**Assignment 1:**

Do a small sized e commerce based web based full stack project, with suitable front end, back code logic and a simple sql based database and using VS Code. This should create multiple files and edit them simultaneously with your prompt

**Approach:**

Prepare well structured prompt with detailed instructions using MS365 Copilot.

Use that prompt with Github copilot in ‘Agent’ mode inside VS Code

Do follow-up instructions for fixing issues and refining website

**Tech Stack used:**

**Frontend**

* React 18 + TypeScript
* Vite (dev/build)
* React Router v6
* @tanstack/react-query (data fetching/cache)
* Zustand (state management, persisted)
* Axios (HTTP)
* Tailwind CSS + PostCSS + autoprefixer (styling)
* Testing: Jest, Testing Library (react, user-event), ts-jest

**Backend**

* Node.js + Express (TypeScript, tsx for dev)
* Prisma ORM (SQLite in dev)
* Auth: jsonwebtoken (JWT), bcryptjs (password hashing)
* Middleware: cors, helmet, morgan, compression, express-rate-limit, express-validator, dotenv
* File uploads and images: multer, sharp
* Testing: Jest, Supertest

**Database**

* SQLite (via Prisma; [dev.db](vscode-file://vscode-app/Users/vasurekha/Applications/Visual%20Studio%20Code.app/Contents/Resources/app/out/vs/code/electron-browser/workbench/workbench.html" \o "))

**Tooling & DX**

* TypeScript 5, ESLint, Prettier
* Docker/Docker Compose
* VS Code Dev Container configuration

**Prompt used to generate prompt:**

Provide me an agentic prompt to develop a small fully functional e commerce website to sell Clay artisan products of different categories to be developed with VS code in dev container environment using components version optimised for my Docker desktop version v24.0.6 running in my Mac Book. Make sure the dev container has all required privilege to read, write, install required components and data base. The website shall have colour full and pleasant layout with appropriate icons for all menu items, free stock images from internet for few sample products

**First prompt used with Github Copilot Agent:**

|  |
| --- |
| # Agentic Prompt: Clay Artisan E-commerce Website Development  ## Project Overview  Develop a fully functional e-commerce website to sell clay artisan products across different categories. The development environment should be VS Code with dev containers optimized for Docker Desktop v24.0.6 on macOS.  ## Technical Requirements  ### Development Environment  - \*\*IDE\*\*: VS Code with Dev Container extension  - \*\*Container Platform\*\*: Docker Desktop v24.0.6 (macOS compatible)  - \*\*Container Privileges\*\*: Full read/write/install permissions for development needs  - \*\*Database Access\*\*: Complete database management capabilities within container  ### Technology Stack Selection  Choose and implement a modern, scalable technology stack such as:  - \*\*Frontend\*\*: React.js/Next.js or Vue.js/Nuxt.js with responsive design  - \*\*Backend\*\*: Node.js with Express.js or Python with FastAPI/Django  - \*\*Database\*\*: PostgreSQL or MongoDB with proper indexing  - \*\*Authentication\*\*: JWT-based authentication system  - \*\*Payment Integration\*\*: Stripe or PayPal sandbox integration  - \*\*Image Storage\*\*: Local storage with optimization  ### Dev Container Configuration  Create a comprehensive `.devcontainer/devcontainer.json` with:  - Base image compatible with Docker Desktop v24.0.6  - All necessary extensions (ESLint, Prettier, Database viewers)  - Required runtime environments and package managers  - Database services (PostgreSQL/MongoDB)  - Port forwarding for development servers  - Volume mounts for persistent data  - Proper user permissions and sudo access  ## Functional Requirements  ### Core E-commerce Features  1. \*\*Product Management\*\*  - Product catalog with categories (bowls, vases, decorative items, kitchenware, etc.)  - Product variants (size, color, glaze type)  - Inventory tracking  - Product search and filtering  2. \*\*User Management\*\*  - User registration and authentication  - User profiles and order history  - Admin panel for product and order management  3. \*\*Shopping Experience\*\*  - Shopping cart functionality  - Wishlist/favorites  - Product reviews and ratings  - Related product suggestions  4. \*\*Order Processing\*\*  - Checkout process with multiple payment options  - Order confirmation and tracking  - Email notifications  - Invoice generation  ### Database Schema Design  Design comprehensive database schema including:  - Users (customers, admins)  - Products (with categories, variants, pricing)  - Orders (with line items, status tracking)  - Reviews and ratings  - Inventory management  - Shopping cart sessions  ## Design Requirements  ### Visual Design  - \*\*Color Scheme\*\*: Warm, earthy tones reflecting clay artisan aesthetic (terracotta, cream, sage green, warm browns)  - \*\*Layout\*\*: Clean, modern, mobile-first responsive design  - \*\*Typography\*\*: Readable fonts that complement artisan theme  - \*\*Navigation\*\*: Intuitive menu structure with breadcrumbs  ### UI/UX Elements  - \*\*Icons\*\*: Implement appropriate icons for all menu items using icon libraries (Feather Icons, Heroicons, or Font Awesome)  - \*\*Product Display\*\*: Grid and list views with high-quality image galleries  - \*\*Interactive Elements\*\*: Hover effects, smooth transitions, loading states  - \*\*Accessibility\*\*: WCAG 2.1 AA compliance  ### Sample Content Integration  - \*\*Product Images\*\*: Source and integrate free stock images from:  - Unsplash (pottery, ceramics, clay products)  - Pexels (artisan crafts, handmade products)  - Pixabay (clay art, pottery workshop images)  - \*\*Product Categories\*\*:  - Dinnerware (plates, bowls, mugs)  - Decorative (vases, sculptures, wall art)  - Functional (planters, storage, lighting)  - Custom/Personalized items  ## Implementation Tasks  ### Phase 1: Environment Setup  1. Create dev container configuration with all required tools  2. Set up database with proper schema  3. Configure development server with hot reloading  4. Implement basic project structure and routing  ### Phase 2: Backend Development  1. Create REST API endpoints for all CRUD operations  2. Implement authentication and authorization  3. Set up database models and relationships  4. Create admin dashboard functionality  ### Phase 3: Frontend Development  1. Build responsive UI components  2. Implement product catalog and search functionality  3. Create shopping cart and checkout flow  4. Design and implement user dashboard  ### Phase 4: Integration & Testing  1. Connect frontend with backend APIs  2. Implement payment gateway integration  3. Add email notification system  4. Perform comprehensive testing (unit, integration, e2e)  ### Phase 5: Optimization & Deployment  1. Optimize images and implement lazy loading  2. Add SEO meta tags and structured data  3. Implement caching strategies  4. Prepare production deployment configuration  ## Quality Assurance  ### Code Quality  - Implement ESLint and Prettier configurations  - Use TypeScript for type safety (if applicable)  - Follow established coding standards and conventions  - Include comprehensive code comments  ### Testing Strategy  - Unit tests for all major functions  - Integration tests for API endpoints  - End-to-end tests for critical user flows  - Cross-browser compatibility testing  ### Performance Optimization  - Image optimization and compression  - Code splitting and lazy loading  - Database query optimization  - Caching implementation  ## Documentation Requirements  - Comprehensive README with setup instructions  - API documentation (if building custom API)  - Database schema documentation  - User guide for admin panel  - Deployment instructions  ## Security Considerations  - Input validation and sanitization  - SQL injection prevention  - XSS protection  - Secure authentication implementation  - HTTPS enforcement  - Rate limiting for API endpoints  ## Deliverables  1. Complete dev container setup with all dependencies  2. Fully functional e-commerce website with all specified features  3. Admin dashboard for product and order management  4. Responsive design working across all devices  5. Sample products with free stock images  6. Complete documentation package  7. Testing suite with good coverage  Begin development by setting up the dev container environment, then proceed with backend API development, followed by frontend implementation, ensuring each phase is thoroughly tested before moving to the next.  Prompt me for confirmation to proceed making major changes |

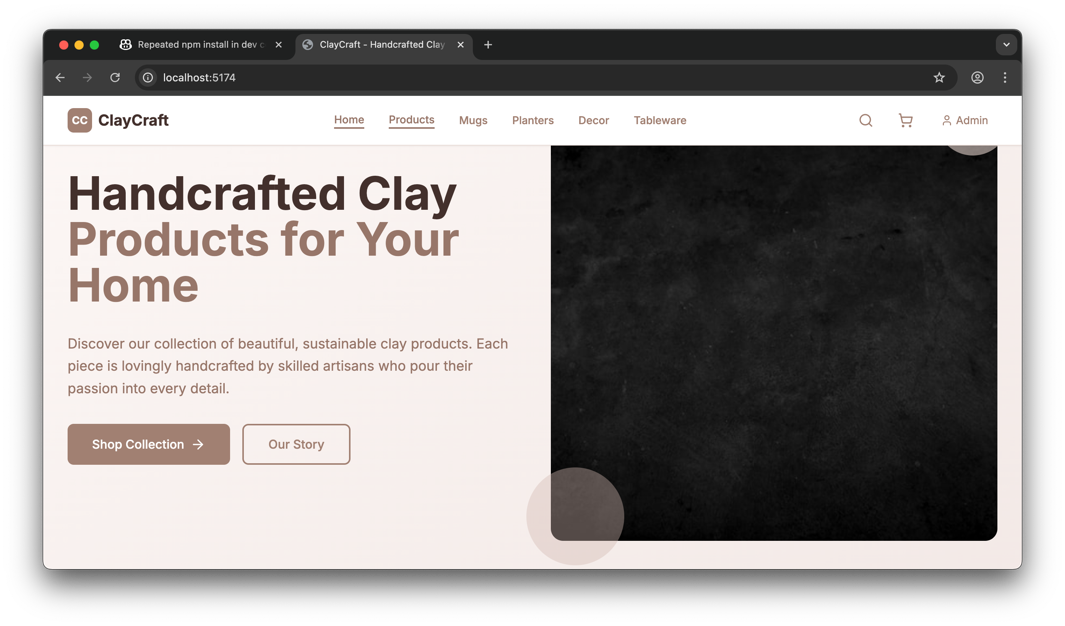
**Working with Gitgub Copilot Agent:**

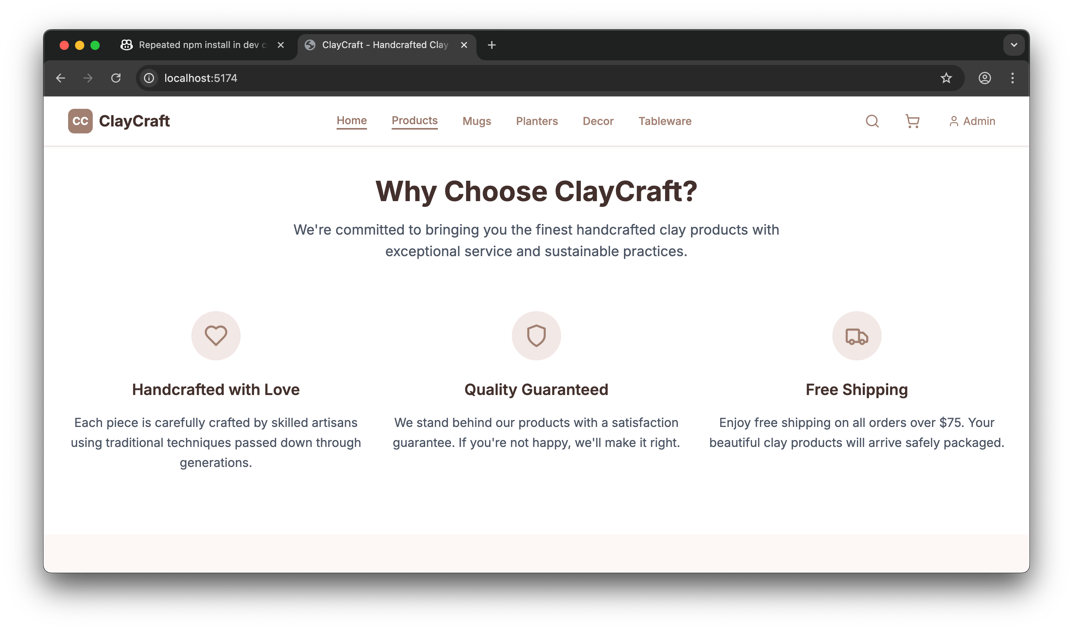
After working with agent in setting up development container, mount volume permission issues, fixing start up issues, able to get the web site up.

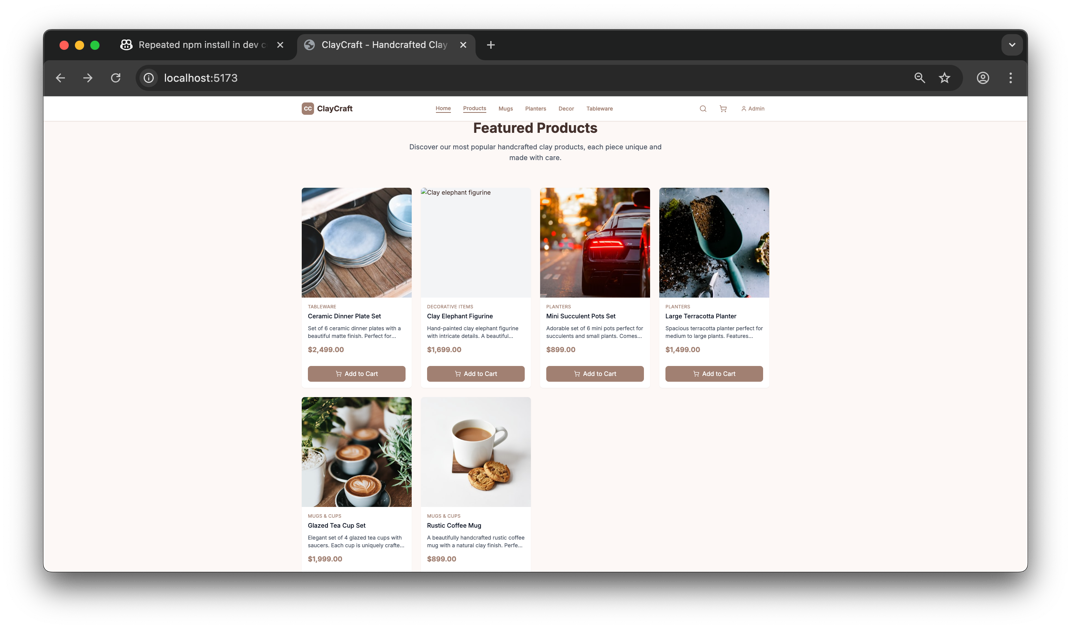
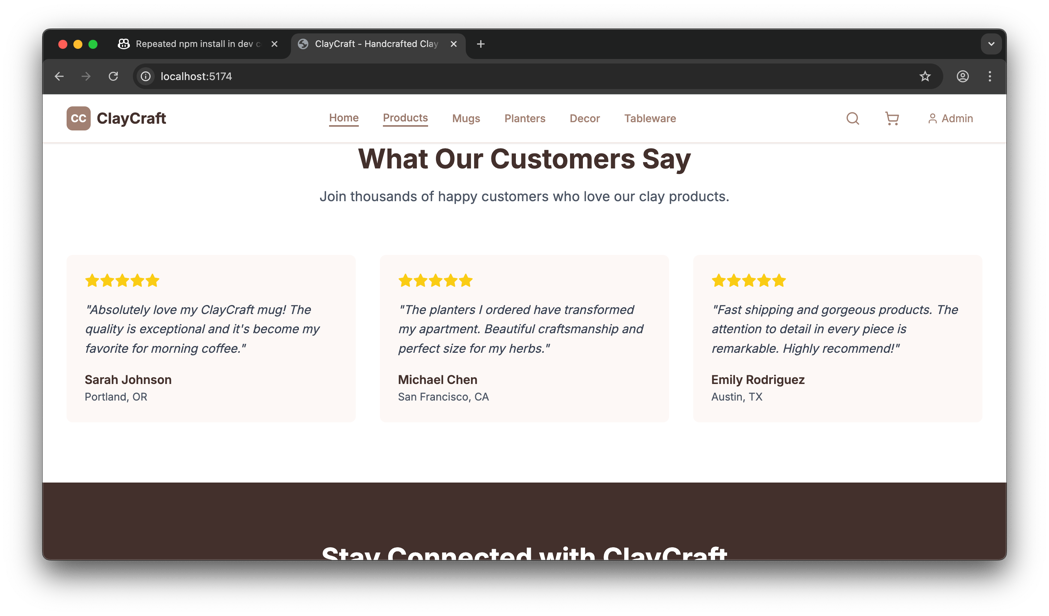
Used Claud Sonnet 3.5 with the Agent most of the time and had switched to GPT-5 later during midway of the project.

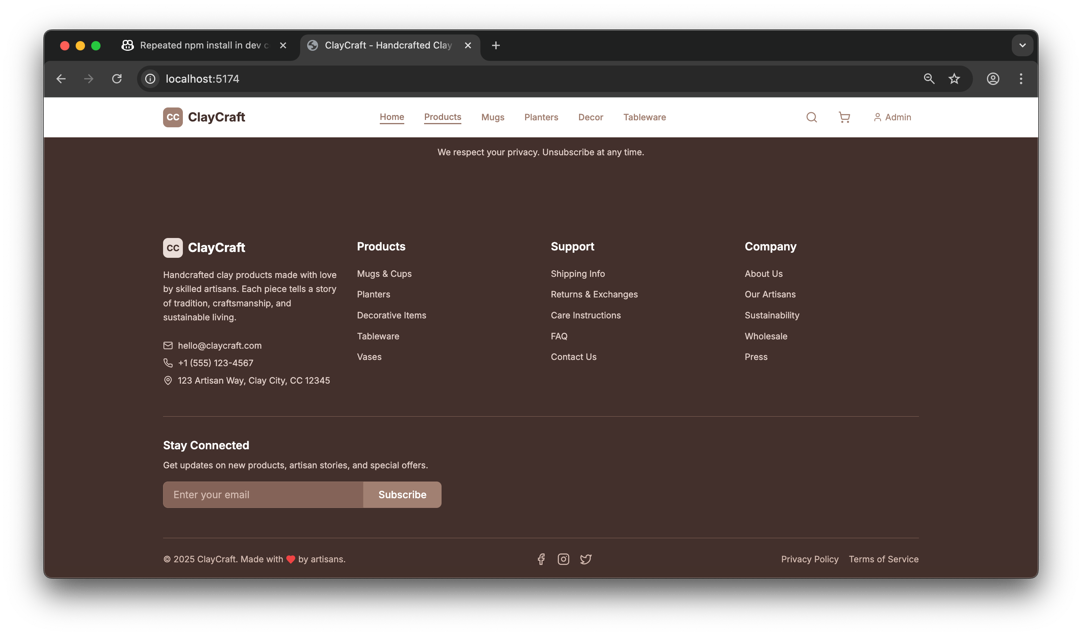
Then had tested different pages of the website and asked agent to fix issues identified. Asked it to load few items in the inventory. Amazed with the final output given the minimal amount of effort from myself never ever had developed a web application.

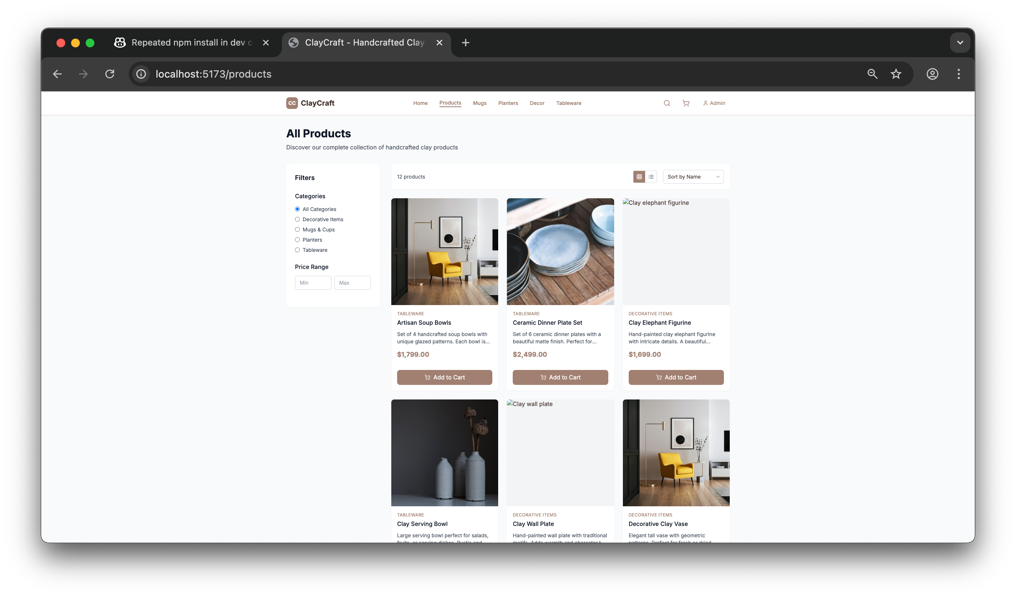
**Few screenshots from the website developed:**

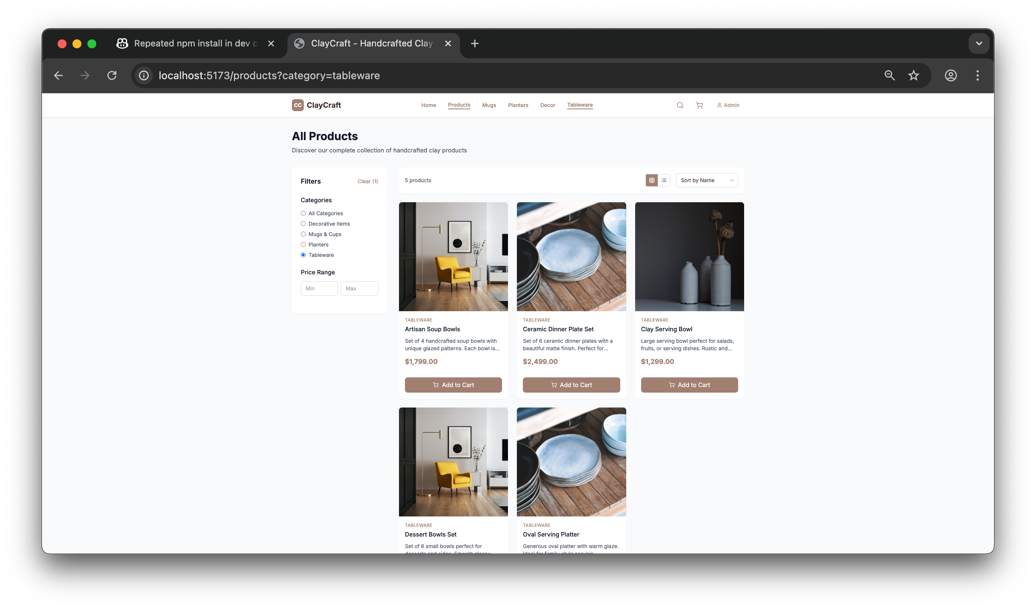


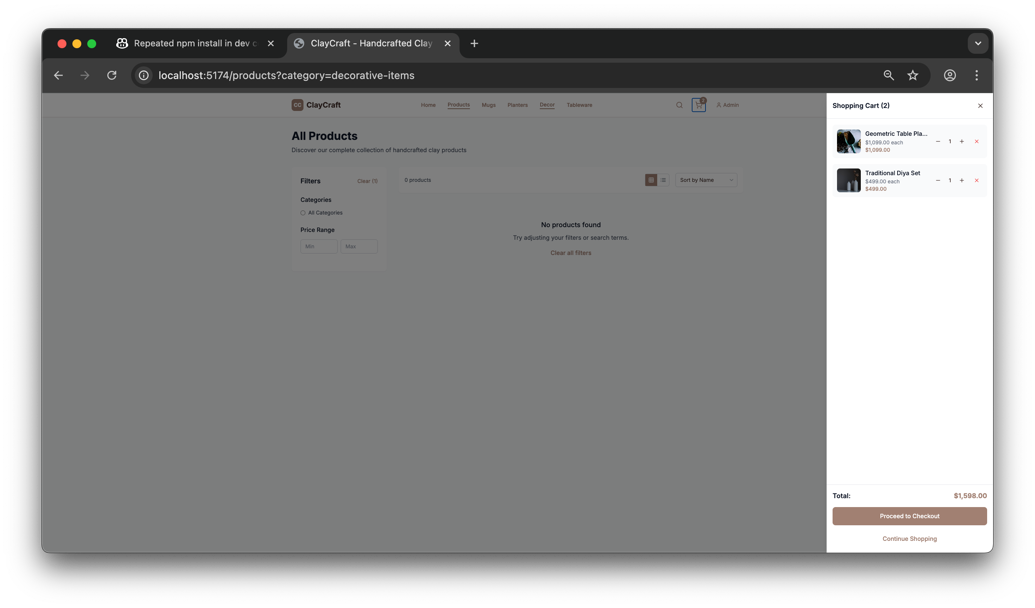


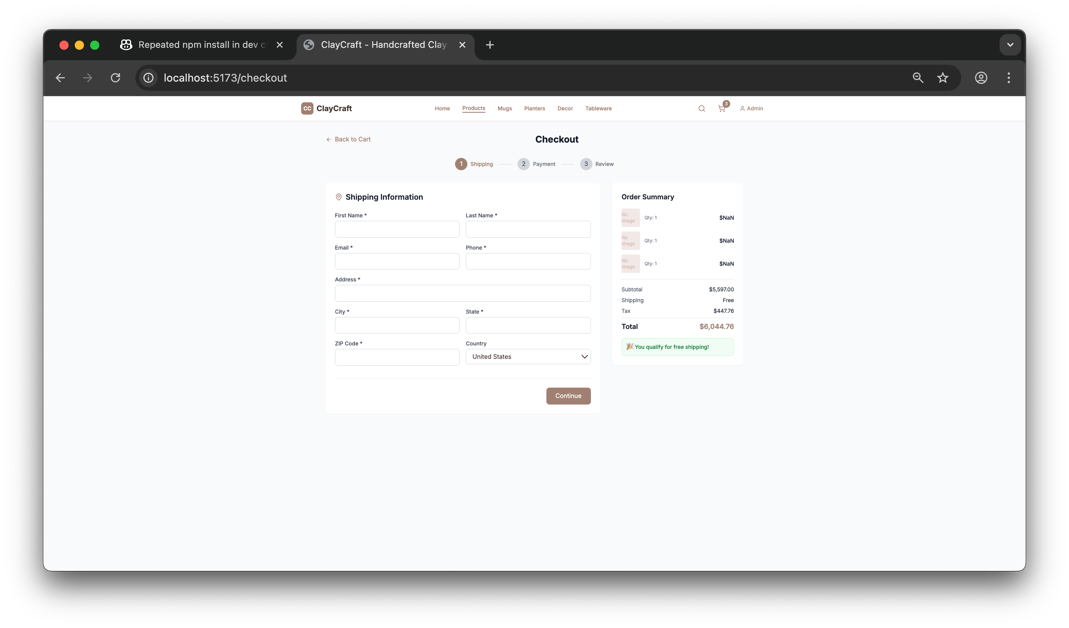


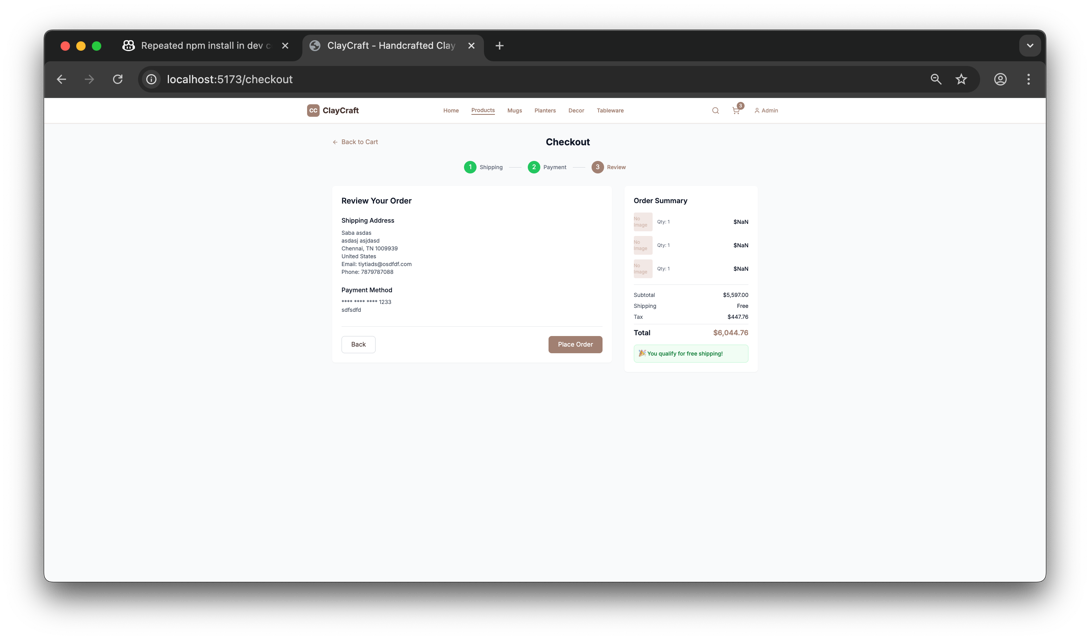


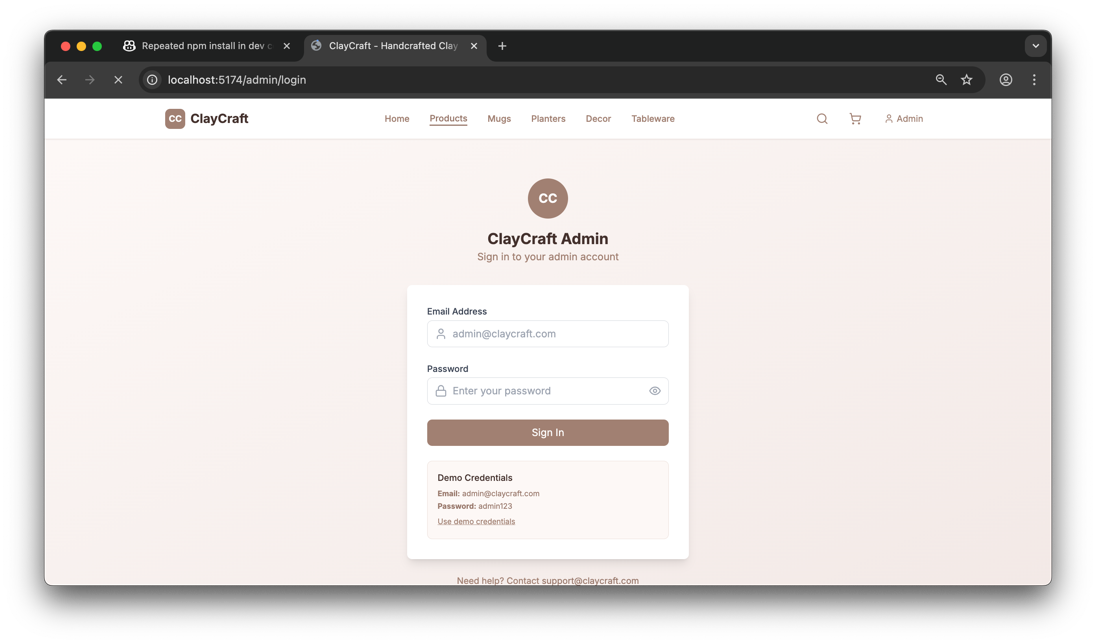












Code for the website:

<https://github.com/yourssabari/AI-Pair-Programming-Assignments/tree/main/Assignment%201%20ecomm-website>