

BarterDB App - Barter Buddy

Samuel Beal
beal6782@vandals.uidaho.edu
University of Idaho
Moscow, Idaho, USA

Aiden Shepard
shep8121@vandals.uidaho.edu
University of Idaho
Moscow, Idaho, USA

ABSTRACT

This paper presents an android app using a standard database to build an anonymous marketplace. The design is meant to allow a quick and easy digital trader that can be used on the go no matter where in the world you are as long as you have a connection to the internet. The system will allow users to work with partners, and select either from the buy or sell section of the app, as you are only allowed to have access to one. These exchanges will be secured through a database using Express and MySQL hosted on XAMPP.

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

React Native, Express, MySQL, Android, JavaScript, TypeScript, Database, Marketplace, Anonymous.

ACM Reference Format:

Samuel Beal and Aiden Shepard. 2024. BarterDB App - Barter Buddy. In *The 38th ACM/SIGAPP Symposium on Applied Computing (SAC '23), March 27-March 31, 2023, Tallinn, Estonia*. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3555776.3577652>

1 INTRODUCTION

Barter Buddy is a user-friendly marketplace where individuals work with partners to trade items anonymously through a bulletin board. Barter Buddy is divided into two sections: Barter Buy and Barter Sell. A user chooses which account they would like during account creation and then has access to purchasing or selling items. The main focus of this app is to properly equate the value of different items being placed for sell for users as well as abstracting users so they remain anonymous. The difficulty in such a task comes from the exchange between two users needing to be secured and guaranteed despite the users not being able to know the other exists.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SAC '23, March 27-March 31, 2023, Tallinn, Estonia

© 2024 ACM.

ACM ISBN 978-1-4503-9517-5/23/03

<https://doi.org/10.1145/3555776.3577652>

1.1 Features

In this paper, we propose Barter Buddy will have these core features.

- The app will be built for the android platform, separated into two sections of BarterBuy and BarterSell. All users will have a set of shared tabs including settings, dashboard at the very least.
- The database will make sure that the individuals involved in the transaction remain anonymous. For example, if pair one consists of person A and person B, and pair two consists of person X and person Y, then neither pair in the exchange are aware of the composition of the other pair.
- The system will include a bartering marketplace that acts as an anonymous bulletin board, an individual will post what items they want in exchange for with an item either they or their partner have. Inversely, other people will post what they want and are willing to give. The system will then connect the participants.
- Once a viable trade has been found the system will create a 16-digit hash code as a key to represent the trade. The two parties involved will be given half the key and the database itself will be given the product P, the initial trade item. The party who wants product P will then give the database product E and their half of the code and the initial party will send a request to exchange to the system and their half of the code. If the combined halves match the 16-digit hash code, then the exchange commences and the system gives the appropriate party their designated item.

1.2 Tools and Languages used for Development

1.2.1 Front-End Development.

- HTML and CSS will be used in combination with JavaScript and TypeScript to implement the interface with React Native. CSS will be very heavily used with React Natives UI framework to modify the look of everything.
- JavaScript and TypeScript will be used to program with React Native and Express to run the entire front-end functionality.
- React Native is the framework we will be using to host the entire app, it is well-known and built to be essentially the typical 'React', but for building mobile apps. We will also be using it for its async storage allowing us to maintain logins through refreshes and restarting the app.
- Expo is the framework used with React Native to connect to and run our mobile app environment. It is the main router of the app, allowing us to connect all of the pages together in different ways.
- Android studio will be our testing environment for this project. Android Studio is "the official integrated development environment for Google's Android operating system"

(Wikipedia, 2024). It will be vital in testing across multiple phones and platforms to ensure our app is great.

1.2.2 Back-End Development.

- XAMPP is where the database will be hosted. It is a powerful tool to be able to easily put in SQL queries and build databases that are able to be seen and looked at in depth with great visuals.
- Express will be used to connect to our database using JavaScript with get and post calls using useEffect. It is a free build-off of Node.js that is quick and easy to setup, and we could also use it to for either SQLite or MySQL on its own, but we are actively using XAAMP.
- MySQL will be our language of choice for querying and working within a database. We are currently using a full fledged database run on a server using XAAMP.
- We are not actively working with Ian and Lucas which is a team doing BarterDB, but we are using the database scheme they gave us as we wanted to partially work together at least.

1.3 Timeline and Plan

Phase I - Through Sep-24.

- Conceptualizing design
- React Native w/ Expo setup
- ER diagram
- Get connected with another group for BarterDB

Phase II - Through Oct-24.

- Rough prototype
- Connect front-end to database
- Draft front-end for Android app

Phase III - Through Nov-24.

- Fully working Database
- Mostly finalized front-end
- All design requirements fulfilled

Phase IV - 05-Dec-24.

- Polishing
- Fully developed system
- Full implementation report

2 COMPONENTS

2.1 Overview

Barter Buddy currently has a functioning registration and login page. After we either login to BarterBuy or BarterSell we can create buy or sell posts, and see a list of the items available to do so.

2.2 Database Design

Barter Buddy has a database that is being set up in XAMPP with MySQL. It is comprised of numerous tables that hold the users valuable information. The currently standing tables and all their properties will be detailed below and also why we chose to implement them in the way we did.

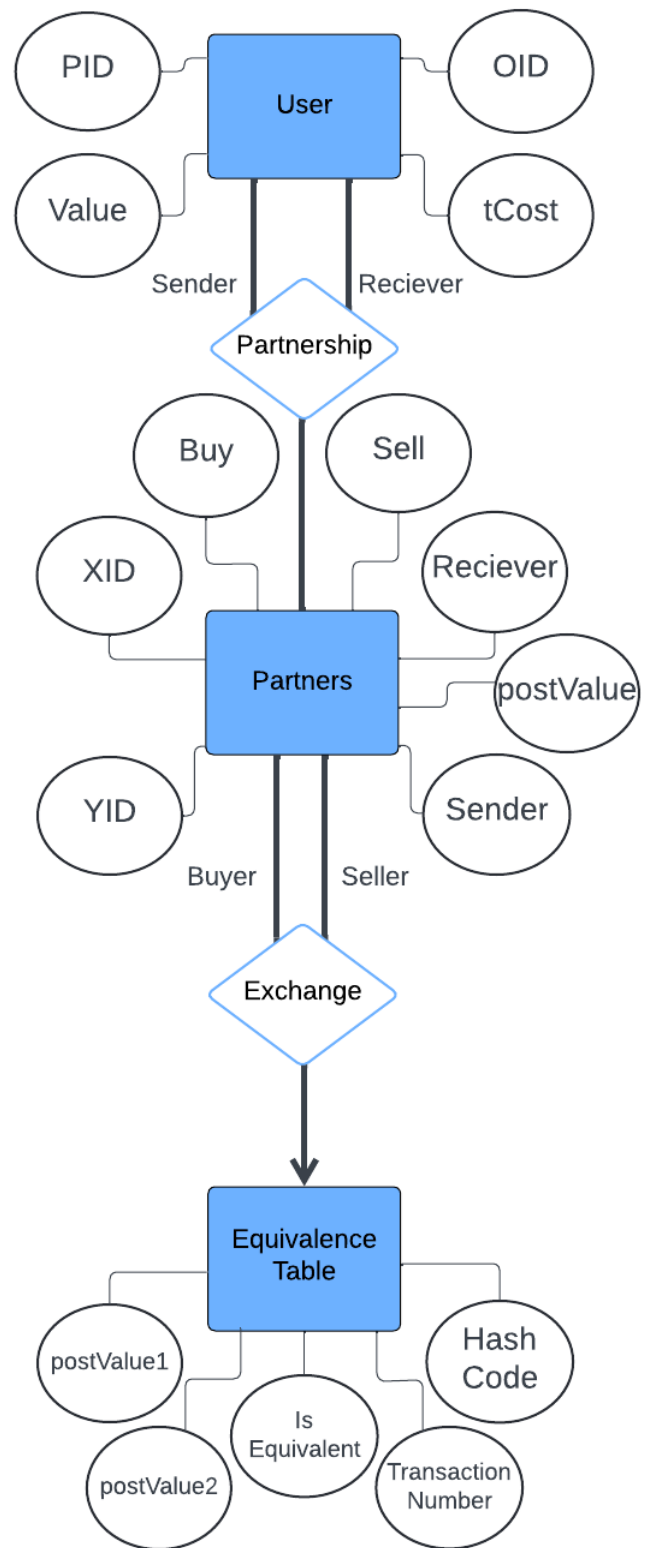


Figure 1: Entity relationship model of Barter Buddy's transaction scheme

- **Accounts**

- **Table Overview:** The Accounts table is where the users account information is stored a row in the Accounts table is created when they register for BarterDB and when logging in, the login information is checked against existing accounts in the table.
- **id:** Id is the in system name for the user, other users that are not partners with the signed-in user will only see this instead of a username.
- **username:** Username is the name the user signs up for the program under, this is only viewed by the user and their partners.
- **password:** Password is established when the user registers and is how they access their account.
- **role:** Role contains either 'BarterBuy' or 'BarterSell' to identify what type of user the client is.

- **Commodities**

- **Table Overview:** The Commodities table is where items are stored, there is a section on the profile tab on the main page of BarterDB where users can add new items to their account. The current version of the system only requires the user to include a name and value, but the table is equipped to handle quantity and type, which will eventually be implemented.
- **commodity id:** This is the unique id of the item within the system and acts as the primary key of the Commodities table.
- **id:** Id is the identifier that tells the system who is in possession of the item in this row.
- **commodity name:** This is the name of the item and this is the attribute that is viewable by the user and other users, this is also how users will refer to items.
- **commodity type:** Commodity type will categorizes different items in the system to help organize the entries. For example, sheet metal and petroleum barrels may go under Industrial, while carrots and barley may fall under Agriculture.
- **quantity:** This is the attribute that identifies the amount of something.
- **value:** Value is the attribute that is weight during the exchanges between users, it is currently the only other attribute besides commodity name that is entered in by the user.

- **Exchanges**

- **Table Overview:** The exchanges table is where the in progress transactions are stored, and used to be displayed as posts on the marketplace.
- **exchange id:** This is an automatically incrementing number that acts as the primary key for the table.
- **seller id:** The seller id is not used in the current state of our design, but in the future will be used to determine the user of a partnership that accepted the post.
- **buyer id:** The buyer id is set as the user who initially submits the post for the exchange of goods on the site.
- **seller partner id:** The seller partner id is based on the partner that was chosen by the user that accepted the post.

- **buyer partner id:** The buyer partner id is set as the id of the partner that the user originally creating the post chose when creating it.
- **commodity id:** The commodity id is the id of the commodity that the user creating the post is listing for trade.
- **offer commodity id:** The offer commodity id is the id of the commodity that the user creating the original post wants to get from the trade.
- **hash code:** This is the hash code that will be generated for each post and used to ensure that the trade between partners is completed.
- **status:** The status will determine whether or not the post is visible. It is either 'pending' or 'completed' and will not be queried to be shown as a post if it is 'completed'.
- **desired commodity:** Desired commodity is used to hold the name of the commodity that the user creating the post wants from their trade. This is used as a placeholder to show the desired commodity until the trade can be accepted and completed, at which point the commodity id can be populated with the id of the item being given to the original poster.

- **Partnerships**

- **Table Overview:** The Partnerships table holds the systems currently established partnerships. Partnerships are created under the Profile tab by entering in the id of a user you wish to be partners with, this information is only known by the user and is not public so, in order to be partners with some one you must personally know them and have them tell you their id.
- **partnership id:** Partnership id is how the system;s unique id for the specific relationship between two users.
- **id:** Which ever user inserted the the other users id into the add partner system will have their personal id used as the id in the Partnerships table.
- **partner id:** The user who is entered into the partnership system by another will have their id copied into the partner id field, this has no functional difference between id, it just so the system can hold both people.
- **status:** The status will determine the state of partnership. It is either 'pending' or 'completed' and a partnership will not be moved to 'completed' until both parties have agreed to the partnership.

2.3 Branding and Icons

We had extensive branding for our application as we need multiple types of logos for the different loading screens and pages of a mobile app. We have a main logo, named logo, and the splash screen logo. We have also included our color palette of red and blue with an off black and off white, as well as the icons for each page. The considerations for logo creation wanted to draw attention to the loop of items being bartered and traded using the Barter Buddy app. The arrows signify the loop of giving and taking as you accept trade deals. Using our primary and secondary colors it creates a nice distinction between the two different arrows. Due to the simplicity of the design as well the logo functions and is effective in an all black and white situation.

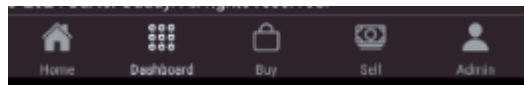


Figure 2: Icons for each Tab



Figure 3: Main Logo



Figure 5: Splash Screen Logo

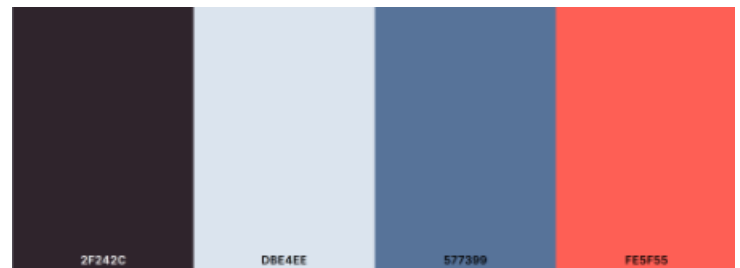


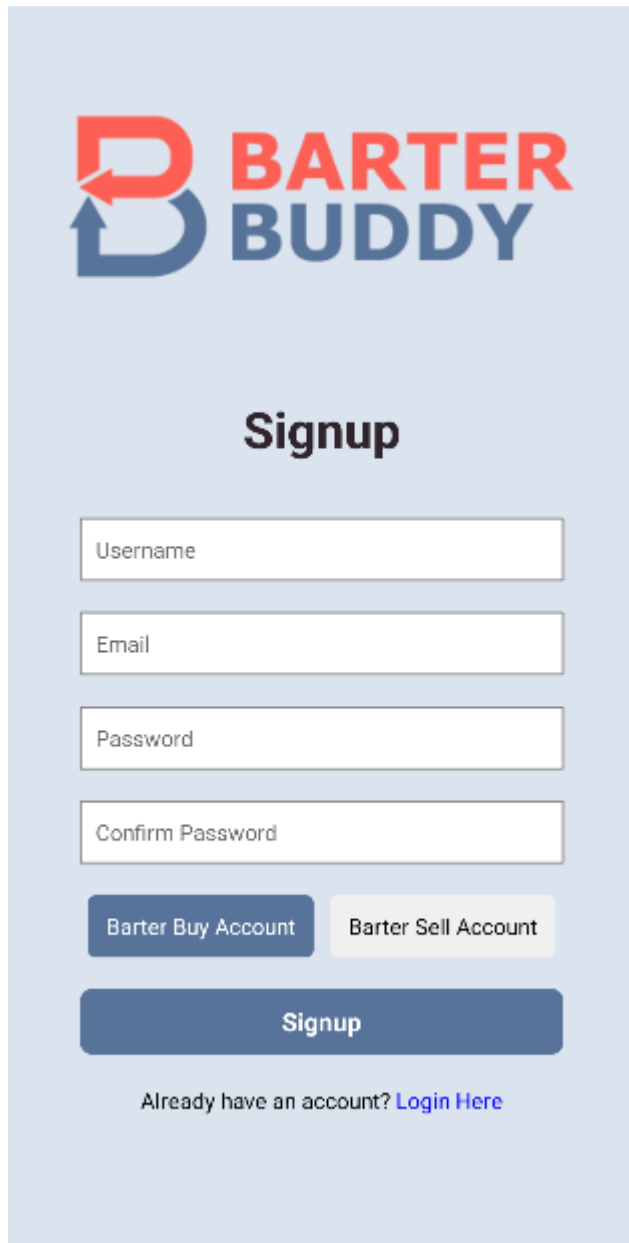
Figure 6: Color Palette



Figure 4: Named Logo

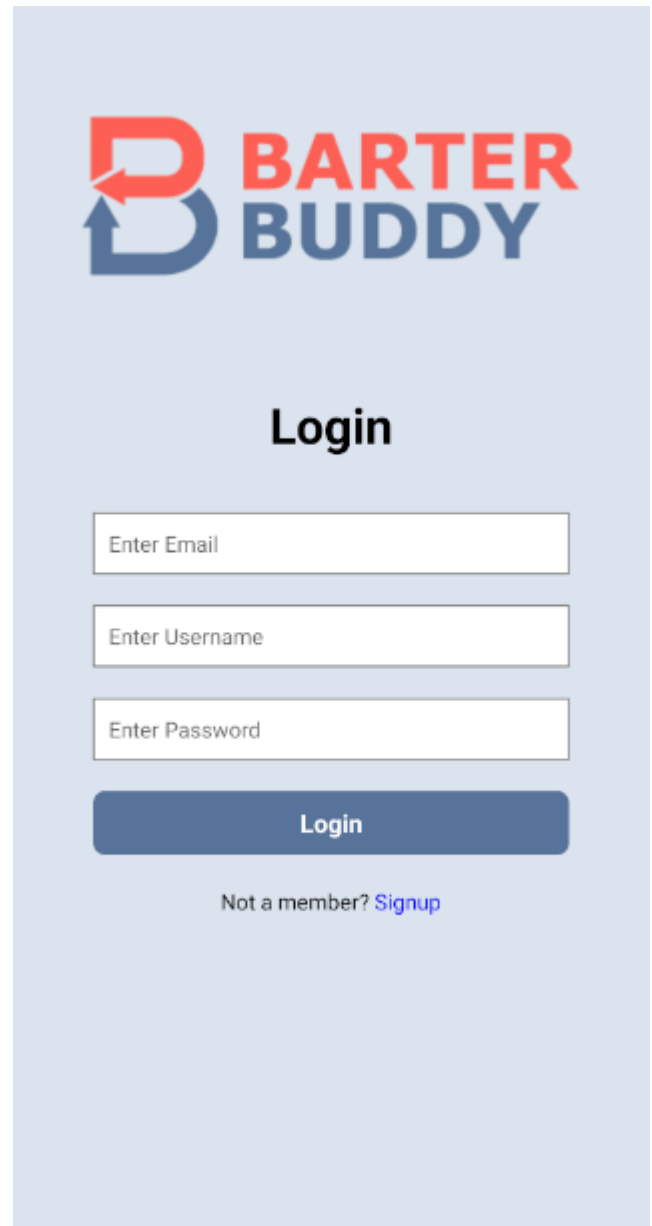
2.4 Registration and Login

The registration and login page will be one of the earliest pages encountered by a user as creating and signing into an account is mandatory before accessing the rest of the site. Both sites are navigatable based on whether the user already has an account or not. Registration requires a username, email, password, and the type of account. Error messages become present if a user inputs an incorrect string, such as an incorrectly formatted email address.



The image shows the Barter Buddy Signup page. At the top is the Barter Buddy logo, which consists of a red 'B' with a blue arrow forming a cycle around it, followed by the text 'BARTER BUDDY' in red and blue. Below the logo is the title 'Signup' in bold black text. The form contains four input fields: 'Username', 'Email', 'Password', and 'Confirm Password'. Below these fields are two buttons: 'Barter Buy Account' (dark blue) and 'Barter Sell Account' (light blue). At the bottom of the form is a large dark blue 'Signup' button. Below the button is a link: 'Already have an account? [Login Here](#)'.

Figure 7: Barter Buddy's Signup Page



The image shows the Barter Buddy Login page. At the top is the Barter Buddy logo, which consists of a red 'B' with a blue arrow forming a cycle around it, followed by the text 'BARTER BUDDY' in red and blue. Below the logo is the title 'Login' in bold black text. The form contains three input fields: 'Enter Email', 'Enter Username', and 'Enter Password'. Below these fields is a large dark blue 'Login' button. Below the button is a link: 'Not a member? [Signup](#)'.

Figure 8: Barter Buddy's Login Page

The login page redirects the user to the main functions of the app based on the account they signed into. A similar function is performed when a user creates an account from the registration page to prevent unnecessary steps. Since usernames are unique to each account, those are the only necessary inputs required for the login process. An error message will occur if either the username or password is incorrect but won't specify which to decrease brute force attempts.

2.5 Homepage Features

The homepage is accessed when you first load into the app. There are a few quality of life drop down menus with frequently asked questions, terms and conditions, use information, and our copyright footer, which will be extended to the rest of the app. As below, we see all of these features except for the footer.

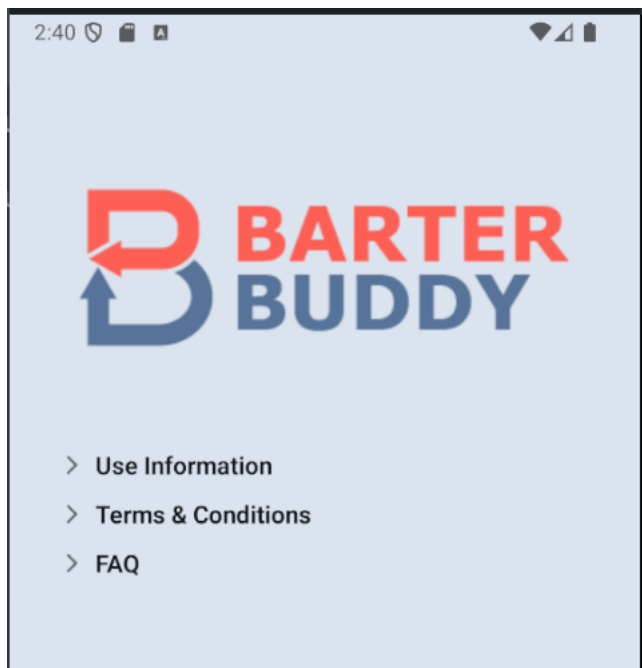


Figure 9: Top half of Barter Buddy's Homepage

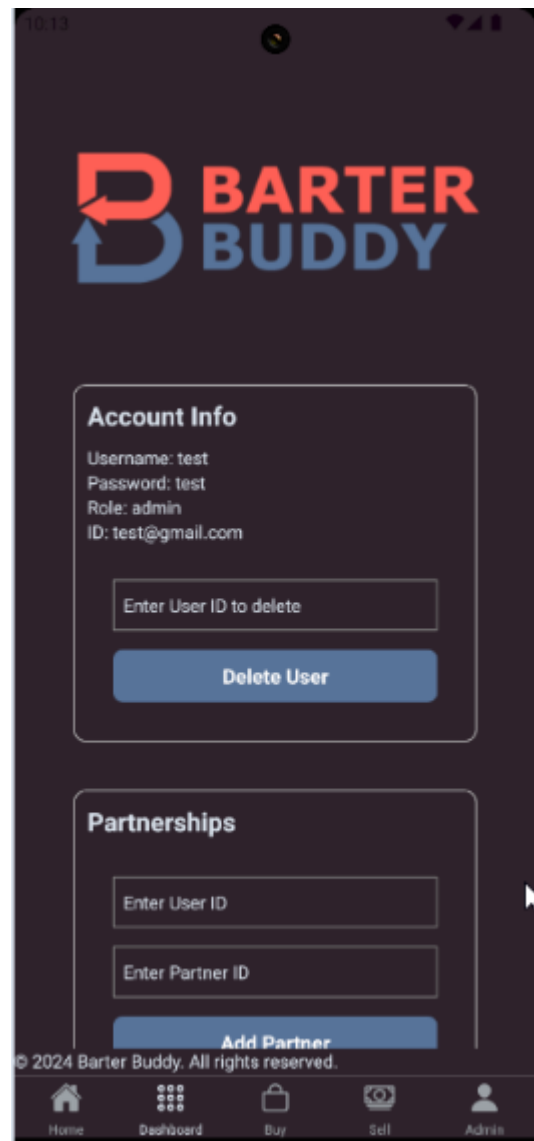


Figure 10: Top half of Barter Buddy's Dashboard

2.6 Dashboard

The Dashboard page is accessed once we are logged in. The user can currently change the color theme, log out, then they can access their account, delete their account, see, add, and delete their partnerships. The color theme only has two current themes, light and dark, but it is extremely easy to add more if our user base wanted that.

2.7 Not Found page

The not found page is accessed when you end up on a page that does not exist. Has a navigation link back to our home page which is `app/index.tsx`. Plans to include a footer and header are in the making.

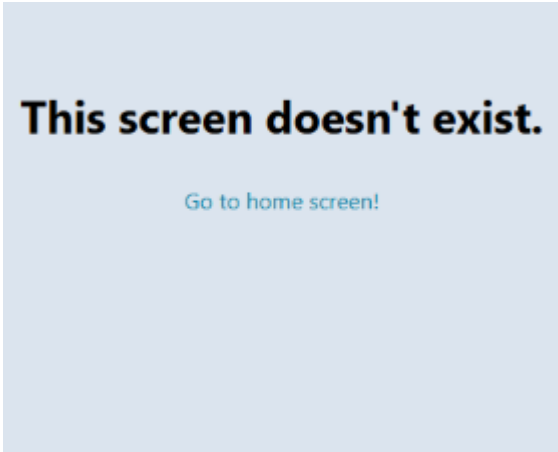


Figure 11: Not Found Page

2.8 Admin Dashboard

The admin page is accessed after a user is logged in with an admin role. This page has access to all database data and most functions of CRUD, with a focus on deletion and reading all the data. Essentially as of now a user on this page can access all the account info and partnerships in the database, and delete and add to that data.

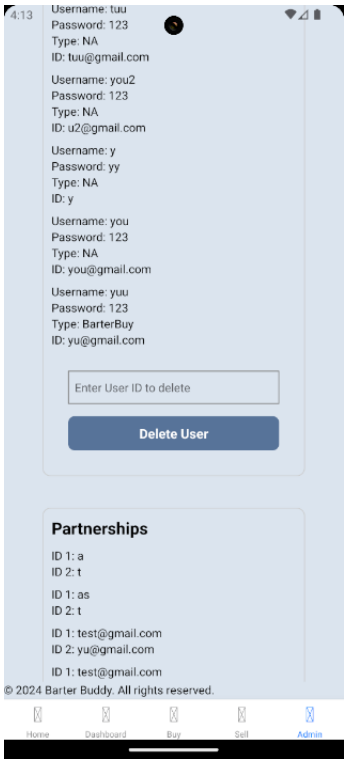


Figure 12: Top half of Barter Buddy’s Admin Page

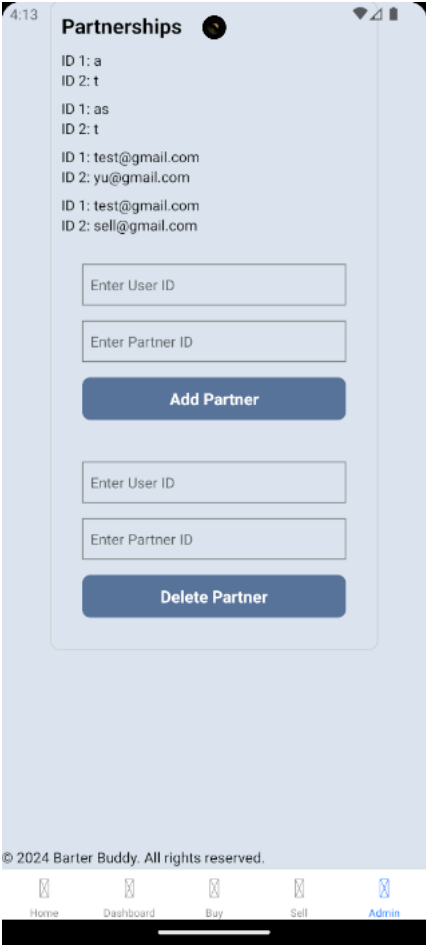


Figure 13: Bottom half of Barter Buddy’s Admin Page

2.9 BarterBuy

The BarterBuy page is accessed after a user is logged in with a BarterBuy role. This page has access to a bulletin board where they can view sell posts and buy using items they have.

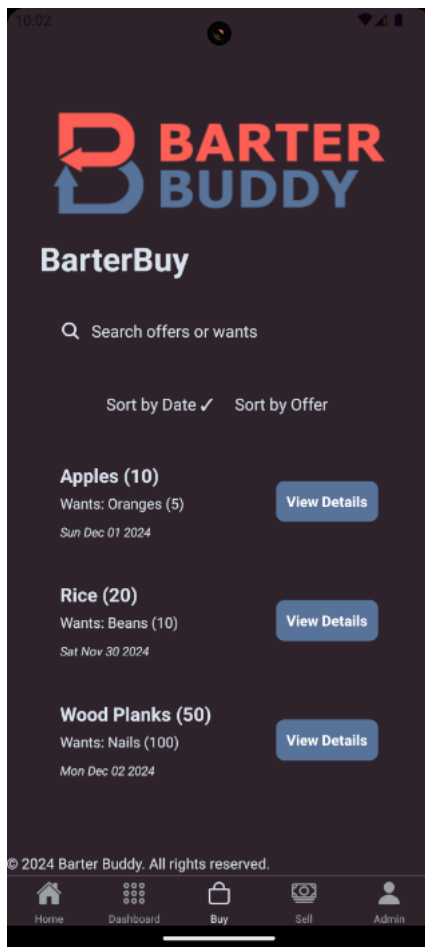


Figure 14: Barter Buddy's BarterBuy Page

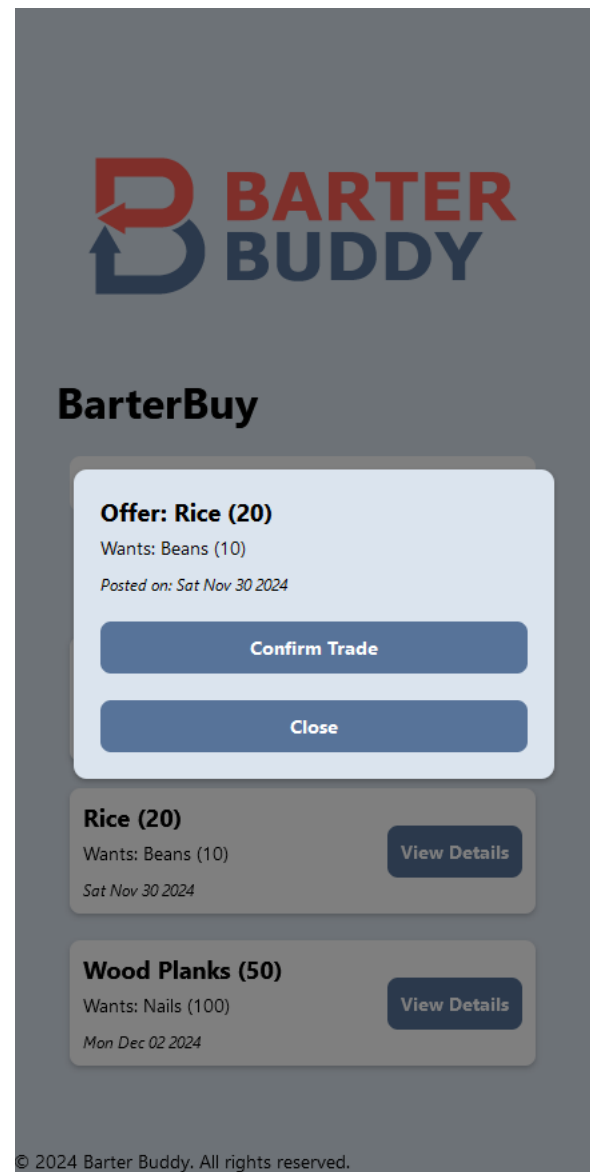


Figure 15: Barter Buddy's BarterBuy Page

2.10 BarterSell

The BarterSell page is accessed after a user is logged in with a BarterSell role. This page has the ability to create sell posts that get posted to the bulletin board.

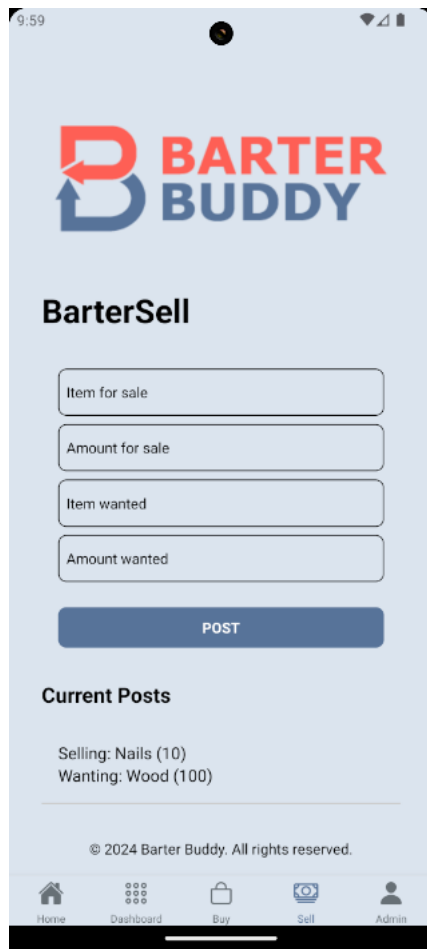


Figure 16: Barter Buddy's BarterSell Page

3 CONCLUSION

Barter Buddy fixes the issue of real-time trading on the go through providing an easily accessible app that can be used whether your mountain biking in Utah or taking a walk in Central Park. With this we conquer the complex system of the digital marketplace, allowing users to sell and buys goods anonymously with a quick and easy bulletin board. The design of Barter Buddy's database and application provides an exceptional platform for just that; real-time trading.

4 SOURCE CODE

The source code for this project is available at https://github.com/ashep04/CS360_Barter_App

5 REFERENCES

- [1] Hasan Jamil. CS 360: Database Systems. Retrieved September 1, 2024 from <https://canvas.uidaho.edu/courses/31052>
- [2] Wikipedia. Android Studio. Retrieved October 21, 2024 from Wikipedia
- [3] Expressjs. Express. Retrieved October 21, 2024 from <https://expressjs.com>