E Commerce Web Application

1. INTRODUCTION

1.1 Project Name

Project - ShopEZ : E-commerce Application

1.2 Purpose

This project report details the development and security analysis of the ShopEZ E-commerce Application. The primary purpose of this application is to provide a user-friendly and secure platform for customers to browse, select, and purchase products online.

Abstract-

The abstract of this project is to create a responsive, full-stack e-commerce solution with essential features like user authentication, product management, and a seamless checkout experience.

Scope of the project-

The scope of the project includes the design and implementation of both the frontend user interface and the backend server-side logic, as well as a preliminary security assessment using industry-standard tools.

List of teammates-

S.no	name	collage	contact
1	Dnyaneshwari Bhauso Patil	DY Patil Agriculture and Technical University, Talsande	8087509694
2	Gauri Bhauso Tanugade		8625936671
3	Shruti Sandip Patil		93220 19014
4	Aarati Ankush Pilaware		95031 76019

E Commerce Web Application

2. IDEATION PHASE

2.1 Thought Behind the Project

The core idea behind the ShopEZ project was to create a modern, scalable, and responsive e-commerce platform from scratch. The team focused on several key concepts: a minimalist user interface for a clean shopping experience, a robust MERN stack backend to handle data efficiently, and a modular design to allow for future feature expansion. Early discussions centered on what makes a great online store, including fast loading times, intuitive navigation, and reliable security.

2.2 Features

The following core features were identified and developed for the application:

- **User Authentication:** Secure sign-up, login, and logout functionalities.
- **Product Catalog**: A comprehensive list of products with details, images, and pricing.
- **Shopping Cart**: The ability for users to add, remove, and update products in their cart.
- **Checkout Process**: A simple, multi-step checkout to finalize purchases.
- **Order History:** A user-specific page to view past orders and their status.
- **Admin Dashboard**: A secure portal for administrators to manage products, users, and orders.

2.3 Empathy Map

User Persona: The Busy Professional

- **SAYS:** "I need to find what I want quickly." "Is this website secure?" "I'm always running out of time."
- **THINKS:** "I hope this process isn't too complicated." "I want to get in and get out with my purchase." "Is this a brand I can trust?"
- **FEELS**: Impatient, concerned about security, relieved when the checkout is fast.
- **DOES:** Skims product descriptions, uses search functionality, checks for reviews, abandons cart if the process is too long. The empathy map guided the design to prioritize speed, simplicity, and clear security messaging.

3. REQUIREMENT ANALYSIS

3.1 List of Vulnerabilities

A threat model was developed to identify potential vulnerabilities, including:

- **Cross-Site Scripting (XSS):** Allowing malicious scripts to be injected into the application.
- **Insecure Direct Object References (IDOR):** Exposing internal object IDs, allowing unauthorized access.

- **Injection Flaws:** SQL injection, which can be a risk with MongoDB if queries are not properly sanitized.
- **Broken Authentication s Session Management:** Weak session token generation or storage.
- **Weak Password Hashing**: Using insecure algorithms for storing user passwords.

3.2 Solution Requirement

To address the identified vulnerabilities, the following security requirements were defined:

- **Input Validation:** Strict server-side validation for all user inputs to prevent injection attacks.
- **Data Sanitization**: Sanitizing user input to prevent XSS attacks.
- **Secure Authentication:** Using JSON Web Tokens (JWT) for stateless authentication and secure storage.
- **Password Hashing:** Employing robust hashing algorithms like bcrypt to securely store user passwords.
- **Authorization Checks:** Implementing checks on the backend to ensure users can only access their own data.

3.3 Technology Stack

The project was developed using the MERN stack, a popular full-stack JavaScript framework.

- **Frontend**: React.js for building a dynamic and responsive user interface.
- **Backend**: Node.js and Express.js for creating a RESTful API to handle business logic.
- **Database:** MongoDB, a NoSQL database, for flexible and scalable data storage.
- **Security Testing Tool:** Nessus, a widely used vulnerability scanner, was used for a static and dynamic analysis of the application.

4. PROJECT DESIGN

4.1 Overview of Nessus

Nessus is a powerful and versatile vulnerability scanner. For the ShopEZ project, Nessus was used to perform automated scans on the deployed application to identify security weaknesses. It helped to pinpoint issues like misconfigured HTTP headers, outdated dependencies in the node_modules directory, and potential injection points that were missed during manual code review.

4.2 Proposed Solution

Based on the Nessus scan findings, several low-to-medium severity vulnerabilities were identified. For instance, Nessus flagged a lack of security-focused HTTP headers (X-Content-Type-Options, X-Frame-Options). The proposed solution was to configure Express.js to include these headers in all responses. Additionally, a vulnerability related to an older version of a thirdparty npm package was found, which was resolved by updating the dependency to a secure version.

4.3 Understanding of E-commerce Application Security

The security of an e-commerce platform is paramount. Key concepts like a Security Operations Center (SOC) and a Security Information and Event Management (SIEM) system are crucial for a mature application. An SOC is a centralized unit that continuously monitors and analyzes an organization's security posture. A SIEM tool, like Splunk or ELK Stack, aggregates and analyzes log data from various sources to detect and alert on potential security threats. While not implemented in the project's initial scope, understanding these concepts is vital for the long-term security of the ShopEZ application.

5. PROJECT PLANNING s SCHEDULING

5.1 Project Planning

The project was planned using an agile methodology with three major sprints.

- **Sprint 1 (Weeks 1-2):** Focused on foundational elements, including setting up the MERN stack environment, implementing user authentication, and designing the core database schema.
- **Sprint 2 (Weeks 3-4):** Focused on developing the main e-commerce features, such as the product catalog, shopping cart logic, and the user's order history page.
- **Sprint 3 (Weeks 5):** Focused on the admin dashboard, finalizing the checkout process, and conducting the initial security scan and functional testing.

6. FUNCTIONAL AND PERFORMANCE TESTING

6.1 Vulnerability Report

A comprehensive vulnerability report was generated after running Nessus scans on the staging environment. The report categorized findings by severity. Initial scans revealed issues such as:

- **Outdated Node.js Packages:** A handful of low-severity vulnerabilities in dependencies were found.
- **Lack of Security Headers**: Nessus flagged the absence of several important HTTP security headers.
- **Directory Traversal**: A minor misconfiguration that could potentially lead to directory traversal.

The impact of these findings was assessed, and all issues were successfully patched by updating packages, adding middleware to handle security headers, and correcting the server configuration.

7. RESULTS

7.1 Findings and Reports

The project successfully delivered a functional e-commerce application. The functional testing confirmed that all features, from user registration to checkout, worked as intended. The final vulnerability assessment report from Nessus, after mitigation efforts, showed no high-severity vulnerabilities and only a few informational findings. This indicates that the application is secure enough for initial deployment and provides a solid foundation for future development.

8. ADVANTAGES s DISADVANTAGES

Advantages (Pros)

- **Unified Language:** Using JavaScript/TypeScript across the entire stack (frontend and backend) streamlined development and team collaboration.
- **Performance:** The MERN stack is well-suited for building fast and high-performance applications.
- **Scalability:** MongoDB's NoSQL nature allows for flexible schema design, making the application highly scalable.

Disadvantages (Cons)

- **Complexity:** Managing a full-stack JavaScript application with multiple moving parts can be complex.
- **Dependency Management:** Reliance on a large number of npm packages can introduce potential security risks and maintenance overhead if not managed carefully.

G. CONCLUSION

In conclusion, the ShopEZ E-commerce Application was successfully developed using the MERN stack, fulfilling all primary feature requirements. The project followed an agile methodology, allowing for efficient development and iterative improvements. Through a structured security assessment using Nessus, key vulnerabilities were identified and mitigated, resulting in a robust and secure application ready for deployment. The project provides a strong foundation for future enhancements and serves as a testament to the power and flexibility of the MERN stack.

10. FUTURE SCOPE

- **Payment Gateway Integration:** Implement a secure payment gateway (e.g., Stripe, PayPal) for real-world transactions.
- **Recommendation Engine:** Integrate a machine learning model to provide personalized product recommendations to users.
- **CI/CD Pipeline:** Automate the build, test, and deployment process for faster and more reliable updates.
- **Advanced Security**: Implement more advanced security measures such as a Web Application Firewall (WAF) and integrate with a SIEM tool for ongoing security monitoring.

11. APPENDIX

- **GitHub Link: https://github.com/gauritanugade/E-commerce**
- **Project Demo Link: https://shopez-cucq.onrender.com**