Maverick Home Center

CSE 5325-004: Software Engineering: Analysis, Design, and Testing

Group No. 13 - Mavericks

Team Members

S. No.	Student Name	UTA ID
1.	Aditya Vikram Bhat	1002014494
2.	Akash Biswas	1002055500
3.	Amit Gupta	1002066302
4.	Ravi Rajpurohit	1002079916

Maverick Home Center

Maverick Home Center is an app that provides home décor, used furniture, and all other accessories that one needs/wants to study. This app is for students at the University of Texas at Arlington and living close to the campus. This would allow students to have cheap (possibly free) and affordable furniture and would save the time and resources it would take to buy them firsthand or from other sellers. Maverick Home Center is here to help you whether you want to sell things you no longer need or if you're seeking a certain item at a good price.

There are many online marketplaces for buying and selling used goods, including eBay, Facebook Marketplace, Etsy, and Ebid. However, the main issue with the worldwide market is that anyone can pose as a buyer or a seller, which has the potential to defraud other users. This would ultimately result in a loss of faith in the system. Additionally, the user interface and user experience of the current market are so overwhelming that any first-time user could run into problems while attempting to complete fundamental tasks on the application.

The main advantage of having this application is that it is made by students with the aim to provide a seamless and easy-to-access place where one can get all the items one needs to have a convenient and easy journey to studies and have an immersive college life experience. Maverick Home Center, which is only available to students at The University of Texas at Arlington, overcomes all the aforementioned problems by gathering all the products in one location. Here, a buyer and seller can interact to lay the groundwork for negotiation and to have a better understanding of the state of the commodity. There would be no chance of being duped because the application only allows students who have been validated by the university to register. Additionally, the user can sell or buy a thing with just a few clicks because of the interactive design's simple user interface.

Overall Design Approach

The application would be developed by following the Object Oriented Design. Object-oriented programming (OOP) is a programming methodology that emphasizes the use of objects and classes to represent real-world entities and concepts. Object-oriented design works around the entities and their characteristics instead of the functions involved in the software design. There would be three major entities involved in the project -

- Buyer
- Seller
- Platform Maintainer

The application will support certain features which will allow the user to do the following actions -

- Searching for the product
- Placing a buy request
- Communicating with the sellers

Also, the sellers will be able to do certain actions -

- Upload product images
- Insert product descriptions
- Communicate with the buyer

The development would follow a Top-Down design approach by utilizing Agile Methodology. The entire application would be broken down into multiple sub-systems and components. Then, those sub-systems would be analyzed and worked upon until the entire application is built. The tentative sub-systems defined are mentioned below -

- The primary goal is to create a personalized onboarding and login system to collect initial custom data. This data will be utilized to develop the following sub-systems.
- This would be followed by the development of the landing or items screen subsystem.
- The final step is to develop the payment gateway so that the payment record is created and stored.
- At the end, the process of testing would take place to ensure quick deployment of the completed subsystem

System Design

Database Design:

The application would utilize Firebase Database to store all the data. The data would be stored as

JSON objects as a tree. For example:

```
{
        "users": {
                "1002012345": {
                        "firstName": "Adam",
                        "lastName": "Smith",
                        "mavId": 1002012345,
                        "phoneNo": 6821234321,
                        "dob": 929173819738,
                        "phoneExt": "+1"
                },
                "1002012346": {
                                …
                }
        }
}
```

User Interface Aspects:

Inputs:

The system enables both buyers and sellers to interact with it in multiple ways. It allows users to upload media files, including product images, and record geographical data in terms
of latitude and longitude. Additionally, buyers can fill out forms to input product specifications.

Outputs:

Users can search for specific products based on selected filters and search criteria. The application also sends push notifications to both buyers and sellers when a message is received. Buyers have the ability to track their product listings.

User Interface Design:

Login/ Sign-Up Screen:
The login or Sign-up window will enable authentication for the application's user and also allow new users to onboard the application.
Sign-up would require basic details such as First Name, Last Name, UTA Email, UTAID, Date of Birth, and phone number.
Once the user has signed up, they can always log in with their registered details.

Dashboard Screen:
The initial dashboard view for both buyers and sellers will show a list of products available for sale in the local vicinity. Additionally, there will be a floating action button that enables users to access the product selling page.

Profile Screen:
The profile screen would contain all the basic actions such as view profile, edit profile, logout, and settings.

Product Feed Screen:
The purpose of this screen is to enable users to generate a product listing. It begins by allowing the user to upload an image of the product, followed by input fields for product details such as name, listing price, specifications, and pick-up location.

Chats Screen:
On this screen, the user can view a list of all active ongoing communications with other users. Each chat item includes information about the last message time and the number of unread messages. Clicking on an item will lead to the direct chat screen.

Notifications Screen:
This screen would display any push notifications and provide an option for the user to mark all notifications as read.

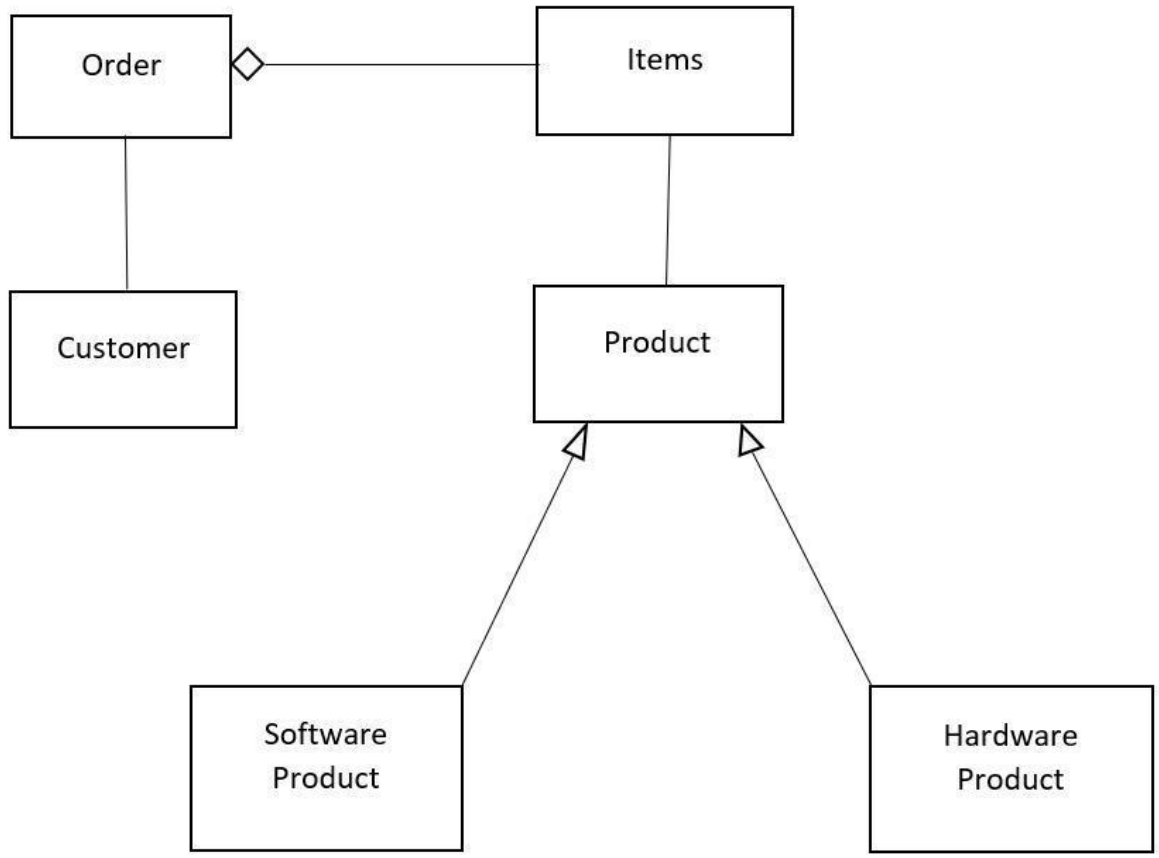
Subsystem Design

In software project management, a subsystem design refers to the process of breaking down a large software system into smaller, more manageable components, or subsystems.

A subsystem is a part of the larger software system that performs a specific function or set of functions. Each subsystem typically has its own set of inputs, outputs, and interfaces with other subsystems.

Subsystem design involves analyzing the requirements of the larger system and identifying the necessary subsystems. The subsystems are then designed, developed, and tested independently, and later integrated into the larger system. The purpose of subsystem design is to improve the modularity, maintainability, and scalability of the software system. It also allows multiple teams to work on different subsystems concurrently, which can speed up the overall development process.

In this project, the software can be divided into the following mentioned subsystems. Once identified, these subsystems will be designed, developed and tested independently.



Requirements

In software project management, requirements refer to the features, functions, and capabilities that a software system must have in order to meet the needs of its users or stakeholders. Requirements can be classified into various types -

Functional requirements:

- **FR1:** Flexible payment gateways like credit/debit cards, UPI payments with financial technological companies like Paypal, Zelle, Venmo etc.
- **FR2:** Should work on different devices like mobiles, tablets, PCs because most users shall try to use it from the portable devices.
- **FR3:** We can also monitor the number of users that interact from different devices to get a rough idea about which device is the most popular among users so that we can work on that more.
- **FR4:** Product features that should be included along with the functioning and uses should be filterable at the start of the site (home page). We can filter out the products based on what they are used for.
- **FR5:** Before buying anything the user should have a login id associated with their account that will be used to track the orders and if they don't have one we should have the feature to make an id.
- **FR6:** We should have a smooth checkout and order tracking flow so that it's easier to buy the things the user wants without any hassle.

Non-Functional requirements:

- **NFR1:** It should include all the social media handles of our e-commerce platform so that the users can check out important updates and announcements that might interest them.
- **NFR2:** It should protect user data like their mobile number, address, payment details and personal information.
- **NFR3:** We should try to load up the data quickly even with all the traffic and site congestion. We can also use a benchmark to track the speed of loading up all the functionalities and integrations of the application.

- **NFR4:** The application should be scalable after the first iteration. The performance shouldn't be affected as we increase the number of functionalities, integrations, features, memory, servers and computations.

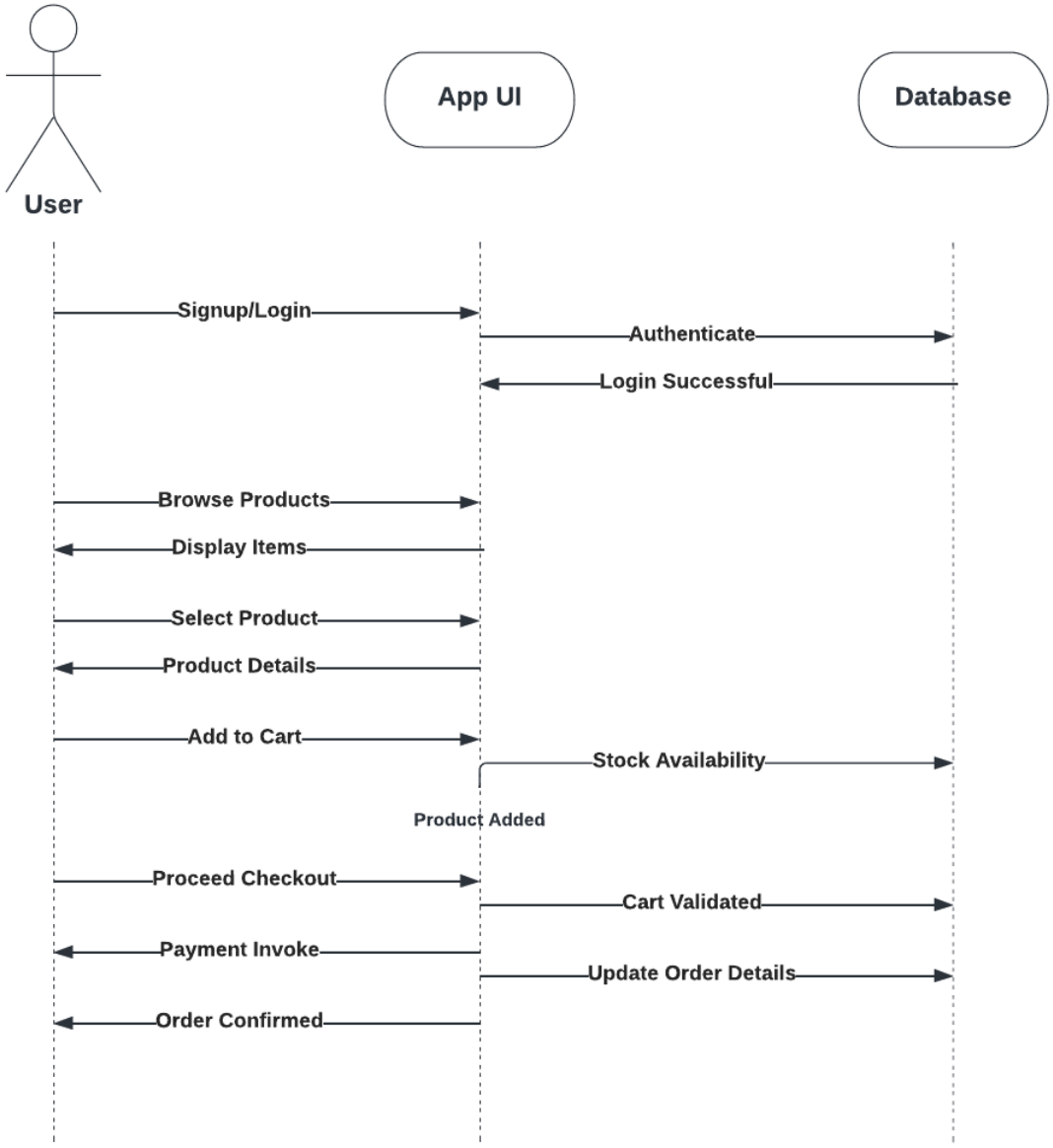
Interface requirements:

- **IR1:** The application should be easy to use for any user whether they have any knowledge about using an application or not (technical knowledge).
- **IR2:** We can also put up a survey after they've been at the application for a while for their feedback on the usability, how easy it was to find all the things they were looking for, did they complete what they initially came to the application for, where to go when you don't know what are you looking for.
- **IR3:** The seller should ideally select at least three images for the product he intends to sell to show its physical condition.
- **IR4:** We can additionally ask the seller for a demo video of the product whether it works, is there any physical or internal damage, how good it's condition is.

Sequence Diagram

A sequence diagram is a type of UML (Unified Modeling Language) diagram that illustrates how objects in a system interact with each other over time. It is a behavioral diagram that shows the flow of messages or method calls between objects in a specific scenario or use case.

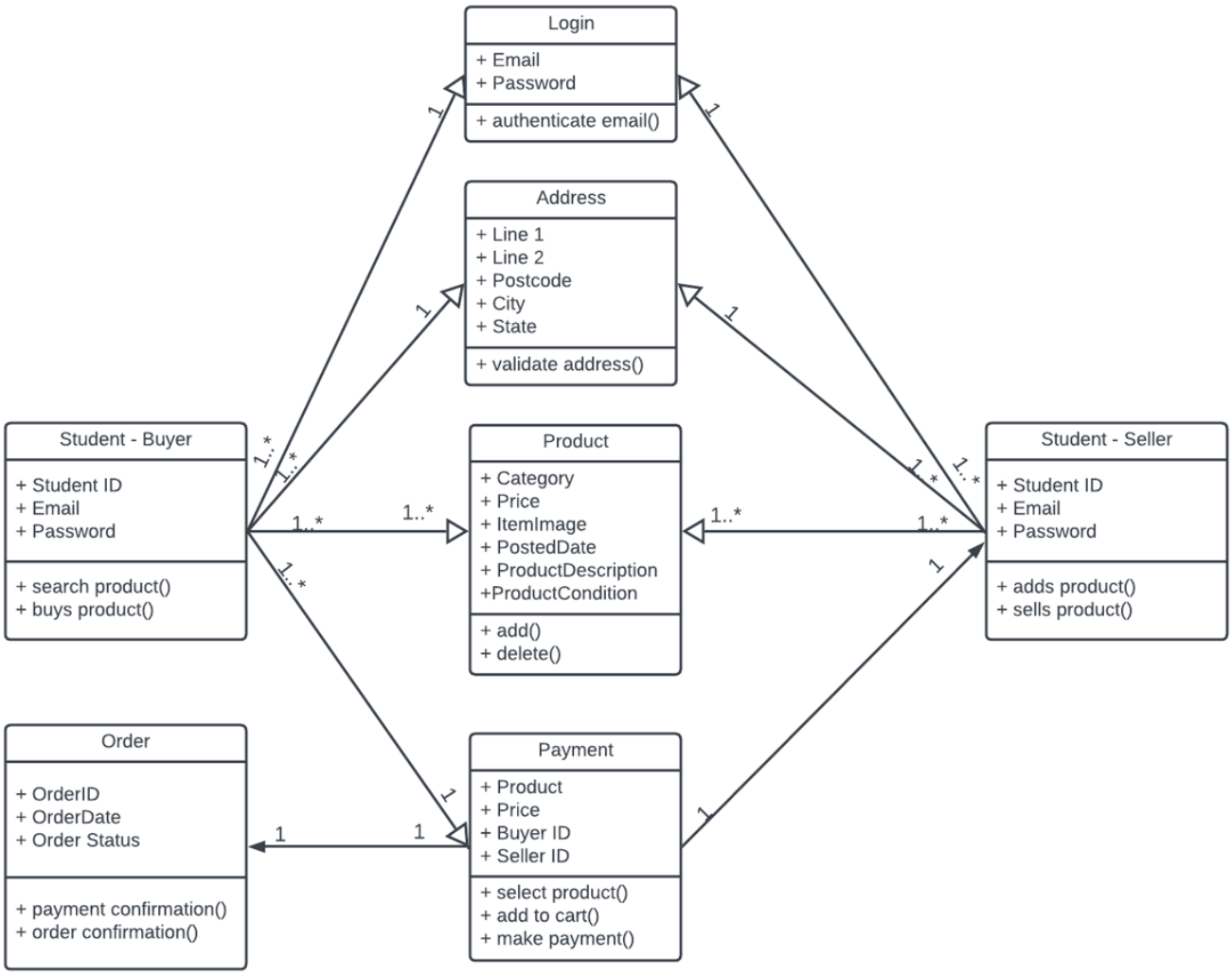
For this project, the sequence diagram covers the interaction between different entities involved in the software. The detailed steps are mentioned in the below diagram -



Design Class Diagram

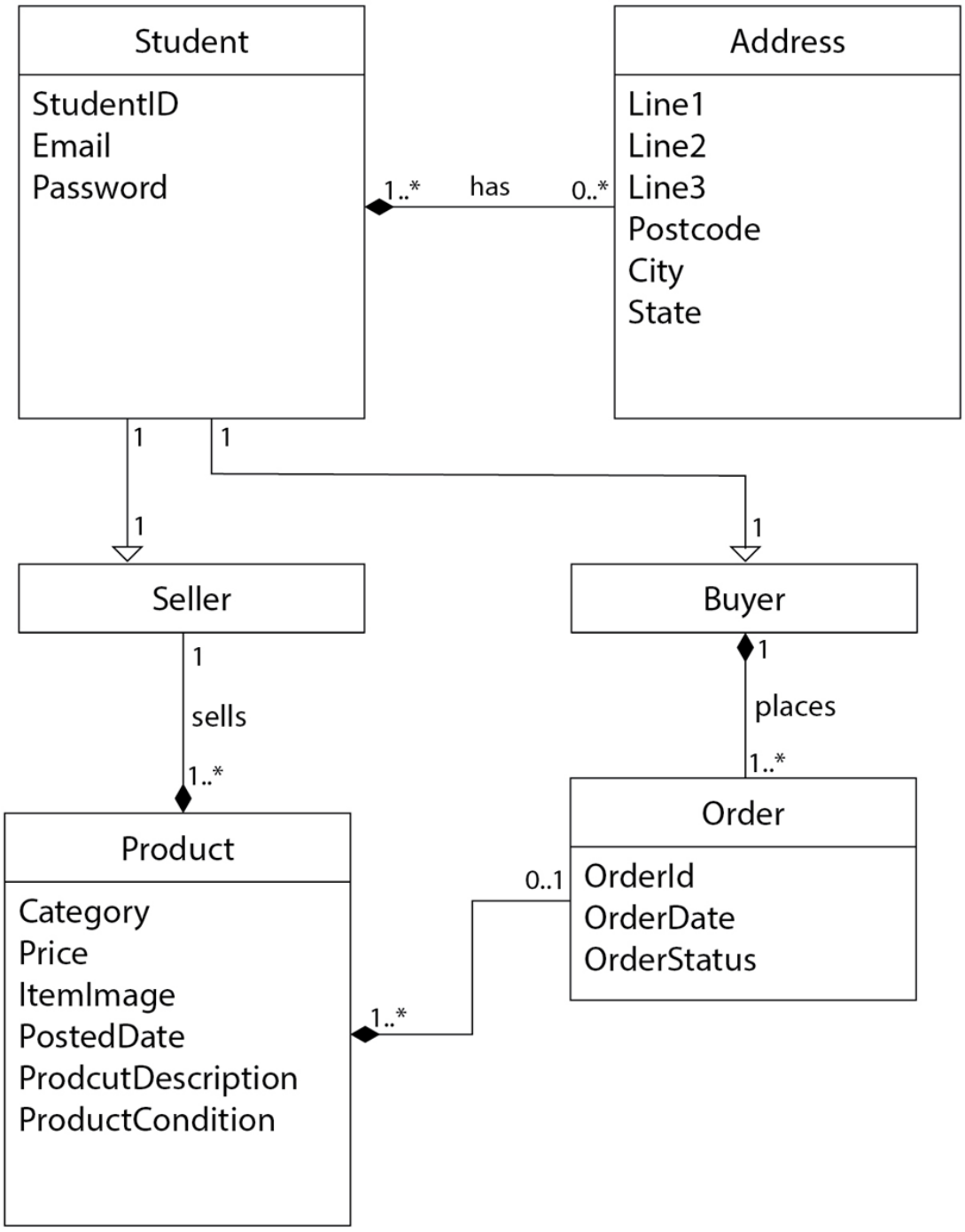
A class diagram is a type of diagram in software project management that depicts the structure of a system by showing the system's classes, their attributes, methods, and the relationships between them. It is a visual representation of the code and helps developers and stakeholders understand the design and architecture of the system. Class diagrams are a key part of object-oriented design and are often used in the early stages of software development to plan and communicate the system's structure and behavior.

In this project, the class diagram and the software elements are designed as following -



Domain Model

A domain model contains conceptual classes, associations between conceptual classes, and attributes of a conceptual class.
For this project, the associations and flow between various entities is specified in the domain model diagram below -

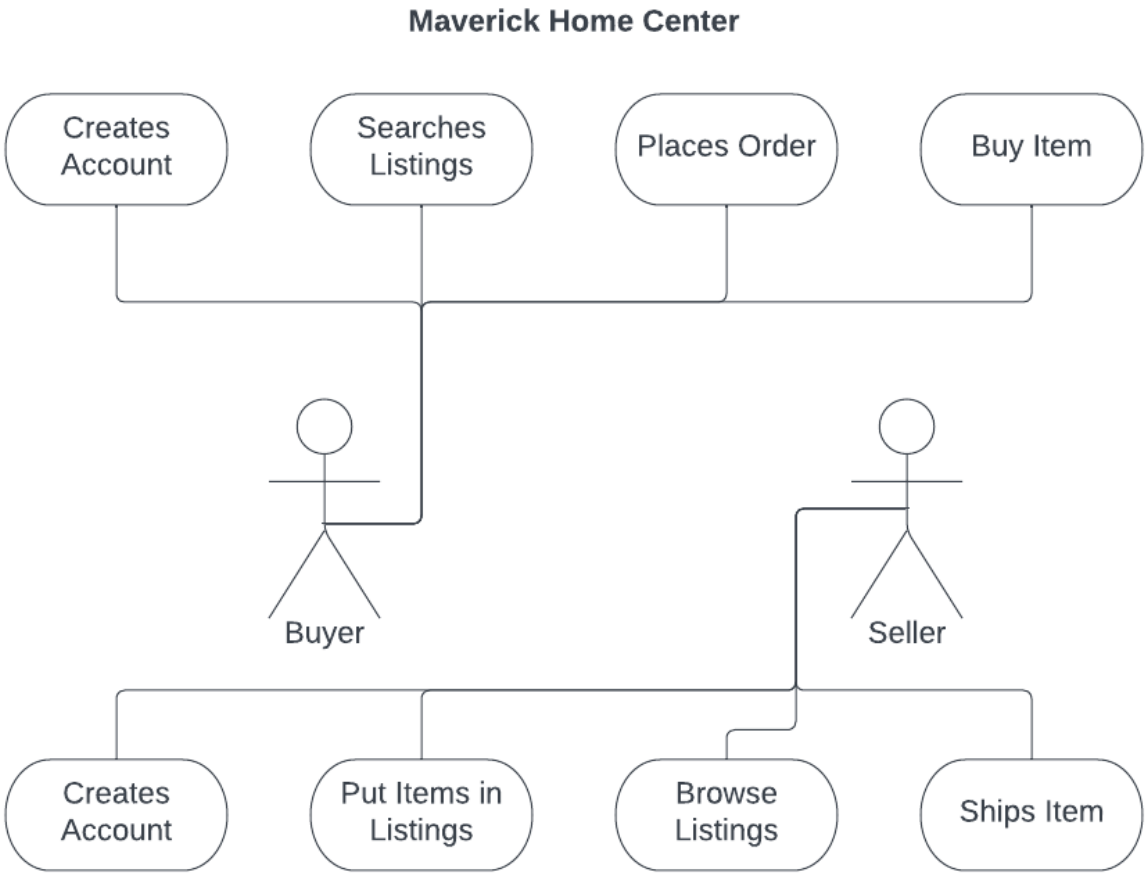


Actor System Interaction Model

An actor system interaction model is a representation of the interactions between actors and the system under design. It is used to capture the different ways in which actors can interact with the system and the behavior of the system in response to these interactions.

This model helps to identify the functional requirements of the system, refine the system's behavior, and ensure that the system meets the needs of its users. It is often created during the requirements analysis and design phases of the software development life cycle.

For this project, the actor system interaction can be designed as following -



Use Cases

In software engineering and system design, use cases are a strategy for identifying, analyzing, and documenting the interactions between users (or actors) and a system that are necessary to complete a specific job or goal. They often take the shape of a flowchart or a diagram and define the processes or actions that a user takes to accomplish a certain goal.

Use cases assist in defining a system's functional requirements, spotting any issues or inconsistencies in the design, and guaranteeing that the system is user-friendly. They are a crucial tool for software designers, developers, and other stakeholders to comprehend a system's requirements and successfully convey those requirements to others.

The use cases we encountered in the development of this application are:

- **UC1: Register**
- **UC2: View Items**
- **UC3: Make Purchase**
- **UC4: Check Out**

UC1: Register

Actor	System
2. User clicks on the “Register” button on the home page.	1. Displays homepage.
4. User enters their details such as name, email, and password.	3. Displays the registration form.
	5. Validates the entered information.
	6. Creates a new user account for the user.
	7. Displays the confirmation page with a message to verify the email address.
9. TUCEV user receives the verification email.	8. Send a verification email to the user's email address.
10. User clicks on the verification link in the email.	11. Confirms the email verification and displays the login page.
12. Users log in with their newly created account.	13. Displays the user dashboard.
14. Users can update their profile information from the dashboard.	15. TUCEV users can access all the features of the application with their account.

UC2: View Items

Actor	System
2. User navigates to the "Items" section on the homepage.	1. Displays homepage.
4. Users can filter and sort the items based on various criteria.	3. Displays the list of available items.
5. User clicks on an item to view more details.	6. Displays the item details page.
7. Users can add the item to their cart or Wishlist.	
8. Users can continue browsing other items or proceed to checkout.	

UC3: Make Purchase

Actor	System
2. User navigates to the "Items" section on the homepage.	1. Displays homepage.
4. Users can filter and sort the items based on various criteria.	3. Displays the list of available items.
5. User clicks on an item to view more details.	6. Displays the item details page.
7. Users can add the item to their cart or Wishlist.	8. Displays the cart page with the item added.
9. Users can update or remove items from the cart.	
10. User proceeds to checkout.	11. Displays the checkout page.
12. User enters shipping and payment information.	13. Confirms the order and displays the order summary page.
14. Users can view their order history and track the shipment.	15. TUCEW: User receives the order confirmation email.

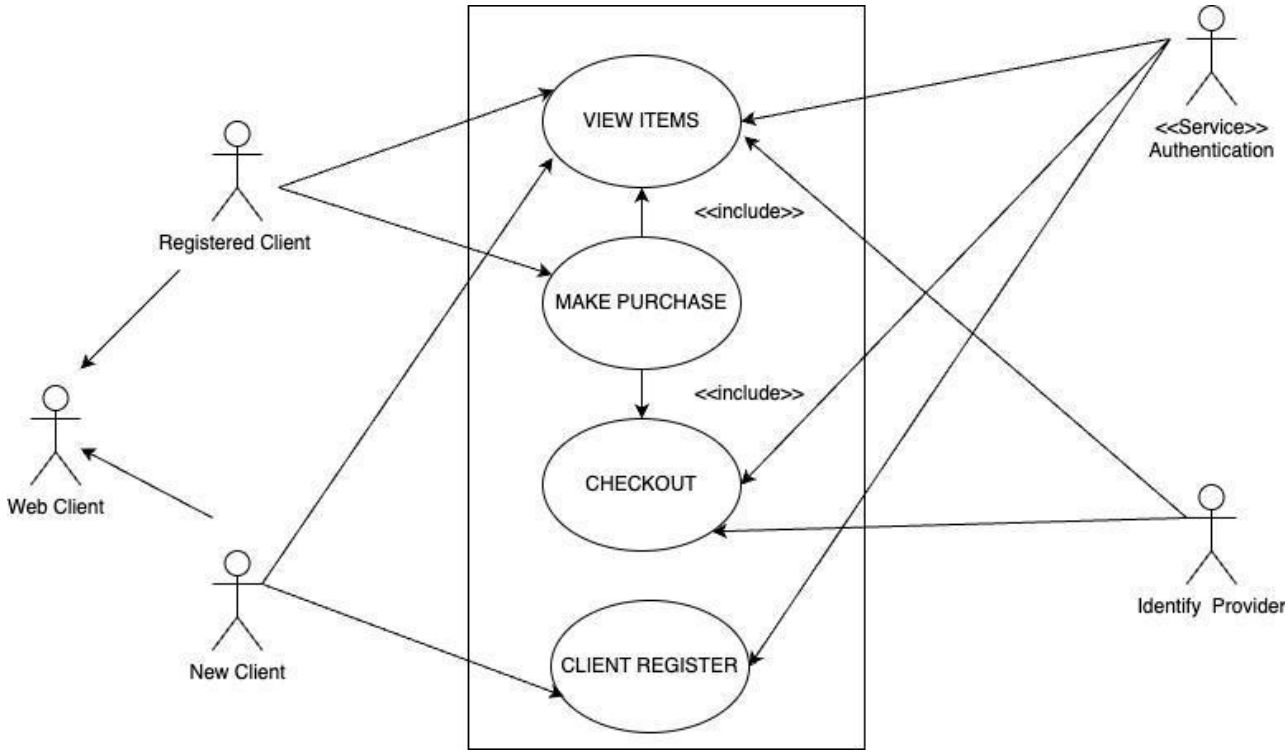
UC4: Check Out

Actor	System
1. Users add items to their cart.	2. displays the cart page with the item(s) added.
3. Users can update or remove items from the cart.	
4. User proceeds to checkout.	5. Displays the checkout page.
6. User enters shipping and payment information.	7. Validates the entered information.
	8. displays the order summary page with the shipping and payment details.
9. User confirms the order.	10. Processes the payment and confirms the order.
	11. Send a confirmation email to the user.
	12. TUCEV user receives the confirmation email.

High level Use Cases

High-level use cases typically consist of a brief description of the main interactions between the actors and the system, along with a list of the main functions or features provided by the system. They are often used as a starting point for more detailed use case development and help to ensure that the system meets the needs of its users. High-level use cases are also useful for communicating the system's functionality to stakeholders and for identifying areas of the system that require further analysis and development.

For this project of maverick home center, the high level use cases can be defined as -



HUC1: User Registration and Authentication.

Scenario	· User registration and authentication.
Triggering event	· A user wants to access the features of the application that require authentication.

Actors	· Guest: That is not yet registered in the system. · User: A registered user who wants to log in to the system.
Related use cases	· Register account. · Login.
Stakeholders	· Users: Who want to use the application features. · System administrators: who need to manage user accounts and ensure security.
Pre-condition	· The system is functional and running. · The user has access to the internet and a compatible device.
Post-condition	· The user is registered in the system and can access the features that require authentication.
Flow of events	· The user navigates to the login page of the application. · The application displays the login form. · The user enters their username and password and submits the form. · The system verifies the username and password. · If the username and password match a registered user, the system authenticates the user and grants them access to the features that require authentication. · If the username and password do not match a registered user, the system displays an error message and prompts the user to try again or register a new account. · If the user is a guest, they can register a new account by clicking the "Register" button and providing their information. · The system verifies the information provided by the user and creates a new account if everything is valid.

	· The system authenticates the new user and grants them access to the features that require authentication.
Exception	· If the system is down, the user cannot register or log in to the application. · If the user forgets their password, they can request a password reset via email.
TUCBW & TUCEW	· TUCBW: The user can begin the registration or login process by clicking on the appropriate button on the mobile app. · TUCEW: The user can end the registration or login process by being logged in to the system.

HUC2: Item Listing and Search.

Scenario	Item Listing and Search.
Triggering event	· A user wants to sell an item or find an item to buy.
Actors	· Seller: a user who wants to sell an item. · Buyer: a user who wants to buy an item. · Administrator: a user who has administrative privileges and can manage item listings.
Related use cases	· Add Listing · Edit Listing · Delete Listing · Search Item · Filter Item · View Item · Contact Seller

Stakeholders	· Users: who want to buy or sell items. · System administrators: who need to manage item listings and ensure quality control.
Pre-condition	· The system is functional and running. · The user has access to the internet and a compatible device. · The user is logged in and authenticated.
Post-condition	· The user can successfully list an item for sale or find an item to buy.
Flow of events	· A seller logs in to the system and navigates to the "Add Listing" page. · The seller enters the details of the item they want to sell, such as title, description, price, and images. · The system verifies the information provided by the seller and creates a new item listing if everything is valid. · The seller can edit or delete their listings at any time. · A buyer logs in to the system and navigates to the "Search Item" page. · The buyer enters keywords or filters to narrow down the search results. · The system displays the search results, along with the item details, such as title, description, price, and images. · The buyer can view the details of a specific item and contact the seller if they are interested in purchasing it. · The seller receives a notification of the buyer's message and can reply to it. · The buyer can rate the seller after the transaction is completed.

Exception	· If the system is down, the user cannot list an item or search for items. · If the seller provides invalid or inaccurate information, the system may reject the listing or remove it later. · If the buyer has an issue with the item or the seller, they can report it to the system administrator for resolution.
TUCBW and TUCEW	· TUCBW: The user can begin listing or searching for items by clicking on the appropriate button on the website or mobile app. · TUCEW: The user can end the item listing or search process by completing the transaction or exiting the platform.

HUC3: Make Purchases

Scenario	Make Purchases.
Triggering event	· A buyer wants to purchase an item listed by a seller.
Actors	· Buyer: a user who wants to purchase an item. · Seller: a user who listed the item for sale. · Administrator: a user who has administrative privileges and can resolve any disputes.

Related use cases	· Add to Cart · Checkout · Confirm Purchase · Cancel Purchase
Stakeholders	· Users: who want to buy or sell items. · System administrators: who need to manage transactions and ensure quality control.
Pre-condition	· The system is functional and running. · The user has access to the internet and a compatible device. · The user is logged in and authenticated.
Post-condition	· The buyer successfully purchases the item, and the seller receives payment.
Flow of events	· The buyer finds an item they want to purchase and clicks the "Add to Cart" button. · The system adds the item to the buyer's cart and displays the cart contents. · The buyer clicks the "Checkout" button and enters their shipping and payment information. · The system verifies the information and processes the payment. · The system sends a notification to the seller that the item has been purchased. · The seller ships the item to the buyer. · The buyer receives the item and confirms that it matches the description and is in good condition.

	· The system releases the payment to the seller. · The buyer can rate the seller after the transaction is completed.
Exception	· If the system is down, the user cannot make a purchase. · If the buyer or seller provides invalid or inaccurate information, the system may reject the transaction or delay the payment. · If there is an issue with the item or the transaction, the buyer can cancel the purchase and request a refund. The system administrator may be involved in resolving any disputes.
TUCBW and TUCEW	· TUCBW: The user can begin the purchase process by clicking on the "Checkout" button on the website or mobile app. · TUCEW: The user can end the purchase process by completing the payment and receiving confirmation of the purchase.

HUC4: Customer Service

Scenario	Customer Service.
Triggering event	· A user needs help with an issue related to the platform or a transaction.

Actors	· User: a buyer or seller who needs assistance. · Customer service representative: a trained professional who can assist with user issues.
Related use cases	· Submit a support ticket. · View support ticket. · Close support ticket.
Stakeholders	· Users: who may need assistance with using the platform or resolving issues related to transactions. · Customer service representatives: who are responsible for providing timely and effective support to users.
Pre-condition	· The system is functional and running. · The user has access to the internet and a compatible device. · The user is logged in and authenticated.
Post-condition	· The user's issue is resolved or escalated to the appropriate team or individual.

Flow of events	· The user encounters an issue and clicks the "Submit Support Ticket" button. · The system opens a form for the user to provide details about the issue, such as a description of the problem, relevant transaction or item details, and any supporting documentation or screenshots. · The user submits the form, and the system creates a support ticket with a unique identifier. · A customer service representative receives the ticket and reviews the details. · The customer service representative works with the user to resolve the issue or escalate it to a higher level of support if necessary. · The representative updates the ticket with any relevant information or actions taken. · The user can view the status of their support ticket at any time and provide additional information or feedback. · Once the issue is resolved, the representative closes the ticket.
Exception	· If the system is down, the user cannot submit a support ticket. · If the user provides incomplete or inaccurate information, the representative may need to ask for additional details, which can delay the resolution of the issue.
TUCBW and TUCEW	· TUCBW: The user can begin a customer service request by clicking on the "Contact us" button on the website or mobile app. · TUCEW: The user can end the customer service process by closing the support ticket or chat.

Expanded Use Cases

Expanded use cases typically consist of a series of steps or actions that the actor performs, along with the system's responses and any alternate paths that may occur. They may also include preconditions, postconditions, and other details that help to refine the use case and ensure that it is complete and accurate.

Expanded use cases are often created after high-level use cases and are used to further refine the system's functionality and behavior. They are also useful for identifying areas of the system that may require additional development or testing. Expanded use cases can be used to develop test cases, design the system's user interface, and to communicate the system's functionality to stakeholders.

For this project, the expanded use cases cover the detailed aspects for each use case. They are as mentioned below -

- **EUC1: Sign UP**
- **EUC2: Log IN**
- **EUC3: Browse Items**
- **EUC4: View Item**
- **EUC5: Make Purchase**
- **EUC6: Checkout**

EUC1: Sign UP

Actor - Student New User	System - MHC
1. The use case begins with the user launching the MHC app.	2. The system displays the Login page.
3. The user clicks on the Sign up link.	4. The system launches thye sign up page.
5. The user enters all the details in the Sign Up form and clicks on the Sign up page.	6. The system validates the data and adds the user in the database.

EUC2: Log IN

Actor - Student User	System - MHC
1. The use case begins with the user launching the MHC app.	2. The system displays the Login page.
3. The user then enters their UTA email address and password.	4. The system then verifies whether the entered email address is in the correct email format.

5. The user then clicks on LOGIN button	6. The system then authenticates and if the user has an account then it redirects to the home screen.

EUC3: Browse Items

Actor - Student User	System - MHC
1. The use case starts with the user logging in.	2. The system displays the Home screen with all the products listed.
3. The user scrolls through the categories by horizontally scrolling through the categories menu.	4. The system displays all the categories available.
5. The user then clicks on a particular category.	6. The system displays all the products under that category.
7. The user then clicks on the home button available at the bottom of the screen.	8. The system navigates the user back to the Home screen.
9. The user then scrolls down the home screen.	10. The system displays all the products available in a listed manner.

EUC4: View Item

Actor - Student User	System - MHC
1. The use case starts with the user being in the Home page of the app and clicks on a product thumbnail.	2. The system launches the product page where all the product details are listed.
3. Then the user clicks on the Home button available at the bottom of the page.	4. The system navigates the user to the Home page.
5. The user then clicks on the search bar	6. The system displays the keyboard for the user to type in the product they are looking for.
7. The user types in the specific product they are looking for.	8. The system then displays the thumbnail of the product that the user has searched for.

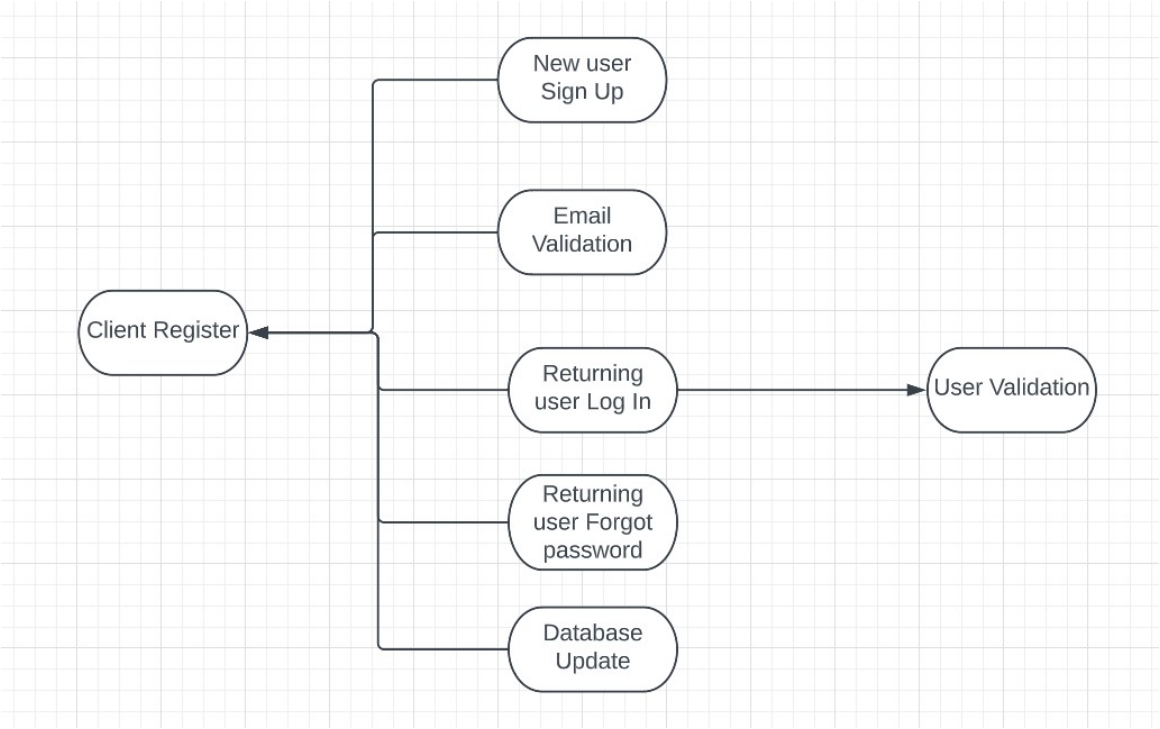
EUC5: Make Purchase

Actor - Student User	System - MHC
1. The use case starts with the user being in the home page and then clicks on a product thumbnail.	2. The system launches the product page where all the product details are listed.
3. The user then verifies all the product details and clicks on the Add to cart button.	4. The system creates a new cart for the user and adds the product to the cart.
5. The user clicks on the cart icon at the top of the screen.	6. The system launches the cart page and displays the product added to cart and the Total cart amount.

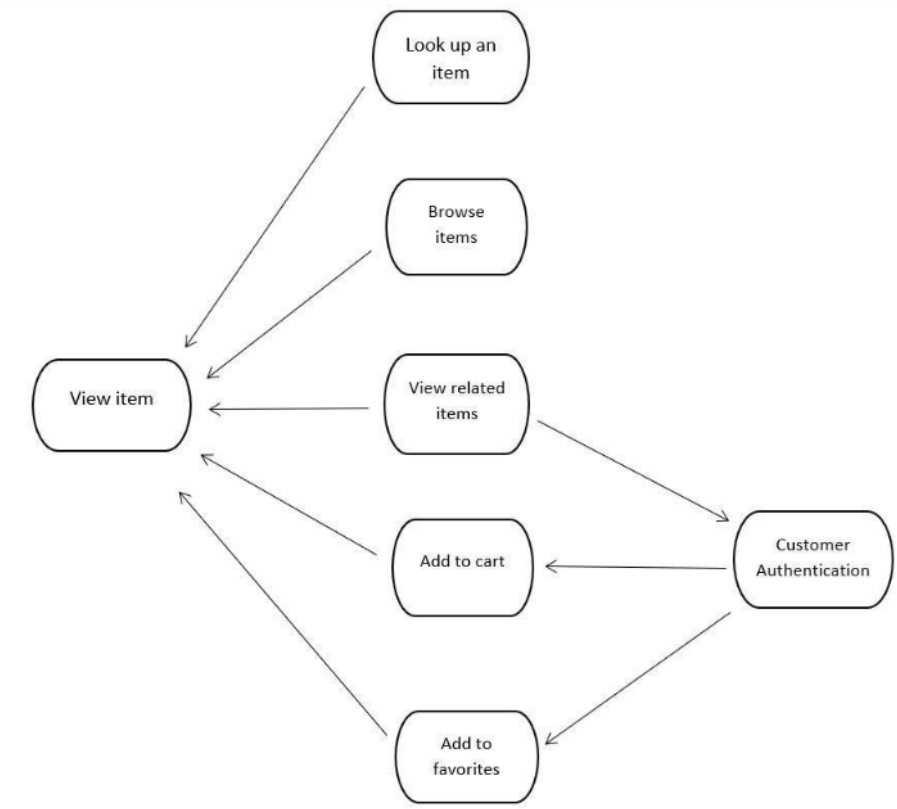
EUC6: Checkout

Actor - Student User	System - MHC
1. The use case starts with user being in the cart page and clicks on checkout	2. The system checks with the database and updates the availability of the product. It also prompts the user to enter payment details.
3. The user then enters their card details and confirms checkout.	4. The system authenticates and authorizes the payment. It also makes a reservation of the product and updates the stock in db. Additionally, it confirms the user of order placement.

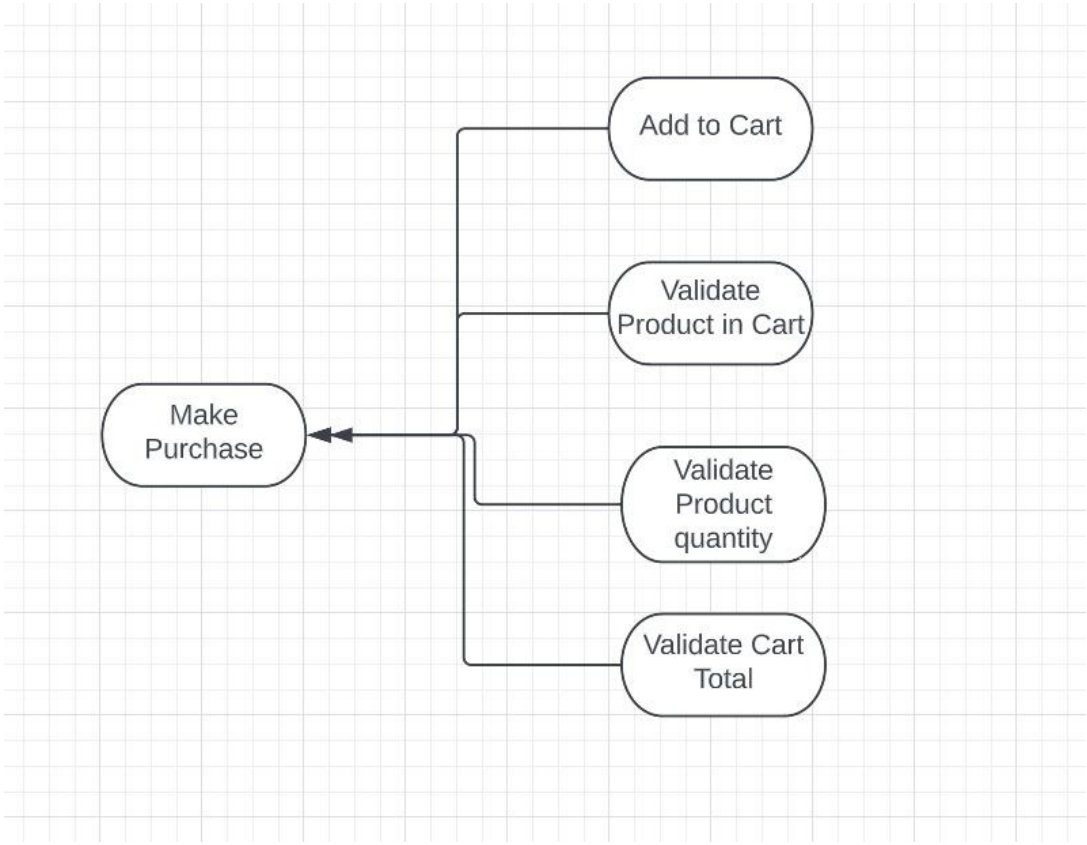
Sign Up and Login -



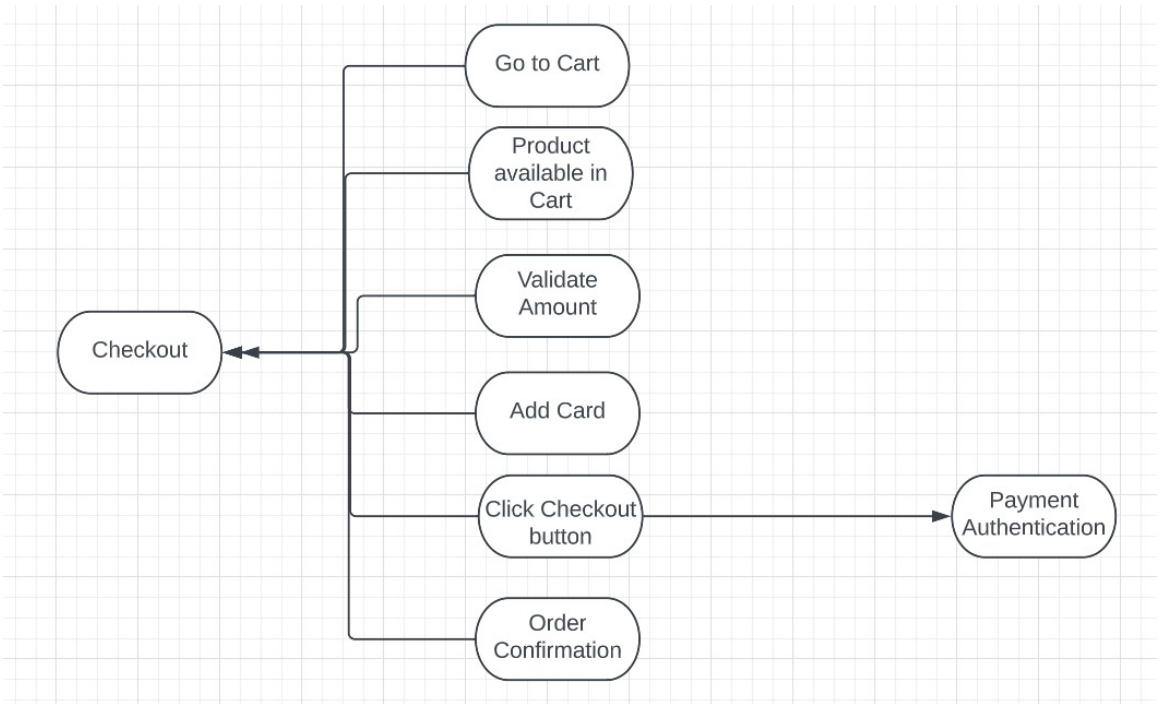
Browse and View Items -



Make Purchase -



Checkout -



Requirements Use Case Traceability Matrix

A Requirements Use Case Traceability Matrix is a tool used in software project management to ensure that all requirements have a corresponding use case, and all use cases have been properly traced to their corresponding requirements. It is a matrix that shows the relationship between the requirements and use cases, and helps to ensure that the system meets the needs of its users by tracing each requirement to the use cases that satisfy it. The matrix can also be used to track changes to the requirements and use cases throughout the software development life cycle.

The matrix also contains the priority level of each requirement, which helps in planning the project properly with a defined timeline. Following is the RUCTM for this project -

Requirement No.	Priority Weight	UC1	UC2	UC3	UC4
FR1	5		x	x	
FR2	5	x			
FR3	5		x		x
FR4	5				
FR5	4				
FR6	4	x			
NFR1	4				
NFR2	3				
NFR3	3			x	x
NFR4	3				
IR1	2				
IR2	2				
IR3	1				
IR4	1				
UC Priority		9	10	8	8

Note: Priority weight: 1 denotes low, 5 denotes highest priority

Increment Matrix

An increment plan is a tool used in Agile software development to outline the work that will be completed in a particular iteration or increment of development. It typically includes a list of user stories or requirements that will be implemented in the upcoming iteration or increment, along with estimates of the time and resources required to complete each task. The iteration plan or increment plan is used to help the development team plan and execute their work, as well as to communicate progress and status to stakeholders.

The iteration plan or increment plan is often updated and revised throughout the software development process as new information is gathered and requirements change. It is a key component of Agile project management and helps to ensure that the software is developed in an iterative and incremental manner, with frequent feedback and adaptation to changing requirements.

The below mentioned increment matrix diagram shows the priority and effort estimated based on the team strength for multiple iterations with respect to this project.

Use Case	Priority	Effort (Person-week)	Depends On	ITR 1 3wks	ITR 2 3wks	ITR 3 3wks	ITR 4 3wks
UC1	9	3	None		12		
UC2	10	3	None			12	
UC3	8	3	None				12
UC4	8	3	None	12			
Total Effort		12		12	12	12	12

Team Size = 4 People

Effort Estimation

First of all we need to know what factors count towards the effort estimation of a software engineering life cycle of an application.

Effort estimation refers to the process of estimating, planning, and monitoring the time and resources required to complete a project. In the context of mobile app development, effort management is critical to ensure that the project is delivered on time, within budget, and with the desired quality.

In order to estimate the cost of the application we are developing, we need to consider several factors that affect the overall performance of the app and the software as a whole.
These can be classified into the following categories:

1. **Platform**: The platform that we have chosen to develop this application if Flutter as it is easier to develop a cross platform application using the same otherwise we would have to build two separate native apps for iOS and Android, which would increase the development time and cost of the overall project and hence would require more efforts to manage, maintain and function.

2. **Features**: The features that we have added are similar to the functional, non-functional and user interface requirements that we have mentioned above. Account creation, browsing and filtering items according to their types (furniture, decor, electronics, etc), payment gateways, data safety, Buyer can search for all the items that are up for sale/ for giveaway, seller can choose whether they want to deliver the items or ask the buyer to pick them up from a certain place. Since we have a number of features to make the user interface and the overall application user friendly, we would require significant development from our team in order to meet these feature requirements.

3. **Overall Design**: The user interface design and overall “look” of the application would also be an essential factor in determining the efforts that it would need in development. This is done keeping in mind that the application home page and the overall flow should be user friendly, that is, it should be easier to use even for people with little to no technical knowledge of the field. If we are delegating the design to a team member, we do that while being mindful of the efforts that it would require, so it would be done using a well defined series of tasks that a

typical user will go through. This would help us target the right areas where the user might get stuck or want to start a new purchase.

4. **Integration**: First and foremost step would be to integrate our app with third-party services like payment gateways (Paypal, Venmo, Zelle, etc) , shipping providers, and inventory management systems. The effort required to integrate these services can vary depending on the complexity of the integration and the availability of APIs. Since we are using the flutter libraries and APIs, this should be easier than your typical android or iOS development integration process.

5. **APIs**: We also plan to integrate with third-party APIs (such as payment gateways or shipping providers), so in order to do that we have to consider the effort required to integrate and test these APIs. This can sometimes be a complex and time-consuming process.

6. **Content management**: We also need to consider how we will manage the content that appears in the app, such as item descriptions and images. This may involve developing a content management system (CMS) or integrating with an existing CMS. This could take a while because we plan on adding additional details and images for every item that we will add to the system including but not limited to the original packaging (if available), item description from where it was originally bought (example: Amazon, eBay, etc) and the most recent pictures of the item

7. **Localization**: We plan to release the app in a singular region (Arlington) since it is for the students and you need a valid UTA ID to login to the application so not much effort would be required to localize the application as it would already be in the ideal state for our purpose. Additional and possibly a bonus feature that we could add in this would be to offer the item descriptions in a language other than English (example: Spanish, Hindi, Arabic) since we have a lot of diversity here in UTA.

8. **Scalability**: If the app becomes popular and attracts a large user base, we'll need to consider the effort required to scale the app to handle the increased traffic and demand. This may involve upgrading the server infrastructure or implementing additional features to support scalability.

9. **Analytics**: We also might integrate analytics tools into the app to track user behavior, measure engagement, and identify areas for improvement. This will require some additional effort to set up and configure the analytics tools.

Using the Work Breakdown Structure (WBS) technique, we have estimated the time required for each task, as shown in the table below:

Task	Estimated Time (hours)
Requirement gathering	6
UI/UX design	15
Development of the app for iOS	20
Development of the app for Android	30
Payment Gateway integration	10
Testing and Quality Assurance	60
Deployment and Maintenance	50

Based on these factors, the effort required to build an app that sells refurbished items to local students could range from a **few weeks** to **several months**. It's important to have a clear project plan and timeline to help manage the effort estimation process. Additionally, we have to consider ongoing maintenance and support costs for the app once it's launched.

People Management

We divided the work between the four of us in the following way:

1. We communicated between the team and made the initial requirements, defined the goals, the overall design and the integration part of all the functionalities with each other and made sure to focus on the main aspects of the software project management and maintenance part of the application.

2. We also assigned and defined the roles and responsibilities among ourselves to clear up the uncertainties.

3. We built a diverse and complementary team in such a way that we put the right tasks to the right person who has experience in it and is good at the task in hand.

4. For the design, all of us worked on the development of the application, particularly Amit who worked on the design and the Flutter development part, worked extensively on the design and development.

5. For front end development, Amit made sure all the images and the functionalities were showing up in the right order and had a significant contribution in the overall look of the application.

6. For the back end deveopment, we worked in the firebase linkup, database design and its structure and fetched basically all the data that we required in the making of the application.

7. For integration testing, Ravi had a particularly essential contribution in making sure the right functionalities were showing up in our application and was mainly responsible for understanding the working and design that matched the vision and the expectations we had from our application.

8. As for the project management, we worked interchangeably among ourselves to make sure we were on the right track and had the right plan in mind from the start till the end. Akash and Aditya were responsible for the same and had excellent management and maintenance experience in the design and development of the application.

9. We also had standup meetings, review meetings and brainstorming sessions in person and online to ensure we were designing and developing the right way. We also implemented the right functionalities.

10. We also fostered a positive and healthy environment which had encouraging and supportive feedback in the overall development of the application in such a way so that it was a good and positive learning experience for all of us.

11. We also continuously monitored the progress and the processes we performed along the way so that we were redirected in the right path if we accidentally or unknowingly deter from the path.

Quality Management

Quality Management refers to the process of ensuring that the project meets the desired quality standards. In our mobile app development project quality management is very important or rather critical to make sure that the application built is reliable, usable and meets the customer expectations.

Quality Management Steps :

1. **Set Quality Parameters** - The very first step in Quality Management of any project is to clearly set the quality parameters. The quality parameters should be based on customer requirements, industry best practices and regulatory standards. We are using various techniques such as user surveys, customer feedback and competitive analysis to identify the quality parameters.

2. **Develop Test Plan** - Once the quality parameters are set we will be developing an extensive test plan. The test plan will include various types of test such as Unit Testing or whitebox testing, Functional Testing or Spring Testing, Usability Testing, Compatibility Testing, Performance Testing, Security Testing and User Acceptance Testing. The test plan will specify the test criteria, test data and test environment.

3. **Execute Test** - Once the Test Plan is developed the next step will be to execute the tests as per the plan. We will be using various testing tools such as automated testing tools, manual testing tools, load testing tools and performance testing tools to run the tests. The tests will be recorded and any issue or bug found will be tracked and fixed.

4. **Monitor and Improve** - The final step will be to monitor the quality of the app and continuously improve it. We will be vigilant on the app's functionality, usability and user reviews. The quality of the app will be tracked using a variety of methods including user surveys, analytics and feedback tools. Any issues encountered will be fixed right away and the software should be updated regularly depending on user comments.

Best Practices -

Quality management is a vital component of developing mobile apps. Following are some best practices that we are going to use to ensure the app's quality:

1. Use a Test-Driven Development (TDD) strategy: TDD entails developing tests prior to developing code. This strategy will aid in ensuring that the app satisfies the requirements for quality.

2. Use Automated Testing: Using automated testing will help to increase testing's efficacy and efficiency. To conduct tests efficiently and accurately, automated testing technologies will be employed.

3. Performing Usability Testing: Conducting Usability Testing will help us to understand the app's ease of use, learnability and user satisfaction. It will also ensure that the app satisfies the user's needs and expectations.

Testing and Quality Assurance Activities:

The following testing and quality assurance activities will be conducted:

1. Functional Testing - Functional testing will be conducted to ensure that the app functions as intended. The following tests will be performed:

- User registration
- User login
- Payment processing

2. Usability Testing - Usability testing will be conducted to evaluate the user experience of the app. The following tests will be performed:

- Navigation testing
- Design and layout testing
- User feedback and suggestions

3. Compatibility Testing - Compatibility testing will be conducted to ensure that the app works on different devices and operating systems. The following tests will be performed:

- Device compatibility testing
- OS compatibility testing

4. Performance Testing - Performance testing will be conducted to ensure that the app performs well under different usage scenarios. The following tests will be performed:

- Load testing
- Stress testing
- Performance benchmarking

5. Security Testing - Security testing will be conducted to ensure that the app is secure from potential threats. The following tests will be performed:

- Authentication and authorization testing
- Data protection testing
- Network security testing

6. User Acceptance Testing - User acceptance testing will be conducted to ensure that the app meets user requirements and expectations. The following tests will be performed:

- User testing with a group of representative users
- Feedback and suggestions from users

Test Cases

Project Name	*MAVERICK HOME CENTRE*						
Created by	Ravi Rajpurohit Akash Biswas Aditya Bhat Amit Gupta						
Creation Date	April 3, 2023						
			SPRINT - 1				
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
USER INTERFACE TESTING							
UI01	Verify the login screen has rendered correctly with all items appearing as expected on the User Interface (UI)	1.Enter valid username 2.Enter valid password 3.Click on login button	Need valid username and password for login and Valid URL	Username: Password:	You should able to see the Home Centre application's homepage	Successful Login	Pass
UI01	Verify the login screen has rendered correctly with all items appearing as expected on the User Interface (UI)	1.Enter valid username 2.Enter valid password 3.Click on login button	Need valid username and password for login and Valid URL	Username: Password:	You should able to see the Home Centre application's homepage	Successful Login	Pass
UI01	Verify the login screen has rendered	1.Enter valid username	Need valid username and	Username: Password:	You should able to see the Home Centre	Successful Login	Pass

	correctly with all items appearing as expected on the User Interface (UI)	2.Enter valid password 3.Click on login button	password for login and Valid URL		application's homepage		
UI01	Verify the login screen has rendered correctly with all items appearing as expected on the User Interface (UI)	1.Enter valid username 2.Enter valid password 3.Click on login button	Need valid username and password for login and Valid URL	Username: Password:	You should able to see the Home Centre application's homepage	Successful Login	Pass
UI01	Verify the login screen has rendered correctly with all items appearing as expected on the User Interface (UI)	1.Enter valid username 2.Enter valid password 3.Click on login button	Need valid username and password for login and Valid URL	Username: Password:	You should able to see the Home Centre application's homepage	Successful Login	Pass
UI02	Verify the login screen throws error if username or password do not match	1. Enter invalid username 2. Enter invalid password 3. Click on login button	Username and password should not have registered Or Username or password should not match	Username: Password:	You should see an error for logging in with incorrect credentials	Unsuccessful login	Pass
UI02	Verify the login screen throws error if username or password do not match	1. Enter invalid username 2. Enter invalid password 3. Click on login button	Username and password should not have registered Or Username or password	Username: Password:	You should see an error for logging in with incorrect credentials	Unsuccessful login	Pass

			should not match				
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
UI02	Verify the login screen throws error if username or password do not match	1. Enter invalid username 2. Enter invalid password 3. Click on login button	Username and password should not have registered Or Username or password should not match	Username: Password:	You should see an error for logging in with incorrect credentials	Unsuccessful login	Pass
UI02	Verify the login screen throws error if username or password do not match	1. Enter invalid username 2. Enter invalid password 3. Click on login button	Username and password should not have registered Or Username or password should not match	Username: Password:	You should see an error for logging in with incorrect credentials	Unsuccessful login	Pass
UI02	Verify the login screen throws error if username or password do not match	1. Enter invalid username 2. Enter invalid password 3. Click on login button	Username and password should not have registered Or Username or password should not match	Username: Password:	You should see an error for logging in with incorrect credentials	Unsuccessful login	Pass
UI03	Verify if Account tab shows the user logged in	1. Go to the account tab 2. Check the user details	User should have logged in	Username: Password:	You should see the user details on the account tab	User can see the login details	Pass

Test cases-ID	Description	Test Steps	Pre-conditions	Test data	Post-condition	Expected Result	Test Pass/Fail
UI03	Verify if Account tab shows the user logged in	1. Go to the account tab 2. Check the user details	User should have logged in	Username: Password:	You should see the user details on the account tab	User can see the login details	Pass
UI03	Verify if Account tab shows the user logged in	1. Go to the account tab 2. Check the user details	User should have logged in	Username: Password:	You should see the user details on the account tab	User can see the login details	Pass
UI03	Verify if Account tab shows the user logged in	1. Go to the account tab 2. Check the user details	User should have logged in	Username: Password:	You should see the user details on the account tab	User can see the login details	Pass
UI03	Verify if Account tab shows the user logged in	1. Go to the account tab 2. Check the user details	User should have logged in	Username: Password:	You should see the user details on the account tab	User can see the login details	Pass
UI04	Verify if the homepage renders the images correctly	1. Login to the application 2. Get connected to internet	User should have logged in	Products should have been uploaded for display	You should see the products and images successfully	User can see the images rendered well	Fail
UI04	Verify if the homepage renders the images correctly	1. Login to the application	User should have logged in	Products should have been uploaded for display	You should see the products and images successfully	User can see the images rendered well	Pass

		2. Get connected to internet					
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
UI04	Verify if the homepage renders the images correctly	1. Login to the applicatio n 2. Get connected to internet	User should have logged in	Products should have been uploaded for display	You should see the products and images successfully	User can see the images rendered well	Pass
UI04	Verify if the homepage renders the images correctly	1. Login to the applicatio n 2. Get connected to internet	User should have logged in	Products should have been uploaded for display	You should see the products and images successfully	User can see the images rendered well	Pass
UI04	Verify if the homepage renders the images correctly	1. Login to the applicatio n 2. Get connected to internet	User should have logged in	Products should have been uploaded for display	You should see the products and images successfully	User can see the images rendered well	Pass
UI05	Verify if the user can see recent searches below the search bar	1. Search for an item in the search bar	User should have searched a few things first to create a search history	Product searches	You should see the recent searches as text below the search bar	User can see the recent searches	Pass
UI05	Verify if the user can see recent searches below the search bar	1. Search for an item in the search bar	User should have searched a few things first to create a search history	Product searches	You should see the recent searches as text below the search bar	User can see the recent searches	Pass

Test cases-ID	Description	Test Steps	Pre-conditions	Test data	Post-condition	Expected Result	Test Pass/Fail
UI05	Verify if the user can see recent searches below the search bar	1. Search for an item in the search bar	User should have searched a few things first to create a search history	Product searches	You should see the recent searches as text below the search bar	User can see the recent searches	Pass
UI05	Verify if the user can see recent searches below the search bar	1. Search for an item in the search bar	User should have searched a few things first to create a search history	Product searches	You should see the recent searches as text below the search bar	User can see the recent searches	Pass
UI05	Verify if the user can see recent searches below the search bar	1. Search for an item in the search bar	User should have searched a few things first to create a search history	Product searches	You should see the recent searches as text below the search bar	User can see the recent searches	Pass
FUNCTIONAL TESTING							
FT01	Verify if the user is able to enter their credentials	1. Try putting text in username and password fields	User should open the application	-	User should be able to type text as well as password	User can type text and password	Pass
FT01	Verify if the user is able to enter their credentials	1. Try putting text in username and password fields	User should open the application	-	User should be able to type text as well as password	User can type text and password	Pass

Test cases-ID	Description	Test Steps	Pre-conditions	Test data	Post-condition	Expected Result	Test Pass/Fail
FT01	Verify if the user is able to enter their credentials	1. Try putting text in username and password fields	User should open the application	-	User should be able to type text as well as password	User can type text and password	Pass
FT01	Verify if the user is able to enter their credentials	1. Try putting text in username and password fields	User should open the application	-	User should be able to type text as well as password	User can type text and password	Pass
FT02	Verify if the Login button works	1. Try logging in using any username and password	User should type in their credentials	-	The login button should be active	The login button works to login or throw error	Pass
FT02	Verify if the Login button works	1. Try logging in using any username and password	User should type in their credentials	-	The login button should be active	The login button works to login or throw error	Pass
FT02	Verify if the Login button works	1. Try logging in using any username and password	User should type in their credentials	-	The login button should be active	The login button works to login or throw error	Pass
FT02	Verify if the Login button works	1. Try logging in using any username and password	User should type in their credentials	-	The login button should be active	The login button works to login or throw error	Pass
FT02	Verify if the Login button works	1. Try logging in using any	User should type in their credentials	-	The login button should be active	The login button works to login or throw error	Pass

		username and password					
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
FT03	Verify registration section is not displayed when the user enters valid credentials for an existing account	1. Signup using valid credentials 2. Login using credentials	User should sign up	Username Password	The user should not see register if already signed up	The user sees login button	Fail
FT03	Verify registration section is not displayed when the user enters valid credentials for an existing account	1. Signup using valid credentials 2. Login using credentials	User should sign up	Username Password	The user should not see register if already signed up	The user sees login button	Pass
FT03	Verify registration section is not displayed when the user enters valid credentials for an existing account	1. Signup using valid credentials 2. Login using credentials	User should sign up	Username Password	The user should not see register if already signed up	The user sees login button	Pass
FT03	Verify registration section is not displayed when the user enters valid credentials for an existing account	1. Signup using valid credentials 2. Login using credentials	User should sign up	Username Password	The user should not see register if already signed up	The user sees login button	Pass
FT03	Verify registration	1. Signup using	User should sign up	Username Password	The user should not see	The user sees login button	Pass

	section is not displayed when the user enters valid credentials for an existing account	valid credentials 2. Login using credentials			register if already signed up		
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
DATABASE TESTING							
DB01	Check the table is available in database	available in database	Users should have signed up	Username Password	Data entry should be happening in DB	Database updates user data	Pass
DB01	Check the table is available in database	available in database	Users should have signed up	Username Password	Data entry should be happening in DB	Database updates user data	Pass
DB01	Check the table is available in database	available in database	Users should have signed up	Username Password	Data entry should be happening in DB	Database updates user data	Pass
DB01	Check the table is available in database	available in database	Users should have signed up	Username Password	Data entry should be happening in DB	Database updates user data	Pass
DB01	Check the table is available in database	available in database	Users should have signed up	Username Password	Data entry should be happening in DB	Database updates user data	Pass
DB02	Check the product database is updated properly	The database is updated when user buys/sells products	Products should be there to load/render	Product information - Description and images	Database should keep track of products	Database updates product details after transaction	Fail
DB02	Check the product database is updated properly	The database is updated when user buys/sells	Products should be there to load/render	Product information - Description and images	Database should keep track of products	Database updates product details after transaction	Pass

		products					
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
DB02	Check the product database is updated properly	The database is updated when user buys/sells products	Products should be there to load/render	Product information - Description and images	Database should keep track of products	Database updates product details after transaction	Pass
DB02	Check the product database is updated properly	The database is updated when user buys/sells products	Products should be there to load/render	Product information - Description and images	Database should keep track of products	Database updates product details after transaction	Pass
DB02	Check the product database is updated properly	The database is updated when user buys/sells products	Products should be there to load/render	Product information - Description and images	Database should keep track of products	Database updates product details after transaction	Pass
			SPRINT - 2				
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
User Interface Testing							
UI11	Verify that the splash screen appears with the application logo	Install and Open the applicatio n	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass
UI11	Verify that the splash screen appears with the application logo	Install and Open the applicatio n	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass
UI11	Verify that the splash screen appears with	Install and Open the	-	-	The application sign-in page should appear	The user sees the splash	Pass

	the application logo	applicatio n				screen with the logo	
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
UI11	Verify that the splash screen appears with the application logo	Install and Open the applicatio n	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass
UI12	Verify that the sign in button appears	Open the app and sign up	Sign up	-	-	Sign in button appeared	Pass
UI12	Verify that the sign in button appears	Open the app and sign up	Sign up	-	-	Sign in button appeared	Pass
UI12	Verify that the sign in button appears	Open the app and sign up	Sign up	-	-	Sign in button appeared	Pass
UI12	Verify that the sign in button appears	Open the app and sign up	Sign up	-	-	Sign in button appeared	Pass
UI13	The password should be hidden by default	Run the applicatio n	-	password	-	Password is written as hidden text	Fail
UI13	The password should be hidden by default	Run the applicatio n	-	password	-	Password is written as hidden text	Pass
UI13	The password should be hidden by default	Run the applicatio n	-	password	-	Password is written as hidden text	Pass
UI13	The password should be hidden by default	Run the applicatio n	-	password	-	Password is written as hidden text	Pass
UI14	Show password button should be disabled by default	Run the applicatio n	-	password	-	Show password is default disabled	Pass

Test cases-ID	Description	Test Steps	Pre-conditions	Test data	Post-condition	Expected Result	Test Pass/Fail
UI14	Show password button should be disabled by default	Run the application	-	password	-	Show password is default disabled	Pass
UI14	Show password button should be disabled by default	Run the application	-	password	-	Show password is default disabled	Pass
UI14	Show password button should be disabled by default	Run the application	-	password	-	Show password is default disabled	Pass
UI14	Show password button should be disabled by default	Run the application	-	password	-	Show password is default disabled	Pass
UI14	Show password button should be disabled by default	Run the application	-	password	-	Show password is default disabled	Pass
FUNCTIONAL TESTING							
FT11	Verify that the splash screen appears with the application logo	Install and Open the application	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass
FT11	Verify that the splash screen appears with the application logo	Install and Open the application	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass
FT11	Verify that the splash screen appears with the application logo	Install and Open the application	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass

Test cases-ID	Description	Test Steps	Pre-conditions	Test data	Post-condition	Expected Result	Test Pass/Fail
FT11	Verify that the splash screen appears with the application logo	Install and Open the applicatio n	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass
FT11	Verify that the splash screen appears with the application logo	Install and Open the applicatio n	-	-	The application sign-in page should appear	The user sees the splash screen with the logo	Pass
FT12	The product details are presented when clicked on the specific product on the homepage	Open the applicatio n and click on any product	sign in to the user account	-	Product details should be loaded	Product details appear on a new screen	Fail
FT12	The product details are presented when clicked on the specific product on the homepage	Open the applicatio n and click on any product	sign in to the user account	-	Product details should be loaded	Product details appear on a new screen	Fail
FT12	The product details are presented when clicked on the specific product on the homepage	Open the applicatio n and click on any product	sign in to the user account	-	Product details should be loaded	Product details appear on a new screen	Pass
FT12	The product details are presented when clicked on the specific product on the homepage	Open the applicatio n and click on any product	sign in to the user account	-	Product details should be loaded	Product details appear on a new screen	Pass
FT13	Log out button should log the user out	Open the applicatio	Sign up and login to the application	Username Password	User needs to sign in again	User gets logged out	Fail

		n and sign in. Go to account window and click on logout					
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
FT13	Log out button should log the user out	Open the applicatio n and sign in. Go to account window and click on logout	Sign up and login to the application	Username Password	User needs to sign in again	User gets logged out	Pass
FT13	Log out button should log the user out	Open the applicatio n and sign in. Go to account window and click on logout	Sign up and login to the application	Username Password	User needs to sign in again	User gets logged out	Pass
FT13	Log out button should log the user out	Open the applicatio n and sign in. Go to account window and click on logout	Sign up and login to the application	Username Password	User needs to sign in again	User gets logged out	Pass
API TESTING							
API11	Products images are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can browse the products with images	User can check the visuals of the products	Fail

Test cases-ID	Description	Test Steps	Pre-conditions	Test data	Post-condition	Expected Result	Test Pass/Fail
API11	Products images are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can browse the products with images	User can check the visuals of the products	Fail
API11	Products images are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can browse the products with images	User can check the visuals of the products	Pass
API11	Products images are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can browse the products with images	User can check the visuals of the products	Pass
API11	Products images are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can browse the products with images	User can check the visuals of the products	Pass
API12	Names of the products are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can recognize the products with names	User can identify the products by name	Pass
API12	Names of the products are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can recognize the products with names	User can identify the products by name	Pass
API12	Names of the products are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can recognize the products with names	User can identify the products by name	Pass
API12	Names of the products are loaded on the homepage	Open the applicatio n and sign in	Sign up	Username Password	User can recognize the products with names	User can identify the products by name	Pass
API13	Email verification works for new users	Sign up to the applicatio n Try to login	Sign up and choose a username and password	Email address	User can verify their email address to use the application	Email verification link is sent by the server	Fail

Test cases-ID	Description	Test Steps	Pre-conditions	Test data	Post-condition	Expected Result	Test Pass/Fail
API13	Email verification works for new users	Sign up to the applicatio n Try to login	Sign up and choose a username and password	Email address	User can verify their email address to use the application	Email verification link is sent by the server	Fail
API13	Email verification works for new users	Sign up to the applicatio n Try to login	Sign up and choose a username and password	Email address	User can verify their email address to use the application	Email verification link is sent by the server	Fail
API13	Email verification works for new users	Sign up to the applicatio n Try to login	Sign up and choose a username and password	Email address	User can verify their email address to use the application	Email verification link is sent by the server	Pass
API13	Email verification works for new users	Sign up to the applicatio n Try to login	Sign up and choose a username and password	Email address	User can verify their email address to use the application	Email verification link is sent by the server	Pass
DATABASE TESTING							
DB11	Test that the database can store and fetch the product details correctly	Open the applicatio n Login as a registered user Browse product and select one	Products should be added already	Username Password	Users can see the product details	The database fetches the product details	Fail
DB11	Test that the database can store and fetch	Open the applicatio n	Products should be	Username Password	Users can see the product details	The database fetches the product details	Fail

	the product details correctly	Login as a registered user Browse product and select one	added already				
Test cases-ID	**Description**	**Test Steps**	**Pre-conditions**	**Test data**	**Post-condition**	**Expected Result**	**Test Pass/Fail**
DB11	Test that the database can store and fetch the product details correctly	Open the applicatio n Login as a registered user Browse product and select one	Products should be added already	Username Password	Users can see the product details	The database fetches the product details	Fail
DB11	Test that the database can store and fetch the product details correctly	Open the applicatio n Login as a registered user Browse product and select one	Products should be added already	Username Password	Users can see the product details	The database fetches the product details	Fail
DB12	Check that the database can handle user authentication, including storing and retrieving user passwords securely	Open the applicatio n and sign up	User must sign up in order to create an entry	Username Password	Users can validate accounts and use the application	The database allows authorized users to sign in	Pass
DB12	Check that the database can handle user authentication, including storing and	Open the applicatio n and sign up	User must sign up in order to create an entry	Username Password	Users can validate accounts and use the application	The database allows authorized users to sign in	Pass

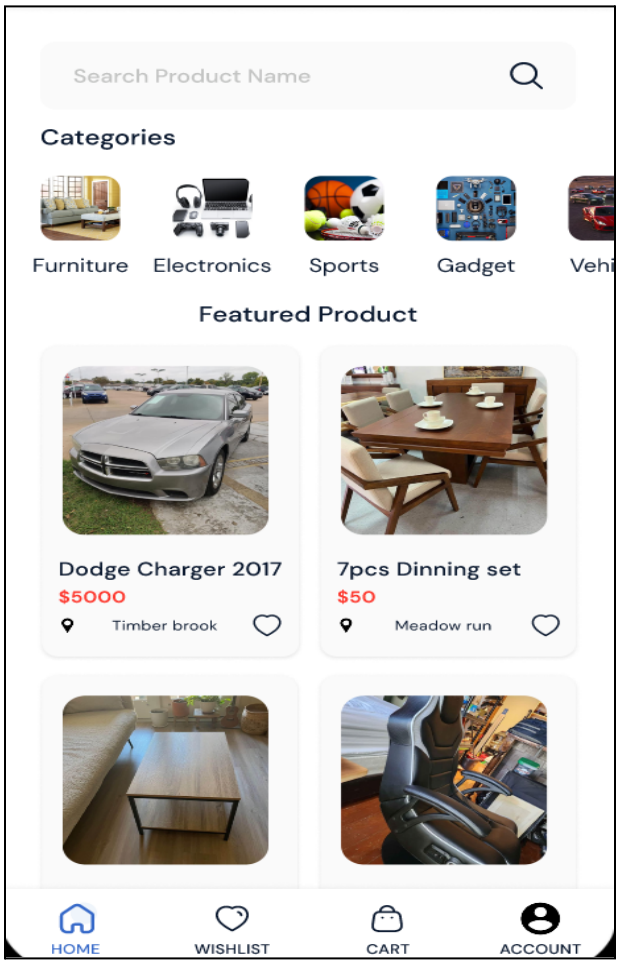
	retrieving user passwords securely						
DB12	Check that the database can handle user authentication, including storing and retrieving user passwords securely	Open the applicatio n and sign up	User must sign up in order to create an entry	Username Password	Users can validate accounts and use the application	The database allows authorized users to sign in	Pass
DB12	Check that the database can handle user authentication, including storing and retrieving user passwords securely	Open the applicatio n and sign up	User must sign up in order to create an entry	Username Password	Users can validate accounts and use the application	The database allows authorized users to sign in	Pass
DB12	Check that the database can handle user authentication, including storing and retrieving user passwords securely	Open the applicatio n and sign up	User must sign up in order to create an entry	Username Password	Users can validate accounts and use the application	The database allows authorized users to sign in	Pass

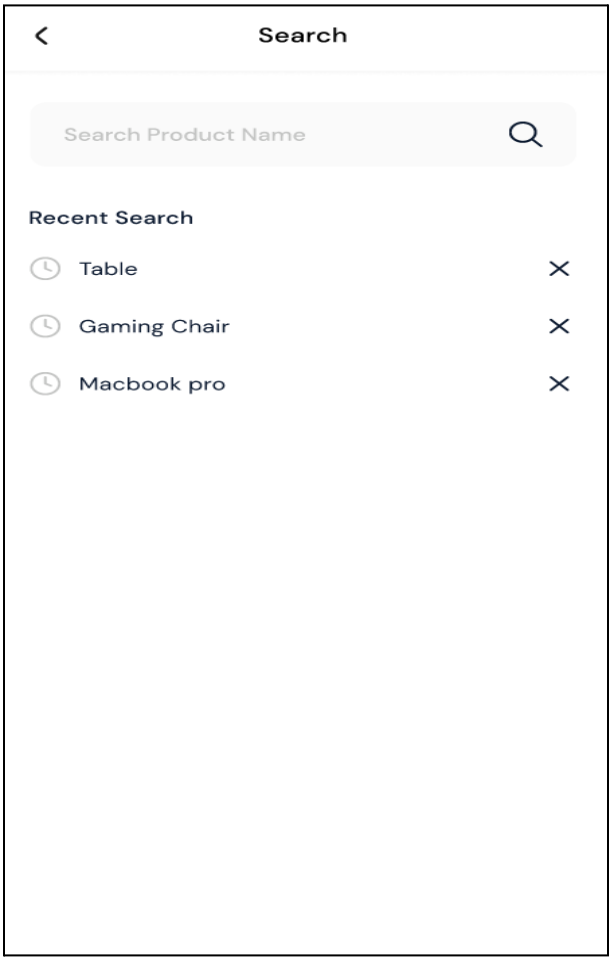
Testing Traceability Matrix

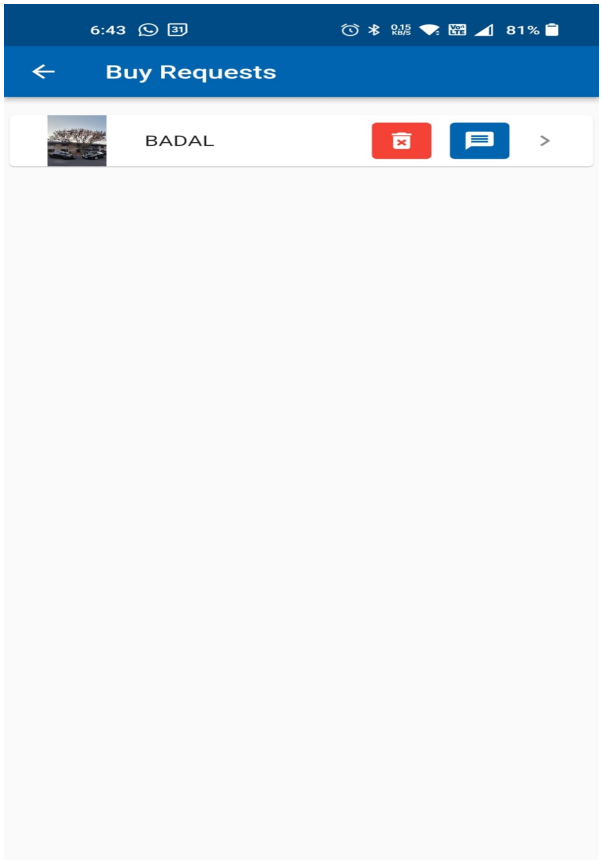
	Register	Log In	Account Tab	Home Page	Search Bar	Splash screen	Product Details	Log Out	Database Connection	Database Availability	Database Data retrieval	Database Update	API Get Product Image	API GET Product Details	API POST Email
Requirements	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15
Test Cases															
UI01	X														
UI02		X													
UI03			X												
UI04				X											
UI05					X										
UI11						X									
UI13	X														
FT11	X														
FT01	X														
FT02		X													
FT12						X	X								
FT13								X							
DB01									X						
DB02										X					
DB11											X				
DB22												X			
API11													X		
API12														X	
API13															X

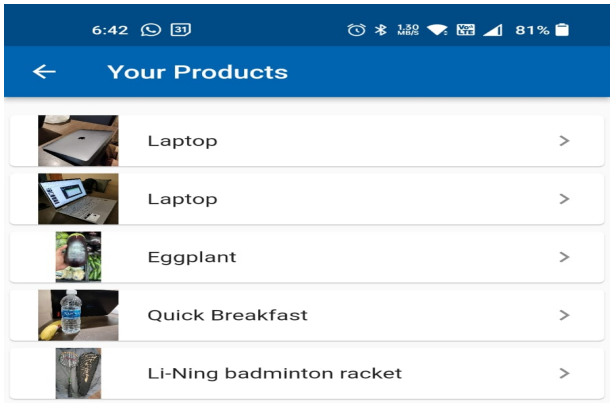
Features Implemented

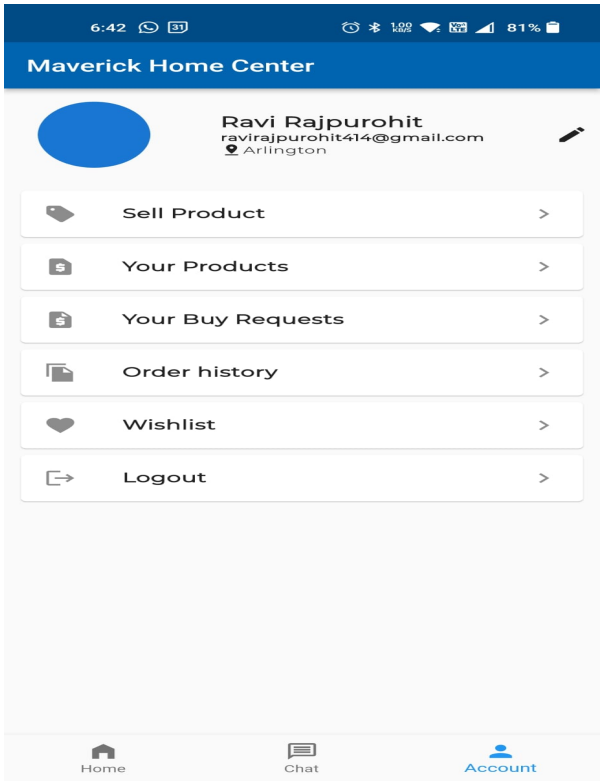
- Home Page.
- Search Feature.
- Buy Requests.
- Sell a product.
- Profile Setup.
- Your products.
- Chat screen.
- Items on sale.

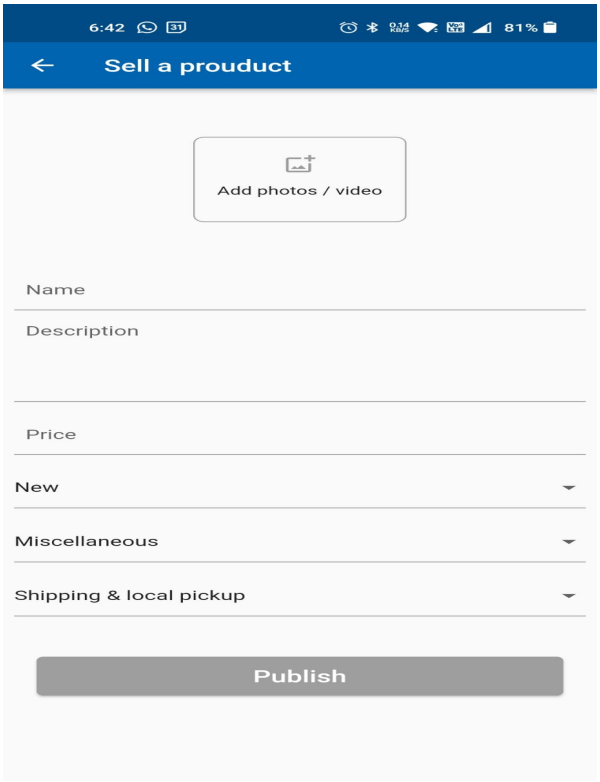


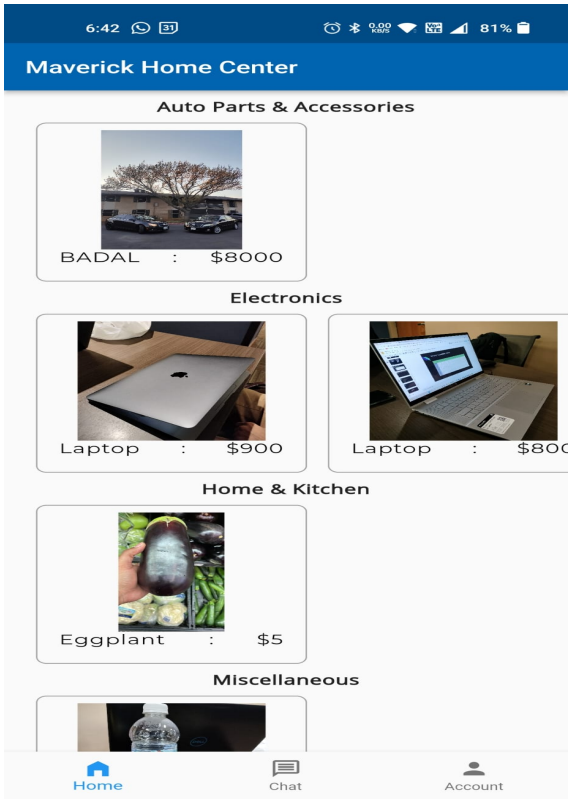


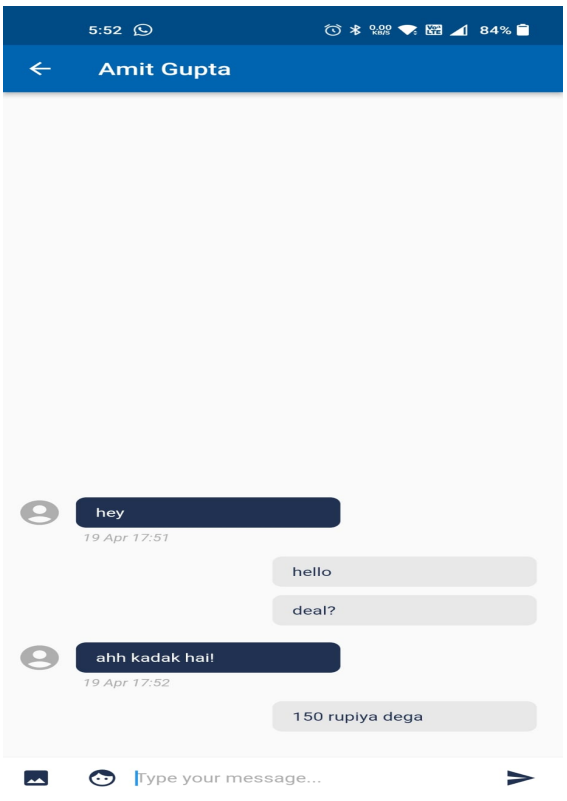

6:43
81%
Buy Requests
BADAL


6:42
81%
Your Products
Laptop
Laptop
Eggplant
Quick Breakfast
Li-Ning badminton racket


6:42
81%
Maverick Home Center
Ravi Rajpurohit
ravirajpurohit414@gmail.com
Arlington
Sell Product
Your Products
Your Buy Requests
Order history
Wishlist
Logout
Home
Chat
Account


6:42
81%
Sell a prouduct
Add photos / video
Name
Description
Price
New
Miscellaneous
Shipping & local pickup
Publish


6:42
81%
Maverick Home Center
Auto Parts & Accessories
BADAL : $8000
Electronics
Laptop : $900
Laptop : $80
Home & Kitchen
Eggplant : $5
Miscellaneous
Home
Chat
Account


5:52
84%
Amit Gupta
hey
19 Apr 17:51
hello
deal?
ahh kadak hai!
19 Apr 17:52
150 rupiya dega
Type your message...

Academic Integrity Honor Code

I pledge, in my honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence. I promise that I will submit only the work that I personally create or contribute to group collaborations, and I will appropriately reference any work from other sources. I will follow the highest standards of integrity and uphold the spirit of the Honor Code.

Signature: Aditya Bhat
Name: Aditya Vikram Bhat
UTA ID: 1002014494

Signature: Akash Biswas
Name: Akash Biswas
UTA ID: 1002055500

Signature: Amit Gupta
Name: Amit Munna Gupta
UTA ID: 1002066302

Signature: Ravi Rajpurohit
Name: Ravi Rajpurohit
UTA ID: 1002079916

Conclusion

In conclusion, building an app that sells refurbished items to local students requires careful consideration of several key factors, including the purpose of the **app**, **user profiles**, **security**, **communication**, and **ratings** and **reviews**. Additionally, implementing quality management in the app is critical to ensure a high-quality user experience, including **testing**, **user feedback**, **continuous improvement**, **security**, and compliance with **legal and regulatory requirements**.

When estimating the effort required to build such an app, it's important to consider additional factors such as **API integration**, **content management**, **localization**, **scalability**, and **analytics**. Being thorough and detailed in the estimation process can help avoid underestimating the time and resources required to build a successful app that meets the needs of its users.

Overall, building an app that sells refurbished items to local students can be a complex and time-consuming process, but with careful planning and execution, it can provide an engaging and valuable service to students and the local community.