Group 20: CloakedCommerceDB

Owen McDaniel
University of Idaho
USA
mcda0107@vandals.uidaho.edu

Braydyn Proctor
University of Idaho
USA
proc3080@vandals.uidaho.edu

ABSTRACT

This paper presents CloakedCommerceDB, a database-driven website designed for the CS360 course at the University of Idaho. This website allows users to post, browse, and trade items while safeguarding their own anonymity. The system uses MySQL with encryption methods to ensure data security while maintaining integrity for these transactions. The final result of this paper is be a fully functional website using MySQL to trade between other users. This system will offer a secure and anonymous platform for users to exchange goods efficiently and privately.

CCS CONCEPTS

• **Information systems** → Service discovery and interfaces; RESTful web services; Query representation; *Query intent*; **Query reformulation**; **Query languages for non-relational engines**; Semi-structured data; **Middleware for databases**; **Database query processing**; **Query languages**; • **Human-centered computing** → *Human computer interaction (HCI).*

KEYWORDS

Schema Graph, Biological Databases, Data Integration, Ad hoc Querying, Schema Abstraction, Query Reformulation.

ACM Reference Format:
Owen McDaniel and Braydyn Proctor. 2025. Group 20: CloakedCommerceDB. In *The 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), March 27-March 31, 2023, Tallinn, Estonia.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3555776.3577652

1 INTRODUCTION

CloakedCommerceDB is an anonymous bartering platform designed to facilitate the exchange of goods and services without revealing users' identities. In our current time, we face a new issue: the issue of privacy and security. Many websites that offer bartering platforms do not have safeguards for protecting their users' identities. CloakedCommerceDB addresses this issue by utilizing a 16-digit hash key authentication system to ensure complete anonymity during these transactions. Unlike other marketplaces, this system allows users to post, browse, and trade items while safeguarding their personal information.

The platform will be accessible in any web browser through our carefully constructed website, designed using vanilla HTML, CSS, Bootstrap, and JavaScript to provide a consistent user experience across all devices. Our back end is powered by Node.js with Express, enabling us to create a secure and efficient data processing system. This, paired with our encrypted MySQL databases, provides a reliable and secure platform.

Overall, CloakedCommerceDB will provide a secure platform for trading that redefines the standards followed by similar platforms.

2 E.R DIAGRAM

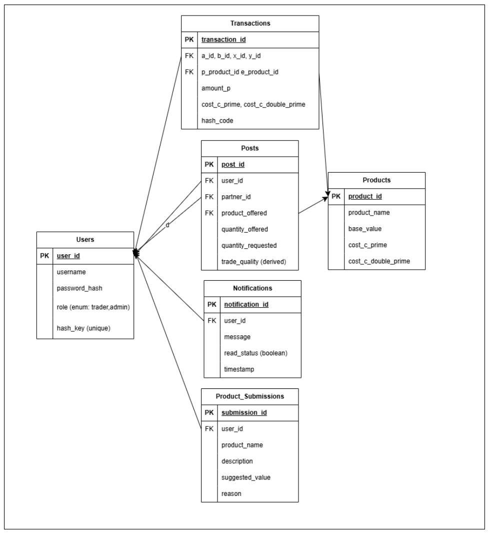


Figure 1: ER Diagram

Our ER diagram outlines the logical structure of the CloakedCommerceDB database & system, illustrating how data is managed and communicated for anonymous bartering. It includes six key entities: Users, Transactions, Products, Posts Product_Submissions, and Notifications

The Users entity contains primary key `user_id`, as well as `username`, `password_hash`, a unique `hash_key` for anonymity, and an `enum:role`, defining each user's role in the barter process such as a trader or admin. The Transactions entity captures details of each exchange with foreign keys `a_id`, `b_id`, `x_id`, and `y_id` referencing Users, `p_product_id` and `e_product_id` linking to Products, plus attributes `hash_code`, `amount_p`, `amount_e`, `cost_c_prime`, `cost_c_double_prime`

The Products entity includes primary key `product_id`, as well as `product_name`, `description`, and `base_value`. This represents items available for barter, with transfer costs `cost_c_prime` and `cost_c_double_prime` tracked in Transactions.

The Posts entity with primary key `post_id` links to Users via `user_id` and `partner_id`, and to Products via `product_offered` and `product_requested`, capturing barter offers and request Relationship sets include Users-Transactions as one-to-many, Products-Transactions as one-to-many, Users-Posts and Products-Posts as one-to-many. Additionally, The users have one-to-many relationships with both Product_Submissions and Notifications.

The Product_Submissions table allows users to suggest new products. It includes the primary key `submission_id` as well as a foreign reference to the user table `user_id`, `product_name`, `description`, `suggested_value`, `reason`. These submissions allow administrators to review the submissions and decide whether or not they can be added to the products table. The notifications table allows admins to send messages to users. It contains the primary key `notification_id`, as well as a foreign key reference to Users `user_id`, message, read_status, and timestamp.

3 TOOLS

Our main development tool for both the front-end and back-end will be JavaScript. HTML & JS will be used for the user interface and front-end. TO help with the back end of the website, we will be using Node.js which will allow us to use JavaScript for connecting to the database aspect. The barter system is designed without real-world payment processing to maintain complete anonymity. CloakedCommerceDB will be using MySQL for the database management system in order to securely store and manage data from users.

For working on the project we will be using Visual Studio Code. VSCode has built-in integration for Git/Github as well as numerous other extensions to ease the programming process, which makes it a fantastic choice for this project. VSCode also includes automatic formatting support for HTML, Php, MySQL, and JavaScript which significantly impacts the time it takes to implement.

For version control, we will be using Git and GitHub so both partners can work on the CloakedCommerceDB in a standardized environment.

4 PLAN / TIMELINE

Development of CloakedCommerceDB will be done in phases with updated documentation and presentations to demonstrate progress made in implementation and understanding. This project will continue through the course of the University of Idaho Spring 2025 semester and will end on May 1st with a completed project.

4.1 Phase I - February 25th

By February 25, we will have completed two pages of documentation which will include an ER diagram with an explanation of database design. The paper will also include an implementation methodology with details of the tools that will be used for individual parts of the development.

4.2 Phase II - March 21st

By March 21, we will have a working prototype for the CloakedCommerceDB website, with the front-end mostly implemented to demonstrate the implemented functionality. This will also include very basic MySQL implementation to show understanding of database design and querying. The report for the CloakedCommerceDB will be four pages long with more details of implementation.

4.3 Phase II - Results

For our Phase 2 Demo we were able to implement everything we wanted. The front end is heavily developed. We have implemented most of the required pages with minor issues. We changed from react to a basic HTML CSS and bootstrap front end for ease of use. Currently for our back-end development, we are using Node.js /w Express and modules for different libraries that will be discussed later.

We have created a backend SQL database with "dummy data" that is for testing of products. The database takes into account any information that is necessary to save and follows the configuration of the ER diagram. We have a fully implemented Register & Login system, all information is stored within the database's "Users" table. The password is encrypted via Bcrypt which generates a secure hashed password, and it does not showcase the password in plain-text anywhere.

There is a product posting page where users can enter their partners' hash code (a unique user ID that is only known by your partner(s)). This is the space where you can request trades and input what you are offering.

We have an admin panel for users that are marked as an admin in their role, this is done via code & verification through the database. The backend uses authorization routing to change HTML pages in order to have different features. In this deliverable due date the largest struggle was learning how to do back-end routing for sessions, and making the HTML of the normal site change when a user is logged in, versus when they are not logged in.

4.4 Phase III - April 18th

By April 18, a fully functioning prototype will be available with all intended functional features. The prototype will include all the trading interfaces and feature a fully implemented front end. The phase III report will be six pages long further detailing the CloakedCommerceDB implementation.

4.5 Phase III - Results

For our Phase 3 Demo we were able to implement most of the features that we were planning from the beginning. The administrator page was fully implemented featuring new products to be submitting and barter with in CloakedCommerce, a list of products with detailed descriptions and information for trades like c' and c", a list of users with the ability to promote users to administrator, delete users, and send notifications to users, as well as a list of all of the transactions that have happened.

For the SQL database, we have changed the database layout to include six tables with the notifications being a new table. The The ER Diagram has not been updated currently, but will be done so before the final report. The equivalence between the different items has also been changed for them to be compared based on a monetary value assigned to each product.

4.6 Phase IV - May 1st

By May 1st, the fully working final product will be released and the final touches for the product will be implemented. The system should be fully functional with no large errors or bugs for all systems, including administrator tools, account creation, and database queries. There will be an eight-page document specifying implementation of our trading system with encountered challenges being detailed.

4.7 Phase IV - Results

For Phase IV we finalized and polished all features of CloakedCommerceDB, officially we have a fully functional user friendly anonymous bartering platform. The project reflects our original vision at the start of the semester, it has secure authentication, seamless bartering, and well-developed admin tools that are integrated into the system. All font-end pages are finalized with HTML, CSS, Bootstrap, and Js which renders the navbar and dynamic UI elements. The SQL database now contains six fully functioning tables; Users, Products, Posts, Transactions, Notifications, and Product Submissions. As shown in our E.R Diagram The anonymous barter system works end-to-end. Users can create listings using their partner's anonymous hash key that acts as a profileID. The Trade Quality Meter (See Figure 13) Calculates and displays the trade "fairness" based on dollar-value rations, it triggers if one side stands to lose significant value (more than a 20 percent loss) with estimated losses shown in red. We have fully implemented an Admin Dashboard (Figure 14) accessible only to users marked with an admin role. Admins can manage users (promote/demote/delete), view all transactions, add or remove products, approve or reject product submissions or send users notifications via our custom-built notification system. All known bugs were resolved and database queries execute cleanly with input validation for all entries and queries. The UI Responds well to session changes, page protection works, and the hashing and salting system ensures users anonymity and protection across any and all exchanges. Overall the system is officially usable, stable, and ready for presentation.

5 CHALLENGES

5.1 Anticipated Challenges

While Developing the CloakedCommerceDB system we anticipate several challenges, primarily in ensuring anonymity, cross-platform compatibility, data security and accurate cost simulations. To maintain anonymity we will use a 16-digit hash key for authentication anonymously, this will ensure user identities are protected during (simulated) transactions. For Cross-Platform compatibility to work properly we will use vanilla HTML paired with Bootstrap in order to keep a consistent user interface and experience across any device.

Ensuring data security while maintaining integrity for these transactions is another challenge, we will be using MySQL with encryption methods in order to securely manage user information. Finally accurately simulating barter exchanges and cost adjustments will require using an equivalence table (T) for value comparisons and the Node.JS backed logic in order to perform constant real-time calculations to simulate real prices changing for bartering. Overall these solutions will help maintain anonymity, security, and fair value simulations.

5.2 Encountered Challenges

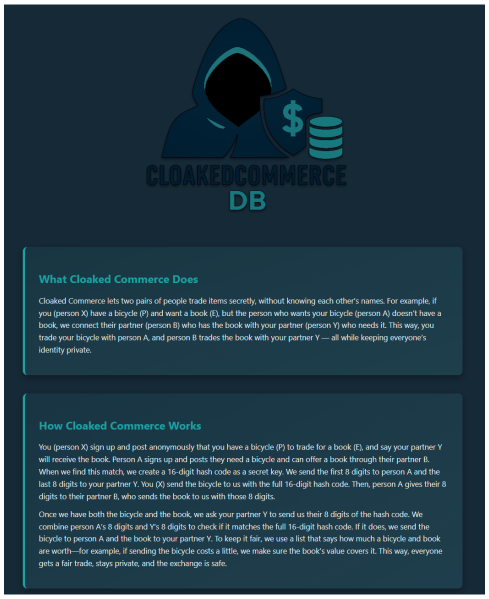
We have encountered various challenges while developing this portion. The first major challenge involved understanding the issue we are trying to solve. It took some time to dissect what we need for the project and what we shouldn't exactly have in it. The second challenge came when having the foreign key constraints whilst creating barter listings. Originally the system required users to input their partners userID directly. However this would defeat the purpose of remaining anonymous as someone could easily tell who they were trading with. So we created a partner hash code for a unique identifier that only the partner knows.

We had first tried using React.js for the front-end, but it seemed like too unnecessary for what we were trying to achieve. It would've worked, but it was adding complexity that wasn't really needed. We were already a lot more used to working with regular HTML, CSS, Bootstrap, and JavaScript, so it made more sense to just stick with that. It ended up saving us a lot of time and made things much easier to test and change as we went. Allowing us to get more done and streamline the whole process.

Dealing with these issues along the way helped the project come together more. It pushed us to work harder on making the platform secure, keeping user info anonymous, and making sure everything worked the way it should.

6 USER INTERFACE

6.1 Homepage & Navbar



What Cloaked Commerce Does

Cloaked Commerce lets two pairs of people trade items secretly, without knowing each other's names. For example, if you (person X) have a bicycle (P) and want a book (E), but the person who wants your bicycle (person A) doesn't have a book, we connect their partner (person B) who has the book with your partner (person Y) who needs it. This way, you trade your bicycle with person A, and person B trades the book with your partner Y — all while keeping everyone's identity private.

How Cloaked Commerce Works

You (person X) sign up and post anonymously that you have a bicycle (P) to trade for a book (E), and say your partner Y will receive the book. Person A signs up and posts they need a bicycle and can offer a book through their partner B. When we find this match, we create a 16-digit hash code as a secret key. We send the first 8 digits to person A and the last 8 digits to your partner Y. You (X) send the bicycle to us with the full 16-digit hash code. Then, person A gives their 8 digits to their partner B, who sends the book to us with those 8 digits.

Once we have both the bicycle and the book, we ask your partner Y to send us their 8 digits of the hash code. We combine person A's 8 digits and Y's 8 digits to check if it matches the full 16-digit hash code. If it does, we send the bicycle to person A and the book to your partner Y. To keep it fair, we use a list that says how much a bicycle and book are worth—for example, if sending the bicycle costs a little, we make sure the book's value covers it. This way, everyone gets a fair trade, stays private, and the exchange is safe.

Figure 2: Home Page

The homepage links to the backend through a session check. If the user is logged in, the navigation bar will dynamically show their name and a profile page via a drop-down. If not logged in the homepage will prompt the login and registration options. This check is performed using a fetch call to `/auth/session`

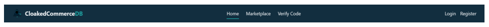


Figure 3: Navigation Bar

The default navigation bar appears at the top of every page and provides access to the main areas of the site. If no session is active, it shows links to the home page, login, and registration. These links help time users to the account creation process.

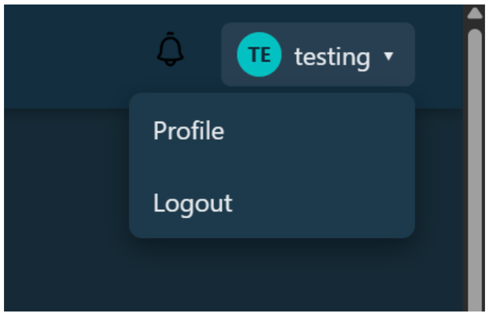
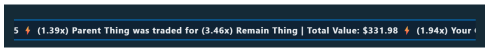


Figure 4: Navigation Bar Profile

Once logged in, the navbar updates to include the user's name drop-down menu if a user is an administrator the admin panel will also appear here. From here, users can navigate to their profile page, or log out. This makes navigation simple and intuitive for users returning to the site and provides a clean UI for profile-related actions.

Figure 5: Navigation Bar Logged In

When logged in, options become accessible in the navbar, such as "Marketplace" and "Verify Code." These options are only available to authenticated users and provide access to all major platform functionality.



5 ⚡ (1.39x) Parent Thing was traded for (3.46x) Remain Thing | Total Value: $331.98 ⚡ (1.94x) Your

Figure 6: Transaction Data Stream

The transaction bar is an animated visual element used on the homepage to highlight the completed trades' data in CloakedCommerceDB. It shows completed trades, the products & quantities of the trade as well as the calculated value of the trades.

Note: The backend implementation that powers the navbar rendering, the profile sessions, and the transaction data stream is discussed in detail in the next section as there is a lot for this specific section. All remaining sections from this point forward will focus on the front-end experience, with backend functionality described alongside each feature.

6.1.1 Navigation & Transaction Data Stream Backend. The navigation system and the homepage behaviors are tied to session-based authentication. This is handled through the backend. When the homepage loads, a `fetch` call is made to the `/auth/session` endpoint.

This endpoint checks if there is a valid session that exists. This uses the middle-wear coded into the backend. If a session is active, the backend will return the user's `username`, `user_id`, and `role`, which the front-end uses to determine what needs to be in the navbar.

The backend stores user sessions using server-side cookies. When a user logs into the site, their session is initialized and tracked. Based off the role returned in the session obj. the front-end determines wether or not to display the normal user references or the admin-level links such as `Admin Panel`.

The profile drop-down and logout option are rendered using this session data. Logging out will trigger a backend route `/auth/logout`, which will destroy the entire session and redirect the user back to the homepage. This pairing ensures that the navigation options are never shown to any non-authenticated users, or any users without role access.

The transaction data stream shown on the homepage is a visual representation of the SQL database. It uses a backend route called via a `<script>` in the HTML that accesses the `Transactions` table. This route queries all transactions in descending order by timestamp, allowing for a dynamic and continuously updating display of recent barters. The front-end then animates this data to simulate a live feed of anonymous trade activity across the platform making the site feel more alive.

6.2 Login / Registration

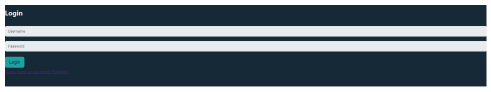


Figure 7: Login Page

The login and registration pages submit form data to the backend through POST requests to `/auth/login` and `/auth/register`. Registration hashes and salts the user's password with bcrypt and generates a 16-character anonymous hash identification key for users to share with their known partner for trading. After a successful login, the session is initialized, and a cookie is stored, afterwards the user is redirected to the homepage. The logout button terminates the session as well. No RAW passwords are ever stored and or transmitted.

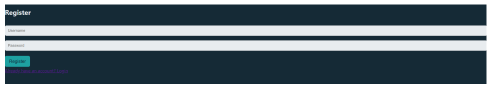
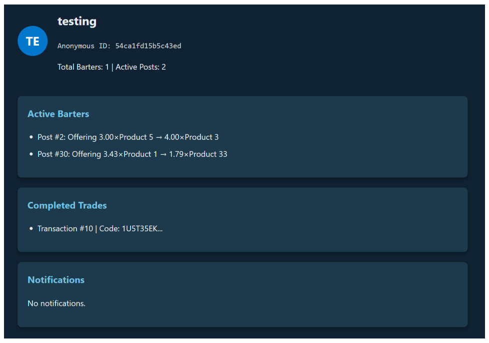


Figure 8: Registration Page

6.3 Profile Page



testing

TE

Anonymous ID: 54ca1fd15b5c43ed

Total Barters: 1 | Active Posts: 2

Active Barters

- Post #2: Offering 3.00×Product 5 → 4.00×Product 3
- Post #30: Offering 3.43×Product 1 → 1.79×Product 33

Completed Trades

- Transaction #10 | Code: 1U5T35EK...

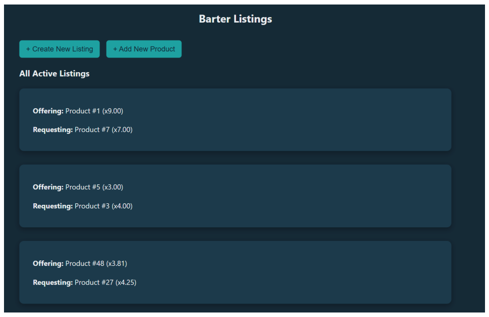
Notifications

No notifications.

Figure 9: Profile Page

The profile page grabs the personal data using a GET request to `/dashboard`. This includes the user's anonymous ID key, username, and role. This page is protected and only accessible if the session is valid. If the user attempts to access it without a valid session, it will return them to the login page.

6.4 Listings Page



Barter Listings

All Active Listings

Offering: Product #1 (x9.00)
Requesting: Product #7 (x7.00)

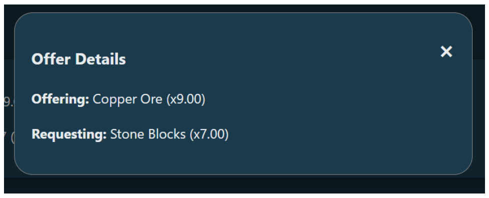
Offering: Product #5 (x3.00)
Requesting: Product #3 (x4.00)

Offering: Product #48 (x3.81)
Requesting: Product #27 (x4.25)

Figure 10: Listings Page

The listings page communicates with the backend to perform several API calls. It fetches the product data from `/barter/products`, loads the existing unfulfilled listings from `/barter/posts`, and submits new listings with a POST request to `/barter/posts`. Users create listings by submitting a partner's anonymous hash key, which the backend uses to figure out the user ID prior to storing the posts.

6.5 Detailed Barter Listings



Offer Details

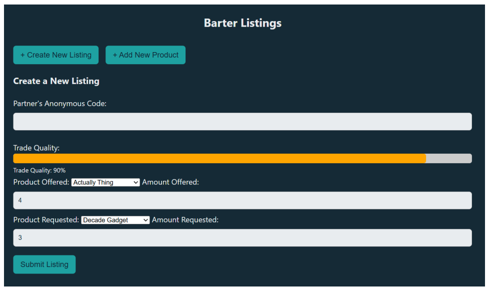
Offering: Copper Ore (x9.00)

Requesting: Stone Blocks (x7.00)

Figure 11: Detailed Barter Listings

The detailed Barter Listings area is a Modal that users can click on to view more detailed information about a trade, such as the offer and requested products name. This is an essential part of the website as users generally don't know the product ID's and if they want to search for a specific one to see if anyone is offering the product they want they can. It connects to the backend via the table `Posts` joined with `Users` and `Products`. The server-side script dynamically fetches this information and displays it as trades are created.

6.6 Create a New Barter



Barter Listings

Create a New Listing

Partner's Anonymous Code:

Trade Quality:

Trade Quality: 90%

Product Offered: Actually Thing Amount Offered: 4

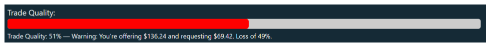
Product Requested: Decade Gadget Amount Requested: 3

Figure 12: Create New Barter on Barter Listing Page

The process of creating a new barter is located at the top of the listings page. Here, a new listing will be created through inputting your partner's anonymous code. Then, you can choose which product you are offering through a drop-down menu and the quantity through either typing into the text box or using the up and down arrows on the right of the text box that appear when hovering your mouse above the text box. The exact process is the same for which product and quantity you are requesting. There is also a trade quality measurement that I will detail in the next section. Finally, to submit the listing, there is a button at the end of the section.

When submitted, the form attempts to make a backend route that inserts the data into the `Posts` table. However, first the input is verified via a `<script>` in the HTML, which prevents SQL injection attacks and XSS attacks, which can cause major security issues. After verification, the barter is inserted with the `user_id`, the partner's `user_id`, offered product and quantity, requested product and quantity, and the calculated trade quality.

6.7 Trade Quality Meter & Fairness Algorithm


Trade Quality:

Trade Quality: 51% — Warning: You're offering $136.24 and requesting $69.42. Loss of 49%.

Figure 13: Trade Quality Meter

The trade quality meter compares the quantities of the products offered and products requested and according to the monetary value of each product. The comparison formula produces a percentage score 0-100 representing the total value ratio between the sides of the trade. The algorithm ensures that both parties are aware and understand any imbalances and provides a visual and numerical feedback i.e offering x requesting y a total loss of Z

6.8 Fairness Algorithm (VL (Value Loss))

$$\begin{aligned}\text{Let:}\mathbf{x_0} &= \text{quantity of product offered}\\ \mathbf{v_0} &= \text{base value of product offered (per unit)}\\ \mathbf{x_1} &= \text{quantity of product requested}\\ \mathbf{v_1} &= \text{base value of product requested (per unit)}\end{aligned}$$

Total Trade value offered & requested are calculated as:

$$\mathrm{V}_o = x_0 \cdot v_0$$
$$V_r = x_1 \cdot v_1$$

Trade percentage is then:

$$\text{Trade Quality (\%)} = \min\left(\frac{V_o}{V_r}, \frac{V_r}{V_o}\right) \cdot 100$$

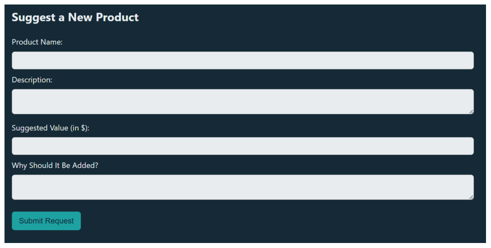
If a user is offering less than requested(i.e., $V_o < V_r$) a warning shows the loss percentage

$$\text{Loss (\%)} = \left(1 - \frac{V_o}{V_r}\right) \cdot 100$$

The visual color feedback is calculated as.

- **Green bar:** Trade Quality $= 100\%$
- **Orange bar:** $80\% \leq$ Trade Quality $< 100\%$
- **Red bar:** Trade Quality $< 80\%$

6.9 New Product Page


Suggest a New Product

Product Name:

Description:

Suggested Value (in $):

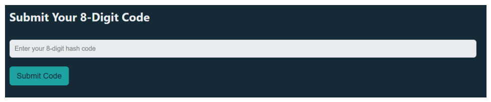
Why Should It Be Added?

Figure 14: New Product Page

To suggest a new product to be offered through the CloakedCommerce website, you must provide a product name, a description of the product, the suggested value in dollars and cents, and a message explaining why the product should be added. To submit the request, you click the button at the bottom of the page.

When the form is submitted, it sends the data to a backend route that inserts the request into the `Product_Submissions` table. The submission includes the `user_id`, `description`, `suggested_value`, and `reason`. This data is later reviewed by administrators via the admin panel.

6.10 Verify Code Page

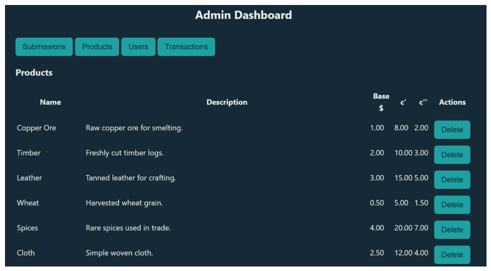

Submit Your 8-Digit Code

Enter your 8-digit hash code

Figure 15: Verify Code Page

Here is the verify code page located in the navigation bar on the top of the website. This page will be used for verifying that you have received the product for your end of the transaction.

6.11 Admin Panel Products


Admin Dashboard

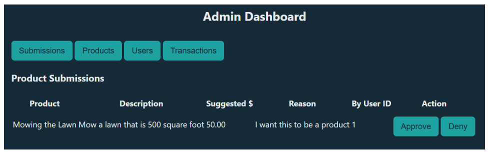
Products

Name	Description	Base $	c'	c''	Actions
Copper Ore	Raw copper ore for smelting.	1.00	8.00	2.00	Delete
Timber	Freshly cut timber logs.	2.00	10.00	3.00	Delete
Leather	Tanned leather for crafting.	3.00	15.00	5.00	Delete
Wheat	Harvested wheat grain.	0.50	5.00	1.50	Delete
Spices	Rare spices used in trade.	4.00	20.00	7.00	Delete
Cloth	Simple woven cloth.	2.50	12.00	4.00	Delete

Figure 16: Admin Panel

The admin panel is a protected section only accessible to administrators. This role is designated within the code, and verified in the database. This allows for protection of user identities, codes, trades etc. If a user has the role of admin they will be able to access this via the Admin section on the drop-down. It accesses the backend via set routes. To remove a user it uses `/admin/remove` in the future you will be able to view peoples trades, and add products via this panel using the admin route `/admin/verifyprod`. This will allow regular users to add their product and admins to view them and verify worth and update the equivalence table.

6.12 Admin Product Submissions



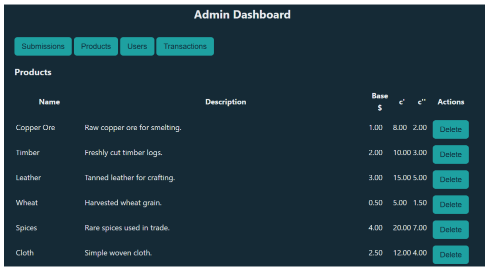
Product Submissions

Product	Description	Suggested $	Reason	By User ID	Action
Mowing the Lawn	Mow a lawn that is 500 square foot	50.00	I want this to be a product	1	Approve Deny

Figure 17: Admin Product Submissions

The products request submissions will be loaded on the administrator dashboard first by default since it is the action most needed from administrators. Like the product request page, you will be able to see the product name, description, suggested dollar value, reason for adding the product, the User ID of the person submitting it, and then approve and deny options for the administrators to decide to add the product. Submissions are pulled from the `Product_Submissions` Table and displayed to admins. Approval inserts the product into the `Products` table while denial deletes the request from the table.

6.13 Admin Products Page



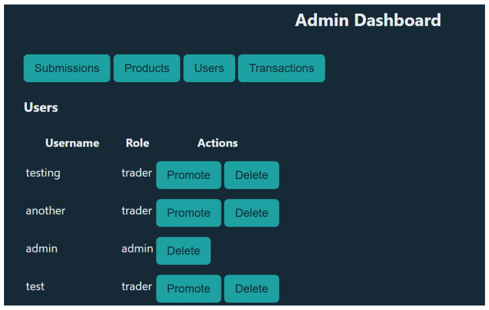
Products

Name	Description	Base $	c'	c''	Actions
Copper Ore	Raw copper ore for smelting.	1.00	8.00	2.00	Delete
Timber	Freshly cut timber logs.	2.00	10.00	3.00	Delete
Leather	Tanned leather for crafting.	3.00	15.00	5.00	Delete
Wheat	Harvested wheat grain.	0.50	5.00	1.50	Delete
Spices	Rare spices used in trade.	4.00	20.00	7.00	Delete
Cloth	Simple woven cloth.	2.50	12.00	4.00	Delete

Figure 18: Admin Products Page

Here, all of the products currently supported are listed with their name, description, base dollar value, and then c’ and c” values to be used for transactions. There is also a delete option to get rid of any product that is currently offered. This page loads all data from `Products` table. Deleting a product sends its ID to a backend route which runs SQL DELETE cmd to remove the entry completely from the table.

6.14 Admin Users Page



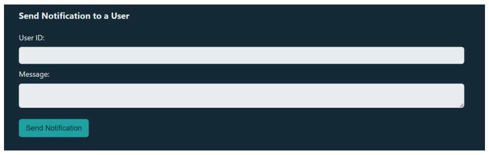
Users

Username	Role	Actions
testing	trader	Promote Delete
another	trader	Promote Delete
admin	admin	Delete
test	trader	Promote Delete

Figure 19: Admin Users Page

The Users page through the administrator panel shows all of the created users with their username, role of trader or administrator, and gives you the actions of elevating a user with trader to administrator or delete any users. This page queries the `Users` table to load usernames and roles of users. Elevating a users role via the `Promote` button will send a backend route to update their role and permissions. Deletion similarly send a backend route to remove their row from the table by running the DELETE command.

6.15 Admin Notifications


Send Notification to a User

User ID:

Message:

Figure 20: Admin Notifications

Located at the bottom of the administrator users page is the ability to send notifications to a user. You will do this by putting in the user ID and then typing the message into the text box and clicking the send notification button. Sending a notification creates a new row in the `Notifications` table with the `user_id`, `message`, `timestamp`, and unread status. These notifications get pulled when the recipient loads into the website.

7 FUTURE DEVELOPMENT

For future development, we would like to implement several new features to enhance usability, privacy, and the overall trading experience.

Friend System: A planned feature is a friend system that allows users to add/manage a list of trusted individuals who they trade with. This would remove the need to manually track all your friends' hash codes. With this feature, a user can initiate trades easier as they would select their friends as their trade partners or optionally enter their hash-code manually.

Trade Analytics Dashboard: A dashboard where users can see their total losses, gains, and all visual statistics associated with their trades. This would allow for users to get an in-depth understanding of their trading patterns and their loss/gain margins.

Hash Refreshing: A security feature that allows users to refresh their hash code. This would remove all of their friends with the exception of the ones they select via a list of their current friends. This feature would completely remake their unique hash-code ID, which would make them completely anonymous again if they ever wanted to get rid of someone who knows their hash-code. This adds an additional layer of security to keep anonymity.

Scheduled Trades: A scheduled trade would allow users to preset barters to occur at a set timestamp or under certain conditions. This would be useful for users who want to automate any repetitive trades or to schedule when they have items available.

8 CONCLUSION

CloakedCommerceDB is a fully developed system that is designed to match users looking to exchange products and services while maintaining complete anonymity. It is accessible through a learnable and user-friendly interface that provides an efficient platform for private transactions between individuals who wish to barter goods.

Users can log into their accounts to view and manage their posts, track ongoing transactions, and access their trading history. Administrators have access to tools that allow them to manage all active & inactive postings, approve or deny product submissions, and oversee platform activity to ensure fair and secure use of the system.

By emphasizing user anonymity, security, and ease of use, CloakedCommerceDB offers a unique and practical approach to modern bartering. The system has been implemented with secure backend protections, intuitive yet simplistic front-end design, and flexible trade logic that allows users to negotiate fairly. With continued refinement, CloakedCommerceDB has the potential to reshape the way anonymous trade is approached in a digital space.

REFERENCES

https://github.com/B-Proctor/CloakedCommerceDB

Bootstrap Framework. https://getbootstrap.com/

MySQL Documentation. https://dev.mysql.com/doc/

Express.js Guide (Used for backend route structure inspiration). https://expressjs.com/