

FINAL REPORT

Principles of Software Engineering, Group 28

Olivia Logan

olivia.n.logan@vanderbilt.edu

Amanda Peppard

amanda.r.peppard@vanderbilt.edu

Lauren Scott

lauren.e.scott@vanderbilt.edu

Hilly Yehoshua

hilly.yehoshua@vanderbilt.edu

TABLE OF CONTENTS

Project Overview	3
User Stories	12
Implementation Report	15
Picture of Success	16
Changes	17
What We Did Well	17
What We Did Poorly	19
Other Lessons	20

Project Overview

Many students at Vanderbilt University are interested in sustainable fashion, but there is no convenient way to find and exchange secondhand clothing on campus. Students can purchase items from international sites such as Depop or eBay, but this feels impersonal and introduces environmental costs associated with shipping that undermine sustainable intentions. Alternatively, students can shop locally using Reuse Vandy, the predominant platform for sharing used goods at Vanderbilt. Reuse Vandy is an unorganized group chat where over four thousand members mention available items and attempt to organize transactions. Hundreds of messages are sent every day, and there is no way to search for an item, check if an item is still available, or easily bid on an item. To make matters worse, scammers are often added to the chat, and students lose money making what they erroneously believed were honest purchases. Due to these limitations, Reuse Vandy is not a viable option to exchange goods at Vanderbilt.

This motivated our team to partner with Vanderbilt Fashion Week, a registered student organization involved in the sustainable fashion space, to create a solution aimed at streamlining clothing exchanges at Vanderbilt. We developed an iOS application entitled Campus Closet that allows students to efficiently find, purchase, and sell secondhand apparel on campus. Our collaboration with Vanderbilt Fashion Week, or VFW, is integral to this project. With the resounding success of their annual fashion shows aimed at celebrating sustainable fashion in Nashville, VFW now seeks to expand the scope of their activities to help Vanderbilt students sustainably purchase apparel, including garments featured at their shows, without leaving campus. Throughout the semester, we have built an app that realizes this vision and helps build a future in which students can easily exchange their fashion items to simultaneously help each other and the environment.

Our app Campus Closet is a fully deployed iOS app that is currently available on the App Store (fig. 1). To develop the app, we used SwiftUI for the UI and UX frontend design and Firebase for the backend database and storage. When a user opens the app, they first see a login screen (fig. 2) that allows them to input their account credentials, reset their password, sign up for an account, or access the marketplace as a guest. If they click the link indicating they forgot their password, they reach a screen where they can enter their email and receive a link to reset the password to their existing account (fig. 3). If they click the link to sign up, they are directed to a separate sign up screen (fig. 4). They can enter their Vanderbilt email and password in the corresponding input fields to create a new account. If they do not enter a valid Vanderbilt email, an error message is displayed. Otherwise, they are directed to a screen informing them that they received an email with a link to verify their new account (fig. 5). By opening this email and clicking the attached link, the user is approved to access Campus Closet. They can easily return to the login screen by clicking the provided button on the app's email verification screen.

The final option for users on the login screen is to enter the app as a guest. This directs the user to a limited version of the app's marketplace (fig. 6) where they can only view items and corresponding details. We implemented a search bar, filters, and sorting options so that users can easily locate desired items. Since our target audience consists of Vanderbilt students, we chose to limit features such as creating a profile, posting items, and

purchasing items to Vanderbilt student users. To access these features, a user must submit their valid credentials on the login screen. This directs them to the complete marketplace home screen (fig. 7) and provides them with access to all app features.

Users can click the person-shaped icon at the bottom right of the screen to view their profile. A profile has two view modes - buyer and seller - that are managed by a toggle button. In buyer mode (fig. 8), a user can see the items they have saved for future reference and the items on which they have bid. In seller mode (fig. 9), a user can see the items they have listed and sold. To update their profile or sign out, the user can press the pencil icon at the upper right, which directs them to the edit profile screen (fig. 10).

There are two core functionalities available to users - posting an item as a seller and purchasing an item as a buyer. The latter also involves capabilities such as viewing an item, placing a bid, and rating a seller from whom they have purchased an item. To post an item, a user must click the plus button at the upper right of the marketplace screen. This instigates a progression of screens on which the user can upload an image of their item (fig. 11), enter required information including its size, price, condition, and description (fig. 12), and then add optional information including its tags (fig. 13). Once an image is posted, the seller is navigated to that image's detail view (fig. 14). This detail screen becomes instantaneously available to all other users in the marketplace view.

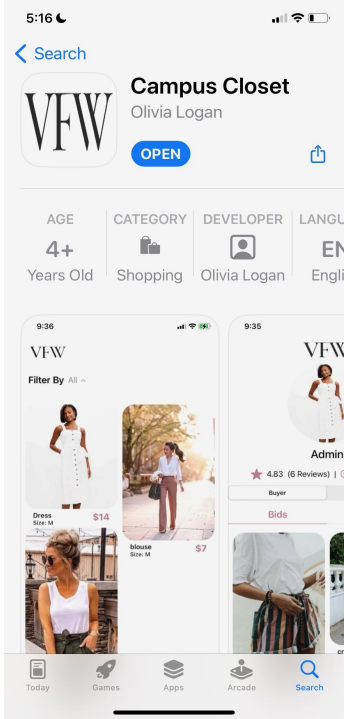
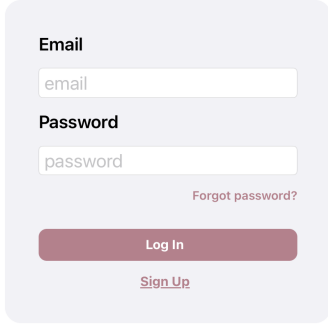
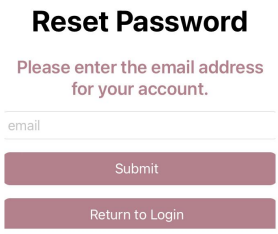
From the buyer perspective, a user can save an item by clicking the heart button on the detail view. This adds the item to the saved items on their profile. They can also view the seller's profile by clicking on their profile picture or bid on the item by pressing the "Place Bid" button. This directs them to a bid page (fig. 16) where they can make an offer. Upon placing an offer, the bidder sees a popup that describes how their decision to shop for secondhand clothing can benefit the environment (fig. 17).

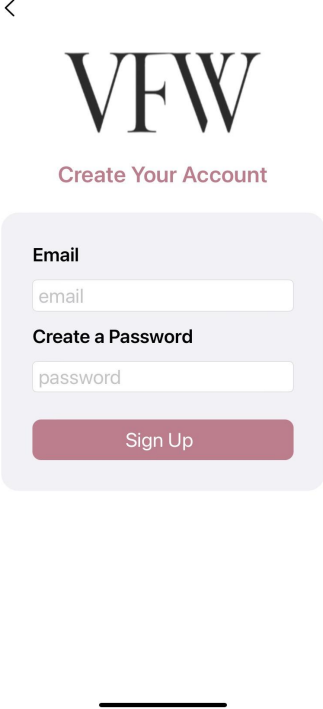
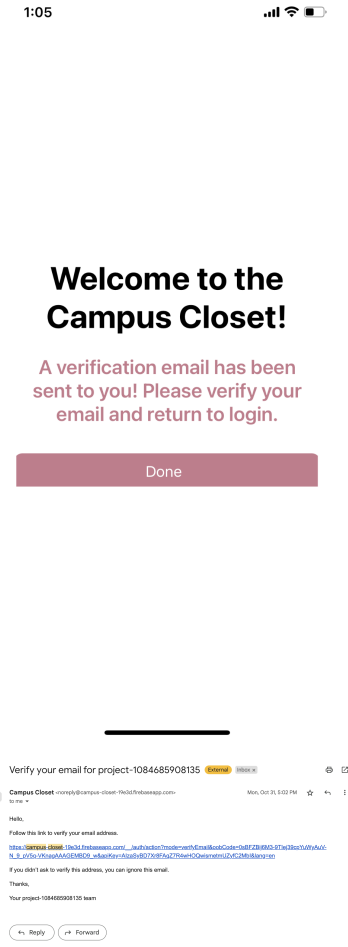
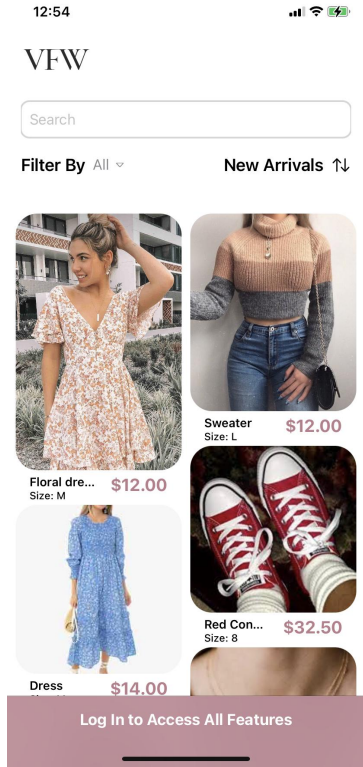
The item's seller receives a push notification once a bid is placed. They can then review the offer and choose whether or not to accept it. After accepting a bid, the seller is directed to message the buyer and coordinate details pertaining to payment and the item exchange (fig. 18). Based on their interactions, the buyer can also leave an optional rating for their seller (fig. 19). This helps potential future buyers make informed decisions about whether they would like to bid on items listed by the seller.

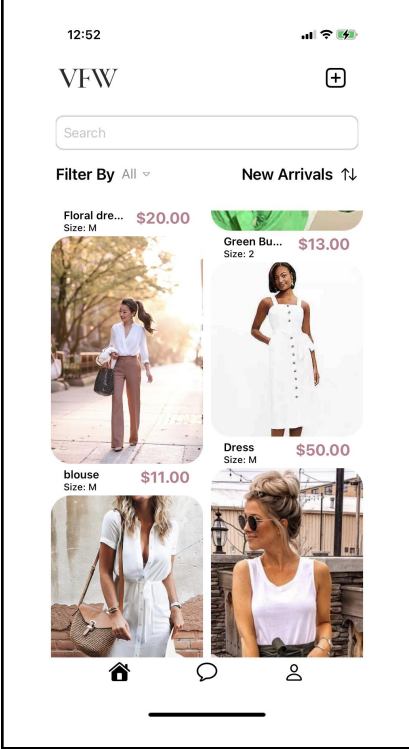
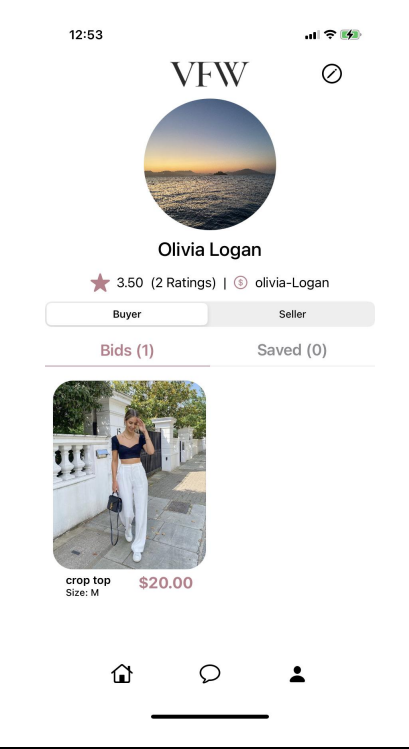
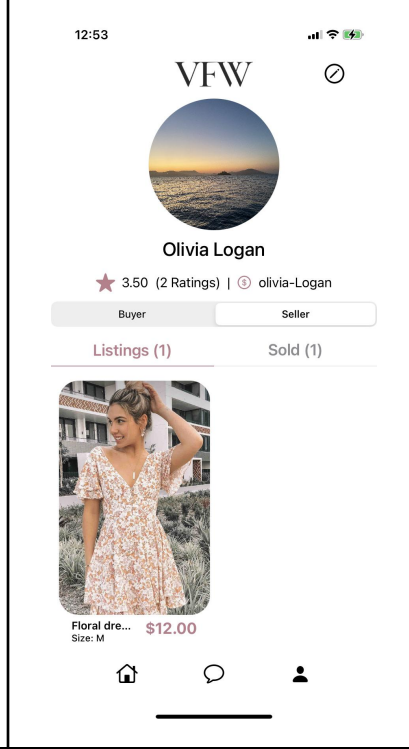
As previously mentioned, buyers and sellers can easily communicate exchange details by using our messaging platform. Users can access this subsystem by clicking the chat icon at the bottom center of the screen, which displays their recent messages (fig. 20) and allows them to send a message to any user (fig. 21). Alternatively, they can message a specific seller by tapping the messaging icon on an item's detail view page (fig. 19).

The goal of the messaging platform, like the overarching goal of Campus Closet, is to provide an experience that is usable, accessible, and safe for Vanderbilt student users. Our team achieved all of our planned user stories except when necessitated by changes in client requirements or priorities. We are extremely satisfied with our product, and it is our hope that this application will prove useful to students looking to overcome the limitations of Reuse Vandy and promote sustainable clothing exchange on the Vanderbilt campus.

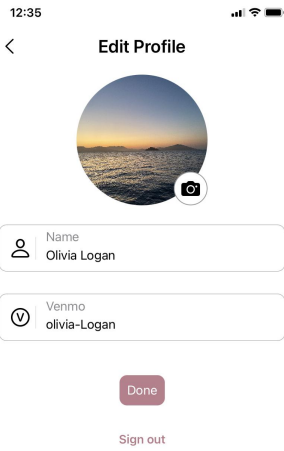

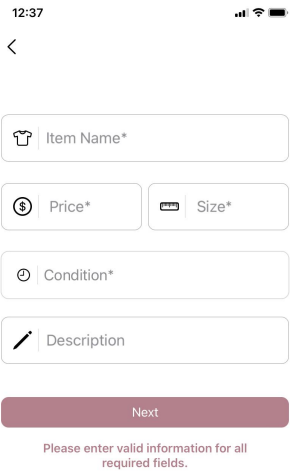
Figures

		
<p><i>Fig. 1:</i> Campus Closet is available on the Apple Store. Any user with iOS 16 can download and use the app for free.</p>	<p><i>Fig. 2:</i> This screen appears upon opening the app. Once a user signs in with valid credentials, their login persists until they sign out.</p>	<p><i>Fig. 3:</i> A user can enter their email to get a link to reset their password. Only emails with Campus Closet accounts are accepted.</p>
<p><i>User Story:</i> As a Vanderbilt student, I need to be able to download the app on the App Store so that I can access it on my Apple device.</p>	<p><i>User Stories:</i> As the client, I need to see the Vanderbilt Fashion Week logo when the app opens so that users can identify the app's sponsor. As a user, I need to be able to log in with my Vanderbilt credentials so that I can access the app as a current student. As a user, I need to have the option to save my login information so that I do not have to log in every time I open the app.</p>	<p><i>User Story:</i> As a user, I need to be able to reset my password so that I do not lose access to my account if I forget my login credentials.</p>

		
<p><i>Fig. 4:</i> This is the sign up page where a user can create a new account with a Vanderbilt email.</p>	<p><i>Fig. 5:</i> This screen notifies a user that a verification link has been sent to their email. After clicking the link, the user can access the app.</p>	<p><i>Fig. 6:</i> The guest mode of the marketplace only allows a user to view some information about currently listed items.</p>
<p><i>User Story:</i> As a user, I need to be able to log in with my Vanderbilt credentials so that I can access the app as a current student.</p>	<p><i>User Story:</i> As a user, I need to be able to log in with my Vanderbilt credentials so that I can access the app as a current student.</p>	<p><i>User Story:</i> As a Vanderbilt student, I need to look at listed items without making an account so that I can decide if I am interested in registering for the app.</p>

		
<p><i>Fig. 7:</i> This screen is the marketplace view seen by a registered user. It offers features not available in guest mode (fig. 6).</p>	<p><i>Fig. 8:</i> This is a user's profile under buyer mode. They can see their saved items and the items on which they have placed a bid.</p>	<p><i>Fig. 9:</i> This is a user's profile under seller mode. They can see their items for sale and the items they have already sold.</p>
<p><i>User Stories:</i></p> <p>As a buyer, I need to be able to use a search bar so that I can find specific listed items that I need.</p> <p>As a buyer, I need to be able to filter items based on price range so that I can choose an item within my budget.</p> <p>As a buyer, I need to be able to choose categories of items that I want to view so that I can browse the types of items I need even if I do not know which specific item I want.</p> <p>As a seller, I need items that are sold to be eliminated</p>	<p><i>User Story:</i></p> <p>As a buyer, I need to have access to a "my cart" so that I can save items that I might be interested in purchasing.</p>	<p><i>User Story:</i></p> <p>As a buyer, I need to see a seller's rating and number of items previously sold on their profile so that I can determine their credibility and ensure that I will not get scammed.</p>

from the app so that I don't get requests on items that are no longer available.
 As a buyer, I need to not see unsold items that were posted more than two weeks ago so that I can view newer, more popular items instead.
 As a user, I need all posts, images, and messages to load quickly so that I can use my shopping time efficiently.

		
<p><i>Fig. 10:</i> This is the edit profile screen. A user can update their name and Venmo, upload a new profile picture, or sign out of their account.</p>	<p><i>Fig. 11:</i> This is the first page that a user sees when posting an item. It lets them upload a picture from the camera roll on their device.</p>	<p><i>Fig. 12:</i> After uploading a picture, the next page allows the seller to specify details about the item such as the name, price, size, condition, and description.</p>
<p><i>User Story:</i> As a user, I need to be able to create a profile that</p>	<p><i>User Story:</i> As a seller, I need to be able to upload an image of an</p>	<p><i>User Stories:</i> As a seller, I need to be able to specify the list price of an</p>

includes my name and picture so that buyers and sellers can easily identify me.

item so that buyers can see what I am selling and make an informed decision about bidding on it.

item I am posting so that potential buyers know how much I expect to receive for the item.
As a seller, I need to be able to add a condition tag (new, lightly used, heavily used) for an item so that potential buyers understand its current state..

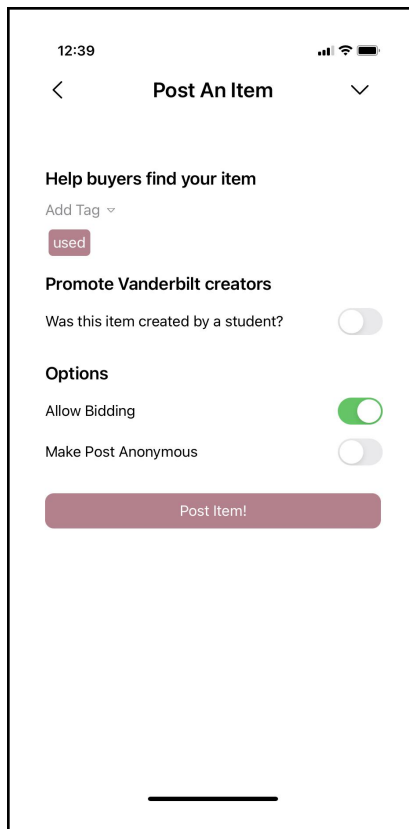


Fig. 13: This is the final page presented before an item is posted to the marketplace. Here, the seller can enable bidding, add tags, and make the post anonymous.

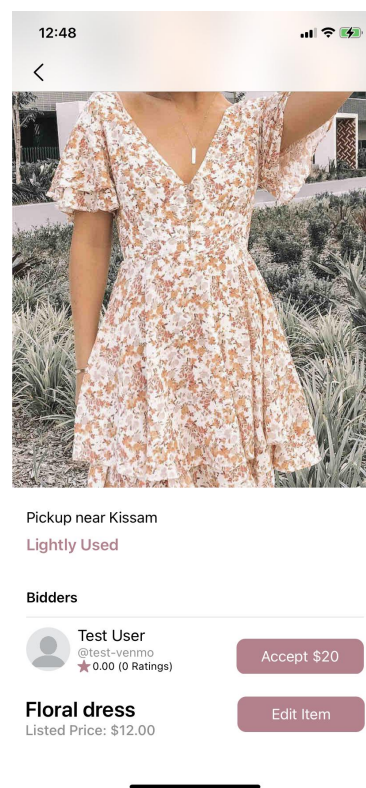


Fig. 14: This is the item detail view seen by the item's seller. They can see how the item looks to potential buyers when they are scrolling through the home page. They can also edit the post and view bids.

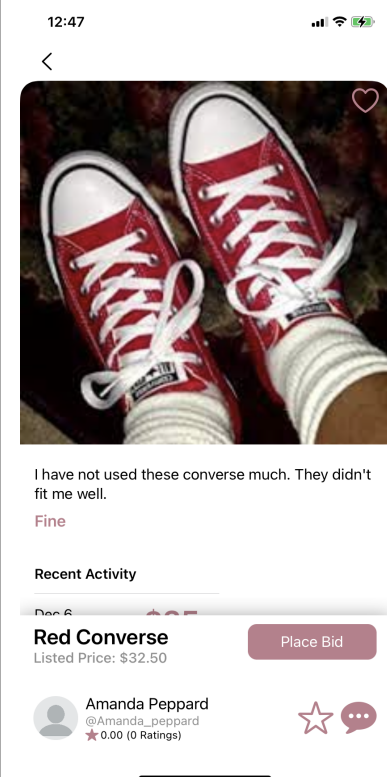
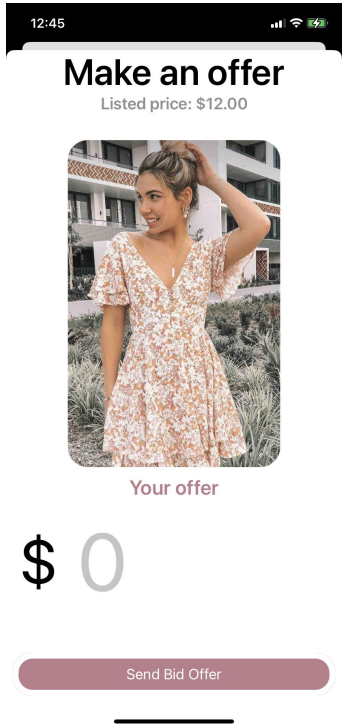
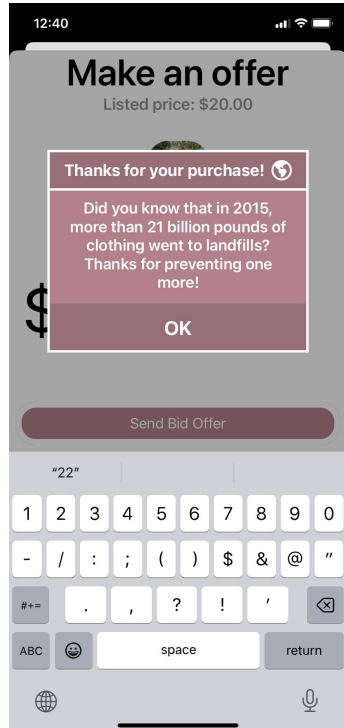
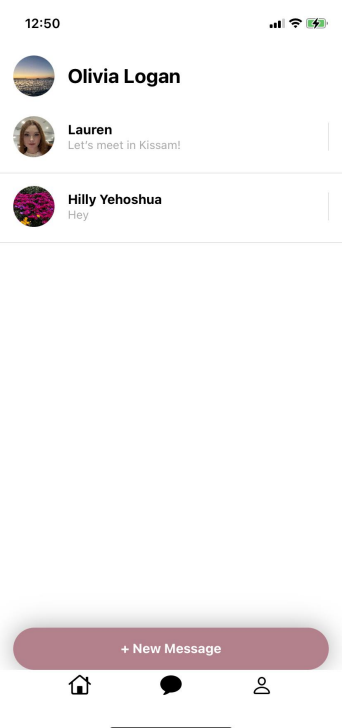
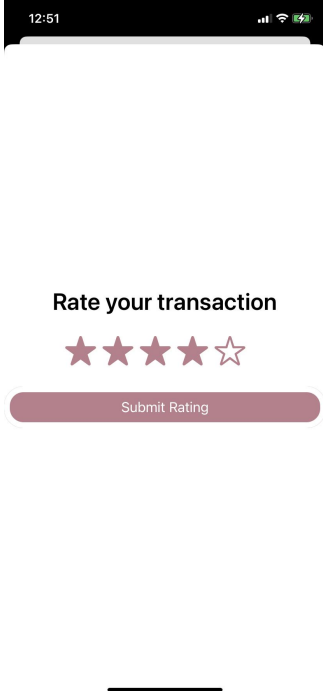
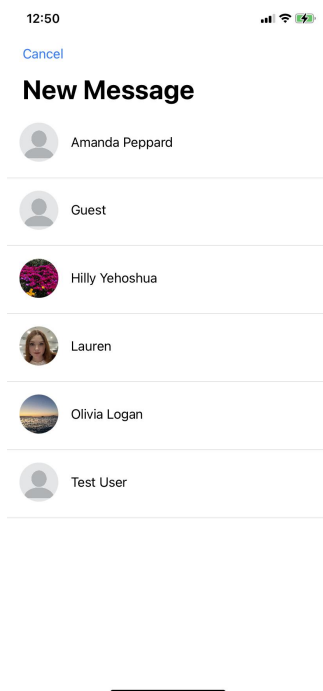
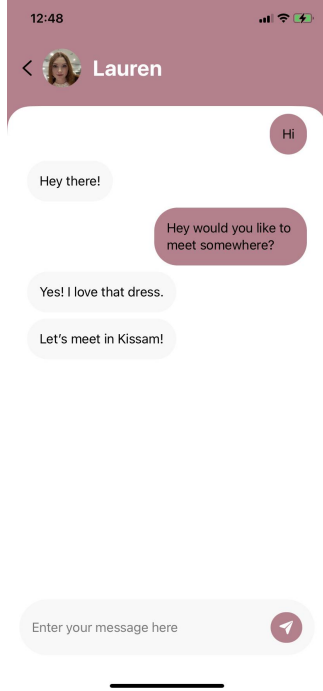


Fig. 15: The item detail view seen by a potential buyer. They can "save" the item by clicking the heart icon on the top right of the picture, identify the seller, see current bids on the item, and place their own bid.

<p><i>User Stories:</i></p> <p>As a seller, I need to be able to enable bidding for an item so that buyers can compete with each other for the item and hopefully pay me a high price.</p> <p>As a seller, I need to be able to “post” an item to the marketplace so that other users can see what I am selling.</p>	<p><i>User Stories:</i></p> <p>As a seller, I need to be able to edit my listings so that potential buyers can see any changes in item details.</p> <p>As a seller, I need to be able to accept, reject, or negotiate a bid so that I can get an acceptable price for an item.</p>	<p><i>User Story:</i></p> <p>As a buyer, I need to be able to bid on items that I want to buy so that I can try to get a bargain.</p>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------

		
<p><i>Fig. 16:</i> After clicking “place bid,” a user can submit an offer for an item. The offer needs to be higher than the listed price or the app displays an error message.</p>	<p><i>Fig. 17:</i> When a user places a valid bid on an item, they see a popup message that describes the environmental impact of their decision to bid on a secondhand item.</p>	<p><i>Fig. 18:</i> This screen shows a user their message history and allows them to start a new conversation. It can be reached from the lower navigation bar.</p>
<p><i>User Stories:</i></p> <p>As a buyer, I need to be able to bid on items that I want</p>	<p><i>User Story:</i></p> <p>As the client, I need every person who makes a</p>	<p><i>User Story:</i></p> <p>As a user, I need to be able to message another user so</p>

<p>to buy so that I can try to get a bargain. As a seller, I need to get a push notification when a bid is placed on an item I posted so that I can look at the proposed price.</p>	<p>purchase to see a small infographic about their sustainability impact so that buyers and sellers are motivated to use this app instead of fast fashion.</p>	<p>that I can communicate logistics regarding payment method and pickup location.</p>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	----------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------

		
<p><i>Fig. 19:</i> On this page, a user can rate their interactions with an item's seller.</p>	<p><i>Fig. 20:</i> This screen allows a user to view other Campus Closet users and send a message to any of them.</p>	<p><i>Fig. 21:</i> This is the screen where two users can send messages to each other.</p>
<p><i>User Story:</i> As a buyer, I need to be able to leave an optional rating for my seller so that their future buyers can accurately determine whether or not they are trustworthy.</p>	<p><i>User Story:</i> As a user, I need to be able to message another user so that I can communicate logistics regarding payment method and pickup location.</p>	<p><i>User Stories:</i> As a user, I need to be able to message another user so that I can communicate logistics regarding payment method and pickup location. As a user, I need to get a push notification when I have received a message so that I can quickly respond.</p>

User Stories

These user stories guided our development tasks. Stories 1-25 were included in our Requirements Analysis report. The remaining stories were added during the semester to reflect additional requirements and ensure compliance with our client's priorities and the Apple Store's policies. The effort estimates for these stories were completed during sprint planning meetings.

Story Description	Estimated Effort	Actual Effort	Priority Level	Status
1. As a user, I need to be able to log in with my Vanderbilt credentials so that I can access the app as a current student.	7 hours	13 hours	Critical	Complete
2. As a user, I need to be able to create a profile that includes my name and picture so that buyers and sellers can easily identify me.	10 hours	17 hours	Critical	Complete
3. As a seller, I need to be able to "post" an item to the marketplace so that other users can see what I am selling.	10 hours	23 hours	Critical	Complete
4. As a seller, I need to be able to specify the list price of an item I am posting so that potential buyers know how much I expect to receive for the item.	2 hours	1 hour	Critical	Complete
5. As a seller, I need to be able to add a condition tag (new, lightly used, heavily used) for an item so that potential buyers understand its current state.	2 hours	2 hours	Minor	Complete
6. As a buyer, I need to be able to use a search bar so that I can find specific listed items that I need.	7 hours	3 hours	Minor	Complete
7. As a user, I need to be able to message another user so that I can communicate logistics regarding payment method and pickup location.	10 hours	20 hours	Major	Complete
8. As a seller, I need to be able to enable bidding for an item so that buyers can compete with each other for the item and hopefully pay me a high price.	2 hours	8 hours	Critical	Complete

9. As a buyer, I need to be able to bid on items that I want to buy so that I can try to get a bargain.	3 hours	5 hours	Critical	Complete
10. As a seller, I need to be able to accept, reject, or negotiate a bid so that I can get an acceptable price for an item.	3 hours	1 hour	Critical	Complete
11. As a buyer, I need to have access to a "my cart" so that I can save items that I might be interested in purchasing.	5 hours	4 hours	Major	Complete
12. As the client, I need every person who makes a purchase to see a small infographic about their sustainability impact so that buyers and sellers are motivated to use this app instead of fast fashion.	2 hours	3 hours	Major	Complete
13. As a seller, I need items that are sold to be eliminated from the app so that I don't get requests on items that are no longer available.	1 hour	1 hour	Minor	Complete
14. As a buyer, I need to have access to a Vanderbilt campus map with the marked pickup location so that I can know where to collect the item.	5 hours	0 hours	Minor	Not Started
15. As a user, I need to have the option to save my login information so that I do not have to log in every time I open the app.	2 hours	<1 hour	Major	Complete
16. As a seller, I need to be able to edit my listings so that potential buyers can see any changes in item details.	1 hour	3 hours	Major	Complete
17. As the client, I need to see the Vanderbilt Fashion Week logo when the app opens so that users can identify the app's sponsor.	1 hour	<1 hour	Major	Complete
18. As a buyer, I need to be able to filter items based on price range so that I can choose an item within my budget.	4 hours	5 hours	Minor	Complete
19. As a buyer, I need to be able to choose	4 hours	9 hours	Major	Complete

categories of items that I want to view so that I can browse the types of items I need even if I do not know which specific item I want.				
20. As a buyer, I need to not see unsold items that were posted more than two weeks ago so that I can view newer, more popular items instead.	3 hours	0 hours	Minor	Not Started
21. As a buyer, I need to be able to leave an optional rating for my seller so that their future buyers can accurately determine whether or not they are trustworthy.	4 hours	4 hours	Minor	Complete
22. As a buyer, I need to see a seller's rating and number of items previously sold on their profile so that I can determine their credibility and ensure that I will not get scammed.	4 hours	3 hours	Minor	Complete
23. As a user, I need to get a push notification when I have received a message so that I can quickly respond.	3 hours	4 hour	Major	Complete
24. As a seller, I need to get a push notification when a bid is placed on an item I posted so that I can look at the proposed price.	3 hours	7 hours	Major	Complete
25. As a user, I need all posts, images, and messages to load quickly so that I can use my shopping time efficiently.	1 hour	<1 hour	Major	Complete
26. As a Vanderbilt student, I need to be able to download the app on the App Store so that I can access it on my Apple device.	2 hours	4 hours	Critical	Complete
27. As a Vanderbilt student, I need to look at listed items without making an account so that I can decide if I am interested in registering for the app.	4 hours	3 hours	Critical	Complete
28. As a user, I need to be able to reset my password so that I do not lose access to my account if I forget my login credentials.	2 hours	5 hours	Critical	Complete

29. As a seller, I need to be able to upload an image of an item so that buyers can see what I am selling and make an informed decision about bidding on it.	2 hours	16 hours	Critical	Complete
--------------------------------------------------------------------------------------------------------------------------------------------------------------	---------	----------	----------	----------

Implementation Report

We decided to create a native iOS application using SwiftUI for iOS 16 and a Firebase database. We also used the Swift Package Manager to install a small number of third party libraries, including those necessary for Firebase and one to display the Toggle in the profile view. We decided to use SwiftUI due to its responsiveness, ease of use, and popularity in iOS development circles. We chose to use a Firestore database because it ensures simple, secure authentication code, and it is highly compatible with Google services such as a map we intend to include in the future.

One major roadblock we encountered was an issue with the image picker. This picker is used to support profile pictures and item images in our application. The problem was that images were getting selected, but due to inconsistencies with asynchronous execution, they were not getting updated in the database. We overcame this problem by collaboratively tackling it in pair programming sessions. After this experience, we were better able to handle async issues that arose during development.

Another problem we experienced involved passing data between views, such as when a bid was placed or an item was posted. We encountered this issue due to our internal structure, which contains separate view models for the child and parent that both need to access the same item. However, we wanted to prioritize separate, readable code, so rather than using one overarching view model to solve the problem, we decided to copy over objects from the parent to the child view.

Additionally, we encountered issues with UI testing and in XCode as a whole. As we came to realize, UI tests in XCode do not work 100% of the time, so it proved difficult to know whether our app had an issue or the UI test was playing incorrectly. XCode would also often spend upwards of twenty minutes loading packages and processing files, sometimes showing errors that didn't actually exist in the app. This initially took away some time from our development, but we eventually realized that periodically restarting XCode, cleaning the build, and re-downloading the app helped. Lastly, we experienced a roadblock with push notifications. Apple's push notification process occurs on the server side and uses HTTP protocols, which made debugging the process difficult because failures only showed a 400 status code rather than a helpful error message. However, we overcame this roadblock by watching tutorials and learning about HTTP protocols in Swift.

We have completed the implementation for all but two of our user stories, which we did not implement due to changes in client requirements. However, there are many improvements that we would like to make in the app before publicizing it to Vanderbilt students. For instance, we would like to ensure that users can only rate other users with whom they are involved in a bid process. Currently, users are able to rate any other users in the app, which may result in inaccurate or misleading ratings. Additionally, when users delete their account, the app does not yet delete all of the data associated with it. Lastly, we

would like to improve navigation in the app to make it highly intuitive and give users the best possible overall experience. We expect to complete this additional work over the course of the winter break and release the app for students early in the spring semester. Afterwards, we have further ideas for additional features. For instance, as previously mentioned, we would like to support a campus map where users can easily choose a pickup location. Also, we plan to connect the app with Venmo to facilitate payment. Lastly, we would like to adapt the app for release on other college campuses.

Picture of Success

When compared to the deliverable we initially proposed in our Requirements Analysis document, the final version of Campus Closet implements nearly all planned capabilities and supports several features beyond the predicted scope of functionality. We developed core features for the app including user authentication, profile creation, item bidding, post searching and filtering, transaction rating, user messaging, and push notifications. All of these were proposed in our initial Requirements Analysis report. The only two features we planned but did not develop were a Vanderbilt campus map that users could reference when designating an exchange location and the automatic timeout of listed items after two weeks. During a progress meeting with our client, the VFW executive board informed us that they no longer desired these features in the app. In particular, they found item timeout to be unnecessarily restrictive since we support sorting items by their post time. Instead, they requested a new feature - the ability to tag items as “Vanderbilt created” to highlight garments made by student creators. Although we did not include this feature in our initial report, it appears in the final version of Campus Closet to satisfy the requirements of our client.

Some of our design choices also changed over the course of the semester. For example, we initially planned to add a link in the navigation bar to a page solely dedicated to viewing one’s bids. We envisioned that another distinct page would display one’s liked items. However, we received valuable feedback from Dr. Singh and some of our first user test participants who expressed that this design felt cumbersome and unintuitive. We responded by combining the pages for viewing bids and likes to appear under a user’s profile. Now, users can easily see all of their sold items, bids, and saved items on one page.

Overall, although our team added new features to Campus Closet and modified several predicted design choices, our final deliverable largely aligns with the Requirements Analysis document that we developed at the beginning of the semester. This alignment is even evidenced by our project timeline. Not only did we achieve most of our development goals, but overall, our progress matched our predicted timeline for each sprint. We approached this project in phases so that each sprint focused on a few related features, and thanks to the consistent efforts and quality work of all team members, we were able to meet our predicted milestones and ultimately deliver a quality product.

Changes

Our project evolved over the course of the semester to fulfill the requirements and reflect the priorities of our stakeholders. In particular, our work changed according to

feedback from our client, the App Store, and potential users including user test participants and our team's internal members.

First, we frequently updated the VFW executive team on our development progress and requested their feedback. In these meetings, we learned of several changes in their initial requirements. For example, VFW initially requested a map feature that would help users find and designate convenient locations on Vanderbilt's campus to exchange items in person. Around mid-semester, they expressed to our team that this was no longer a desired feature for the app. Students are already familiar with the campus layout, and there is little reason to suspect that they would use this feature over existing services like Google Maps or Apple Maps. Instead, they encouraged us to use the effort we would have exerted in developing the map to support a "Vanderbilt created" tag. This tag indicates items that are handmade or altered by Vanderbilt students, allowing them to highlight their own creative work. We implemented this tag to satisfy our client. Another change in client requirements occurred after we demonstrated our implementation of sorting functionality. This caused VFW to rescind their initial request that we remove items from the marketplace if left unpurchased after two weeks. Therefore, we did not develop this functionality. The sorting approach we developed satisfied the initial reason for the client's request - to ensure that users can easily see recently listed items on the app.

Campus Closet also changed according to requirements imposed by the App Store. When we submitted our project for deployment, our first request was rejected since we did not offer a way for users to browse the marketplace without creating an account. Since our app is not intended for a public audience - it is explicitly designed for Vanderbilt students - we approached our client with several ideas to satisfy this request. Together, we decided to support a guest view for the marketplace that only allows users to browse listings. Guest users cannot list their own items, bid on items, message users, or create profiles. This approach pleased our client and satisfied the App Store - our request for app publication was approved after we made the revision.

We also adapted our solution design based on the feedback of potential Campus Closet users. This audience includes participants in our user testing efforts and members of our team. One change we implemented relates to the pages in the app. We initially planned to create separate pages with tabular formats where users could view items they had saved and those they had bid on. However, user feedback revealed that this was confusing and unintuitive. It was also difficult to see the small item images in the compact tabular format. As a response, we modified our solution design to list saved items and those with bids in two lists accessible by a simple toggle. We display large images to ensure that users, including those with sight impairments, can easily see and select items.

What We Did Well

While creating Campus Closet, our team encountered several hurdles that we successfully addressed. The first challenge was that our team member Lauren does not have a Mac, so she does not have access to XCode. In order for our team to develop an iOS application, she used a virtual Mac through MacInCloud. This service cost her approximately \$50 per month, introducing a financial barrier to our team before we started

development. However, this challenge did not impede our progress since Lauren was able to find funding to cover the cost.

During Sprint One, we strove to establish our app's login and sign up capabilities. Our initial plan involved supporting SSO with Vanderbilt emails and Duo, so we contacted Vanderbilt IT in September to request these permissions. We did not hear back from them until November, so we had to pursue another approach to authentication. We chose to require email verification and only send verification links to email addresses with vanderbilt.edu domains. This approach has worked very well and even laid the groundwork for our support of password resets. Our shift to this new approach demonstrates our team's adaptability to unexpected obstacles.

In Sprint Two, we aimed to enable image upload from the camera roll to add profile pictures and item images. This presented another challenge for our app. All our images are stored in Firebase Storage buckets, but the data models associated with each user and listing reside in the Firestore database. Thus, in order to get images to load on the home view, profile view, and item detail view screens, we need to reference the paths stored in database entries, use them to fetch corresponding images from storage, and then reload the app to ensure images become immediately available to the frontend. Despite facing several bugs and setbacks due to asynchronous execution and complex loading logic, we were able to successfully complete this implementation. In particular, researching Swift closures proved extremely helpful and aided our progress. We used closures to force an image to finish uploading to storage before we attempt to fetch it and use it on frontend pages.

In Sprint Three and the last few weeks of class, our team struggled to implement push notifications. The first challenge we faced was that push notifications require a user to have a developer account, so only one team member was able to work on enabling and sending the notifications to users. The other challenge was that, for an unknown reason, push notifications were not consistent or timely during our initial implementation efforts. We gained the knowledge to resolve this issue by watching many tutorials, learning how to perform post requests in Swift, and partaking in lots of debugging in synchronous working meetings and pair programming sessions.

We also faced an obstacle when completing our first submission to Apple's App Store. Our app was initially rejected since we did not provide a guest account for users to view the marketplace without registering for an account. We responded by contacting our client regarding their vision for this functionality, promptly implementing the desired change, and resubmitting our app for reconsideration. After we added the requested functionality, Campus Closet was quickly approved and published on the App Store.

Overall, our team faced many technical and personal challenges over the course of the semester. Our technical challenges are discussed above in detail, but we also faced unexpected personal conflicts such as extended team member illnesses. However, we overcame many of these challenges by learning from online resources, working alongside each other, and sometimes covering for each other when some of us were unable to finish our internally assigned tasks.

What We Did Poorly

Several problems arose this semester due to mistakes made by our team. These include technical issues related to the app's code along with interpersonal issues related to team dynamics. Firstly, our team did not make our code exceptionally generic. The source code contains several repetitive views and elements that could be better consolidated. This issue largely arose due to our frequent prioritization of rapid development over thoughtful design. Plus, limitations in communication sometimes caused team members to implement separate solutions to similar problems instead of defining one generic solution and adapting it to solve both issues. To avoid this mistake in the future, we should focus on templating more of our UI features and certain views in order to avoid duplicate code as much as possible.

Another shortcoming of our development pertains to our handling of the database. Simply, our team did not initially set up our Firestore database very effectively. When creating the app, we often focused on frontend components and imagined how the app would look instead of thoughtfully considering how we would handle data on the backend. If we had considered the backend in greater detail at the beginning of our development process, we would have avoided many issues related to adding database fields and restructuring collections every time we needed to build a new feature. For example, when creating the messaging page, we first wrote it so that messaging would work between two users only. This solution did not scale well once we needed to support users messaging any other user of the app. Resolving this issue proved difficult and time-consuming. We had to refactor the message model for the database and track both the sender and receiver's ID in every message. Problems like this arose many times, and they could have been avoided if our team had created a mockup of our backend data near the beginning of development.

We made additional mistakes related to project and team management. For instance, in our initial time estimates, we did not consider testing efforts. The Gantt charts in our Requirements Analysis report do not include testing, and we did not initially assign testing responsibilities to any team members. This was a significant mistake on our team's part. Ultimately, one team member ended up becoming mostly responsible for the testing efforts. This caused the other members to frequently test by manually updating the database, which caused many errors. If one field was missing or one data type was incorrect in a manual entry, the entire app would crash. If our team had crafted a thorough testing plan in advance, mock data would have not been all added manually, and more time and people could have focused on testing.

If we had the opportunity to write our Requirements Analysis report again, we would also add more specific user stories. We based our time estimates and Trello board on the user stories, and it became difficult to accurately track progress when implementing some user stories required several distinct and complicated steps. This led to an accumulation of smaller unfinished tasks and generated many hours of unexpected work prior to sprint deadlines. For example, we did not consider resetting a password when creating user stories, but at the beginning of Sprint 3, we realized this was core functionality that we needed to implement. We hope to avoid this problem in the future by

employing the knowledge and experience we have now developed by working on an extended software engineering project over the course of the semester.

Lastly, our team should have listened to Dr. Singh and submitted our application to the App Store earlier. We made this mistake because we erroneously believed that our app needed to be perfect or entirely complete before beginning the submission process. In the future, we will make sure to plan for any deployments or publication deadlines far in advance. This will provide us with sufficient time to respond to suggestions and implement any additional functionality requested by the App Store.

Other Lessons

Our team learned many indispensable software engineering lessons through our work on Campus Closet. For instance, we learned that before beginning a project, not only the UI elements need to be mapped out. Instead, any software engineering team should make a plan for their database structure and understand how they align with each of the product's desired features. If the database is set up well from the beginning, it makes the coding easy - it sets up a strong foundation upon which the rest of the application can be effectively built.

We also learned that it is valuable to research different architectures to find the option that is best suited for the project. For example, we initially chose to use the MVC architecture, but we switched to MVVM once we determined that it better suits our app's goals. This shift and lack of clarity regarding the architecture choice complicated Campus Closet, and we encountered many mistakes and had to rewrite database fields and code following our architectural shift. Like planning the database fields beforehand, our team should have done more research on architectures and made an informed choice before beginning development to minimize the likelihood of changing our selection. This is a lesson that we will carry with us to our next software engineering project.

Lastly, we learned that feedback from users is extremely helpful in app design. We received feedback both from our client and from other users who participated in our user testing efforts. This feedback proved indispensable - it helped us align priorities for the app and adjust the frontend and navigation to optimize the user experience that Campus Closet facilitates. For example, we learned from users to add swiping functionality to easily navigate between pages. We also learned that the profile page should have two sides - a seller and a buyer view - to easily switch between functional roles. From the client, we learned to add a tag for Vanderbilt creators and eliminate the Vanderbilt map and item timeouts. Incorporating feedback from various stakeholders allowed us to improve the app and deliver a final version that is far better than any version we could have created without external feedback.