UTM

UNIVERSITI TEKNOLOGI MALAYSIA

SEMESTER I 2022/2023

SCSJ 3104 SEC 2

APPLICATION DEVELOPMENT PROJECT

SPRINT 1

PROJECT TITLE:

UtiTrade Application Development System

LECTURER: ASSOC. PROF. DR. MOHD. YAZID IDRIS

GROUP NAME: Master Controllers

GROUP MEMBERS:

NO	Name	No Matric
1.	ANATASYA HUMAIRA	A20EC0261

Table of Content

1. Introduction To Application

1.1. Introduction

Lately, the growth of the Internet has surged dramatically. Particularly in the realm of online shopping, internet stores remain open round the clock, and a significant portion of households have an internet connection. This thriving online landscape has evolved beyond mere websites into prosperous digital ventures. However, our group's perspective emphasises the amplified performance of the C2C UTM Online Shop. This platform specialises in promoting UTM students' unused goods or perhaps reusable items and endeavours to boost revenue, a feat that might be unattainable for a UTM store reliant on physical presence in specific regions.

The main objective revolves around crafting an engaging tool that simplifies the process for UTM online stores to offer their products and enables customers to showcase their wares, facilitating peer-to-peer transactions through the UTM Online market's C2C utility. Furthermore, this tool heightens consumer satisfaction by expediting UTM product orders with just a few clicks. Our commitment lies in enriching online shopping services, bolstering customer loyalty, and augmenting profits. The financial outlook for the UTM Online Shop C2C venture is both cautious and promising.

Therefore, We intend to develop this e-commerce web application based system called "UtiTrade" in order to help the students to sell their unwanted items in a fun and easy way by uploading a picture and negotiating directly with the buyers within UTM. Unlike desktop apps, web-based apps are not as reliant on the hardware that the students are using. This means that older, newer, faster, slower, bigger and smaller machines can all access and use the program. It also means that web-based apps offer more points of entry and are more efficient in terms of memory usage.

1.2. Problem Statement

The majority of pupils favour opting for used items instead of new ones, given the significantly reduced and budget-friendly cost. Simultaneously, a portion of other students possess items they don't employ and desire to generate some earnings. Reaching out to potential buyers for their products constitutes one of the most challenging predicaments for any student. Establishing a platform for students to trade, both in buying and selling, stands as a crucial necessity for all UTM students and those with an interest in acquiring second hand goods. Unfortunately, currently, no websites are available that cater to students seeking to sell their possessions, encompassing garments, electronic gadgets, furniture, and household articles, and in most instances, new items prove to be pricier.

1.3. The Proposed System

Introducing the UTM C2C online marketplace—an internet-based hub facilitating UTM scholars in enlisting their previously owned commodities they intend to market, while simultaneously identifying fresh clientele. This endeavour would remain unattain.

able sans a virtual venue or a physical presence within distinct geographical confines. Given that certain patrons might be dispersed across diverse regions, satisfying their requisites presents challenges. The UTM online marketplace aspires to fashion an internet-based arena for patrons to vend their

wares digitally. Here, patrons possess the liberty to opt for their desired items and conduct online transactions to procure chosen goods. Subsequently, the merchandise would be dispatched to patrons' locations via the UTM virtual marketplace.

1.4. Technology And Tool Used

The proposed technology and tools used in developing UtiTrade web-based application are as listed below:

- Trello
- Lucid App
- GitHub
- Xampp
- Figma
- Visual Studio Code
- Google Sheets
- Laravel

1.5. Functional Requirements

Functional requirements are a crucial part of software development. Below are the functional requirements of UtiTrade online shopping system, which describe a specific description of a software system's behaviour and the functions it must perform.

1. The system shall provide a user registration process that allows users to create an account by providing their UTM email address.
2. Upon successful registration, the system shall send a verification email to the provided address, which the user must click to confirm their registration.
3. The system shall ensure that usernames and email addresses are unique.
4. The system shall provide user authentication whenever a user logs into the system.
5. The website shall be integrated with a secure payment system.
6. The website shall allow the user to manage and edit their profile.

7. Users shall be able to check their transaction history.
8. The system shall display available products.
9. The system shall provide a search functionality allowing users to search items by name.
10. Search results shall display the names of matching items.
11. When a user clicks on their profile picture, the system shall initiate an action to display the user's profile page.
12. The system shall provide a 'Add to Cart' functionality that enables users to add products to their shopping cart.
13. Users shall be able to manage their cart, remove products from cart, edit quantity of the product, and buy products from the cart.
14. The system shall provide an 'Add Product' functionality that allows users to add new products to the catalog for sale.
15. Users shall be able to add new products, including edit existing product details, and remove products from the catalog.
16. The system shall provide a 'Buy Product' functionality that allows users to purchase products available in the system's catalog.
17. Users shall be able to add new products, edit existing product details, and remove products from the catalog.
18. Users shall be presented with multiple payment methods, including cashless options (credit/debit cards, digital wallets) and cash payment method.
19. Users shall have access to the checkout process from their shopping cart or product page.
20. Users shall receive immediate feedback on the success or failure of the payment authorization.

References:

1. Wei, F., & Zhang, Q. (2018). Design and Implementation of Online Shopping System Based on B/S Model (vol. 246). MATEC Web Conf. https://doi.org/10.1051/matecconf/201824603033
2. MN, A. Z., Junaidi, J., & Putra, R. D. (2018). Design of E-Commerce Payment System at Tokopedia Online Shopping Site (vol.1, no.2). APTISI Transactions on Management. https://doi.org/10.33050/atm.v1i2.666
3. McLaughlin, C., Bradley, L., Prentice, G., et al. (2017). Consumer to consumer (C2C) online auction transaction intentions: an application of the Theory of Planned Behaviour. Dublin Business School. https://esource.dbs.ie/handle/10788/3372

4. Sabir, S. (2010). E-commerce solution : creating a shopping cart using open sources (APACHE/MYSQL/PHP). Turun ammattikorkeakoulu. https://urn.fi/URN:NBN:fi:amk-2010062312480

2. Application Diagram

2.1. Use Case Diagram

Figure 1.0 below represents the use case diagram of UtiTrade with three subsystems which are authentication subsystem, purchasing product subsystem, and selling product subsystem.

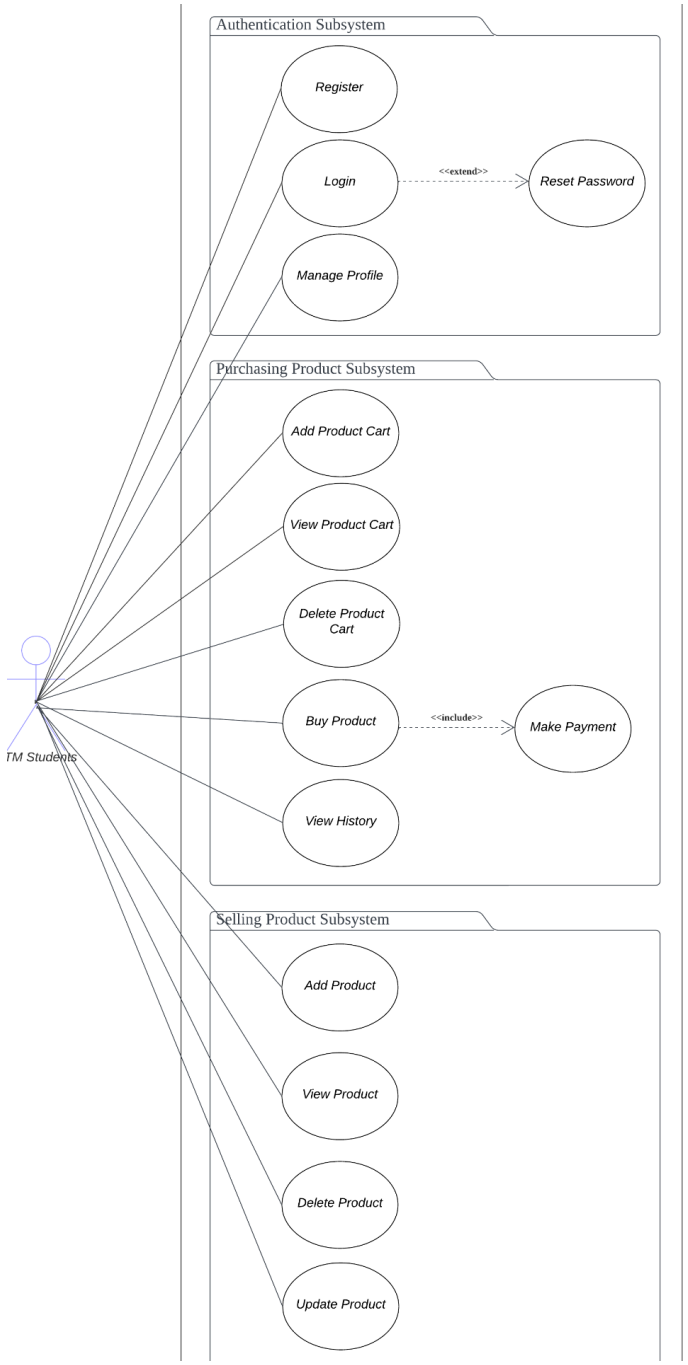


Figure 1.0 : UtiTrade Use Case Diagram

2.2. Architecture Design

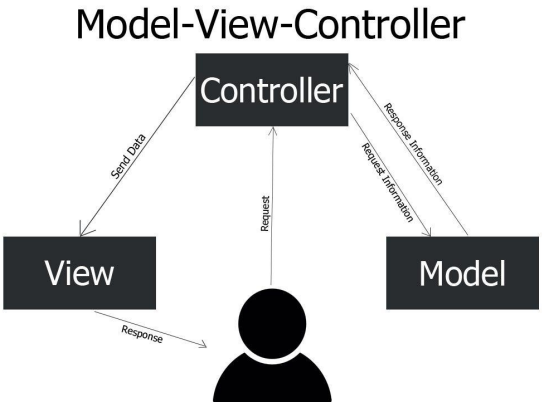


Figure 2.0 : MVC Diagram

The Model-View-Controller (MVC) architectural pattern is a widely employed design approach in software development, especially for constructing user interfaces, managing data, and handling control logic. This pattern is structured into three main components: a Model responsible for maintaining core system data, a View responsible for presenting information to users, and a Controller responsible for managing user input. This project will adopt the MVC pattern due to its suitability for web applications, its capacity to accommodate multiple views, and its ability to facilitate modifications without disrupting the entire underlying model.

The MVC pattern separates the web-based application into three interconnected components including Model, View, and Controller as shown in Figure 2.1. The Model layer here is used to bridge the gap between the database and the application code. It handles database operations, data retrieval, and data updates whereas the View layer includes the web application's user interface (UI), which is typically built using web technologies like HTML, CSS, and JavaScript. The Controller layer consists of application logic that handles user interactions and manages the flow of data between the Model and View. Controllers receive requests from users (e.g., browsing products, adding to cart, making payments) and invoke corresponding actions.

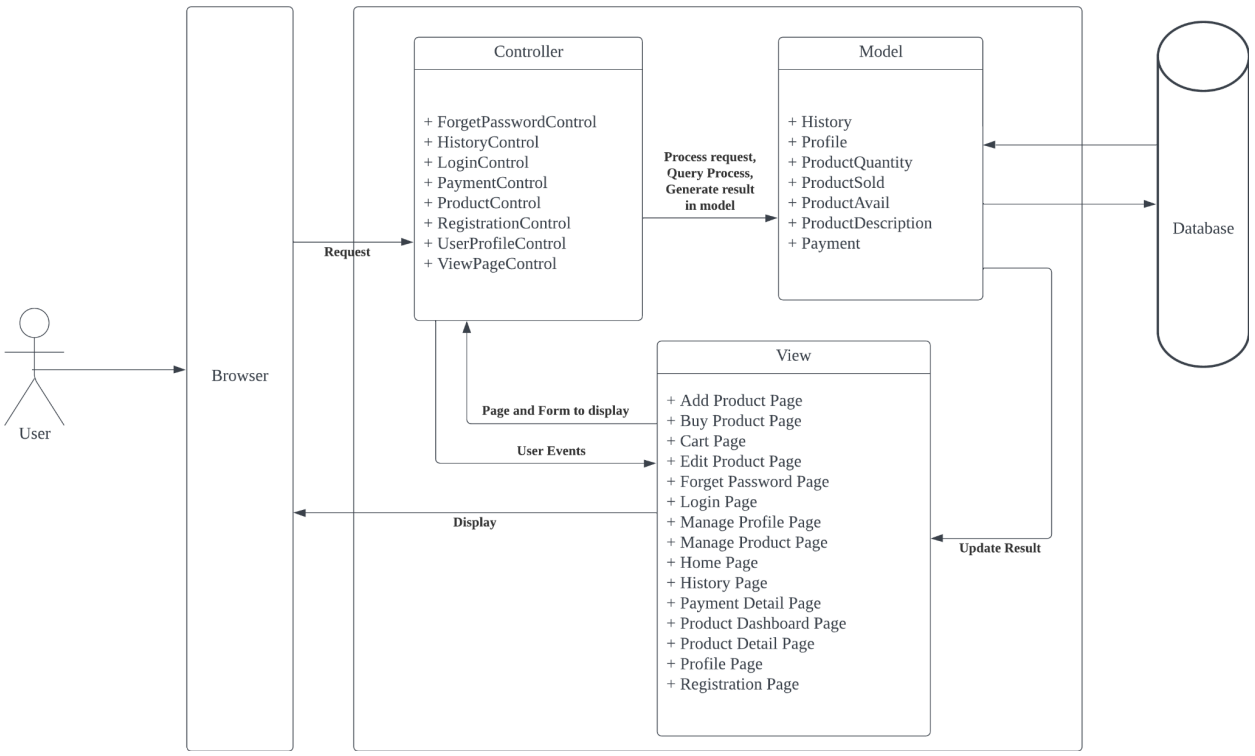


Figure 2.1 : Architecture Diagram UtiTrade System

3. Module 1

3.1. Specific Requirement

The goal of UtiTrade C2C online shopping system is to provide a dynamic platform designed exclusively for UTM students, allowing them to easily resale unneeded products as well as find value items supplied by their peers within the university community. Our approach encourages students to declutter, recycle, and discover economical resources in a sustainable and cost-effective manner, all while creating a lively, student-centric marketplace.

3.1.1. User characteristics

The UtiTrade C2C online shopping system will be used by only one user which is the UTM student. This is because C2C platforms are designed to facilitate direct interactions and transactions between individual customers or users. While the primary user role in this C2C system is the same (UTM students), it is important to note that there are different user characteristics and roles within this category:

3.1.1.1. UTM Students (as buyer)

- Students are expected to still be registered as an active student in UTM.
- Students will require UTM email to register their account to the system for the first time used and login by their UTMID.

- Students have specific preferences and interests, which influence the types of products they search for and purchase.
- Students may use the ‘Product Cart’ feature to bookmark products they are interested in but are not yet ready to purchase. This cart helps them keep track of items they may want to buy later.
- Each student has a user profile that may include their username, full name, phone number, email address, profile picture, and address. Profiles can help build trust between buyers and sellers.
- Students may reset their passwords in the event of a forgotten or compromised password, ensuring uninterrupted access to their accounts and a seamless online shopping experience.

3.1.1.2. UTM Students (as seller)

- Students are expected to still be registered as an active student in UTM.
- Students will require UTM email to register their account to the system for the first time used and login by their UTMID.
- Students have specific products or unwanted items they wish to sell to the other students with lower prices.
- Each student has a user profile that may include their username, full name, phone number, email address, profile picture, and address. Profiles can help build trust between buyers and sellers.
- Students may reset their passwords in the event of a forgotten or compromised password, ensuring uninterrupted access to their accounts and a seamless online shopping experience.

3.1.2. System Features

The UtiTrade C2C online shopping system is a web-based application system that functions on various computing devices, such as desktop and mobile devices, that includes web browsers. For module one, the system features include:

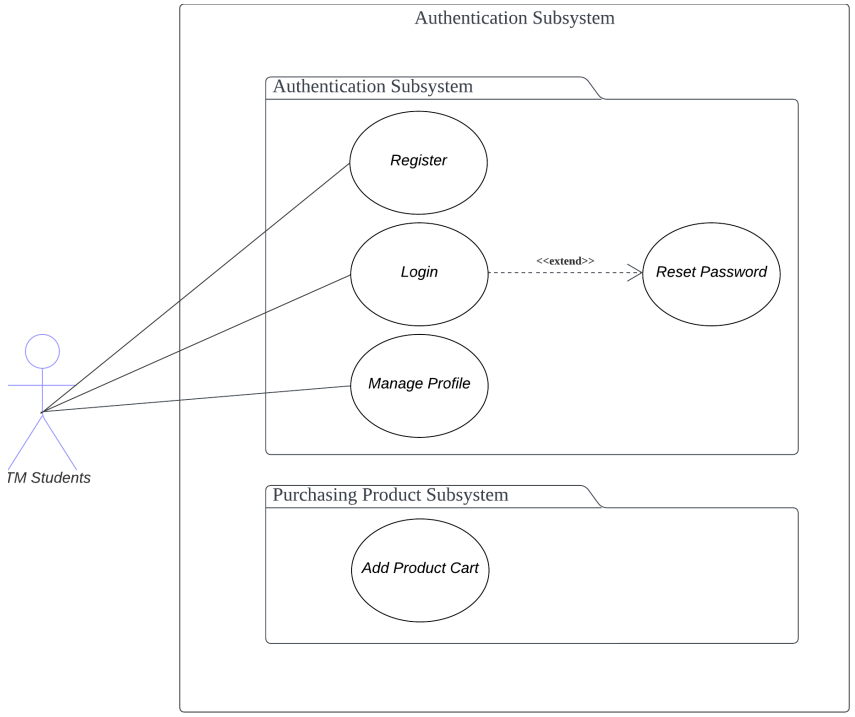


Figure 3.0 : Use Case Diagram for Module 1

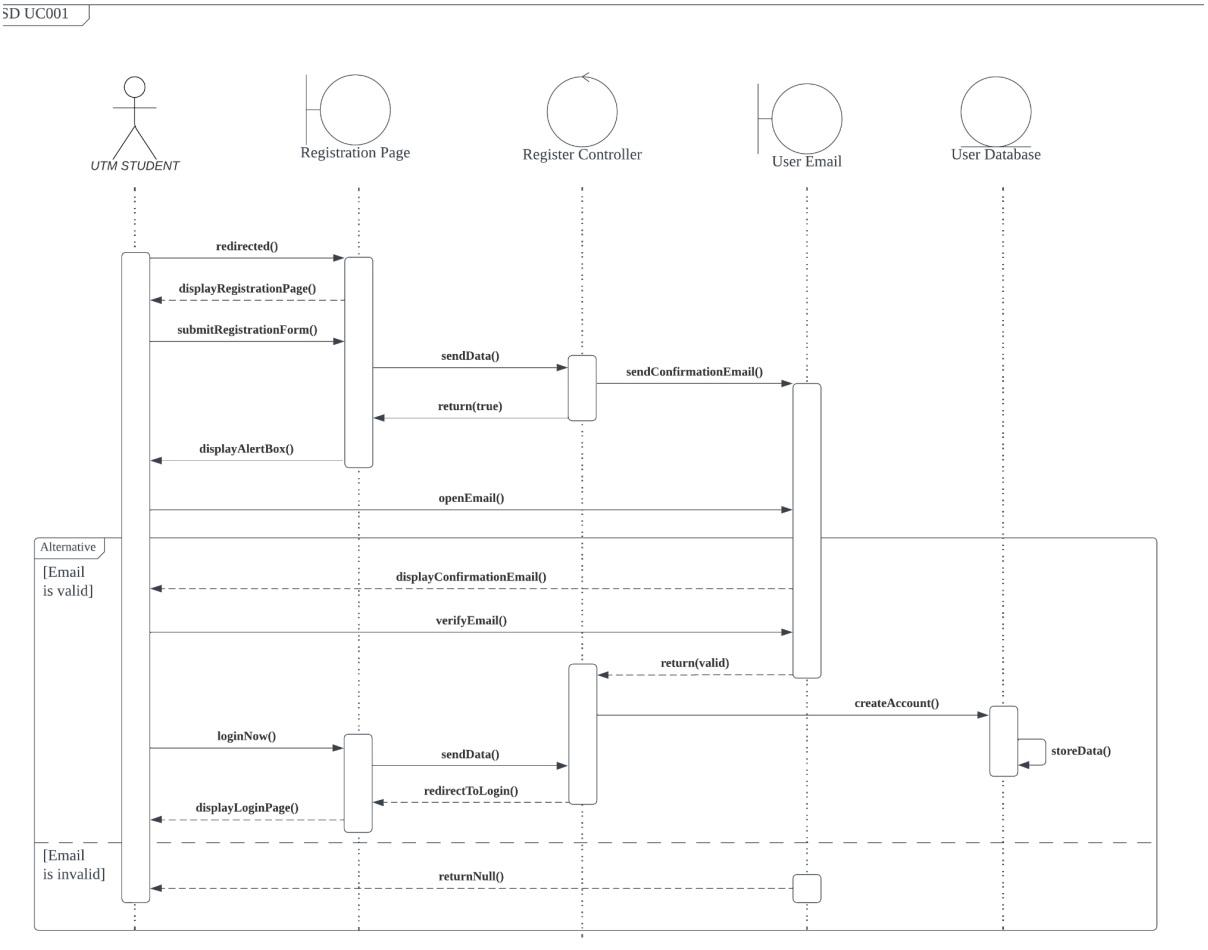
3.1.3. UC001: Use Case <Register> & Seq Diagram

Table 1.0: Use Case Description for <register>

Use case: <register>
ID: UC001
Actors: UTM Student
Preconditions: 1. The user must be registered as an active UTM student. 2. The user must not already be registered to the system.
Flow of events: 1. User navigates to the registration page. 2. The user enters their UTM email address, UTMID, and password.

3. The system displays an alert box which asks the user to verify their email address . 4. User shall open their email. 5. User taps the verification email and confirms their email. 6. User refresh the page and taps on ‘login now’ button 7. System redirects the user to the login page. 8. Use case end.
Exception flow (if any): 1. Email address is invalid.

Figure 3.1: SD001: Sequence Diagram for <register>

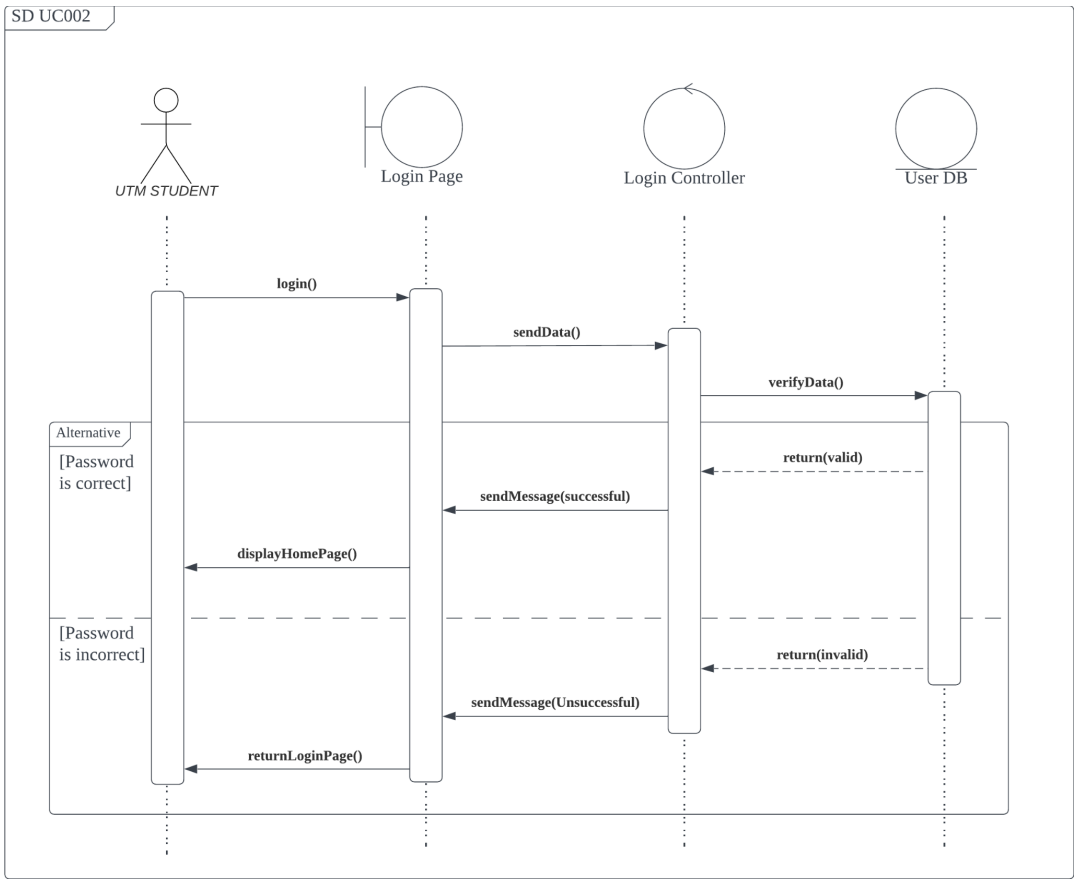


3.1.4. UC002: Use Case <Login> & Seq Diagram

Table 2.0: Use Case Description for <login >

Use case: <login>
ID: UC002
Actors: UTM Student
Preconditions: 1. The user must be registered as an active UTM student. 2. The user must be registered to the system. 3. The user must input the registered UTMID and password.
Flow of events: 1. The system will redirect user to the login page. 2. The system displays the login page. 3. User fills in the required details such as UTMID and password. 4. User taps the login button. 5. System checks and validates the user's login form. 6. System redirects the user to the user's home page. 7. Use case end.
Exception flow (if any): 1. Password is incorrect. - System displays an error message.

Figure 3.2: SD002: Sequence Diagram for <Login>



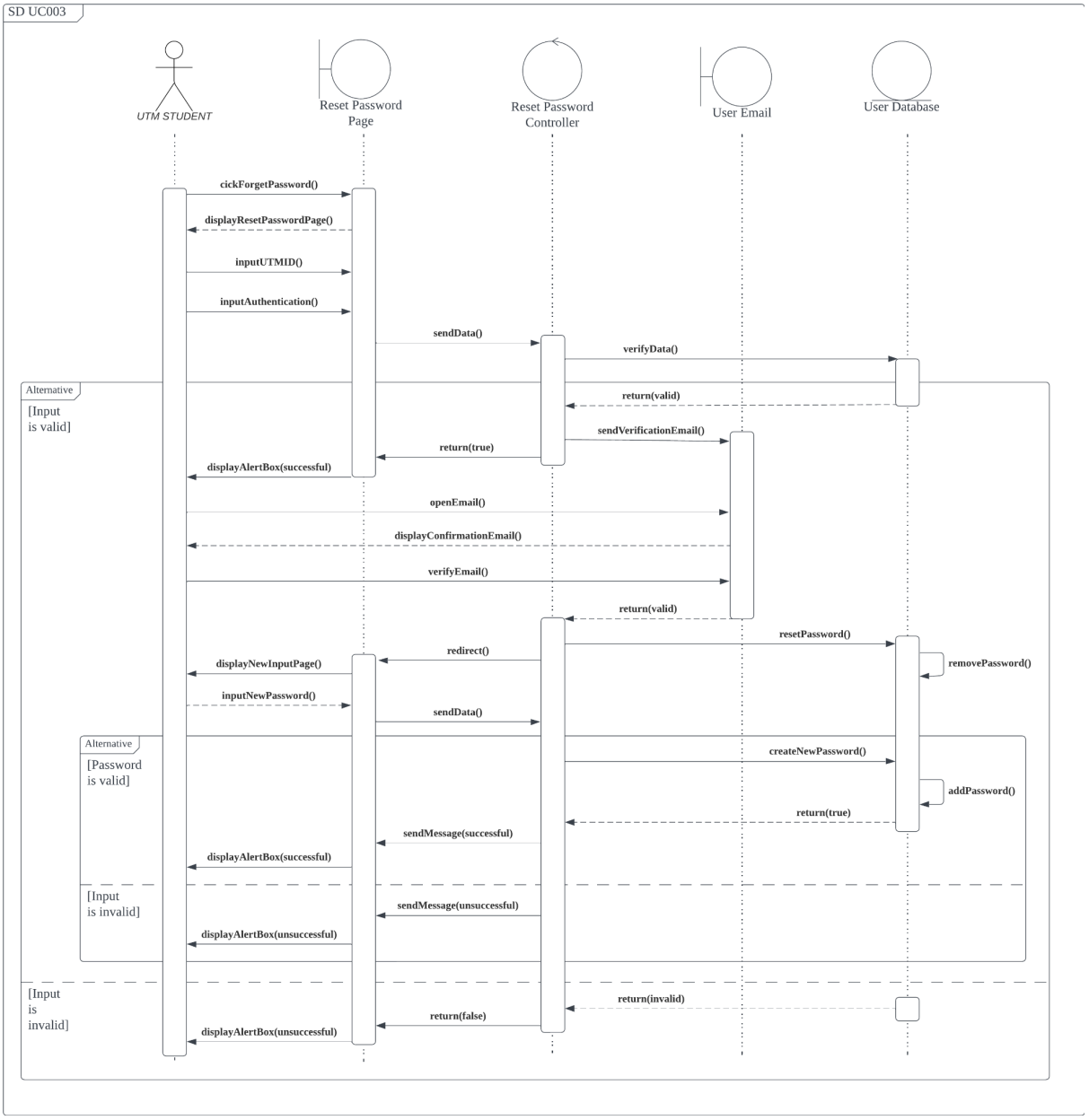
3.1.5. UC003: Use Case <Reset Password> & Seq Diagram

Table 3.0: Use Case Description for <Reset Password >

Use case: <reset password>
ID: UC003
Actors: UTM Student
Preconditions: 1. The user must be registered as an active UTM student. 2. The user must be registered to the system.
Flow of events: 1. The use case starts when user is already in the log in page. 2. User taps on the ‘forget password?’. 3. The system will redirect user to the reset password page.

4. The user input their UTMID as well as the correct answer for authentication purpose. 5. User clicks continue button. 6. System checks and validates the user's submission form. 7. System displays an alert box which tells the user an email has been sent to their email address and asks user to reset the password via email. 8. User opens their UTM email address. 9. User reset their password from the email. 7. Use case end.
Exception flow (if any): 1. Email address or authentication is invalid. - System display unsuccessful 2. Password is invalid - System display unsuccessful

Figure 3.2: SD003: Sequence Diagram for <reset password>



3.1.6. UC004: Use Case <Manage Profile> & Seq Diagram

3.1.7. UC005: Use Case <Add to Cart> & Seq Diagram

ID:	UC003 Add Product Cart
Actors:	Registered User: A UTM student who has created an account on the UtiTrade platform.
Preconditions:	- The user is logged in to their UtiTrade account. - The user has navigated to a product listing page.
Basic Flow:	1. The user navigates to the product listing page where they can view various products available for purchase. 2. The user identifies a product they want to buy and clicks on the product's details to view more information.

	3. The system displays detailed information about the product, including its name, description, price, and an "Add to Cart" button. 4. The user clicks the "Add to Cart" button. 5. The system verifies the availability of the product and adds it to the user's shopping cart. 6. The system confirms the successful addition of the product to the user's cart by displaying a notification or message.
Alternative Flow:	If the product is not available (e.g., it was sold by another user), the system displays an error message indicating that the product cannot be added to the cart.
Postconditions:	- The selected product is added to the user's shopping cart. - The cart icon (or a visual indicator) in the user interface shows the updated count of items in the cart.
Special requirements:	- The "Add to Cart" process should be responsive and user-friendly, ensuring a smooth experience for users on both desktop and mobile devices. - The system should prevent adding the same product to the cart multiple times; instead, it should increase the quantity of the existing cart item. - If the user attempts to add a product to the cart without being logged in, the system should prompt the user to log in or create an account first. - The cart should retain the added products even if the user navigates to different pages or logs out and logs back in
Assumptions:	- The product availability is validated in real-time to ensure accurate cart management. - The system will not allow users to add products to the cart that have been marked as unavailable by the seller. - Product details such as name, description, price, and availability are displayed clearly to the user before they confirm adding the product to the cart.
Constraints:	- The implementation should consider potential performance impacts as the number of products in the cart increases. - The user interface should be intuitive, making it easy for users to locate the "Add to Cart" button and view their cart's contents.

3.2. Data Design

There are 4 entities in the database design of the UtiTrade system during this sprint : User DB, Product DB, Cart DB, ProductAvail DB. Figure 3.2.1 below represents the database design of the UtiTrade system as Table 3.2.1 represents the description of the entity.

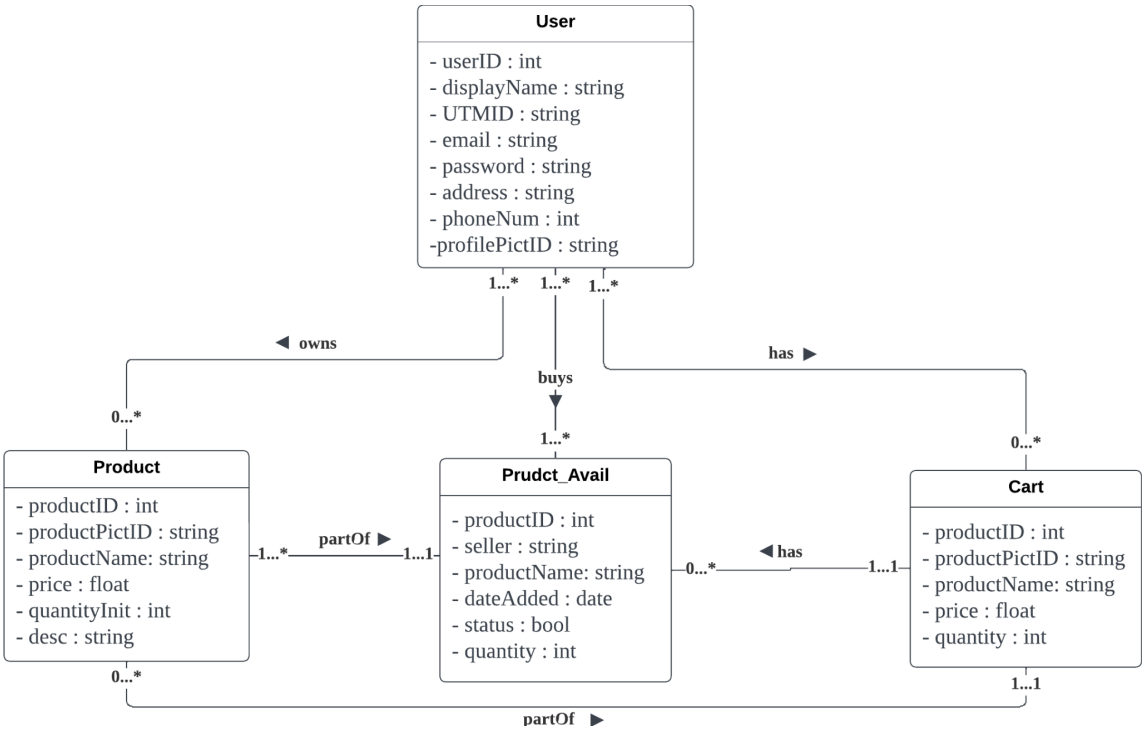


Figure 3.2.1: Entity Relationship Diagram of UtiTrade System.

3.2.1. Data Dictionary & Description

Table 3.2.1.1: Entity Description

Entity	Description
User	Contains the detail information of user
Product	Contains the detail information of a certain product
Product_Avail	Contains the detail information of available products
Cart	Contains the detail information of a product of the user's cart.

Data Dictionary

3.2.1.1. Entity: <User>

Attributes	Type	Description
userID	integer	This attribute serves as a unique identifier for each user in the system. It is used for referencing and distinguishing one user from another.
displayName	string	The displayName attribute stores the user's chosen name and can be edited. It allows users to personalize their profiles and how they appear to others.
UTMID	string	UTMID is an attribute that stores a unique identifier associated with the user's university ID.
email	string	The email attribute stores the user's email address, which is essential for reset password, account authentication, and account confirmation.
password	string	The password attribute stores a securely hashed or encrypted version of the user's password.
address	string	The address attribute stores the user's physical address. which may be used for shipping or location-based services.
phoneNum	integer	It stores the user's contact number. It can be useful for communication.
profilePictID	string	This attribute serves as a reference to the user's profile picture or avatar image.

3.2.1.2. Entity: <Product>

Attributes	Type	Description
productID	integer	This attribute is a unique identifier for each product in the system. It is used to distinguish one product from another and is crucial for database operations.
productPictID	string	The productPictID attribute serves as a reference to the product's main image or picture. Instead of storing the actual image data within the product entity, it links to an image stored elsewhere in the system.
productName	string	productName stores the name or title of the product.
price	float	The price attribute stores the monetary value associated with the product. It indicates the cost of purchasing the item.
quantityInit	integer	This attribute represents the initial quantity of the product available for sale.
desc	string	The desc attribute holds a detailed description of the product. It provides additional information about the item's features, specifications, condition and any other relevant details.

3.2.1.3. Entity: <Product_Avail>

Attributes	Type	Description
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
seller	string	As reference from the User entity. This attribute serves as a unique identifier for each user that sells the particular item in the system.
productName	string	As reference from the Product entity, it stores the name or title of the product. It provides a clear and concise description of the item and is displayed prominently in product listings.
dateAdded	date	It represents the date and time when the product listing was created or added to the platform.
status	bool	The status attribute is of type 'bool' (boolean) and is used to indicate the availability or status of the product listing. A 'true' value typically indicates that the product is available for sale, while 'false' may indicate that the product has been sold or is no longer available.
quantity	integer	It represents the quantity of the product available for sale in the listing. As products are sold, this quantity may decrease to reflect the number of items still available for purchase.

3.2.1.4. Entity: <cart>

Attributes	Type	Description
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
productPictID	string	As reference from the Product entity, the productPictID attribute serves as a reference to the product's main image or picture.
productName	string	As reference from the Product entity, it stores the name or title of the product.
price	float	As reference from the Product entity, the price attribute stores the monetary value associated with the product.
quantity	integer	It represents the quantity of the product in the cart. Users may edit or remove the quantity according to their needs.

3.3. User Interface Design

For the UtiTrade system, there are various pages of interface design that our group managed to create by using a design tool called *Figma*. The system provides two functionality dashboards

which allows the user to be a buyer as well as be a seller at the same time. Several user interfaces of both sides will show as below.

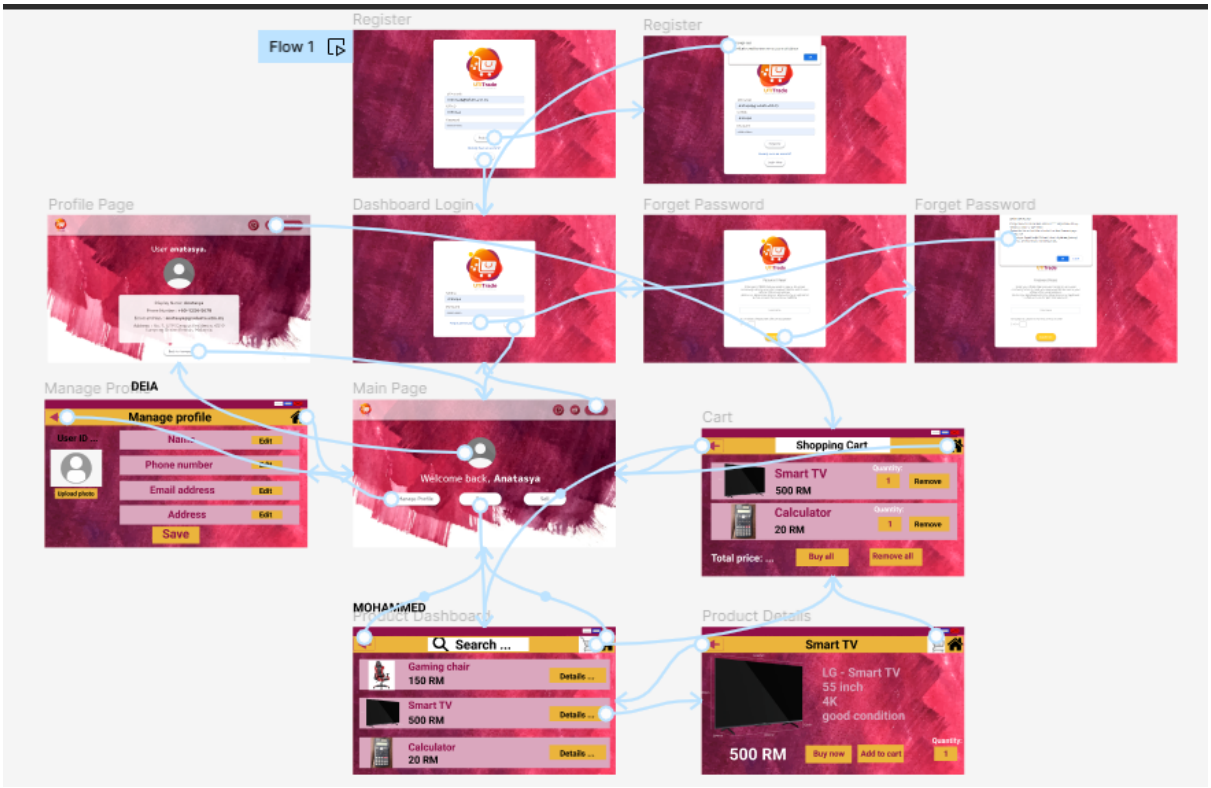


Figure 3.3: Overall System Interface Design

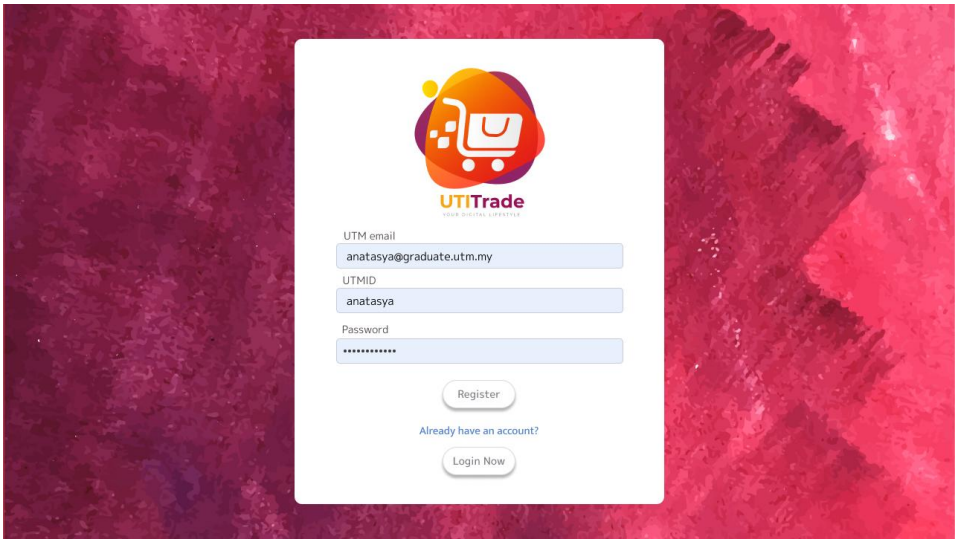


Figure 3.4: Interface Design of Registration Page

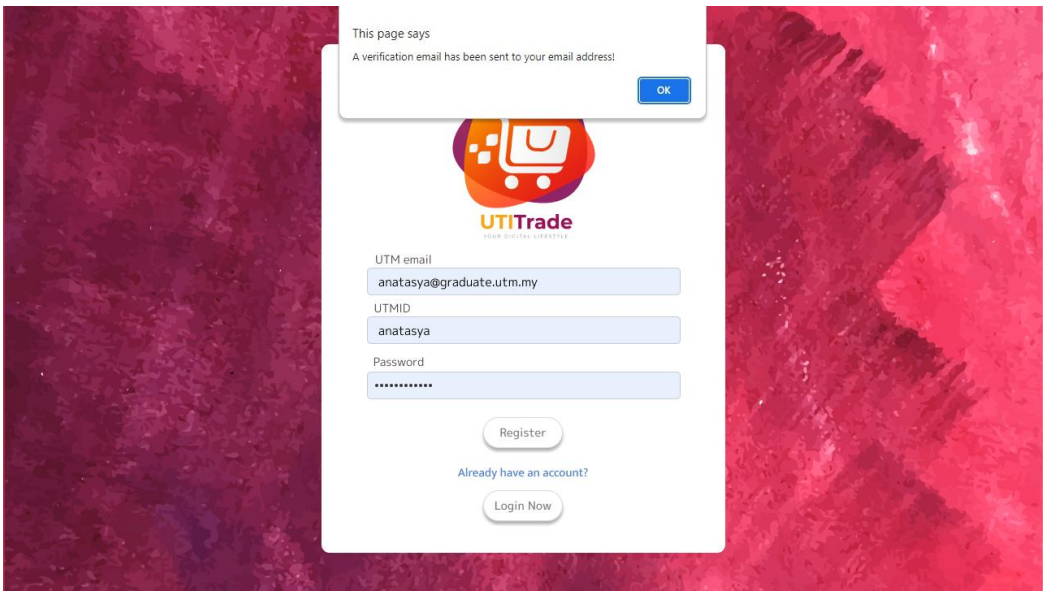


Figure 3.4: Interface Design of Registration Page 2

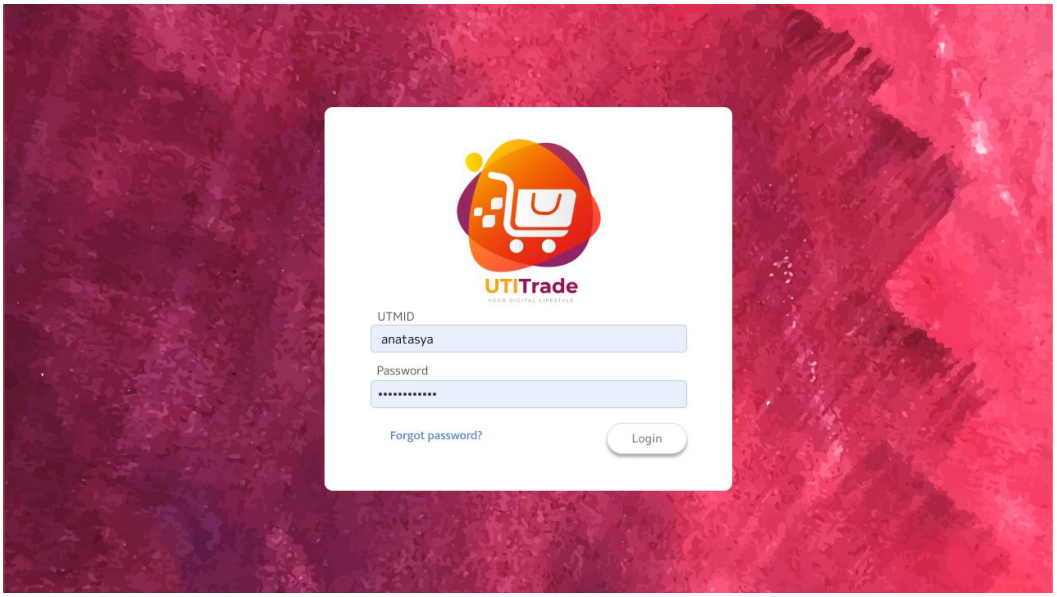


Figure 3.4: Interface Design of Login Page

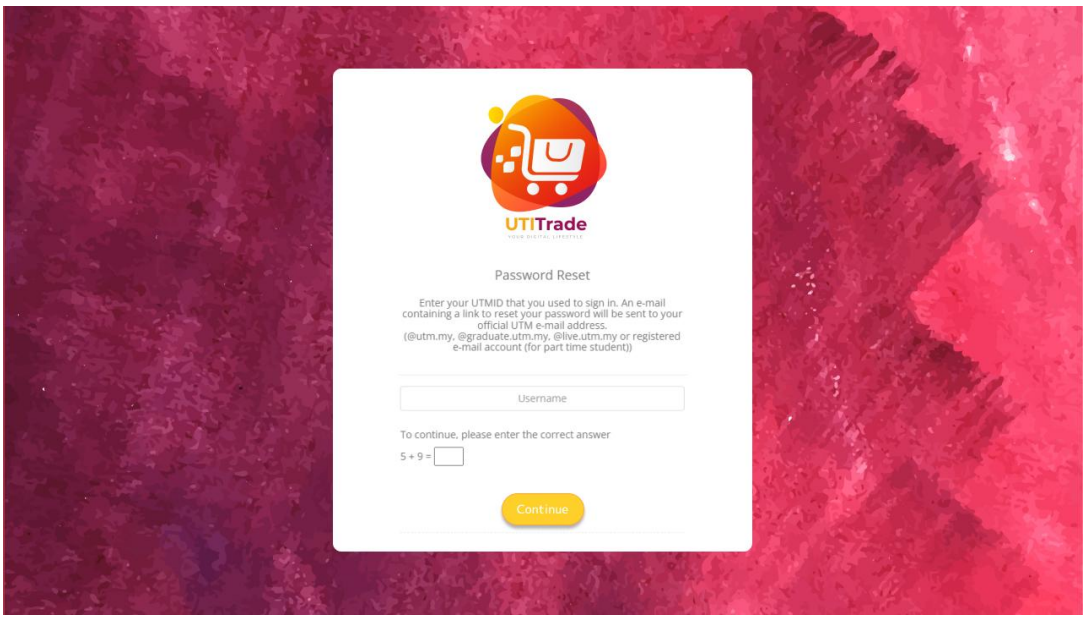


Figure 3.5: Interface Design of Forget Password Page

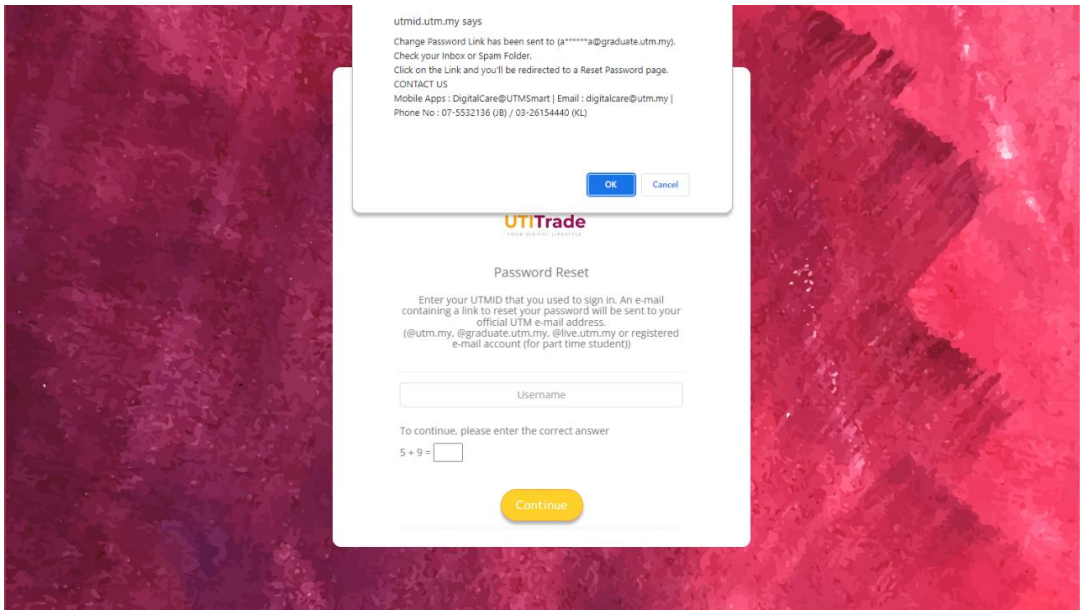


Figure 3.6: Interface Design of Forget Password Page 2

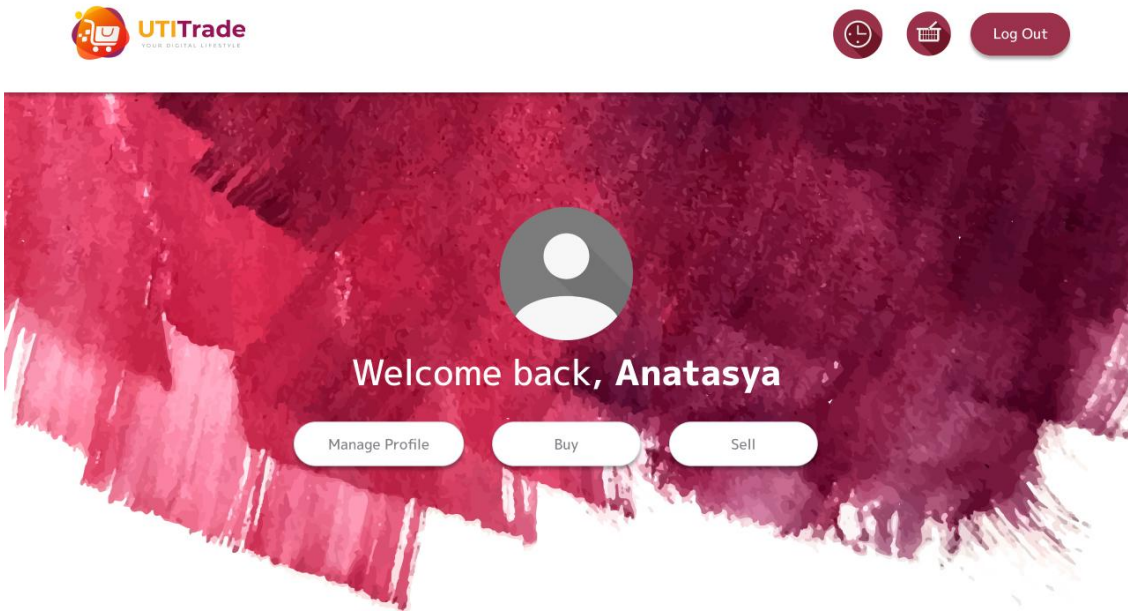


Figure 3.7: Interface Design of Homepage

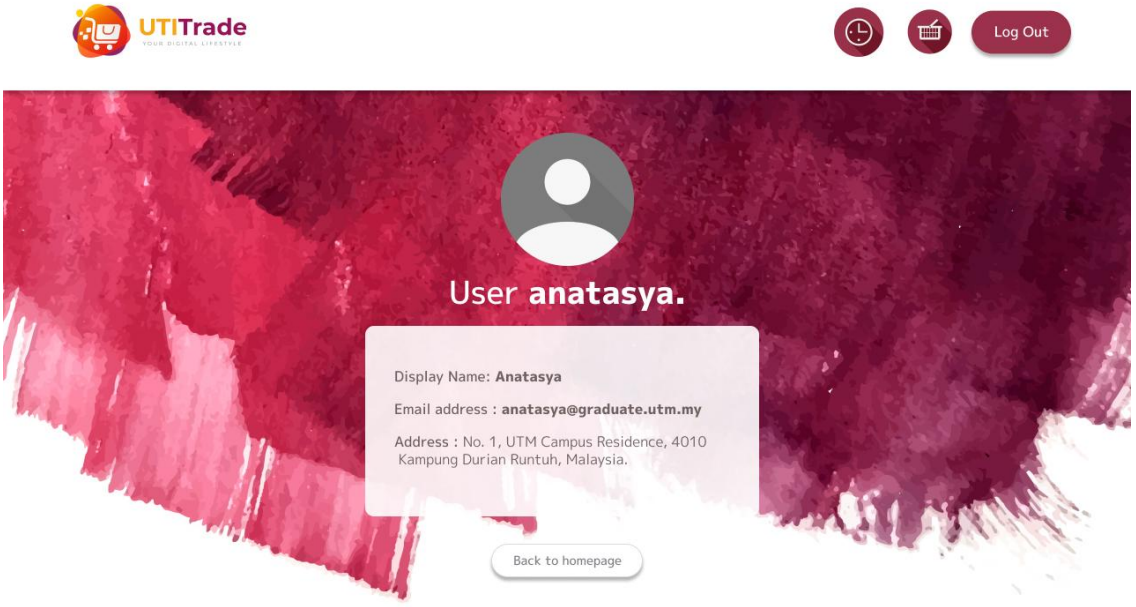


Figure 3.6: Interface Design of Profile Page

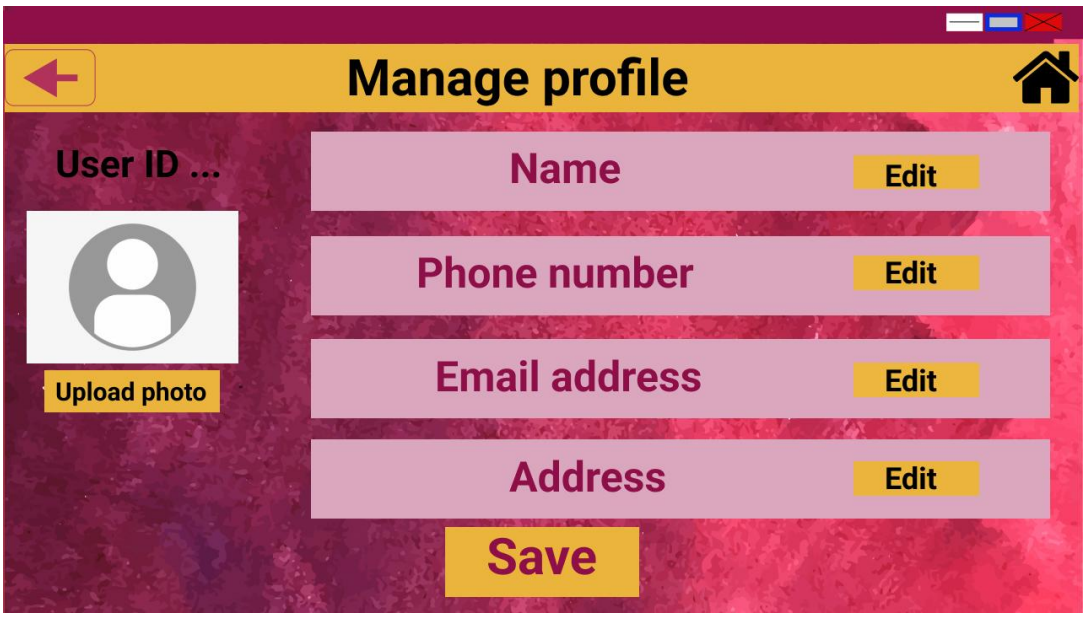


Figure 3.7: Interface Design of Manage Profile

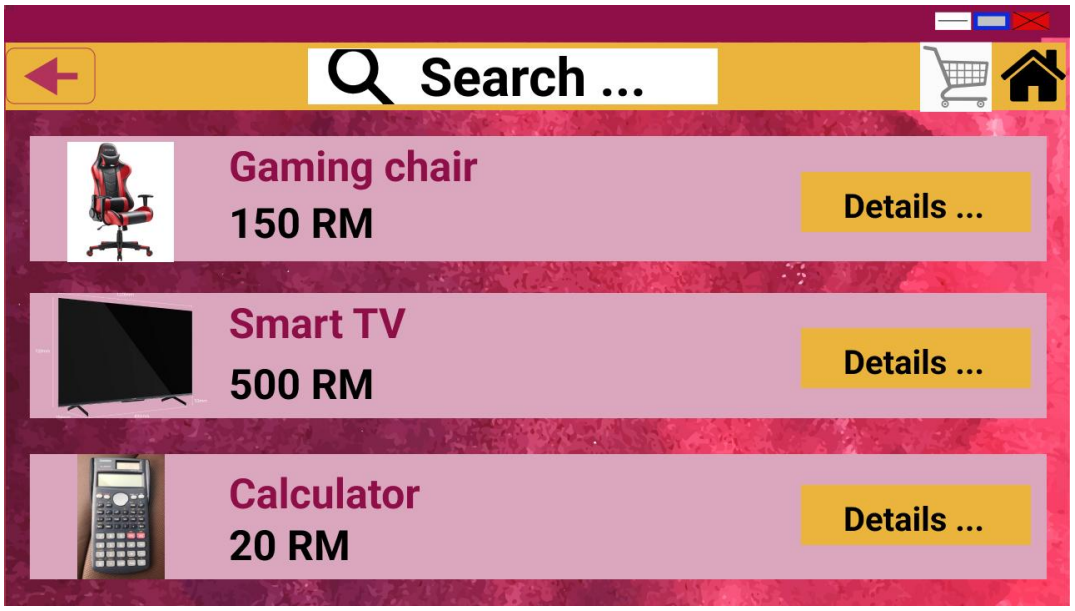


Figure 3.8: Interface Design of Product Catalog Page

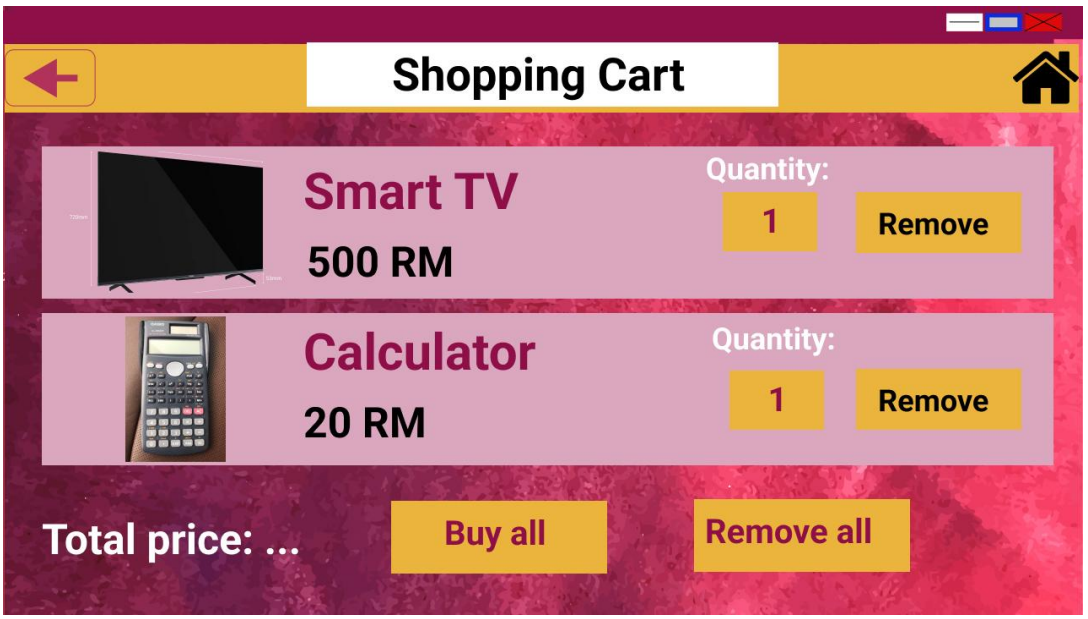


Figure 3.9: Interface Design of Cart Page

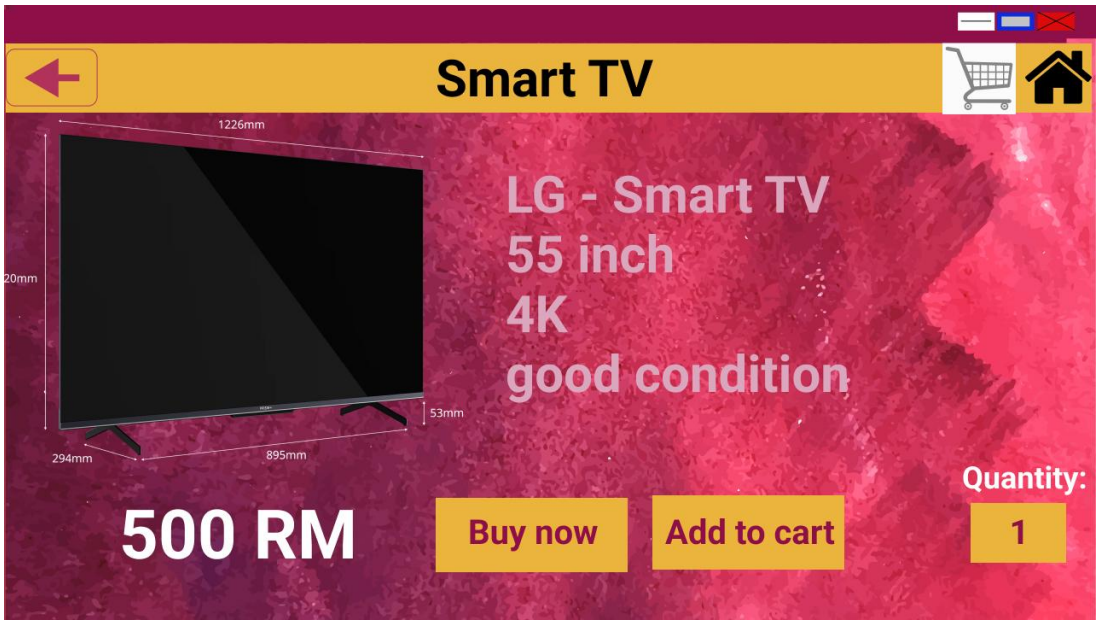


Figure 4.0: Interface Design of Product Detail Page

3.4. Test Cases

3.4.1. TC001: Test <Authentication> Subsystem

This test contains the following test cases:

(a) TC001_01: Test <Scenario of User Registration (SD001)>

(b) TC001_02: Test <Scenario of User Login (SD002)>

(c) TC001_03: Test <Scenario of Reset Password (SD003)>

(d) TC001_04: Test <Scenario of Manage Profile (SD004)>

3.4.2. TC001_01: Test <Scenario of User Registration (SD001)>

Test Case ID	TC001_01	Test Case Description	Test the Register Functionality		
Created By	Anatasya	Reviewed By	Anatasya	Version	2.1
QA Tester's Log	Review comments from Deia incorprate in version 2.1				
Tester's Name	Anatasya	Date Tested	October 9, 2023	Test Case (Pass/Fail/Not	Pass

S #	Prerequisites:
1	The user must be registered as an active UTM student.
2	The user must not already be registered to the system.
3	
4	

S #	Test Data
1	UTM Email = anatasya@graduate.utm.my
2	UTMID = anatasya
3	Password = ath12345
4	

Test Scenario: Verify on entering valid UTM email address, UTMID and password, the user can login

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	User navigates to the registration page.	UtiTrade register page should open	As Expected	Pass
2	Fill in UTMID, UTM email address and password.	Credential can be entered	As Expected	Pass
3	Click Register	User is registered in	As Expected	Pass

3.4.3. TC001_02: Test <Scenario of User Login (SD002)>

Test Case ID	TC001_02	Test Case Description	Test the Login Functionality		
Created By	Anatasya	Reviewed By	Anatasya	Version	2.1
QA Tester's Log	Review comments from Deia incorprate in version 2.1				
Tester's Name	Anatasya	Date Tested	October 9, 2023	Test Case (Pass/Fail/Not	Pass

S #	Prerequisites:
1	The user must be registered as an active UTM student.
2	The user must be registered to the system.
3	The user must input the registered UTMID and password.
4	

S #	Test Data
1	UTMID = anatasya
2	Password = ath12345
3	
4	

Test Scenario: Verify on entering valid UTMID and password, the user can login

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	The system will redirect user to the login page.	UtiTrade login page should open	As Expected	Pass
2	Fill in UTMID and password.	Credential can be entered	As Expected	Pass
3	Click Login	User is logged in	As Expected	Pass

3.4.4. TC001_03: Test <Scenario of Reset Password (SD003)>

Created By	Anatasya	Reviewed By	Anatasya	Version	2.1
QA Tester's Log	Review comments from Deia incorprate in version 2.1				
Tester's Name	Anatasya	Date Tested	October 9, 2023	Test Case (Pass/Fail/Not	Pass

S#	Prerequisites:	S#	Test Data
1	The user must be registered as an active UTM student.	1	username = anatasya
2	The user must be registered to the system.	2	Authentication = 3+3 = 6
3		3	
4		4	

Test Scenario	Verify on entering valid UTMID as username and the authentication number, the user can reset their password through email

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click "forget password" from the system	Reset password page should open	As Expected	Pass
2	Fill in UTMID and authentication number.	Credential can be entered	As Expected	Pass
3	Click continue	User redirected to their email address	As Expected	Pass
4	Open email and proceed the verification	Verification is successful	As Expected	Pass
5	Refresh UtiTrade page	UtiTrade dashboard should refreshed and reopen	As Expected	Pass
6	Fill in UTMID and password	Credential can be entered	As Expected	Pass

3.4.5. TC002: Test <Purchasing Product> Subsystem

This test contains the following test cases:

(a) TC002_01: Test <Scenario of Add Product Cart (SD005)>

4. Module 2

4.1. Specific Requirement

4.1.1. User characteristics

Below are the list of different user characteristics and roles within module 2 known as the purchasing product subsystem:

4.1.1.1. UTM Students (as buyer)

- Students are expected to still be registered as an active student in UTM.
- Users are required to have already reached a purchasing decision and subsequently proceed to the payment phase.
- Each user possesses the autonomy to select a preferred payment method based on personal preference.
- Students might need an interface that is user-friendly and simple to learn since one of the main purposes of the system is to make transactions easier.
- Some users may opt for cash as their primary payment method, notably those who do not possess debit or credit cards.
- Additionally, certain users who are unable to execute direct transactions may opt to employ a card-based payment method instead.

4.1.1.2. UTM Students (as seller)

- Students are expected to still be registered as an active student in UTM.
- Students may have spare time toward engaging in e-commerce activities such as by selling their unwanted products, thereby fostering productivity and accruing valuable experience as online vendors.
- Some students might graduate soon, thus they might want to sell their products at a cheaper price to the other UTM students in need.
- Users possess a range of specific products or superfluous items, which they intend to market to fellow students at preferential pricing.

4.1.2. System Features

For module two, the system features include:

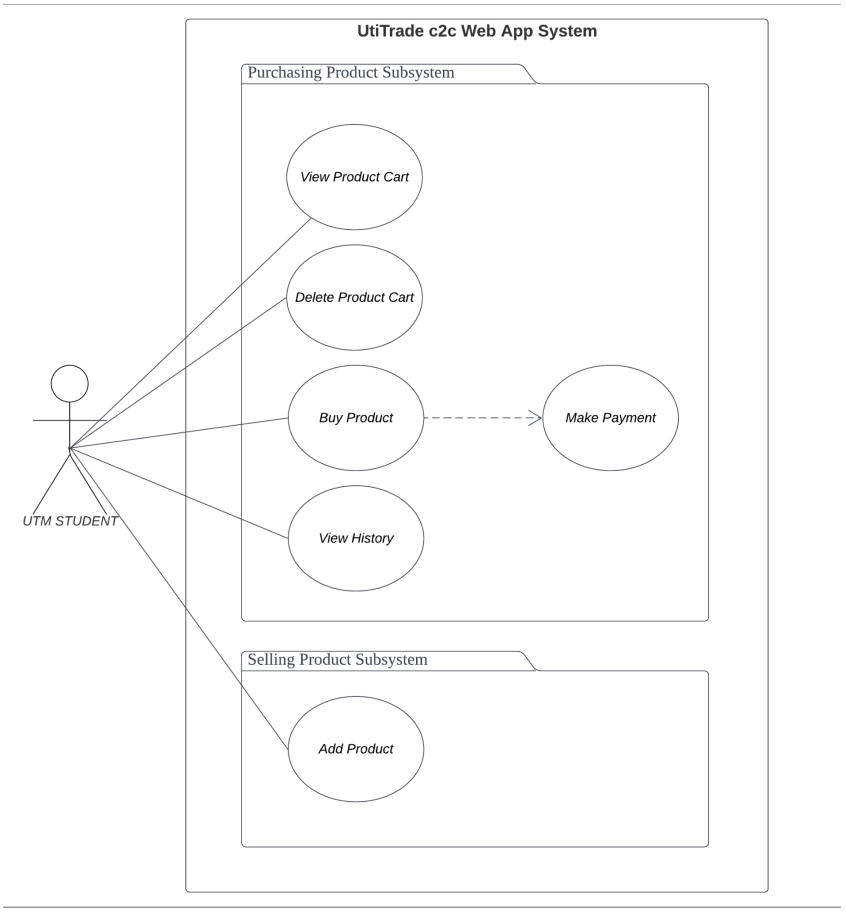


Figure 5.0 : Use Case Diagram for Module 2

4.1.3. UC006: Use Case <View Product Cart> & Seq Diagram

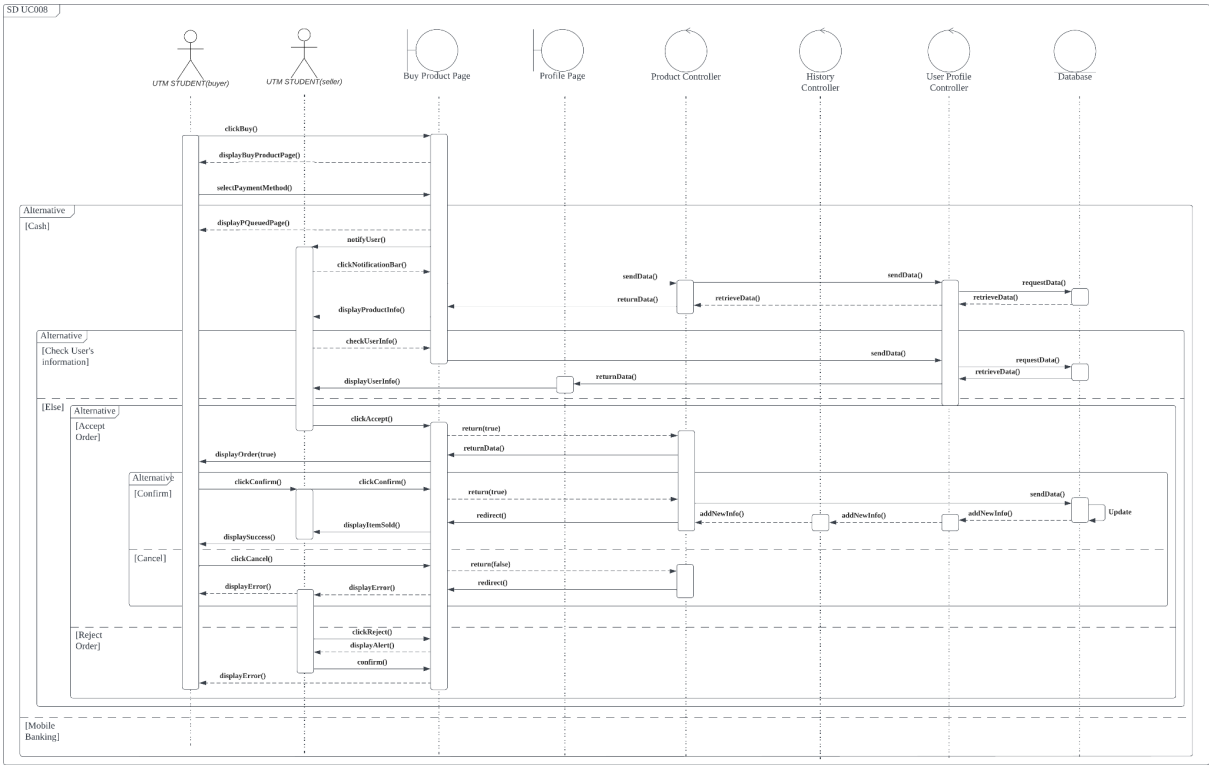
4.1.4. UC007: Use Case <Delete Product Cart> & Seq Diagram

4.1.5. UC008: Use Case <Buy Product> & Seq Diagram

Table 4.1.5: Use Case Description for <buy product>

Use case: <buy product>
ID: UC008
Actors: UTM Student
Preconditions: 1. The user must be registered as an active UTM student. 2. The user must be logged in to the system.
Main flow : 1. User clicks 'buy' from the product page. 2. System redirects the user to the buy product page. 3. The user chooses a payment method, cash. 4. System pops up an alert, confirmation box. 5. User taps ok from the confirmation box. 6. The system notifies the seller. 7. Seller clicks on the notification board and selects the bar. 8. System displays the user and the product they want to buy. 9. Seller accepts the payment method as well as the order. 10. System displays the product on hold to the seller's interface. 11. System update and display buyer's UI, indicates the order has been placed. 12. Seller and buyer managed to communicate as well as receive the ordered item(s) outside the system. 13. Both buyer and seller click on 'confirm order' on each device. 14. System displays the successful transaction to each device. 15. Use Case ends.
Alternate flow : 1. System displays the user and the product they want to buy. 2. Seller taps on 'check user information'. 3. System displays the user's profile page which consists of name, email address, physical address.
Exception flow (if any): 1. Order is cancelled. 2. Seller rejects the order request.

Figure 6.0: SD008: Sequence Diagram for <buy product>



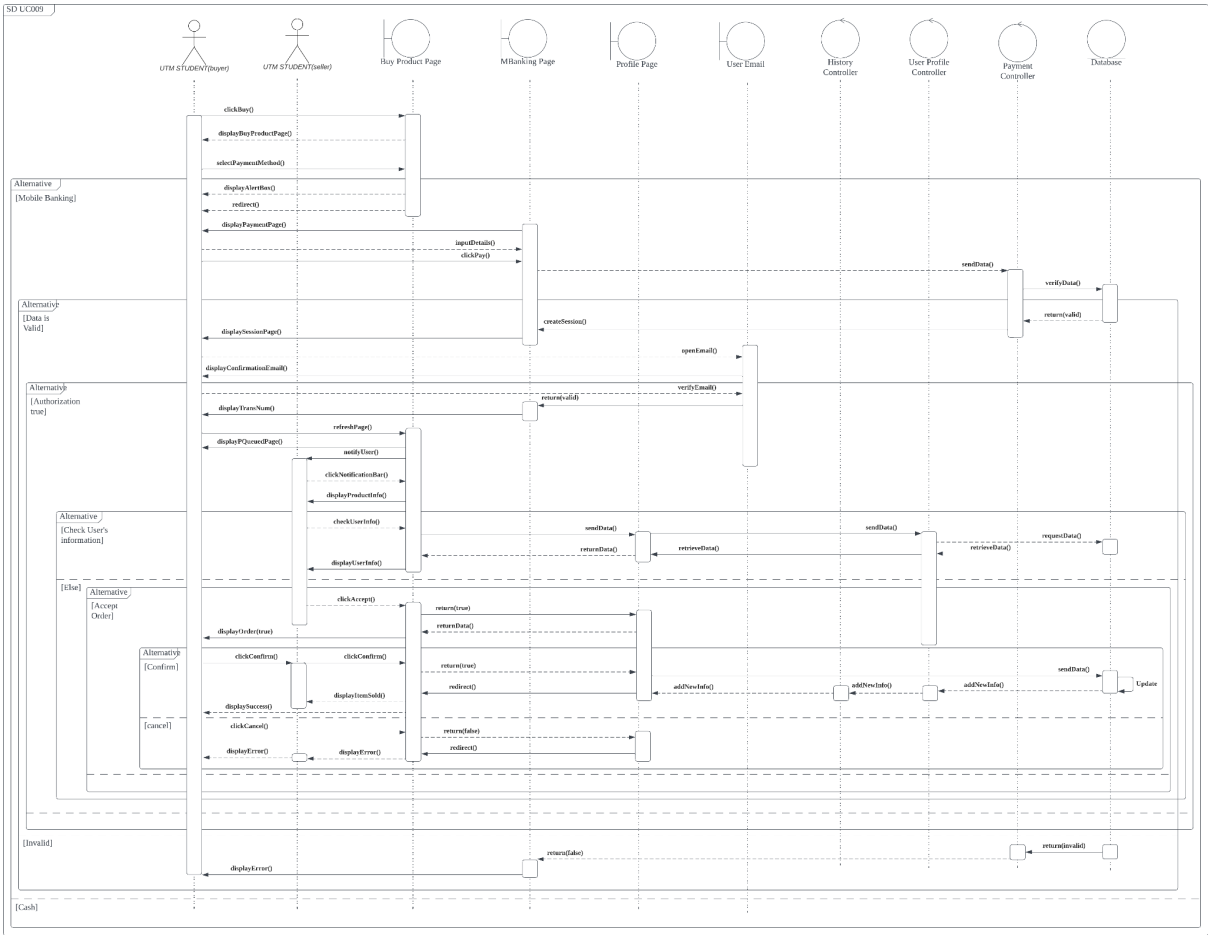
4.1.6. UC009: Use Case <Make Payment> & Seq Diagram

Table 4.1.6: Use Case Description for <Make Payment>

Use case: <make payment>
ID: UC008
Actors: UTM Student
Preconditions: 1. The user must be registered as an active UTM student. 2. The user must be logged in to the system.
Main flow : 1. User clicks 'buy' from the product page. 2. System redirects the user to the buy product page. 3. The user chooses a payment method, online banking. 4. System pops up an alert, confirmation box. 5. User taps ok from the confirmation box. 6. System redirects the user to an online payment gateway and displays the page. 7. User fills up information credentials needed to proceed the payment.

8. The system pops up a confirmation box. 9. User clicks the 'ok' button. 10. System creates a session for the user to complete their authorization process through email. 11. User shall go to their email and confirm the order. 12. The system redirects and displays a successful transaction page. 13. The system redirects the user to the queued request page. 14. . The system notifies the seller. 15. Seller clicks on the notification board and selects the bar. 16. System displays the user and the product they have ordered. 17. Seller accepts the payment method as well as the order. 18. System displays the product on hold to the seller's interface. 19. System update and display buyer's UI, indicates the order has been placed. 20. Seller and buyer managed to communicate as well as receive the ordered item(s) outside the system. 21. Both buyer and seller click on 'confirm order' on each device. 22. System displays the successful transaction to each device. 23. Use Case ends.
Alternate flow : 1. System displays the user and the product they want to buy. 2. Seller taps on 'check user information'. 3. System displays the user's profile page which consists of name, email address, physical address.
Exception flow (if any): 1. Order is cancelled. 2. Authentication fails, payment is unsuccessful. 3. Session is destroyed.

Figure 6.1: SD009: Sequence Diagram for <buy product>



UC010: USE CASE <View History> & SEQ DIAGRAM

UC011: USE CASE <Add Product> & SEQ DIAGRAM

4.2. Data Design

There are 7 entities in the database design of the UtiTrade system during this sprint : User DB, Product DB, Cart DB, ProductAvail DB, ProductSold DB, BankAccount DB, History DB. Figure 4.2.1 below represents the database design of the UtiTrade system whereas Table 4.2.1.1 represents the description of the entity.

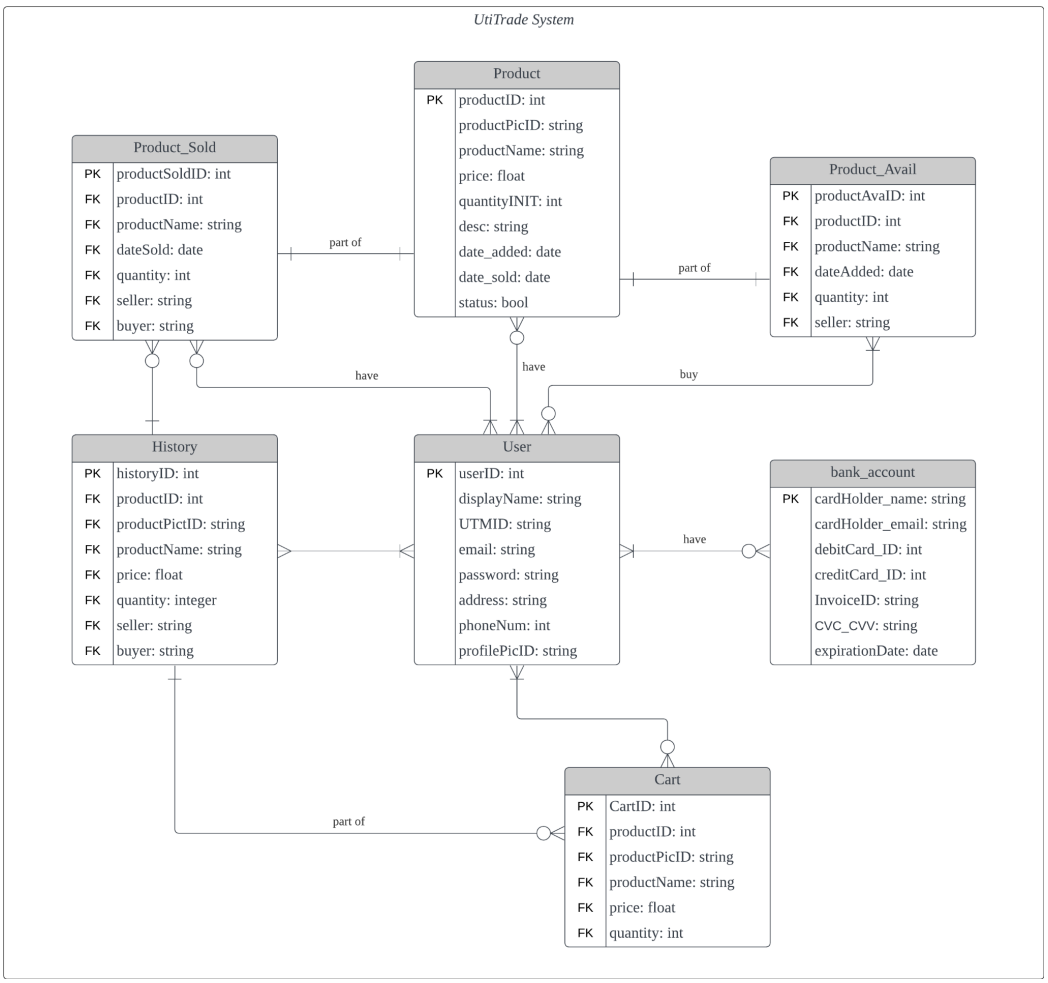


Figure 4.2.1: Entity Relationship Diagram of UtiTrade System.

4.2.1. Data Dictionary & Description

Table 4.2.1.1: Entity Description

Entity	Description
User	Contains the detail information of user
Bank_account	Contains the detail information of user's bank account
History	Contains the detail information of user's transaction as well as notification
Product	Contains the detail information of a certain product
Product_Avail	Contains the detail information of available products
Product_Sold	Contains the detail information of sold out products
Cart	Contains the detail information of a product of the user's cart.

Data Dictionary

4.2.1.1. Entity: <User>

Attributes	Type	Description
userID	integer	This attribute serves as a unique identifier for each user in the system. It is used for referencing and distinguishing one user from another.
displayName	string	The displayName attribute stores the user's chosen name and can be edited. It allows users to personalize their profiles and how they appear to others.
UTMID	string	UTMID is an attribute that stores a unique identifier associated with the user's university ID.
email	string	The email attribute stores the user's email address, which is essential for reset password, account authentication, and account confirmation.
password	string	The password attribute stores a securely hashed or encrypted version of the user's password.
address	string	The address attribute stores the user's physical address. which may be used for shipping or location-based services.
phoneNum	integer	It stores the user's contact number. It can be useful for communication.
profilePictID	string	This attribute serves as a reference to the user's profile picture or avatar image.

4.2.1.2. Entity: <Product>

Attributes	Type	Description
productID	integer	This attribute is a unique identifier for each product in the system. It is used to distinguish one product from another and is crucial for database operations.
productPictID	string	The productPictID attribute serves as a reference to the product's main image or picture. Instead of storing the actual image data within the product entity, it links to an image stored elsewhere in the system.
productName	string	productName stores the name or title of the product.
price	float	The price attribute stores the monetary value associated with the product. It indicates the cost of purchasing the item.
quantityInit	integer	This attribute represents the initial quantity of the product available for sale.
desc	string	The desc attribute holds a detailed description of the product. It provides additional information about the item's features, specifications, condition and any other relevant details.

status	bool	The status attribute is of type 'bool' (boolean) and is used to indicate the availability or status of the product listing. A 'true' value typically indicates that the product is available for sale, while 'false' may indicate that the product has been sold or is no longer available.
dateAdded	date	As reference from the Product Avail entity, it represents the date and time when the product listing was created or added to the platform.
dateSold	date	As reference from the Product Sold entity, it represents the date and time when the product listing was sold to the buyer.

4.2.1.3. Entity: <Product_Avail>

Attributes	**Type**	**Description**
productAvailID	integer	A unique identifier for each available product in the system.
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
seller	string	As reference from the User entity. This attribute serves as a unique identifier for each user that sells the particular item in the system.
productName	string	As reference from the Product entity, it stores the name or title of the product. It provides a clear and concise description of the item and is displayed prominently in product listings.
dateAdded	date	It represents the date and time when the product listing was created or added to the platform.
quantity	integer	It represents the quantity of the product available for sale in the listing. As products are sold, this quantity may decrease to reflect the number of items still available for purchase.

4.2.1.4. Entity: <Product_Sold>

Attributes	**Type**	**Description**
productSoldID	integer	A unique identifier for each sold item in the system.
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
seller	string	As reference from the User entity. This attribute serves as a unique identifier for each user that sells the particular item in the system.
buyer	string	As reference from the User entity. This attribute serves as a

		unique identifier for each user that has purchased the particular item in the system.
productName	string	As reference from the Product entity, it stores the name or title of the product. It provides a clear and concise description of the item and is displayed prominently in product listings.
dateSold	date	It represents the date and time when the product listing was sold to the buyer.
quantity	integer	It represents the quantity of the sold product that is no longer available for purchase.

4.2.1.5. Entity: <bank_account>

Attributes	Type	Description
cardHolder_name	string	This attribute stores the name of the individual or entity to whom the credit or debit card is issued.
cardHolder_email	string	The email address attribute stores the contact email associated with the bank account. It is used for communication with the account holder, sending transaction notifications, and account-related correspondence.
debitCard_ID	integer	Similar to a credit card, this attribute stores the unique identification number of a debit card issued to an account holder. Debit cards are linked to a bank account and are used for making payments and withdrawals directly from the account.
creditCard_ID	integer	This attribute stores the unique identification number of a credit card issued to an account holder.
InvoiceID	string	The Invoice ID attribute is used to link a transaction to a specific invoice or bill. It helps in associating the transaction with the corresponding financial record or order.
CVC_CVV	integer	The Card Verification Code (CVC) or Card Verification Value (CVV) is a security code printed on the back of a credit or debit card. It is used to verify that the cardholder possesses the physical card during online or card-not-present transactions, adding an extra layer of security.
expirationDate	date	This attribute represents the date until which the credit or debit card is valid. After this date, the card cannot be used for transactions, and a new card is typically issued.

4.2.1.6. Entity: <cart>

Attributes	Type	Description
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
productPictID	string	As reference from the Product entity, the productPictID attribute serves as a reference to the product's main image or picture.
productName	string	As reference from the Product entity, it stores the name or title of the product.
price	float	As reference from the Product entity, the price attribute stores the monetary value associated with the product.
quantity	integer	It represents the quantity of the product in the cart. Users may edit or remove the quantity according to their needs.

4.2.1.7. Entity: <history>

Attributes	Type	Description
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
productPictID	string	As reference from the Product entity, the productPictID attribute serves as a reference to the product's main image or picture.
productName	string	As reference from the Product entity, it stores the name or title of the product.
price	float	As reference from the Product entity, the price attribute stores the monetary value associated with the product.
quantity	integer	It represents the quantity of the product in the cart. Users may edit or remove the quantity according to their needs.
seller	string	As reference from the User entity. This attribute serves as a unique identifier for each user that sells the particular item in the system.
buyer	string	As reference from the User entity. This attribute serves as a unique identifier for each user that has purchased the particular item in the system.
dateAdded	date	As reference from the Product Avail entity, it represents the date and time when the product listing was created or added to the platform.
dateSold	date	As reference from the Product Sold entity, it represents the date and time when the product listing was sold to the buyer.
transactionNum	int	As reference from the Product Sold entity, it represents the date and time when the product listing was sold to the buyer.

status	bool	The Status attribute indicates the outcome of a transaction. It can have one of two values: "failed" or "successful."

4.3. User Interface Design

During iteration two, there are various pages of interface design that our group managed to create by using a design tool called *Figma*. The system provides two functionality dashboards which allows the user to be a buyer as well as be a seller at the same time. As for the iteration two, the function is emphasising more on the buyer perspective which allows the user to be roled as the buyer as shown below:

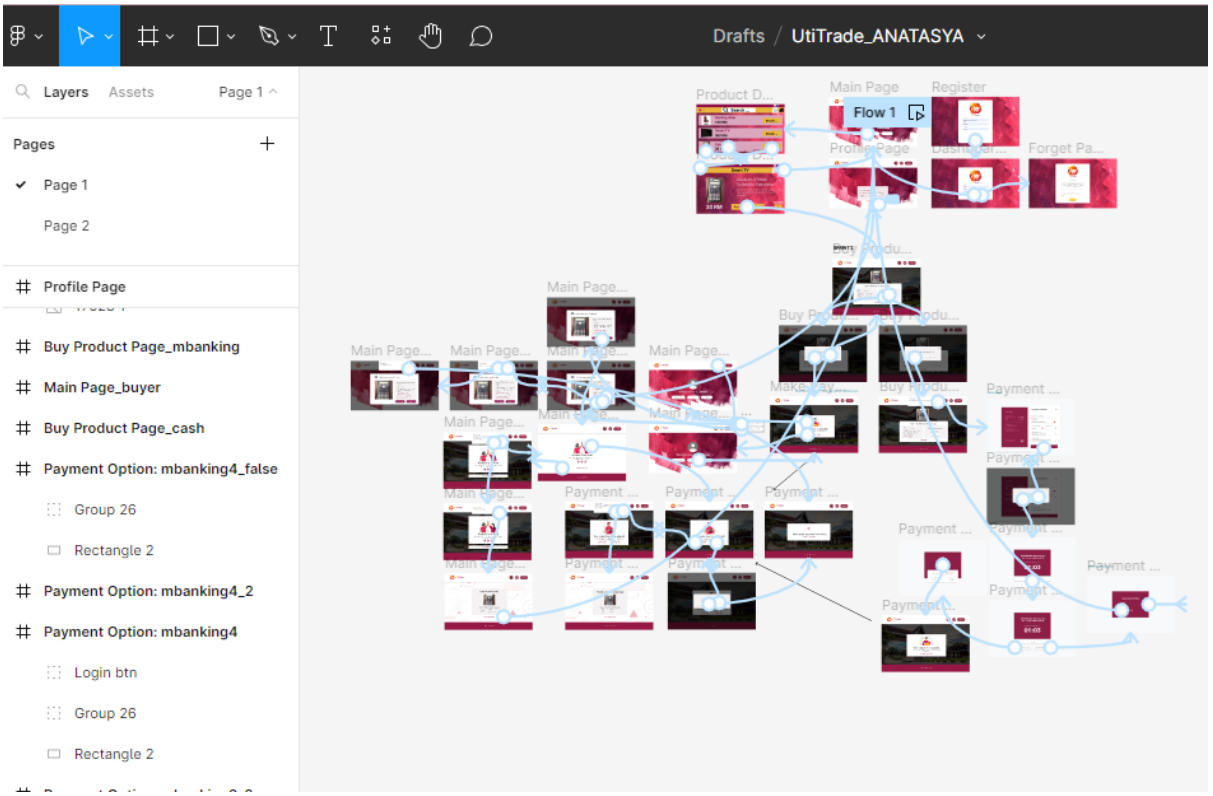


Figure 4.3.1 : Overall Interface Design for <buy product> and <make payment>

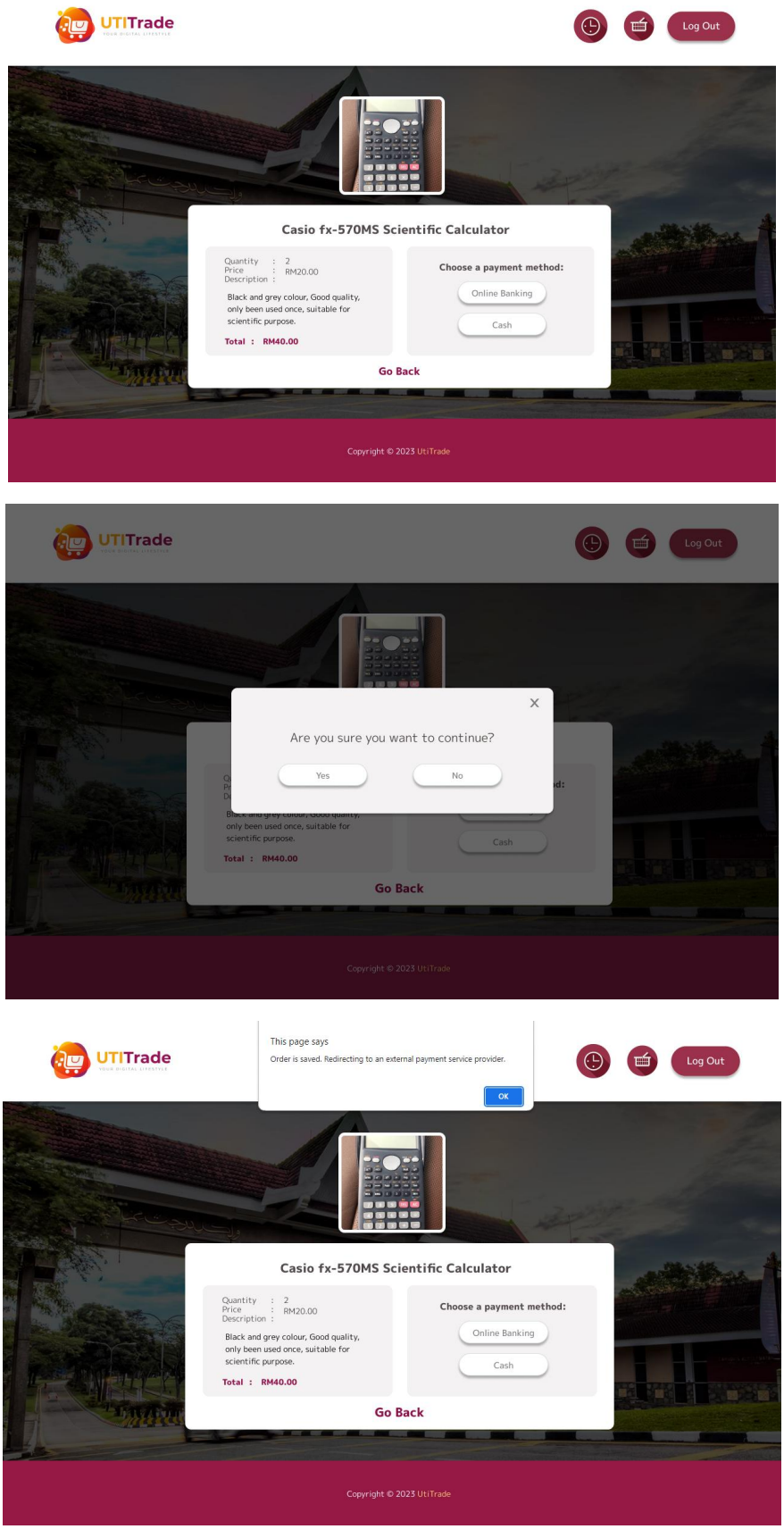


Figure 4.3.2, 4.3.3, 4.3.4 : Interface Design of Buy Product Page

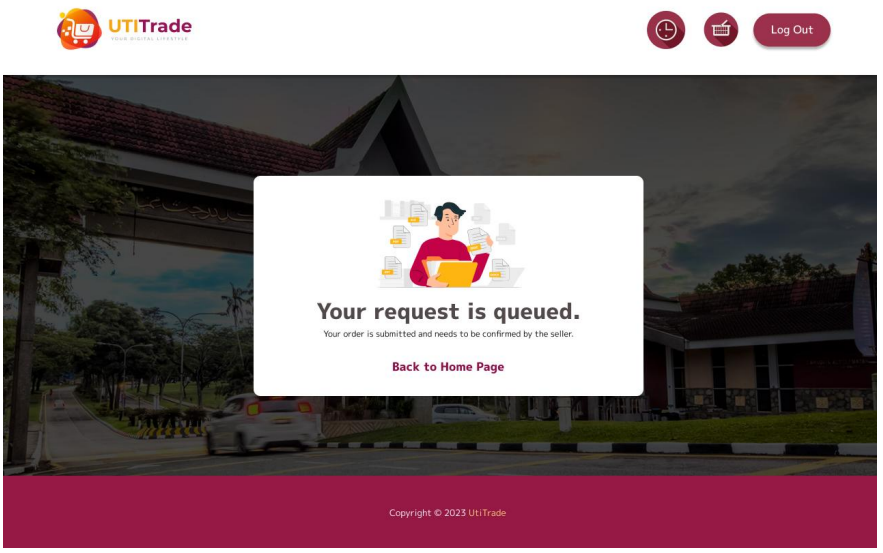


Figure 4.3.5 : Interface Design of Queue Cash Method Page

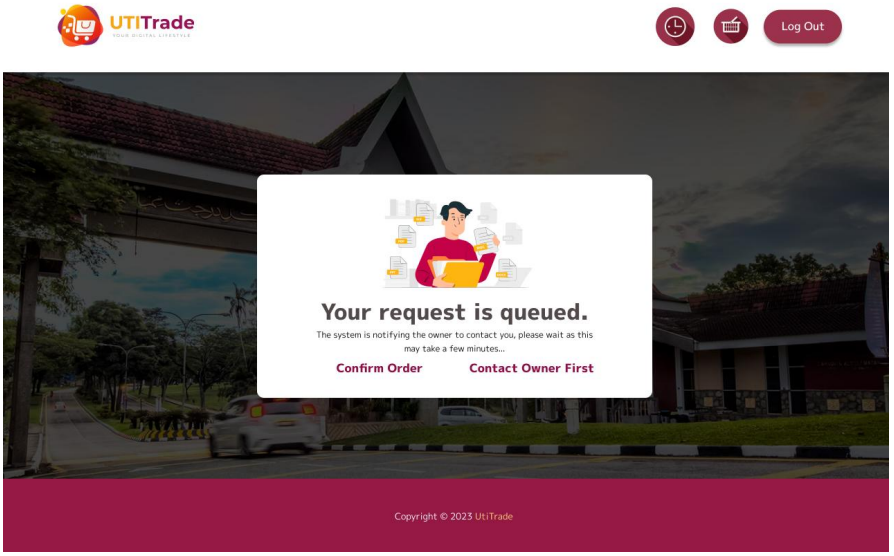
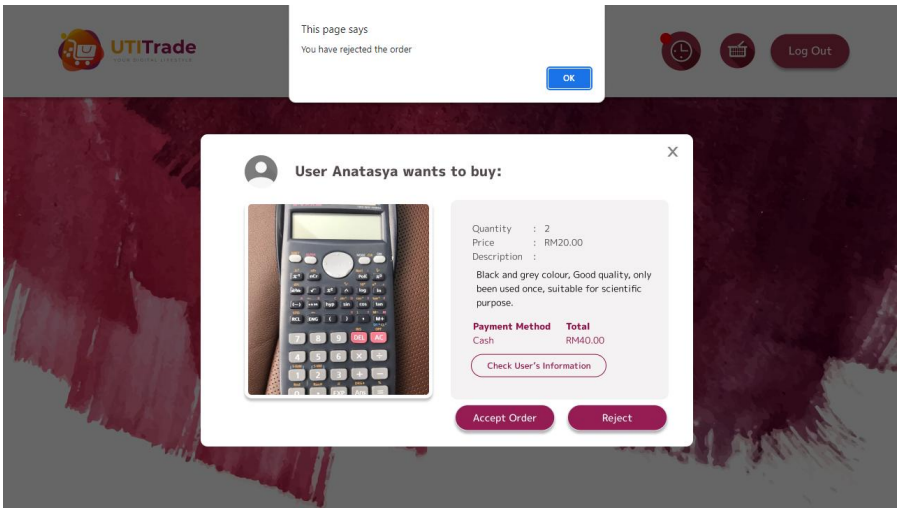


Figure 4.3.6 : Interface Design of Online Payment Method Page



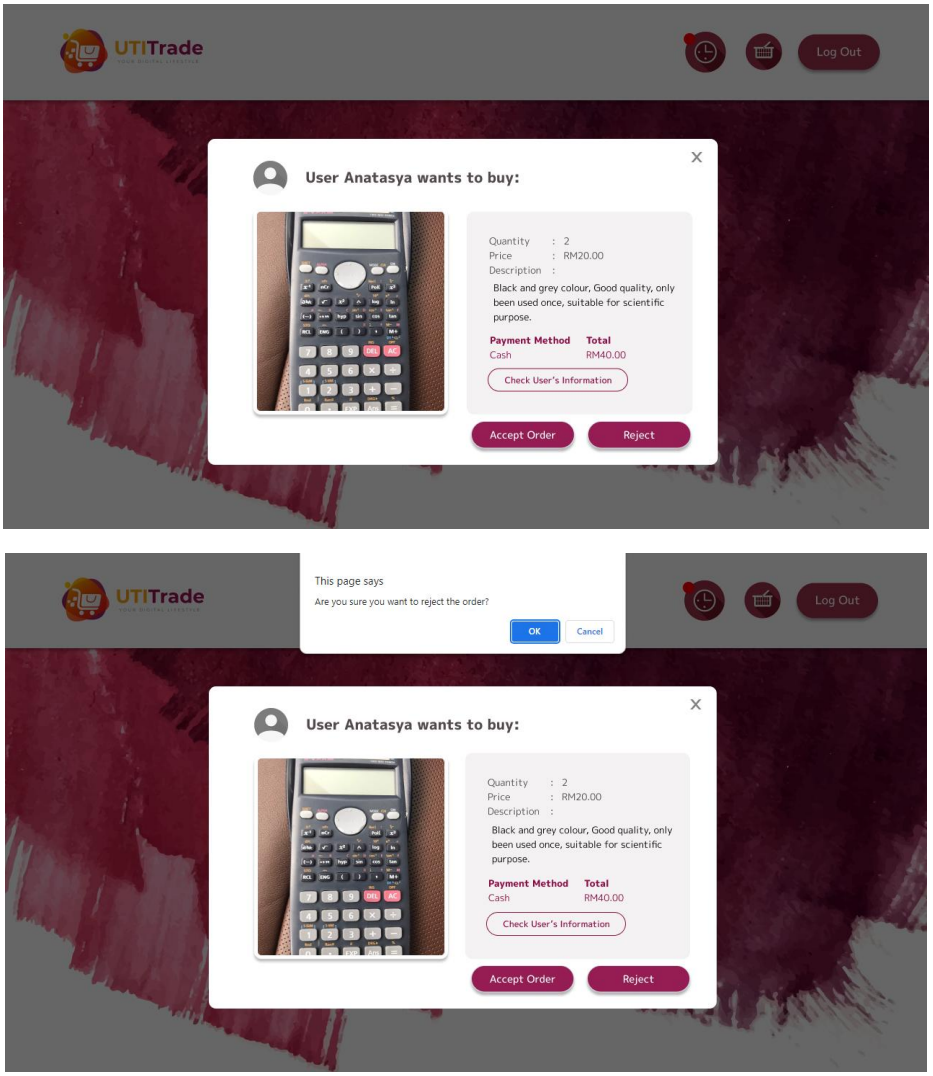


Figure 4.3.7, 4.3.8, 4.3.9 : Interface Design of Buy Product(seller)_cash Page

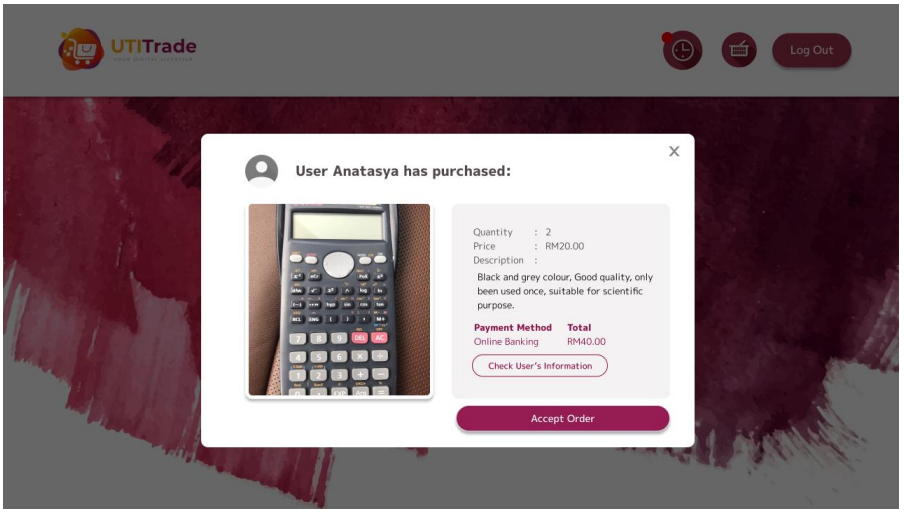
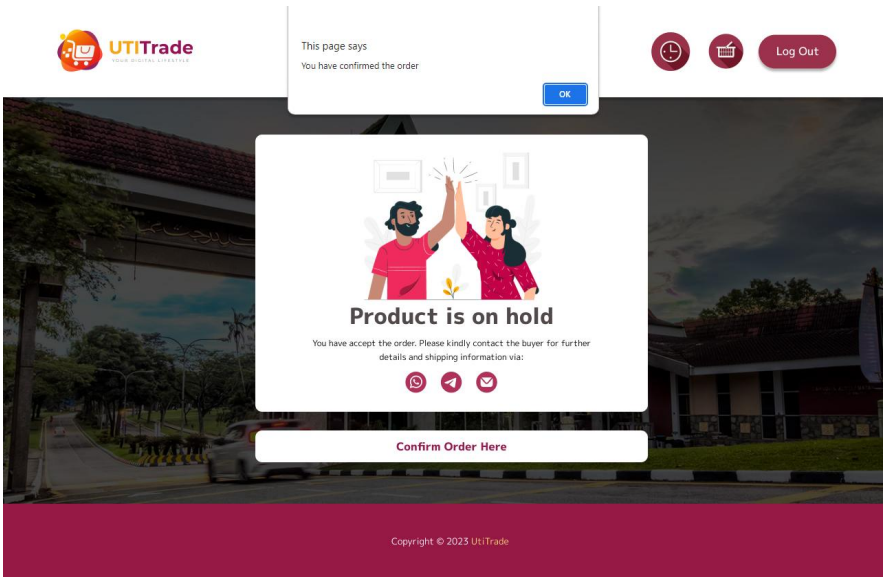
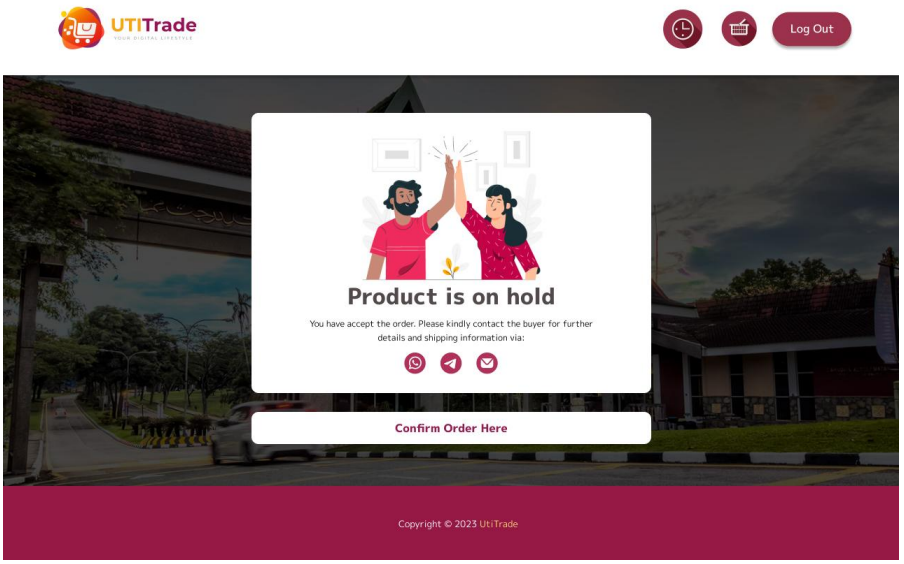


Figure 4.4.0 : Interface Design of Buy Product(seller)_mbanking Page





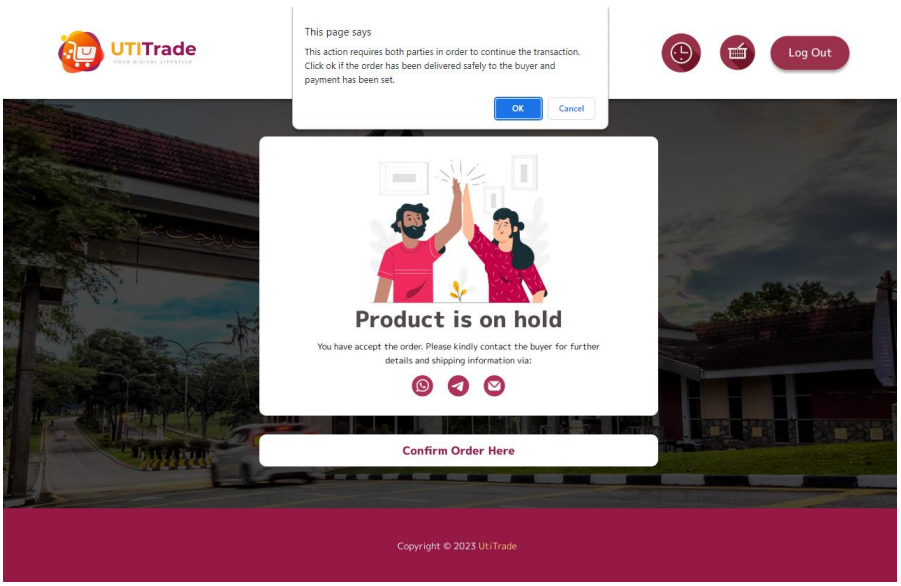


Figure 4.4.1, 4.4.2, 4.4.3 : Interface Design of On Hold Product(seller) Page

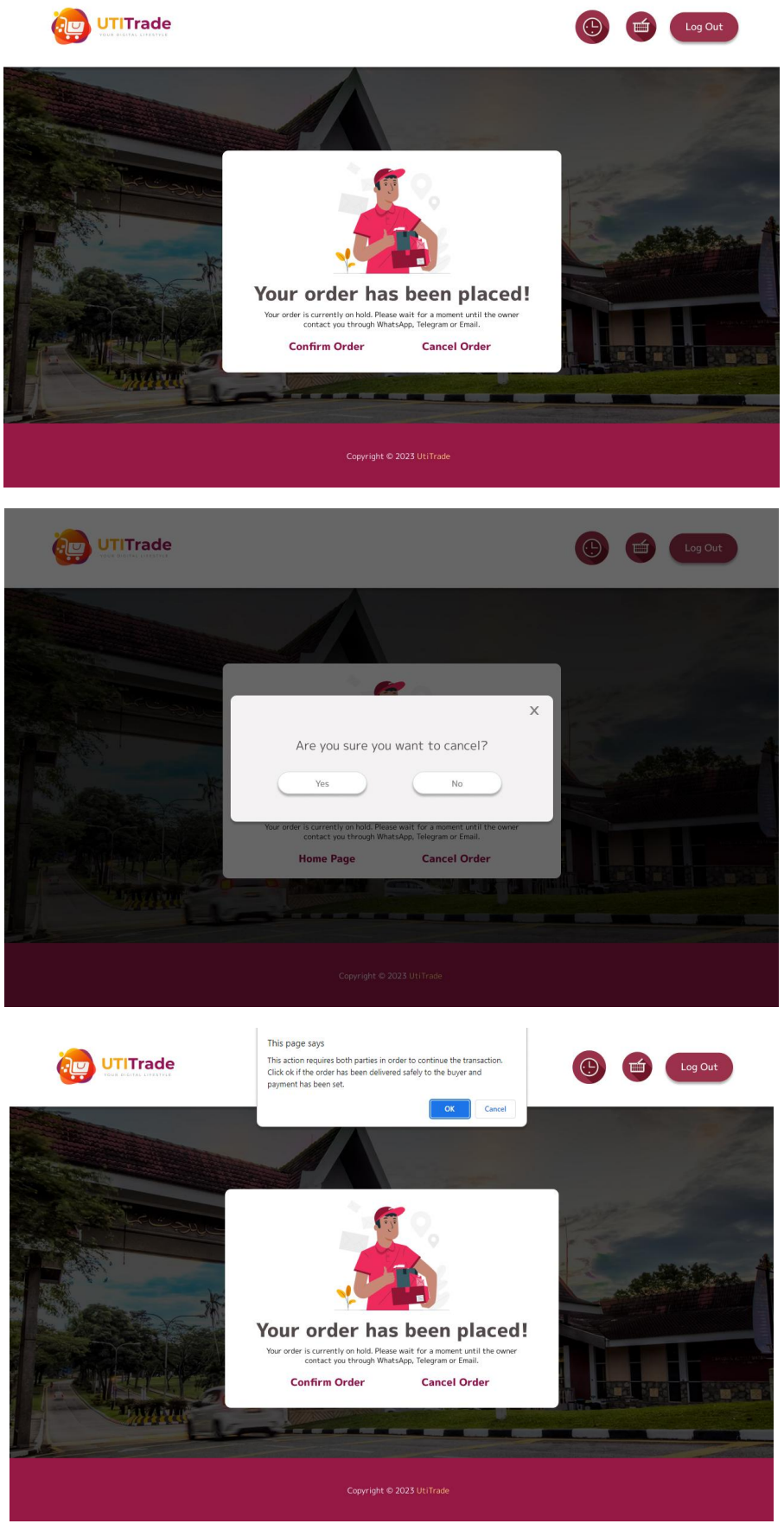


Figure 4.4.4, 4.4.5, 4.4.6 : Interface Design of Place Order Page

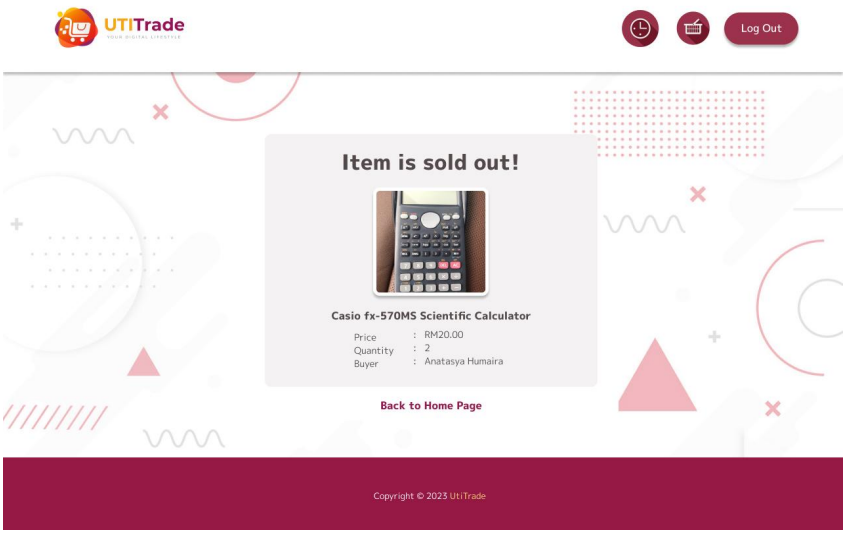


Figure 4.4.7 : Interface Design of Product Sold(seller) Page

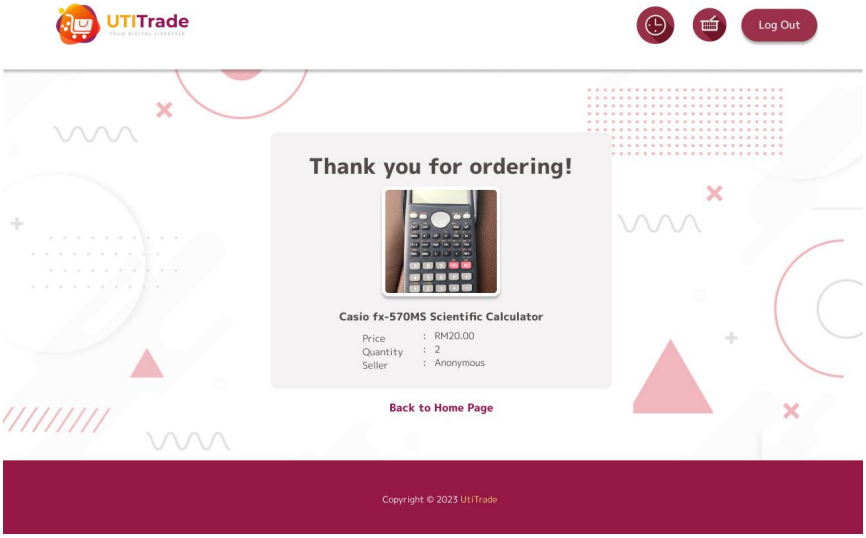


Figure 4.4.8 : Interface Design of Product Sold(buyer) Page

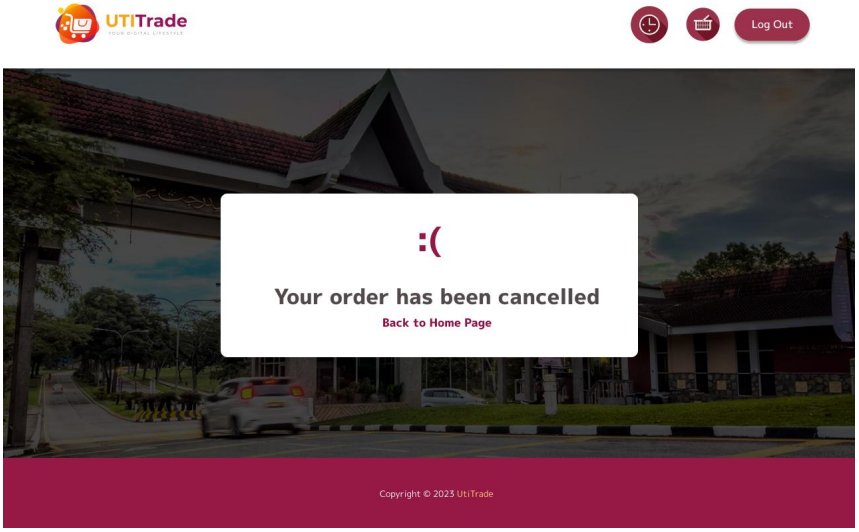
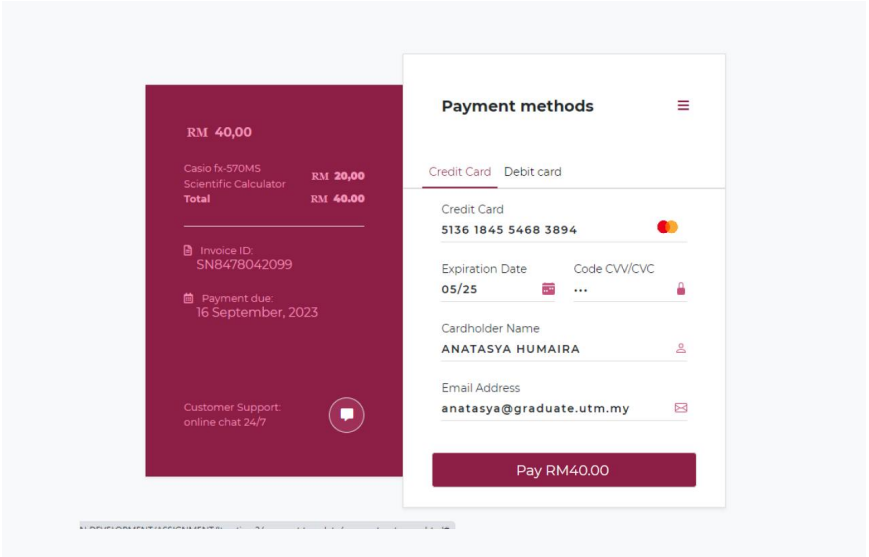


Figure 4.4.9 : Interface Design of Order Cancelled Page



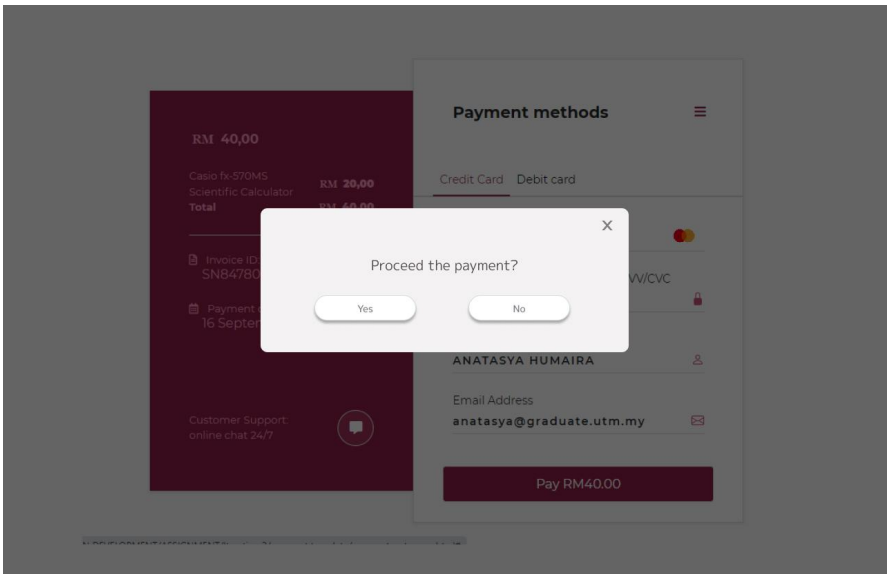


Figure 4.5.0, 4.5.1 : Interface Design of Payment Gateway Page

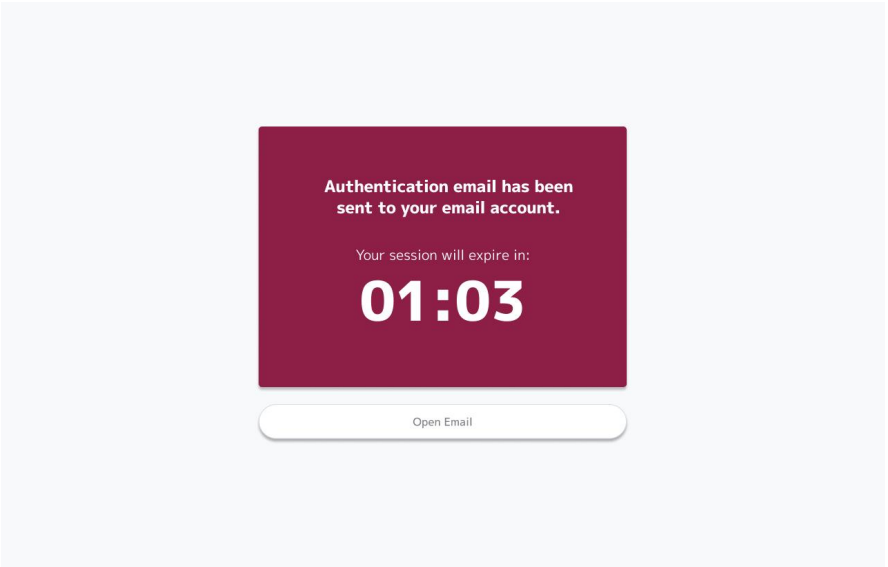


Figure 4.5.2 : Interface Design of Payment Session

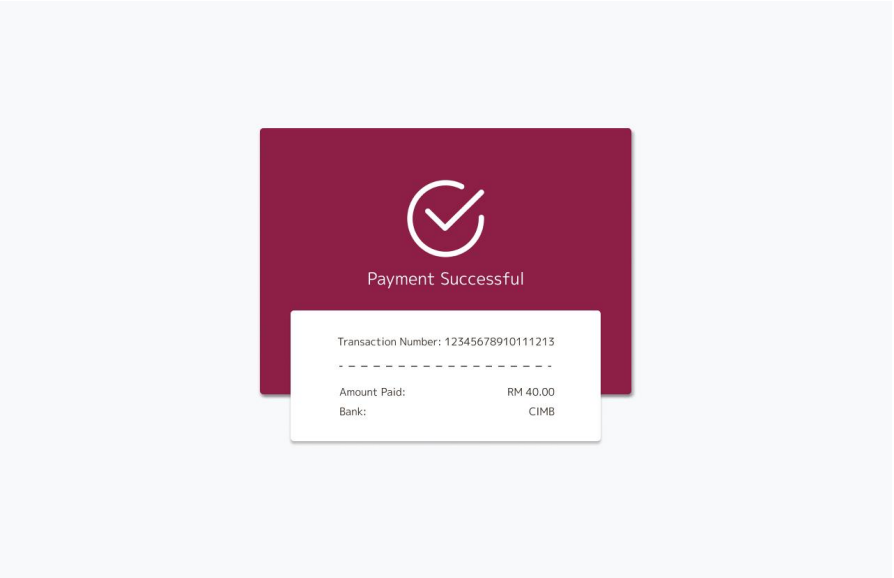


Figure 4.5.3 : Interface Design of Successful Payment Page

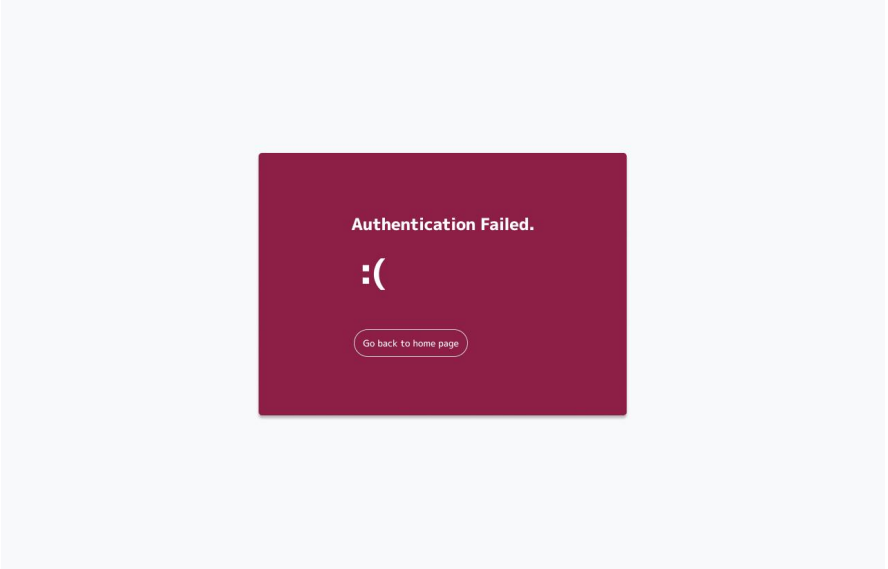


Figure 4.5.4 : Interface Design of Failed Payment Page

4.4. Test Cases

4.4.1. TC002: Test <Purchasing Product> Subsystem

This test contains the following test cases:

(a) TC002_02: Test <Scenario of View Product Cart (SD006)>

(b) TC002_03: Test <Scenario of Delete Product Cart (SD007)>

(c) TC002_04: Test <Scenario of Buy Product (SD008)>

(d) TC002_05: Test <Scenario of Make Payment (SD009)>

(e) TC002_06: Test <Scenario of View History (SD010)>

4.4.2. TC002: Test <Purchasing Product> Subsystem

This test contains the following test cases:

(a) TC003_01: Test <Scenario of Add Product (SD011)>

4.4.3. TC002_01: Test <Alternate Flow>

This test contains the following test cases:

(a) TC002_01_04: Test <Alternate Flow Scenario of Buy Product (SD008)>

(b) TC002_01_05: Test<Alternate Flow Scenario of Make Payment (SD009)>

4.4.4. TC002_04: Test <Scenario of Buy Product (SD008)> : buyer

Case ID		TC002_04	Test Case Description	Test the Buy Product Functionality (buyer)		
ted By		Anatasya	Reviewed By	Anatasya	Version	2.1
ester's Log		Review comments from Deia incorprate in version 2.1				
er's Name		Anatasya	Date Tested	October 16, 2023	Test Case (Pass/Fail/Not	Pass

S#	Prerequisites:
1	The user must be registered as an active UTM student.
2	The user must be logged in to the system.
3	
4	

S#	Test Data
1	username = anatasya
2	product name = Casio fx-570MS Scientific Calculator
3	price = RM20.00
4	quantity = 2

Scenario	Ensure the functionality and usability of an online purchasing system, the user can buy a specific product

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspen
1	Click 'buy' from the product page.	Buy product page should open	As Expected	Pass
2	Chooses a payment method, which in this case is cash.	System pops up an alert, confirmation box.	As Expected	Pass
3	Wait for the order to be confirmed by the seller	System has notified the seller, update and display order has been placed.	As Expected	Pass
4	Click ok from the confirmation box	System update and display order has been placed.	As Expected	Pass
5	Click confirm after transaction was made face to face.	System displays the successful transaction	As Expected	Pass

4.4.5. TC002_04: Test <Scenario of Buy Product (SD008)> : seller

Test Case ID		TC002_04	Test Case Description	Test the Buy Product Functionality (seller)		
Created By		Anatasya	Reviewed By	Anatasya	Version	2.1
QA Tester's Log		Review comments from Deia incorprate in version 2.1				
Tester's Name		Anatasya	Date Tested	October 16, 2023	Test Case (Pass/Fail/Not	Pass

S#	Prerequisites:
1	The user must be registered as an active UTM student.
2	The user must be logged in to the system.
3	
4	

S#	Test Data
1	username = anatasya
2	product name = Casio fx-570MS Scientific Calculator
3	price = RM20.00
4	quantity = 2

Test Scenario	Ensure the functionality and usability of an online purchasing system, the user can buy a specific product

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click the notification bar function	Notification bar should be open	As Expected	Pass
2	Click on the buyer's notification for cash method	A modal box that shows user's ordered product should be display	As Expected	Pass
3	Click accept order	A pop up box should be display for confirmation	As Expected	Pass
4	Click ok from the confirmation box	System will update and display the product is on hold	As Expected	Pass
5	Click confirm after transaction was made face to face.	System displays the successful transaction	As Expected	Pass

4.4.6. TC002_05: Test <Scenario of Make Payment (SD009)>

st Case ID	TC002_05	Test Case Description	Test the Make Payment Functionality		
reated By	Anatasya	Reviewed By	Anatasya	Version	2.1
A Tester's Log	Review comments from Deia incorprate in version 2.1				
ster's Name	Anatasya	Date Tested	October 16, 2023	Test Case (Pass/Fail/Not	Pass

S #	Prerequisites:
1	The user must be registered as an active UTM student.
2	The user must be logged in to the system.
3	
4	

S #	Test Data
1	Card holder name = anatasya humaira
2	Credit card number = 5136 1845 5468 3894
3	Email Address = anatasya@graduate.utm.my
4	Code CVV/CVC = 123

st Scenario	Ensure the functionality and usability of an online purchasing system, the user can make payment though online banking

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspende
1	Click 'buy' from the product page.	Buy product page should open	As Expected	Pass
2	Chooses a payment method, which in this case is online banking.	System pops up an alert, confirmation box.	As Expected	Pass
3	Click ok from the confirmation box	system redirect user to the payment gateway	As Expected	Pass
4	Fill in the necessary information and click pay	a session page should be display afterwards, require the user to continue the authorization process through email.	As Expected	Pass
5	Open email and proceed the verification	Verification is successful	As Expected	Pass
6	Refresh UtiTrade session page	UtiTrade dashboard should refreshed and display transaction is successful	As Expected	Pass

4.4.7. TC002_01_04: Test <Alternate Flow Scenario of Buy Product (SD008)>

est Case ID	TC002_01_04	Test Case Description	Test the Buy Product Functionality (seller)		
reated By	Anatasya	Reviewed By	Anatasya	Version	2.1
A Tester's Log	Review comments from Deia incorprate in version 2.1				
ester's Name	Anatasya	Date Tested	October 16, 2023	Test Case (Pass/Fail/Not	Pass

S #	Prerequisites:
1	The user must be registered as an active UTM student.
2	The user must be logged in to the system.
3	
4	

S #	Test Data
1	username = anatasya
2	product name = Casio fx-570MS Scientific Calculator
3	price = RM20.00
4	quantity = 2

est Scenario	Ensure the functionality and usability of an online purchasing system, the user can buy a specific product

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click the notification bar function	Notification bar should be open	As Expected	Pass
2	Click on the buyer's notification for cash method	A modal box that shows user's ordered product should be display	As Expected	Pass
3	Click check user information	Buyer's profile page should be display	As Expected	Pass

4.4.8. TC002_01_05: Test<Alternate Flow Scenario of Make Payment (SD009)>

Case ID	TC002_01_05	Test Case Description	Test the Buy Product Functionality (seller)		
ed By	Anatasya	Reviewed By	Anatasya	Version	2.1
ester's Log	Review comments from Deia incorprate in version 2.1				
r's Name	Anatasya	Date Tested	October 16, 2023	Test Case (Pass/Fail/Not	Pass

S#	Prerequisites:
1	The user must be registered as an active UTM student.
2	The user must be logged in to the system.
3	
4	

S#	Test Data
1	username = anatasya
2	product name = Casio fx-570MS Scientific Calculator
3	price = RM20.00
4	quantity = 2

Scenario	Ensure the functionality and usability of an online purchasing system, the user can buy a specific product

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspen
1	Click the notification bar function	Notification bar should be open	As Expected	Pass
2	Click on the buyer's notification for cash method	A modal box that shows user's ordered product should be display	As Expected	Pass
3	Click check user information	Buyer's profile page should be display	As Expected	Pass

5. Module 3

5.1. Specific Requirement

5.1.1. User characteristics

Below are the list of different user characteristics as well as itss roles within module 3, known as the selling product subsystem:

5.1.1.1. UTM Students (as seller)

- Students are expected to still be registered as an active student in UTM.
- Users might want to sell either their unused product or any kind of relevant product that they want to sell.
- Students might need an interface that is user-friendly and simple to learn since one of the main purposes of the system is to make transactions easier.
- Users are required to accurately describe the products you are selling, including specifications, features, and any relevant details.
- Users are required to set competitive and reasonable prices for their unused products.
- Users are expected to respond promptly to customer inquiries, messages, and concerns outside the system.

5.1.2. System Features

For module three, the system features include:

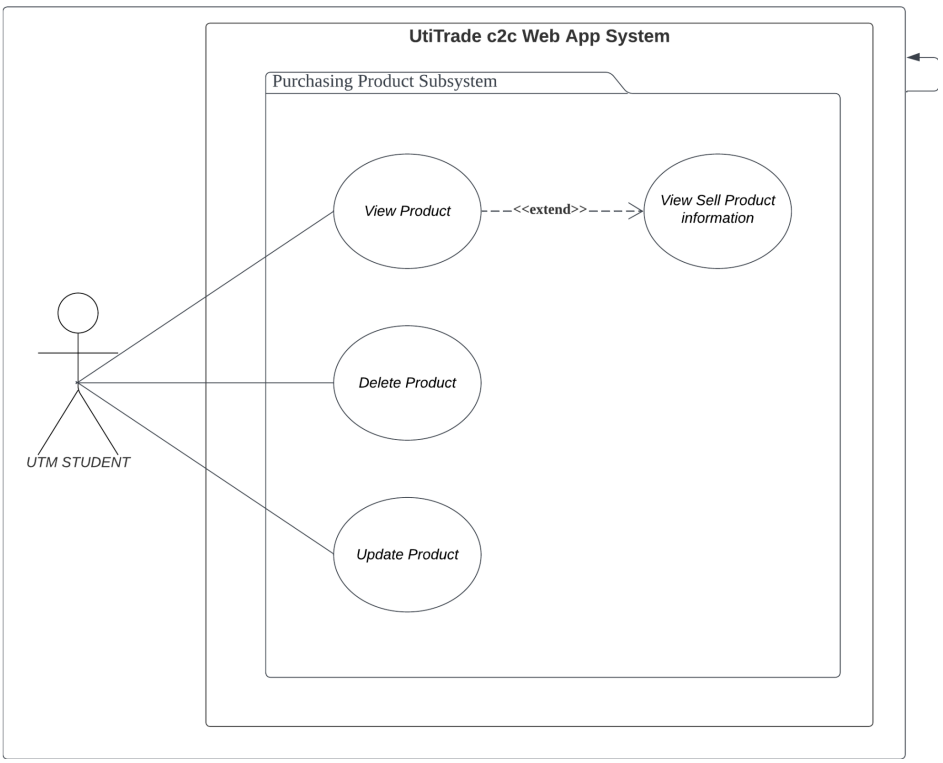


Figure 5.1.2 : Use Case Diagram for Module 3

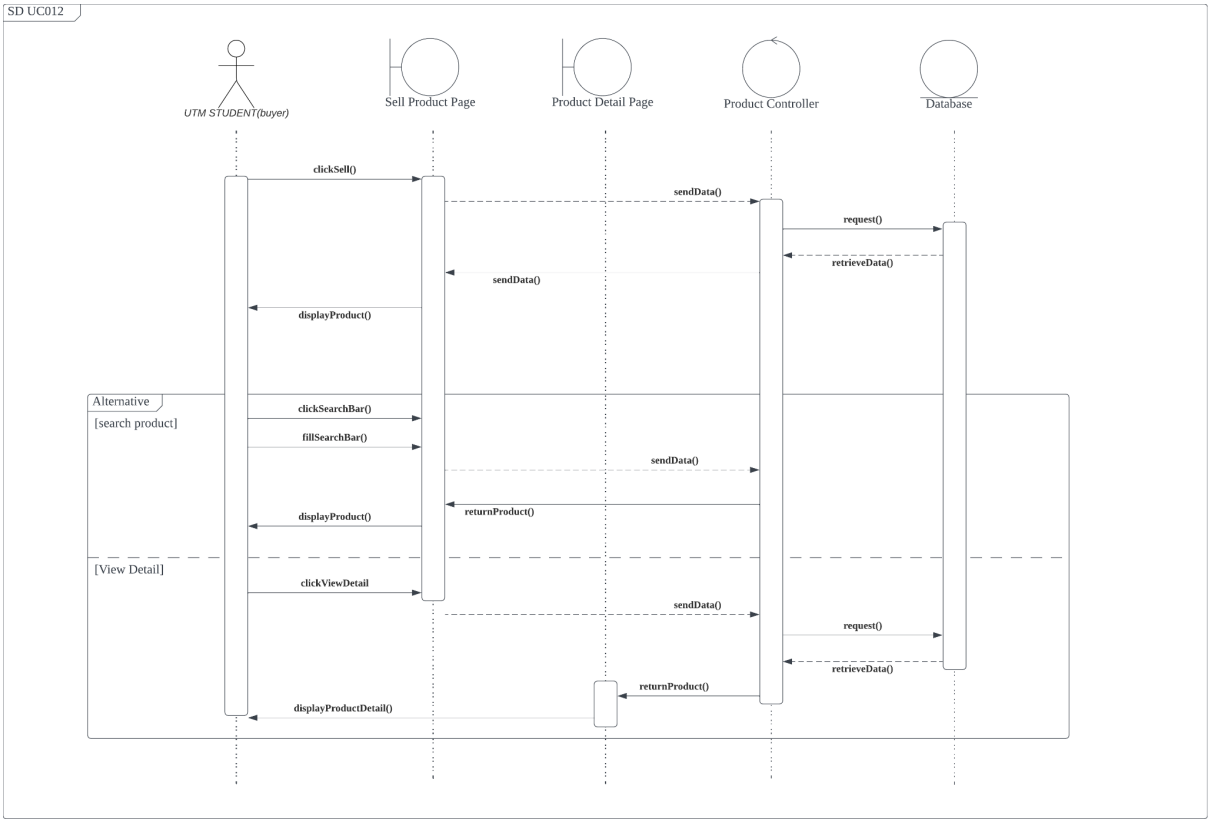
5.1.3. UC012: Use Case <View Product> & Seq Diagram

Table 5.1.3.1 : Use Case Description for <View Product>

Use case: <view product>
ID: UC012
Actors: UTM Student
Preconditions: 1. The user must be registered as an active UTM student. 2. The user must be logged in to the system.
Main flow : 1. User clicks ‘sell product’ from the main page. 2. System redirects the user to the sell product page. 3. The user is provided with many features within the page, including view detail 4. The user can search the prouct which they have added to the system in the searching bar. 5. The users tap the product the wish to see. 6. System redirects the user to the product page, where all the product details are displayed.

7. Use Case ends.
Alternate flow : 1. User taps on next page at the bottom of the page 2. System will display the product list from the next page.
Exception flow (if any): 1. User has not put any product yet to the system.

Figure 5.1.3.1 : SD012: Sequence Diagram for <View Product>



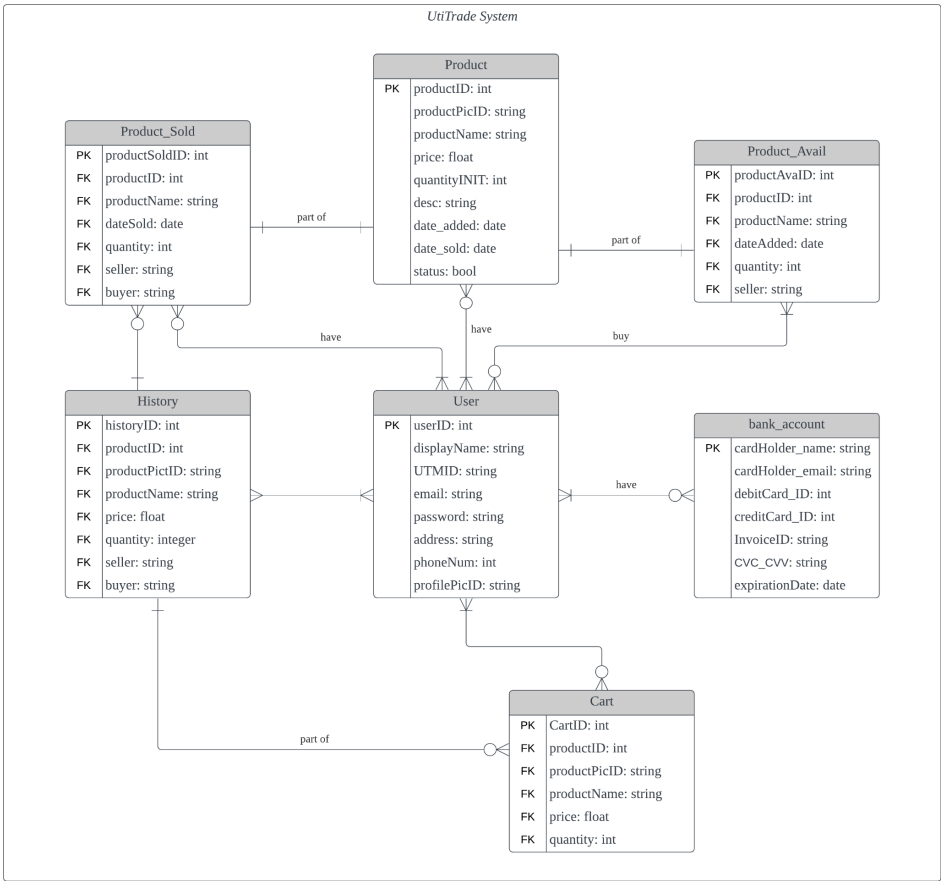
5.1.4. UC013: Use Case <Delete Product> & Seq Diagram

5.1.5. UC014: Use Case <Update Product> & Seq Diagram

5.2. Data Design

There are 7 entities in the database design of the UtiTrade system during this sprint : User DB, Product DB, Cart DB, ProductAvail DB, ProductSold DB, BankAccount DB, History DB. Figure 5.2.1 below represents the database design of the UtiTrade system whereas Table 5.2.1.1 represents the description of the entity.

Figure 5.2.1: Entity Relationship Diagram of UtiTrade System.



5.2.1. Data Dictionary & Description

Table 5.2.1.1: Entity Description

Entity	Description
User	Contains the detail information of user
Bank_account	Contains the detail information of user's bank account
History	Contains the detail information of user's transaction as well as notification
Product	Contains the detail information of a certain product
Product_Avail	Contains the detail information of available products
Product_Sold	Contains the detail information of sold out products
Cart	Contains the detail information of a product of the user's cart.

Data Dictionary

5.2.1.1. Entity: <User>

Attributes	Type	Description
userID	integer	This attribute serves as a unique identifier for each user in the system. It is used for referencing and distinguishing one user from another.
displayName	string	The displayName attribute stores the user's chosen name and can be edited. It allows users to personalize their profiles and how they appear to others.
UTMID	string	UTMID is an attribute that stores a unique identifier associated with the user's university ID.
email	string	The email attribute stores the user's email address, which is essential for reset password, account authentication, and account confirmation.
password	string	The password attribute stores a securely hashed or encrypted version of the user's password.
address	string	The address attribute stores the user's physical address. which may be used for shipping or location-based services.
phoneNum	integer	It stores the user's contact number. It can be useful for communication.
profilePictID	string	This attribute serves as a reference to the user's profile picture or avatar image.

5.2.1.2. Entity: <Product>

Attributes	Type	Description
productID	integer	This attribute is a unique identifier for each product in the system. It is used to distinguish one product from another and is crucial for database operations.
productPictID	string	The productPictID attribute serves as a reference to the product's main image or picture. Instead of storing the actual image data within the product entity, it links to an image stored elsewhere in the system.
productName	string	productName stores the name or title of the product.
price	float	The price attribute stores the monetary value associated with the product. It indicates the cost of purchasing the item.
quantityInit	integer	This attribute represents the initial quantity of the product available for sale.
desc	string	The desc attribute holds a detailed description of the product. It provides additional information about the item's features, specifications, condition and any other relevant details.
status	bool	The status attribute is of type 'bool' (boolean) and is used to

		indicate the availability or status of the product listing. A 'true' value typically indicates that the product is available for sale, while 'false' may indicate that the product has been sold or is no longer available.
dateAdded	date	As reference from the Product Avail entity, it represents the date and time when the product listing was created or added to the platform.
dateSold	date	As reference from the Product Sold entity, it represents the date and time when the product listing was sold to the buyer.

5.2.1.3. Entity: <Product_Avail>

Attributes	Type	Description
productAvailID	integer	A unique identifier for each available product in the system.
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
seller	string	As reference from the User entity. This attribute serves as a unique identifier for each user that sells the particular item in the system.
productName	string	As reference from the Product entity, it stores the name or title of the product. It provides a clear and concise description of the item and is displayed prominently in product listings.
dateAdded	date	It represents the date and time when the product listing was created or added to the platform.
quantity	integer	It represents the quantity of the product available for sale in the listing. As products are sold, this quantity may decrease to reflect the number of items still available for purchase.

5.2.1.4. Entity: <Product_Sold>

Attributes	Type	Description
productSoldID	integer	A unique identifier for each sold item in the system.
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
seller	string	As reference from the User entity. This attribute serves as a unique identifier for each user that sells the particular item in the system.
buyer	string	As reference from the User entity. This attribute serves as a unique identifier for each user that has purchased the

		particular item in the system.
productName	string	As reference from the Product entity, it stores the name or title of the product. It provides a clear and concise description of the item and is displayed prominently in product listings.
dateSold	date	It represents the date and time when the product listing was sold to the buyer.
quantity	integer	It represents the quantity of the sold product that is no longer available for purchase.

5.2.1.5. Entity: <bank_account>

Attributes	Type	Description
cardHolder_name	string	This attribute stores the name of the individual or entity to whom the credit or debit card is issued.
cardHolder_email	string	The email address attribute stores the contact email associated with the bank account. It is used for communication with the account holder, sending transaction notifications, and account-related correspondence.
debitCard_ID	integer	Similar to a credit card, this attribute stores the unique identification number of a debit card issued to an account holder. Debit cards are linked to a bank account and are used for making payments and withdrawals directly from the account.
creditCard_ID	integer	This attribute stores the unique identification number of a credit card issued to an account holder.
InvoiceID	string	The Invoice ID attribute is used to link a transaction to a specific invoice or bill. It helps in associating the transaction with the corresponding financial record or order.
CVC_CVV	integer	The Card Verification Code (CVC) or Card Verification Value (CVV) is a security code printed on the back of a credit or debit card. It is used to verify that the cardholder possesses the physical card during online or card-not-present transactions, adding an extra layer of security.
expirationDate	date	This attribute represents the date until which the credit or debit card is valid. After this date, the card cannot be used for transactions, and a new card is typically issued.

5.2.1.6. Entity: <cart>

Attributes	Type	Description

productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
productPictID	string	As reference from the Product entity, the productPictID attribute serves as a reference to the product's main image or picture.
productName	string	As reference from the Product entity, it stores the name or title of the product.
price	float	As reference from the Product entity, the price attribute stores the monetary value associated with the product.
quantity	integer	It represents the quantity of the product in the cart. Users may edit or remove the quantity according to their needs.

5.2.1.7. Entity: <history>

Attributes	**Type**	**Description**
productID	integer	As reference from the Product entity, this attribute is used as an unique identifier for each product in the system.
productPictID	string	As reference from the Product entity, the productPictID attribute serves as a reference to the product's main image or picture.
productName	string	As reference from the Product entity, it stores the name or title of the product.
price	float	As reference from the Product entity, the price attribute stores the monetary value associated with the product.
quantity	integer	It represents the quantity of the product in the cart. Users may edit or remove the quantity according to their needs.
seller	string	As reference from the User entity. This attribute serves as a unique identifier for each user that sells the particular item in the system.
buyer	string	As reference from the User entity. This attribute serves as a unique identifier for each user that has purchased the particular item in the system.
dateAdded	date	As reference from the Product Avail entity, it represents the date and time when the product listing was created or added to the platform.
dateSold	date	As reference from the Product Sold entity, it represents the date and time when the product listing was sold to the buyer.
transactionNum	int	As reference from the Product Sold entity, it represents the date and time when the product listing was sold to the buyer.
status	bool	The Status attribute indicates the outcome of a transaction.

		It can have one of two values: "failed" or "successful."

5.3. User Interface Design

During iteration three, there are various pages of interface design that our group managed to create by using a design tool called *Figma*. The system provides two functionality dashboards which allows the user to be a buyer as well as be a seller at the same time. As for the iteration three, the fucntion is emphasising more on the seller perspective which allows the user to be roled as the seller as shown below:

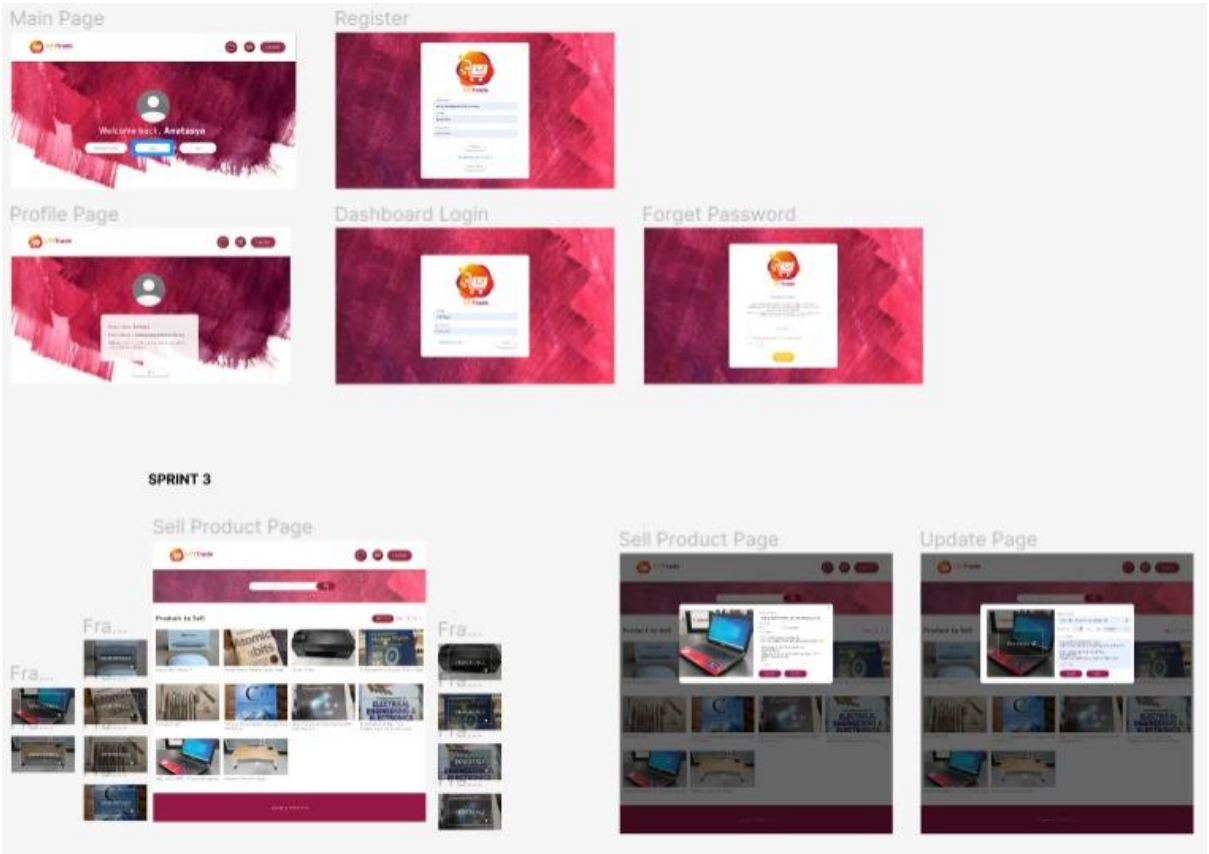
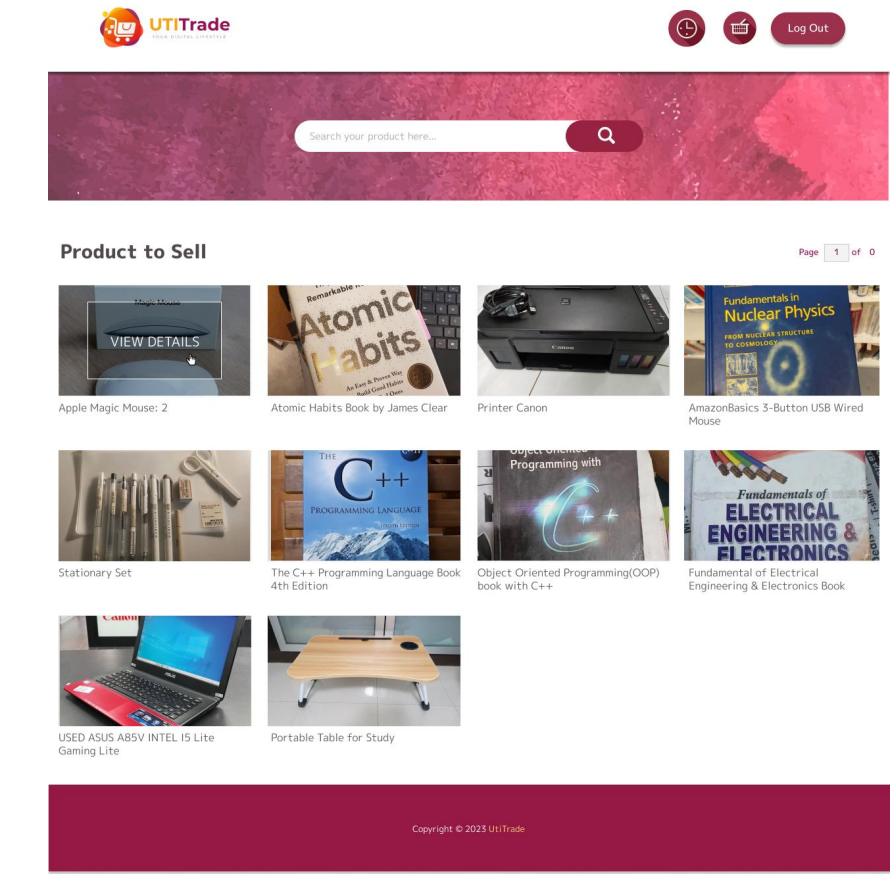


Figure 5.3.1 : Overall Interface Design for <Selling product> subsystem



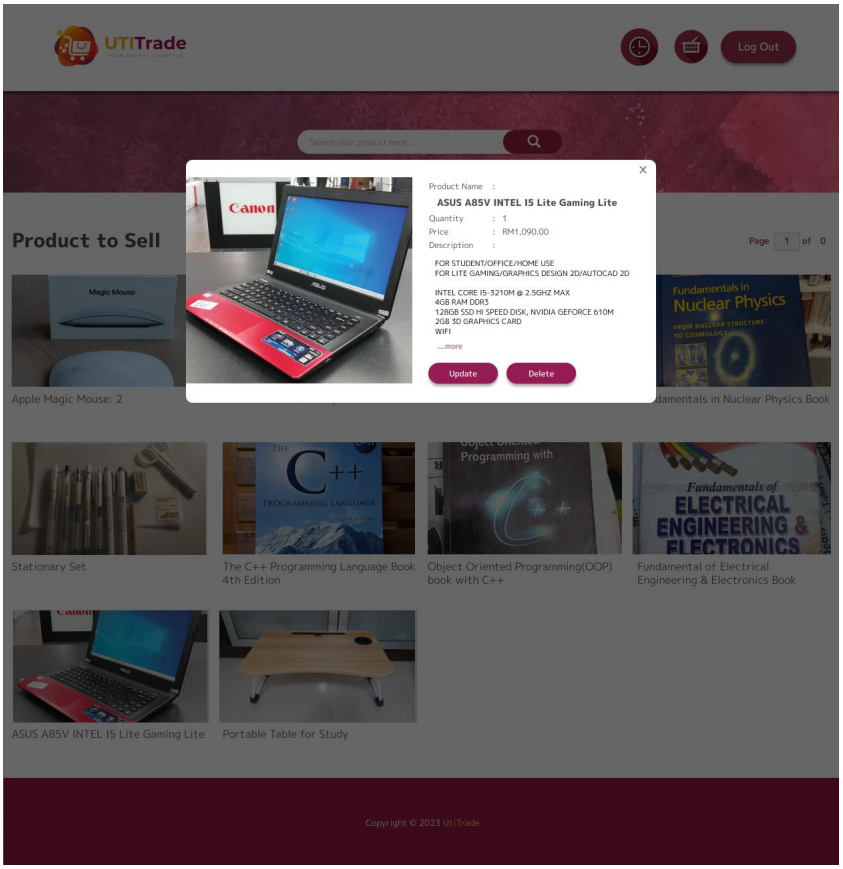


Figure 5.3.1, 5.3.2 : Interface Design of View Product Page

Figure 5.3.3 : Interface Design of Delete Product Page

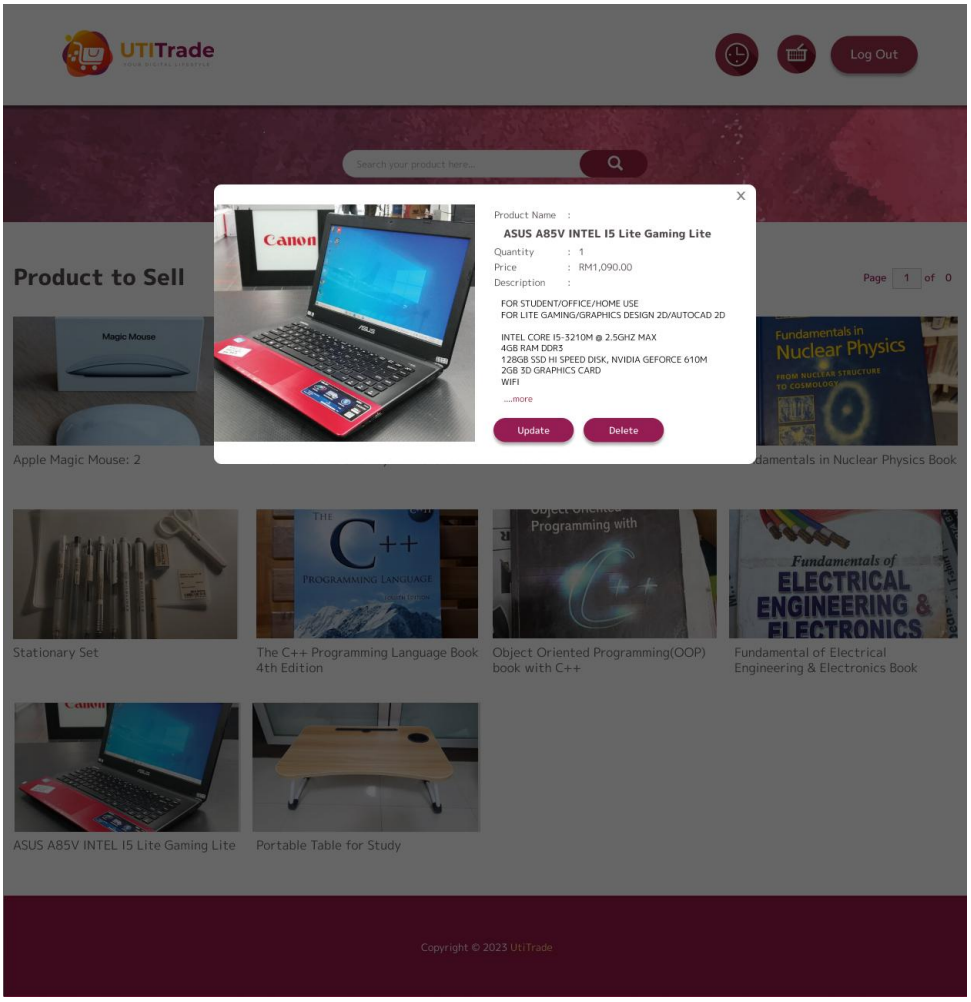


Figure 5.3.4 : Interface Design of Update Product Page

5.4. Test Cases

5.4.1. TC003: Test <Selling Product> Subsystem

This test contains the following test cases:

(a) TC003_02: Test <Scenario of View Product (SD012)>

(b) TC003_03: Test <Scenario of User Login (SD013)>

(c) TC003_04: Test <Scenario of Reset Password (SD014)>

5.4.2. TC002_01: Test <Alternate Flow>

This test contains the following test cases:

(a) TC003_01_02: Test <Alternate Flow Scenario of View Product (SD012)>

5.4.3. TC003_02: Test <Scenario of View Product (SD012)>

Test Case ID	TC003_02	Test Case Description	Test the View Product Functionality		
Created By	Anatasya	Reviewed By	Anatasya	Version	2.1
QA Tester's Log	Review comments from Deia incorprate in version 2.1				
Tester's Name	Anatasya	Date Tested	October 25, 2023	Test Case (Pass/Fail/Not	Pass

S#	Prerequisites:	S#	Test Data
1	The user must be registered as an active UTM student.	1	username = anatasya
2	The user must be logged in to the system.	2	product name = Casio fx-570MS Scientific Calculator
3		3	
4		4	

Test Scenario	Ensure the functionality and usability of an online purchasing system, the user can buy a specific product

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click the "Sell Product" option from the main page.	View product page should be displayed.	As Expected	Pass
2	Click on the searching bar and find the product the want to see	Several options should be provided and displayed to user's screen.	As Expected	Pass
3	Tap on the product and click view detail	System redirect user to the view detail page.	As Expected	Pass

5.4.4. TC003_01_02: Test <Alternate Flow Scenario of View Product (SD012)>

Test Case ID	TC003_01_02	Test Case Description	Test the View Product Functionality		
Created By	Anatasya	Reviewed By	Anatasya	Version	2.1
QA Tester's Log	Review comments from Deia incorprate in version 2.1				
Tester's Name	Anatasya	Date Tested	October 25, 2023	Test Case (Pass/Fail/Not	Pass

S#	Prerequisites:	S#	Test Data
1	The user must be registered as an active UTM student.	1	username = anatasya
2	The user must be logged in to the system.	2	product name = Casio fx-570MS Scientific Calculator
3		3	
4		4	

Test Scenario	Ensure the functionality and usability of an online purchasing system, the user can buy a specific product

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Click the "Sell Product" option from the main page.	View product page should be displayed.	As Expected	Pass
2	Click on the next page button at the end of the page	System displays "none" or "nothing found"	As Expected	Pass
3	Tap on previous page	System displays the previous items of a user	As Expected	Pass