Task Report: FoodTruck Q-Commerce Website Development

Project Overview

Project Name: FoodTruck Q-Commerce Website

Objective: Build a user-friendly, responsive website for ordering fast food. The website will allow users to log in, sign up, place orders, and leave comments, while admins can view reports.

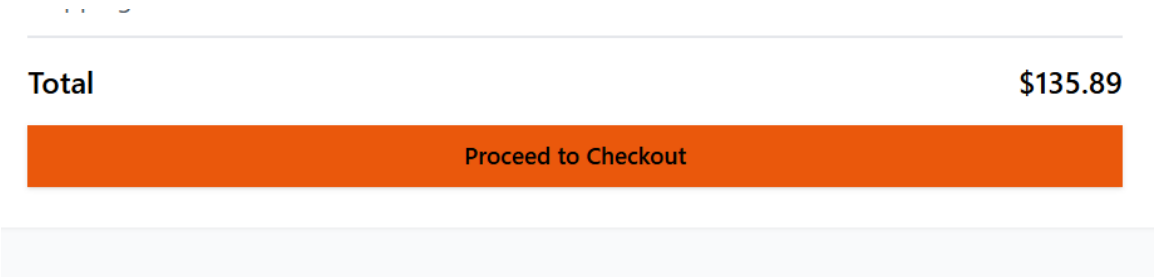
Features Developed

User Features:

```
{ name: "Product", href: "/products"},
{ name: "login", href: "/login"},
{ name: "SignUp", href: "/SignUp"}
```

Login and Signup Pages: Integrated Clerk middleware for easy login and signup.

User Functions:



Users can place orders and add comments.

Once logged in, users won't have to re-enter their details.

Admin Dashboard:

Admins can see:

Daily, monthly, and yearly reports.

User details and order information with timestamps.

Backend Integration:

APIs:

Product data is fetched from Sanity.io.

User and order data are stored in Sanity.

Real-time updates for order information.

Responsiveness:

The website works well on all devices (mobile, tablet, desktop).

Challenges Faced

Time Management:

It was hard to complete the task on time while managing job responsibilities.

Middleware Complexity:

Integrating Clerk middleware was difficult because I had no prior experience with it.

Testing and Deployment

Completed Features:

Login/signup, API integration, and admin reports are ready.

Pending Tasks:

Final testing after deployment.

Deployment on Vercel.

Testing Tools:

Functional testing and API testing will be done once deployed.

Project Insights

Learnings:

Learned how to use Clerk middleware and work with APIs.

Gained a better understanding of the challenges involved in real projects, especially with backend management and responsive design.

Feedback:

The hackathon was a great learning experience, but it showed the need for better time management, especially for students with jobs.

Conclusion

This project was a valuable learning experience, even though the time was limited. The completed features—like middleware integration, backend setup, and responsiveness—provide a solid base for future development. Next steps include final testing, deployment, and performance improvements.