Rental E-Commerce Documentation

Overview:

The Rental E-commerce Platform is designed to connect people who want to rent out their items with those looking to rent them. This project was developed over six days, focusing on creating a secure, user-friendly platform for managing rental transactions, bookings, and payments.

Step 1: Project Conceptualization and Platform Design

Defined Platform Type:

- Established as a peer-to-peer rental marketplace
- Focus on short-term and long-term rentals

Business Goals:

- Enable users to monetize their unused items through rentals
- Provide secure and reliable rental transactions
- Create opportunities for passive income generation

Data Schema Design:

- Entities: Users, Items, Rentals, Reviews, Payments
- Relationships:
- Users can be both renters and owners
- Items link to availability calendars
- Rentals track start/end dates and status

Defined Relationships:

- Users can act as both renters and owners.
- Items are linked to their respective availability calendars.
- Rentals maintain details like start date, end date, and rental status.

Step 2 : Technical Architecture Planning:

Tech Stack Selection:

Frontend:

- Next.js framework combined with Tailwind CSS for efficient styling.

Backend:

- Built using Node.js with Express for server-side logic.

Database:

- chosen for handling relational data.

API Requirements:

Authentication Endpoints:

- /auth/register: To register new users.
- /auth/login: To authenticate existing users.

Item Endpoints:

- /items/create: For owners to add new items for rent.
- /items/:id: To retrieve details of a specific item.
- /items/search: To search for available items.

Booking Endpoints:

- /bookings/create: For renters to create booking requests.
- /bookings/:id/status: To check or update booking status.

Review Endpoints:

- /reviews/create: For users to leave reviews.
- /reviews/user/:id: To fetch reviews for a specific user.

Deployment Strategy:

- Frontend is deployed on Vercel for seamless user access.
- Database is managed on Supabase for secure data storage.

Step 3: Core Features Implantation

Database Implementation:

- Users, Items, and Rentals.

- Established relationships and constraints to ensure data consistency.

User Authentication:

- Email verification system was added to confirm user identity.
- Password reset functionality was introduced for account recovery.

Step 4: Rental Management System

Item Management:

- Developed the Item Card component to display item details.
- Added an image upload feature with a preview option.
- Integrated an availability calendar for each item.

Search and Filters:

- Implemented location-based search using Map box API.
- Added filters for price range, date availability, and item category.

Reusable Components:

DateRangePicker:

- Allows users to select specific rental periods.

PriceCalculator:

- Estimates total rental cost based on duration.

AvailabilityCalendar:

- Displays item availability to renters.

Step 5: Testing and Security Implantation

Testing Suite:

Unit Tests:

- Tested core functionalities like booking logic and payment calculations.

- Validated date-handling mechanisms to avoid conflicts.

Integration Tests:

- Verified end-to-end booking workflows.
- Ensured payment processing was working correctly.
- Confirmed authentication processes were robust.

Security Testing:

- Added input validation to prevent invalid data entries.

Performance Optimization:

- Added lazy loading for images to improve page load speed.
- Introduced API response caching to reduce latency.
- Optimized database queries for faster results.

Step 6: Deployment and Launch Preparation

Deployment Strategy:

- Configured a pipeline using GitHub Actions for automated deployments.
- Set up a staging environment for pre-production testing.
- Implemented Sentry for real-time monitoring of application errors.

Environment Setup:

- Created a production. env template for secure configuration management

Final Testing:

- Conducted load testing using Artillery to ensure scalability.
- Verified cross-browser compatibility for a consistent user experience.
- Ensured mobile responsiveness on various devices.

Conclusion:

- This documentation explains each development phase, highlighting key decisions, implementations, and future plans for the Rental E-commerce Platform.