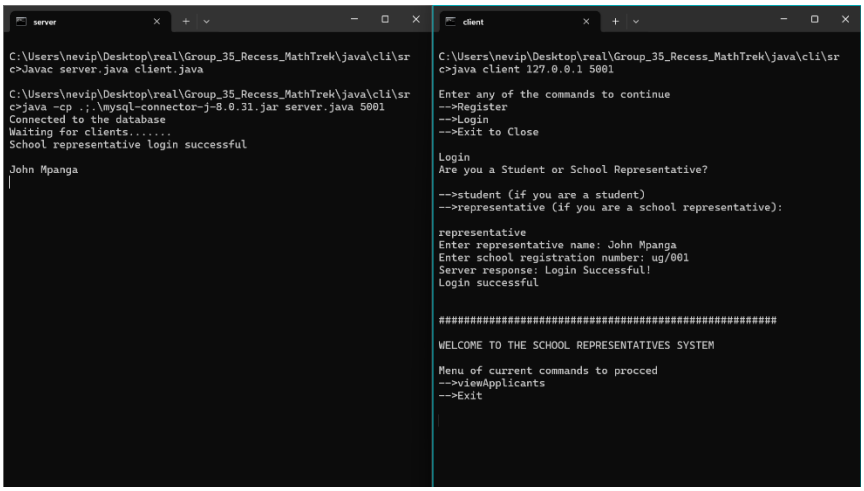

Figure 17 Representative login CLI

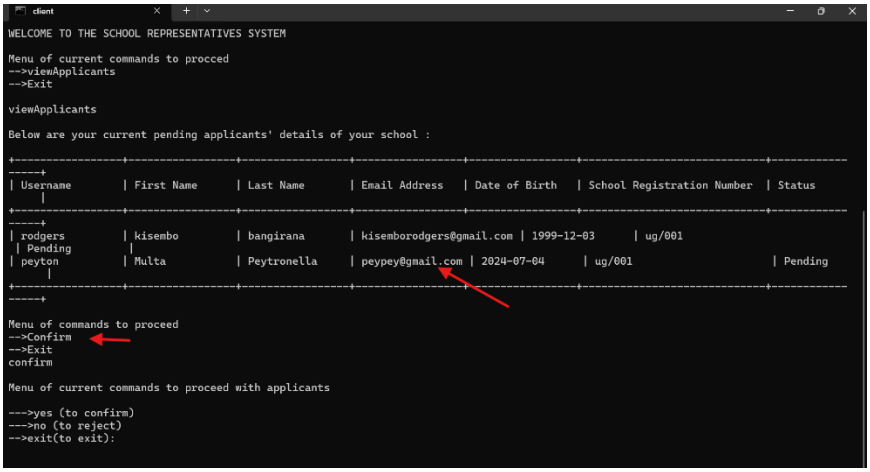

Figure 18 View and confirm applicant

Figure 19 Confirm by username since table has many

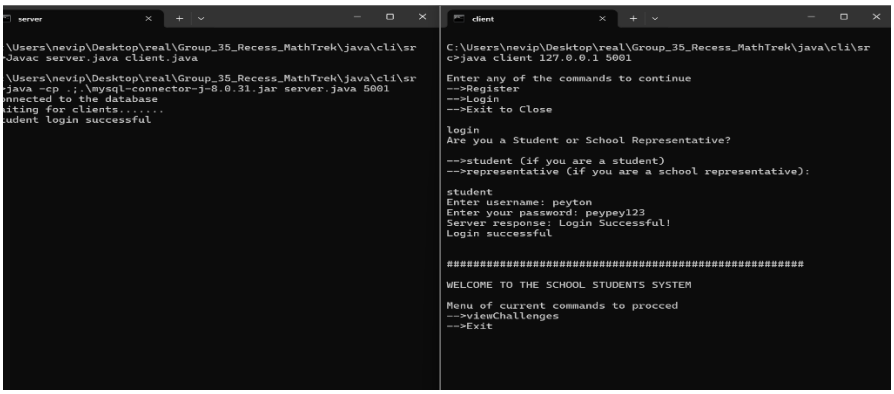

Figure 20 Student login

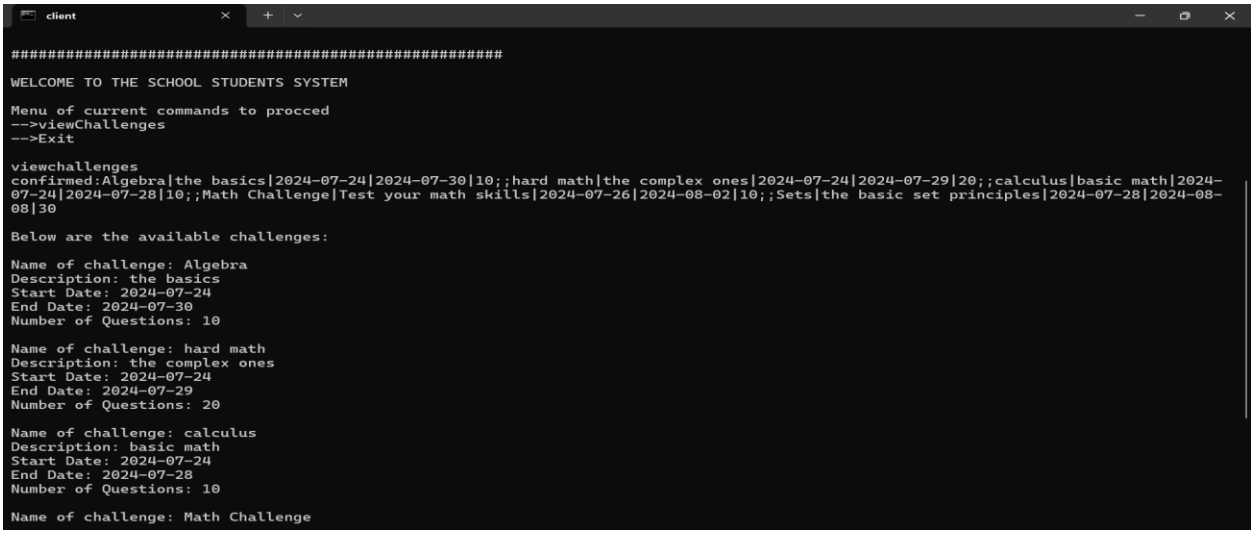

Figure 21 Successful student login and challenge view

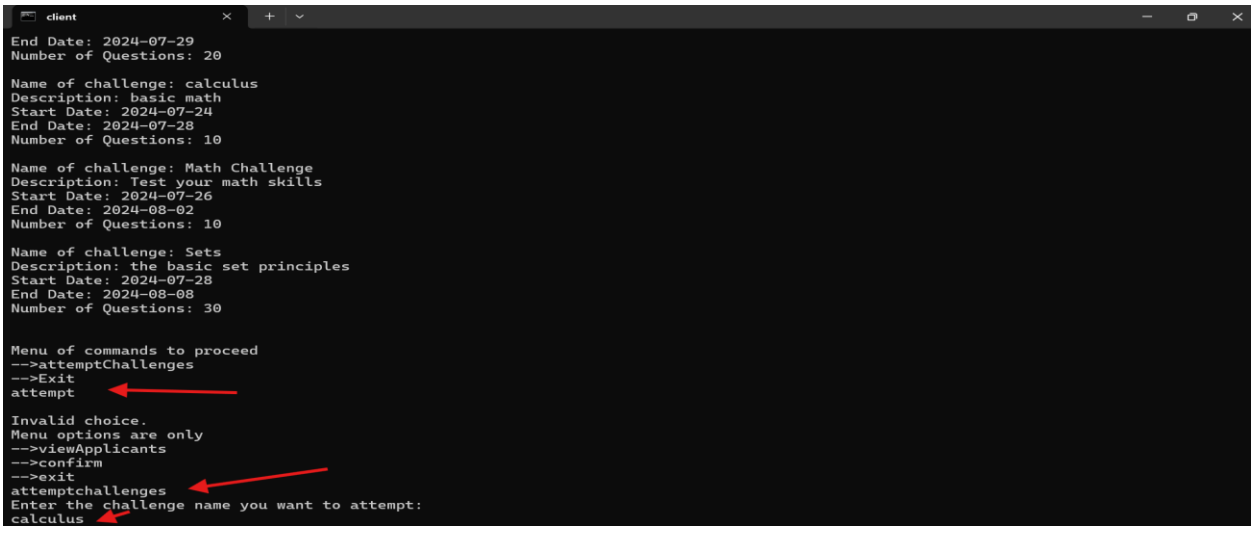

Figure 22 Challenge attempt selection

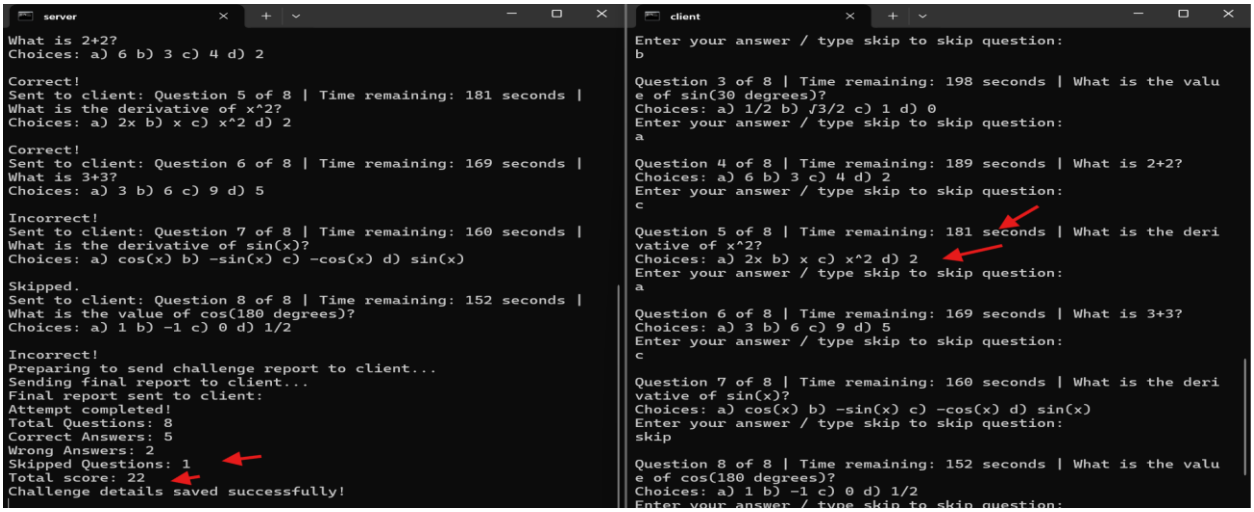

Figure 23 Challenge attempt view on server and client sides

- Reports and Analytics

Figure 24 Analytics view

Challenges and Solutions

Challenge 1: Vite Integration

Issue: Errors while importing chart.js.

Solution: Referred to documentation and online scenarios to ensure correct installation and import paths.

Challenge 2: Data Synchronization

Issue: Keeping Java client and Laravel backend in sync.

Solution: Implemented efficient database tables for data synchronization.

Results and references

Functional System: Successfully implemented a fully functional school management system.

User Satisfaction: Positive feedback from test users who included ourselves and some course-mates.

Installation - Laravel 11.x - The PHP Framework For Web Artisans

Namespaces | Laravel API

"spatie/simple-excel" Excel import/export (laracasts.com)

Mailtrap: Email Delivery Platform

Introduction | browsershot | Spatie

Conclusion

The project successfully met majority of its objectives, providing a robust system for managing schools, participants, and challenges. The integration of analytics and reporting features adds significant value, helping administrators make informed decisions. Future enhancements may include adding more detailed analytics, improving the user interface, and expanding the system's capabilities such as storage and accessibility over a broad network.