Project Proposal

CraftValley: Online Handcrafted Goods Marketplace Platform

Group 9

Berkay Gündüz, 22103402

Ege Şirvan, 22102289

Eren Aslan, 22102329

Kemal Sarper Şahin, 22103801

Öykü Elis Türegün, 21902976

Department of Computer Engineering

CS353 Database Systems

Özgür Ulusoy

26 February 2024

Table of Contents

1. Introduction

CraftValley is an online marketplace connecting small craft businesses and handmade item enthusiasts. It caters to three user groups: Small Businesses, Customers, and Administrators.

Small businesses can create profiles, list products, set prices, manage inventory, and close accounts if needed. Customers can browse, purchase, track orders, provide feedback, and receive recommendations. Administrators oversee operations, generate reports, manage accounts, and ensure compliance.

CraftValley simplifies product discovery with sorting options and maintains transparency with ratings. It employs Django and MySQL for backend operations and jQuery and Element UI for the front end, ensuring a secure and user-friendly experience.

2. Project Description

CraftValley is an online marketplace designed for buying and selling handcrafted items, similar to Etsy. It serves small businesses, customers, and administrators, aiming to make transactions, profile management, and product exploration easy and intuitive.

For small businesses, CraftValley offers tools to set up their digital presence. Artisans can create profiles, display their products, and manage inventory effortlessly. They can customize product listings with images, descriptions, prices, and stock levels.

Customers can explore handcrafted goods with a user-friendly interface. They can easily create profiles, use wishlists and shopping carts, and browse product listings. The buying process is simple, allowing transactions and returns to go smoothly. Customers can also give ratings, promoting trust and transparency.

Administrators can oversee platform operations. They can generate reports on popular products and trends. They control user accounts, including bans or deletions, to ensure platform integrity and security.

CraftValley's database architecture supports efficient data retrieval and manipulation. Users can filter products by name, content, type, and price. The database manages user ratings, transactions, and image uploads.

CraftValley is built on the Django framework with MySQL for database management. It uses jQuery for front-end interactions and the Element UI library for the user interface.

In summary, CraftValley aims to be a premier destination for artisans and enthusiasts, creating a vibrant community around handcrafted goods.

3. Database Usage

For our project, effective and long-lasting data management depends on having a robust and well-organized database. This is required due to the complex data needs of our handcrafted

online marketplace, including details about small businesses, customers, goods, transactions, and administrative duties.

Small businesses must submit essential business information and product details when they register with the marketplace. Administrators will verify these registrations and keep track of the verification procedure, putting relevant documents and their verification statuses in our database system as entities.

Administrators will manage the marketplace's facets, such as product listings, customer communication, and transaction logs. These administrative tasks will be meticulously stored as entities within the database.

In addition, the system needs to effectively handle customer profiles, product listings, and transaction histories to enable customers to browse and buy easily. The database will store transaction records, product details, and customer profiles as separate entities.

Customers can select products depending on their preferences due to the marketplace's search filters and sorting capabilities, which will improve user experience. To provide efficient product browsing and searching, the database must be able to process query requests on time with skill and accuracy.

Designing a well-structured database that can manage data retrieval, manipulation, insertion, and deletion while maintaining data integrity standards is crucial to successfully supporting these tasks.

4. Requirements

4.1. Functional Requirements

4.1.1. Small Business

- Small Businesses will have profiles.
- Small Businesses can add products with amount and price values.
- Small Businesses can list their products.
- Small Businesses can edit products' features.
- Small Businesses can manage their inventory (edit, delete, etc.).

4.1.2. Customer

- Customers will have profiles.
- Customers will have a balance representing the amount of money they can spend.
- Customers can add products from the market that have an amount greater or equal to the desired amount to their cart and see the total price of the items in the cart.
- Customers can purchase items in their cart if their total price is smaller or equal to the Customer's balance.
- Customers will lose money from their balance after a purchase.
- Customers can return purchased products to the Small Business from whom they purchased the item.
- Customers can add products to their wishlist.
- Customers can report Small Businesses.
- Customers can rate items in the market.

4.1.3. Admin

- Admins can create systems analysis to see statistics of the market.

- Admins can create business analysis to see statistics of a small small business.
- Admins can manage reports that are made by Customers.
- Admins can ban Small Businesses' accounts.
- Admins can delete accounts.

4.2. Non-functional Requirements

4.2.1. Scalability

The system should be able to handle requests from users, potentially up to thousands of concurrent requests, without compromising performance. The capacity of the system should be designed so as not to reflect the increasing number of requests to the users. Therefore, there should not be any drop in performance as the number of requests increases.

4.2.2. Security

Given the system involves transactions and payments between users, maintaining the security of the system is imperative. To achieve this, the following precautions should be taken:

- There must be a user authentication system (e.g., 2-factor Authentication using email/phone number) to secure user accounts.
- User data, especially sensitive information, should be encrypted in the database to protect against unauthorized access and breaches.

4.2.3. Performance

The system should rapidly respond to user requests. Therefore, the system should be designed with the following properties:

- Any payment must be processed and completed within several seconds to enhance transaction security.
- Results of the filtering operation must be delivered to the user in less than several seconds, as prolonged operations may lead users to leave the site.
- As the user searches for an item, new items must be fetched from the database in less than several seconds to ensure seamless usability for the user.

4.2.4. Usability

The system needs to be user-friendly to be easily used by everyone. Therefore, the following properties should be embedded in the system at the design stage:

- Contrast values should be adjusted correctly to increase the accessibility of the system.
- The chosen color palette should include colors that do not disturb the users' eyes.
- Colors of the components should be chosen carefully so that they are easy to distinguish from one another.
- Information about the products should be presented clearly (e.g., the rate of the seller will be displayed in both stars and numerical formats).
- There should be a fixed navigation bar consisting of a categories menu button, search bar, profile/sign-in button, and a shopping cart menu.
- Buttons should be highlighted as the user interacts with them.
- The system should use rational sizes to work better on different screen sizes.

4.3 Pseudo Requirements:

- MySQL will be used as the database system of this project. It has an easy-to-use interface, and has all the features that are needed in this project.
- The Django framework of Python will be used in the backend of the application.
- The Django with JavaScript will be used for the front end of the project. In order to have a better-looking UI, jQuery library and Element UI components will be used.

5. Limitations

- Each sale post must have one small business that created it.
- Each item must have a stock size.
- Each customer must enter an email, a password, a name, and a phone number on registration.
- Each small business must enter an email, a password, a name, and a phone number on registration.
- A customer cannot buy an item that is out of stock.
- A customer cannot buy more items than there is in stock.
- A customer cannot buy an item if they do not have enough balance.
- For a sale to occur, there must be a small business, a customer, and a related post.
- Each wishlist must have a customer and a post.
- Each report (as in complaint, etc.) must have a customer and a related business.
- Each report (as in complaint, etc.) must have one administrator associated with it.
- Each report (as in transaction record) must have a customer and a related business.
- Customers cannot return items that they have not bought.

- Only administrators and the owner of a post can remove the post.
- Administrators will have access to best-selling items.
- Each ban must have a related small business and duration.

6. E/R Diagram