# ECO-SHOP

# E-Commerce Web Application Development Report

A green leaf with black text

AI-generated content may be incorrect.

**Course:** Web & Cloud Application Development

**Team Members:** Danillo Sampaio, Shauna Taylor, Thiago Abbud Coutinho

**Submission Date:** 27/05/2025  
**Instructor:** Basel

**Executive Summary**

The ECO-Shop project is a full-stack e-commerce web application designed to simulate a real-world online shopping platform centered on eco-friendly and sustainable products. The platform empowers users to browse and purchase environmentally responsible goods while providing a seamless and secure user experience.

The application adopts modern web technologies, combining a dynamic front-end interface with a robust back-end server and database management system. The system supports user account creation, session management, catalog navigation, product searching, and secure payments, ensuring a complete and professional e-commerce experience.

**1. Project Objectives**

* Develop a user-friendly e-commerce site with emphasis on sustainability.
* Implement responsive design to support various device viewports.
* Ensure secure authentication and data handling.
* Allow full CRUD functionality for user and product data.
* Provide real-time feedback for forms and search operations.
* Foster a modular codebase for future feature expansions.
* Emphasize eco-conscious branding and messaging throughout the interface.

**2. Front-End Development**

Technologies Used

* HTML5, CSS3, JavaScript (ES6)
* Bootstrap 5 for responsive UI design
* Material UI integrated in the React framework
* React.js for reusable component-based structure

Overview of Component Design

The React front-end is structured with modular components for key interface areas such as the navigation bar, homepage, product listings, and shopping cart. Components are reused across different views to ensure a consistent and efficient user interface.

User input forms for login and registration are validated using JavaScript and styled with Bootstrap classes. Shopping cart features support dynamic item addition and updates.

**Front-End Features**

* Responsive layout compatible with desktop and mobile
* Dynamic content routing for single-page application behavior
* Form validation with real-time feedback
* Visual enhancements such as sliders and modals for user engagement

**3. Back-End Development**

Technologies Used:

* Node.js
* Express.js framework
* EJS (when not using React server-side rendering)
* Validator.js for input sanitization
* express-session for managing user sessions

**Server Features**

* RESTful API endpoints for user actions and admin controls
* Middleware for logging, error handling, and CORS support
* Session handling with cookie-based login tracking
* Static file serving and environment-based configuration

Authentication is performed by validating credentials against stored user data and managing login sessions accordingly.

**4. Database Design and Integration**

Platforms Used

* MySQL for primary relational data storage
* Firebase Realtime DB as an optional NoSQL/cloud sync backup

Database Schema Overview

Entities include:

* User: Stores user profile and credentials
* Product: Contains product listings and metadata
* Order: Records user purchase history
* OrderItem: Maps product quantities to specific orders

Relationships are maintained with foreign keys to ensure data integrity and traceability between users, products, and orders.

**5. Entity-Relationship Diagram (ERD)**

**Main Entities**

1. **User**Attributes:
   * user\_id (PK)
   * username
   * email
   * password
   * address
   * phone  
     *Stores user profile details and authentication credentials.*
2. **Product**Attributes:
   * product\_id (PK)
   * name
   * description
   * price
   * stock
   * category
   * eco\_certified  
     *Represents eco-friendly products, including certification and inventory data.*
3. **Order  
   Attributes:**
   * order\_id (PK)
   * user\_id (FK)
   * order\_date
   * status
   * total\_amount  
     *Records each user's purchase transactions.*
4. **OrderItem  
   Attributes:**
   * order\_item\_id (PK)
   * order\_id (FK)
   * product\_id (FK)
   * quantity
   * price  
     *Maps products and quantities to specific orders.*

A diagram of a product

AI-generated content may be incorrect.

Shows the relationships between user accounts, product category, and order records.

**6. Security and Session Management**

* Session management via express-session with secure cookies
* Input validation using Validator.js
* Recommendation to deploy over HTTPS for encrypted data transmission
* Plans for implementing password hashing and role-based access control

**7. Deployment and Testing**

**Deployment**

* Development using localhost with live reload via nodemon
* Firebase Hosting for production deployment
* Docker compatibility considered for future scalability

**Testing Strategies**

* Manual browser testing for layout and functionality
* Unit testing for server routes and functions
* Simulated user journey testing (registration to checkout)
* Responsiveness tests across devices and screen sizes

**8. Future Improvements**

* Integration with online payment systems like Stripe or PayPal
* Admin dashboard for product and order analytics
* User reviews and product ratings
* Personalized recommendations using AI
* Progressive Web App (PWA) capabilities
* Enhanced order tracking and customer notifications
* Wishlist and product save features

**9. Conclusion**

The ECO-Shop project has enabled the team to apply a wide range of web development principles in a practical and meaningful way. The result is a fully functioning e-commerce application with a strong sustainability focus. Future developments can build on this foundation by incorporating advanced features and scalable infrastructure.

**10. Appendix**

* GitHub Repository:
* Firebase Console:
* ERD Diagram:
* Screenshots and Demo Video:
* Component Descriptions:

**11. Development Methodology**

The team followed Agile principles with weekly sprints. Tools like Zoom and Whatsapp were used to manage tasks, and GitHub ensured collaborative version control. Regular reviews and sprint retrospectives helped the team improve iteration by iteration.

**12. Personas and Use Cases**

**Client 1 – Laura, 29, Environmental Activist**

* Goal: Buy eco-friendly home products.
* Interaction: Registers, browses "Homeware" category, adds items to cart, and completes purchase.

**Client 2 – Paul, 35, Small Business Owner**

* Goal: Find suppliers of sustainable goods.
* Interaction: Uses search, filters products, and reviews prices and details.

**13. Sustainability and Social Responsibility**

ECO-Shop promotes environmental awareness by curating products from eco-certified suppliers. Efficient front-end design reduces server load, and planned use of green hosting aligns with our sustainability mission. Features like dark mode also support energy efficiency on OLED displays.

**14. Project Timeline**

|  |  |
| --- | --- |
| Week | Milestone |
| 1 | Scope definition, tech stack selection |
| 2 | Database design and UI wireframing |
| 3 | Authentication and routing setup |
| 4 | Cart system, product CRUD features |
| 5 | Final testing and responsive layout adjustments |
| 6 | Report writing and presentation preparation |

**15. Team Reflection**

Through the ECO-Shop project, we enhanced our skills in full-stack development, teamwork, and agile planning. Each member contributed uniquely to overcome challenges like API integration, UI responsiveness, and data synchronization. The experience strengthened both our technical and collaborative abilities.

**16. References**

* MDN Web Docs — https://developer.mozilla.org/
* React Documentation — https://reactjs.org/
* Express.js Guide — https://expressjs.com/
* Firebase Docs — https://firebase.google.com/docs
* Bootstrap Docs — https://getbootstrap.com/