MERN With Generative AI

Course-End Project

Securing and Enhancing a MERN-Based Car Booking App With AI-Driven Development and Testing

Objective

To design and implement a secure development environment for a MERN-based car booking application by configuring backend infrastructure, managing user access, logging activity, and utilizing AI-driven tools for design, development, and testing

Description:

A real-world scenario is simulated in which a project team is required to secure backend infrastructure to meet compliance requirements and ensure transparent system activity. Using a MERN stack project integrated with Stripe, learners will establish access controls, configure MongoDB, track command history, and log service usage. The project also uses AI-powered tools to improve UI, generate optimized code, and validate application functionality through automated testing. The complete solution ensures end-to-end visibility, secure access, and scalable project delivery.

COURSE-END PROJECT

Problem Statement and Motivation

Real-time scenario:

An enterprise software team is developing a car booking platform powered by AI and integrated with payment gateways. Midway through the project, an internal audit flags risks related to unsecured system directories, missing activity logs, and hardcoded API keys. Additionally, the frontend interface lacks polish, and the QA process relies heavily on manual checks, delaying releases.

To ensure compliance and streamline development, the engineering lead initiates a remediation process. The goal is to enforce secure file access, configure database environments, track user actions via command history, and generate web-based reports. AI tools are leveraged to modernize the user interface, automate backend logic creation, and validate functionality through natural language-based test automation.
This approach provides a robust, scalable, and secure foundation for real-time collaborative projects.

Industry Relevance

The following tools and technologies used in this project reflect current best practices for secure and efficient software delivery:

1. **MERN Stack (MongoDB, Express.js, React, Node.js):** Enabling full-stack development for modern web applications
2. **Visual Studio Code and GitHub Copilot**: Providing integrated development and AI-assisted coding support
3. **Stripe API**: Managing secure payment integration and test transactions
4. **Uizard**: Supporting AI-driven UI design and component generation
5. **testRigor**: Facilitating no-code, natural language test creation and automated validation
6. **Command Line Interface and MongoDB Shell**: Handling secure backend configuration, logging, and directory management

Tasks

The following tasks outline the steps to secure backend services, manage access, enhance design, and validate features:

1. Clone and set up the backend
2. Configure MongoDB and the backend environment
3. Install frontend dependencies
4. Set up Stripe API key and start services
5. Test the application functionality
6. Enhance UI and generate AI-assisted code
7. Integrate GitHub Copilot and use AI prompts
8. Generate backend schemas and frontend components using AI
9. Add custom logic and optimize code using GitHub Copilot
10. Analyze and test the application using AI tools

COURSE-END PROJECT

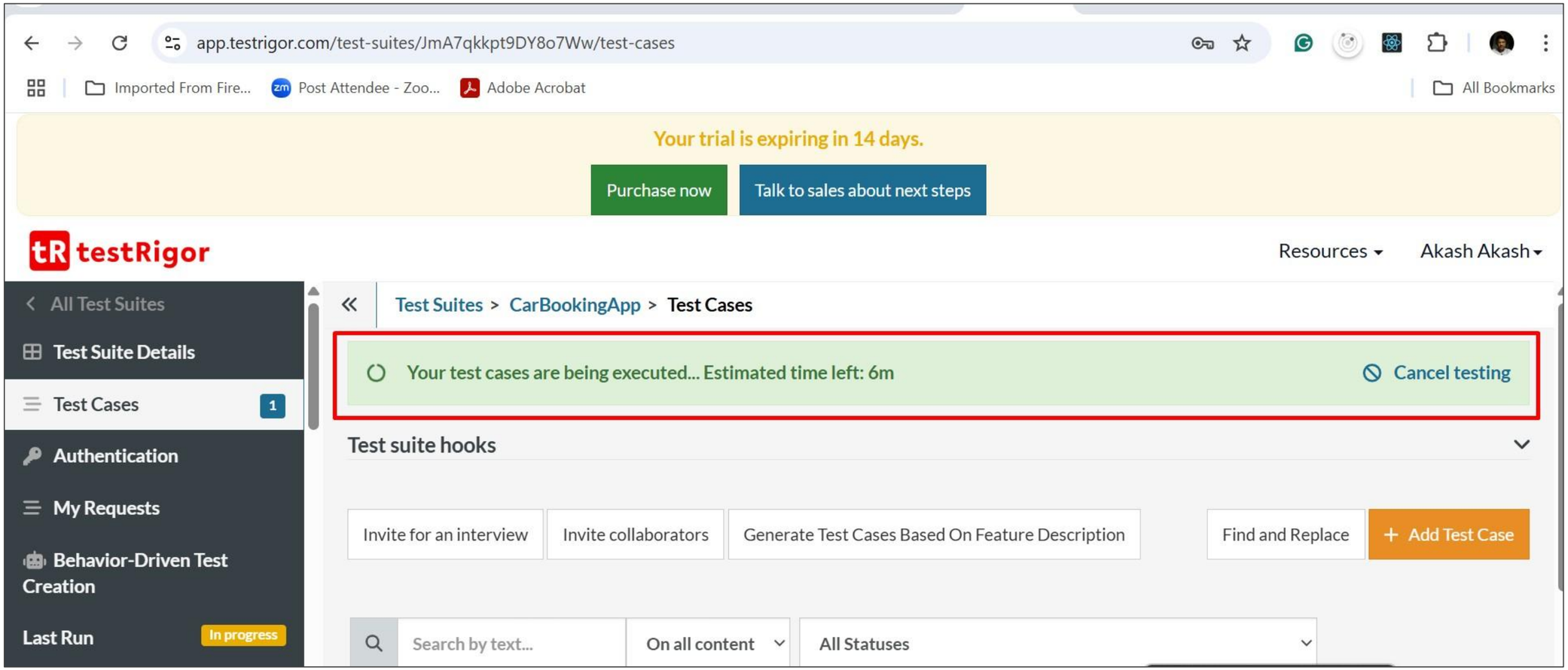
Project References

- Task 1: Lesson 1 and 2
- Task 2: Lesson 3
- Task 3: Lesson 3
- Task 4: Lesson 5
- Task 5: Lesson 2
- Task 6: Lesson 4 and 5
- Task 7: Lesson 2, 5, and 6
- Task 8: Lesson 6 and 7
- Task 9: Lesson 7
- Task 10: Lesson 6

COURSE-END PROJECT

Output Screenshots

AI-generated test case in testRigor:



Thank you