

CSIT321

FINAL YEAR PROJECT REPORT

Project Progress Report

SMART CONTRACT USING BLOCKCHAIN

(CSIT-20-S4-13)



UNIVERSITY OF WOLLONGONG

SCHOOL OF COMPUTING AND INFORMATION TECHNOLOGY

Group Number: FYP-20-S4-17

Assessor: Mr. Tian Sion Hui

Supervisor: Mr. Tan Kheng Teck

Requirements Specification Document

Project: Smart Contracts using Blockchain

Group: FYP-20-S4-17

Name	Student No.	Email
Huang Junde	6347046	jhuang062@mymail.sim.edu.sg
Goh Yong Xing	6354634	yxgoh014@mymail.sim.edu.sg
Eng Yong Tee	6347186	yteng007@mymail.sim.edu.sg
Christian Lemuel Espere Pahilan	6737171	pahilan001@mymail.sim.edu.sg
Ashira Wee	5918169	awee003@mymail.sim.edu.sg

Document Control

Version	Date	Summary of Changes	Author
1.0	13/11/2020	Phase one submission	Huang Junde
2.0	02/01/2021	Project Progress Report	Goh Yong Xing Eng Yong Tee Christian Lemuel Espere Pahilan Ashira Wee

Table of Contents

1. Introduction	4
1.1 Background	5
1.2 Purpose	5
1.3 Scope	5
1.4 Roles and Responsibilities	6
1.5 Project Timeline	7
1.6 Assumptions	7
1.7 Constraints	7
2. Software Development Methodology	9
3. Risk Management Plan	10
4. Research	13
4.1 Market Research	13
4.2 Proposed Software and Programming Languages	14
4.2.1 Ethereum	14
4.2.2 Solidity	15
4.2.3 Front-End	15
5. Requirements	16
5.1 Use Case Diagram	17
5.2 User Stories	16
5.3 Use Case Description	16
5.4 BCE Class Diagrams	29
5.5 BCE Sequence Diagrams	34
5.6 Wireframes	42
5.7 Functional Requirements	48
5.7.1 Description	48
5.7.2 Elaboration	49
5.8 Non-Functional Requirements	52
5.8.1 Availability	52
5.8.2 Data Integrity	52
5.8.3 Integrity	52
5.8.4 Performance	52
5.8.5 Recoverability	53
5.8.6 Reliability	53
5.8.7 Security	53
6. Project Progression	54

6.1 Week 1 - 15 Oct to 21 Oct	54
6.3 Week 2 - 22 Oct to 28 Oct	54
6.3 Week 3 - 29 Oct to 4 Nov	54
6.4 Week 4 - 5 Nov to 11 Nov	54
6.5 Week 5 - 12 Nov to 18 Nov	54
6.6 Week 6 - 19 Nov to 25 Nov	55
6.7 Week 7 - 26 Nov to 2 Dec	55
6.8 Week 8 - 3 Dec to 9 Dec	55
6.9 Week 9 - 10 Dec to 16 Dec	55
6.10 Week 10 - 17 Dec to 23 Dec	55
6.11 Week 11 - 24 Dec to 30 Dec	55
6.11 Week 12 - 31 Dec to 6 Jan	55
7. References	57

1. Introduction

1.1 Background

Due to the COVID-19 pandemic and the lockdowns, people are staying home and are performing more online transactions than ever before. Although the e-commerce space has seen positive growth, it has also resulted in an increase in e-commerce fraud. The aim of this project is to provide a blockchain-based e-commerce platform to reduce the occurrence of e-commerce fraud.

1.2 Purpose

The purpose of this project is to incorporate the use of escrow smart contracts on an online marketplace, where both sellers and buyers mutually fund an escrow account with collateral in addition to the buyer's payment. To prevent losing their collaterals, both parties are required to behave honestly. This enables both parties to transact with higher security, trust and transparency.

1.3 Scope

The team will develop a web-based marketplace that utilises blockchain technology to allow buyers and sellers to establish smart contracts. The application will turn terms of negotiation between both parties into code and transfer of payment will be executed automatically when a certain criteria is met. The web application will allow users to purchase and sell items, initiate and establish a smart contract, pay and receive cryptocurrency when purchasing or selling items.

1.4 Roles and Responsibilities

Name	Role	Organisation
Mr Tian Sion Hui	Assessor	CROW, University of Wollongong
Mr. Tan Kheng Teck	Project Supervisor	CROW, University of Wollongong
Huang Junde	Project Leader	CROW, University of Wollongong
Goh Yong Xing	Back End Software Developer	CROW, University of Wollongong
Eng Yong Tee	Back End Software Developer	CROW, University of Wollongong
Christian Lemuel Espere Pahilan	Front End Software Developer	CROW, University of Wollongong
Ashira Wee	Front End Software Developer	CROW, University of Wollongong

1.5 Project Timeline

Sprint Release Plan			Detailed Timeline		
Sprint	Start	End	Task	Start	End
1	2020-11-15	2020-12-05	Design - Wireframe, BCE, Sequence Diagram	15-Nov-20	19-Nov-20
			Create Test Cases	20-Nov-20	1-Dec-20
			Build	20-Nov-20	1-Dec-20
			Test	3-Dec-20	5-Dec-20
2	2020-12-06	2021-01-01	Design - Wireframe, BCE, Sequence Diagram	6-Dec-20	12-Dec-20
			Create Test Cases	13-Dec-20	25-Dec-20
			Build	13-Dec-20	25-Dec-20
			Test	26-Dec-20	1-Jan-21
3	2021-01-02	2021-01-17	Design - Wireframe, BCE, Sequence Diagram	2-Jan-21	6-Jan-21
			Create Test Cases	7-Jan-21	14-Jan-21
			Build	7-Jan-21	14-Jan-21
			Test	15-Jan-21	17-Jan-21
4	2021-01-18	2021-01-31	Design - Wireframe, BCE, Sequence Diagram	18-Jan-21	21-Jan-21
			Create Test Cases	22-Jan-21	28-Jan-21
			Build	22-Jan-21	28-Jan-21
			Test	29-Jan-21	31-Jan-21
5	2021-02-01	2021-02-19	Design - Wireframe, BCE, Sequence Diagram	1-Feb-21	4-Feb-21
			Create Test Cases	5-Feb-21	16-Feb-21
			Build	5-Feb-21	16-Feb-21
			Test	17-Feb-21	19-Feb-21

1.6 Assumptions

- Users are assumed to have basic understanding of how a marketplace and contract functions to run the program initially.
- Target audience are assumed to mainly consist of teenagers and working adults but not restricted to other age demographic.
- The smart contract will be set in place only when the buyer and seller has come to an agreement on the terms and placed their respective collateral fees as well as payment(buyer) in the escrow account.
- The smart contract is a self-executing contract that will release payment held to the seller upon verification done by the buyer. In the event that either party fails to meet the objective set out initially, their collateral will be absorbed.
- The negotiation of the terms of the smart contract will be done solely on the marketplace to ensure that the set up will run smoothly.
- The currency that the smart contract uses will be obtained from the MetaMask wallet that allows Ether or multiple types of ERC-20 tokens.

1.7 Constraints

- The smart contracts will be deployed to a testnet instead of the mainnet because the cost to deploy contracts will be too high for the team to afford.

- The smart contract will be bound to the underlying technology of the existing blockchain network.
- Due to limited time, the team will not be implementing its own token for use as it will divert the focus of the project.

2. Software Development Methodology

Scrum methodology will be adopted to manage the project, where the development of the web application will be completed via a series of sprints. In this case, our project will have six sprints with each lasting between two to three weeks. The team has prepared the product backlog based on the user stories identified and the user stories have been prioritised based on what the application must have, should have and could have.

The sprints will begin with a sprint planning meeting, where a sprint backlog will be created, consisting of tasks the team needs to complete in order to deliver the functionalities committed. As prioritisation of user stories has been completed, the team will be able to identify the user stories to work on during each sprint in an efficient manner.

During the sprint, the team will create design documents such as wireframes, sequence diagrams and BCE diagrams. After design documents are completed, the team will work on front-end and back-end development of the web application. Test cases will also be created to facilitate testing after development. Due to time constraint, the team will carry out a weekly scrum meeting instead of a daily meeting. In that meeting, members will update on their progress for the past week, what they will work on during the week, and raise any issues that could hinder their progress.

At the end of each sprint, a sprint review meeting will be conducted and the team will demonstrate the new functionalities to the supervisor. During the sprint review meeting, the team will seek feedback to ensure that the functionalities built meets the supervisor's expectations and also identify what could be done better in the next sprint.

3. Risk Management Plan

Project Risk Management				
Risk/Issue	Category	Prevention Measures / Potential Responses	Probability	Impact
Non-user friendly system - user may encounter difficulties when interacting with the platform	Technical Risk	Project team has to think from the user's perspective and come out with a user guide.	Low	Medium
Developed platform's function and the project requirements do not match	Technical Risk	Start the coding process early so that the team has enough time to seek feedback from the supervisor during sprint review meetings, where new functionalities will be demonstrated.	Medium	High

Inadequate performance and quality	Technical Risk	Periodically perform testing to ensure optimal quality assurance.	Medium	High
Significant changes in project requirements and scope	Technical Risk	Implement immediate actions to meet the latest requirements and changes.	Low	High
Proficiency in programming	Technical Risk	Everyone is encouraged to participate in the software development process.	High	High
Lack of interest from potential users for project	User Risk	Include a marketing/business strategy for our online platform.	Low	Medium
Failure of external extension (eg: MetaMask payment issue)	External Risk	Periodically perform compatibility and connection testing with payment gateway extension to ensure smooth user experience.	Medium	High

Lack of communication	Internal Risk	<p>Common communication platform and repository (shared drives) to ensure version control and most updated information.</p> <p>Increase the number of meetings so that everyone is involved.</p>	Low	Medium
Poor schedule planning	Internal Risk	Take note of the project deadlines and keep track of schedules.	Low	Medium
File corruption	External Risk	Team will keep all documents/programs stored and backed up online with the use of cloud servers such as google drive, github.	Low	High

4. Research

4.1

Market

Research

Features	CROW	QuuBe	Carousell
List item	X	X	X
Escrow	X	X	X
Native Digital Currency		X	
Blockchain-based	X	X	
Dual-Deposit Logic	X		

In our research, QuuBe and Carousell were identified as our main competitors as they are both online marketplaces which provide users a platform to buy and sell items or services. QuuBe utilises a smart contract function to handle verification and settlement of payment between sellers and buyers. It also provides escrow, where they temporarily hold buyer's payment until the seller is able to provide sufficient evidence, which can be buyer's verification or proof of delivery. QuuBe has its own digital currency, Q*coin that QuuBe users have to use to transact on the platform. This reduces the operational cost as it removes the need to outsource the payment verification and settlement to third parties. Although QuuBe's business targets the business-to-consumer (B2C) segment and CROW's model targets consumer-to-consumer (C2C), the technology used in both applications are blockchain. Similarly, Carousell also provides escrow services, Carousell Protection, to their users. Carousell Protection is an optional add on for sellers, where sellers can choose to sell with that feature and buyers have to pay a fee of 2% of the item and delivery cost to utilise this feature.

All three applications provide similar features such as listing items and escrow services. The main difference between CROW and its competitors is the addition of dual-deposit logic. In order to establish a smart contract after negotiations, both the seller and buyer are required

to contribute a collateral that is dependent on the cost of the item. The amount paid by both parties will then be held in an escrow account until the buyer verifies that he/she has received the item. The use of dual-deposit logic is to encourage the buyer to provide confirmation that the order has been fulfilled, as well as to not raise disputes unnecessarily and to encourage the seller to provide the item as listed and to ship in a timely manner. As both parties have collateral at stake if they do not hold up their end of the transaction, they are incentivised to act in an honest manner.

4.2 Proposed Software and Programming Languages

We are using industry related software and programming languages that are used to meet the specifications and requirements of the system's needs. These softwares and languages comprise of :

4.2.1 Ethereum

Ethereum is a blockchain platform that supports smart contracts. Smart contracts are pieces of code that perform general purpose computations. Ethereum is an open access to digital money and data-friendly services for everyone. It is a community built technology that is behind the cryptocurrency ether. Ethereum can be used to leverage blockchain to decentralize any function or activity. Overall, Ethereum would be able to aid us in developing an escrow smart contract platform to meet the project requirements.

4.2.2 Solidity

Solidity is a contract oriented high-level programming language that is used to create smart contracts. Solidity is designed to target the implementation of smart contracts on blockchain platforms. It does the processes of verifying and enforcing constraints at compile-time as opposed to run-time. Smart contracts programs can be written solidity and are one of the best means to exchange money, shares or anything of value. Solidity is able to develop smart contracts that have obligations met automatically when requirements are met in the agreement.

4.2.3 Front-End

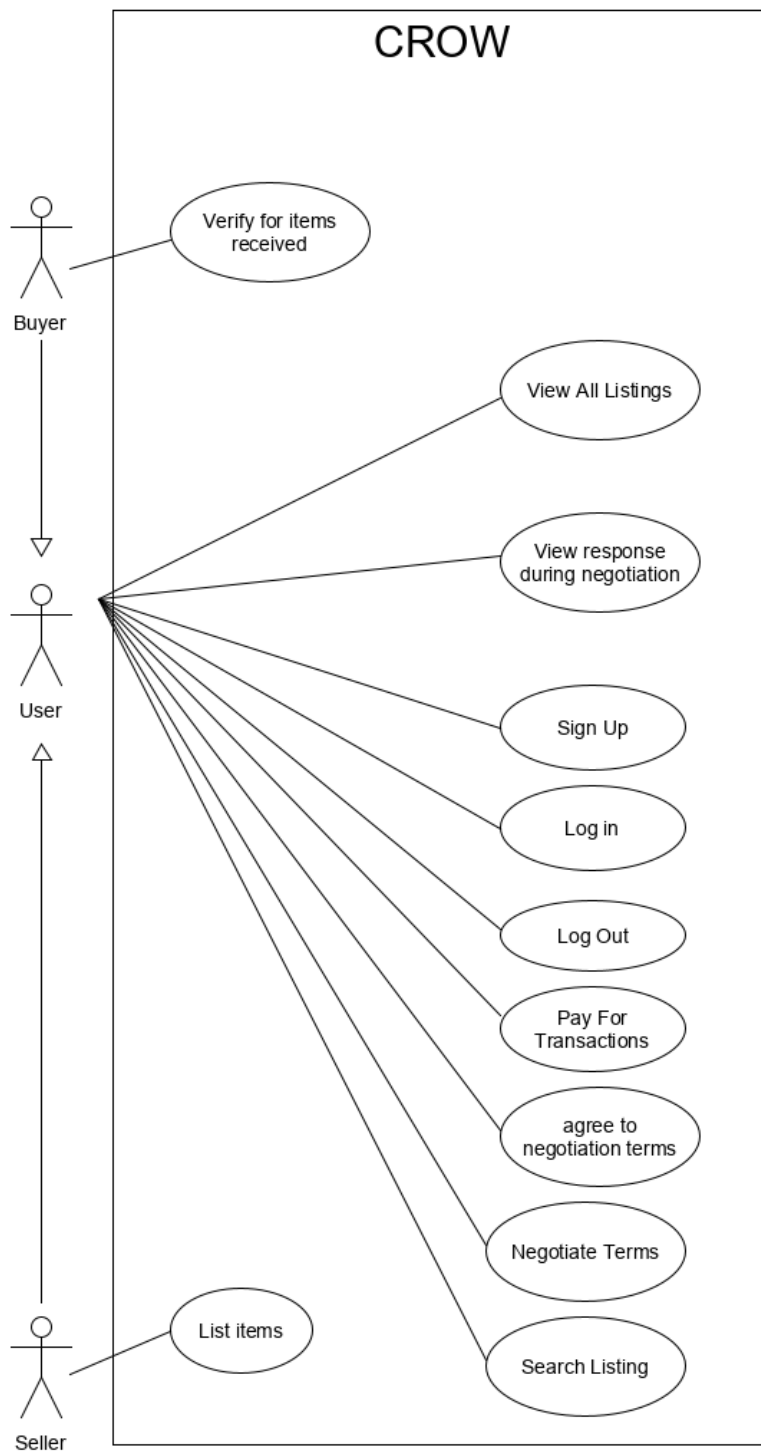
The react framework will be used for the front-end development of the web application. In order to enable interaction between front-end to the blockchain, a library called web3 will be used.

5. Requirements

5.1 User Stories

No.	User Story
U1	As a buyer, I want to be able to pay for transactions so that I can buy listings from the site.
U2	As a buyer, I want to be able to verify that I have received my items so that I can close and end a successful transaction.
U3	As a user, I want to be able to view all listings so that I can browse through what is available.
U4	As a user, I want to be able to view responses during negotiation so that I know what the other party is offering.
U5	As a user, I want to be able to sign up so that I can have an account to use the site.
U6	As a user, I want to be able to log in so that I can use the site.
U7	As a user, I want to be able to log out so that I can terminate my current session.
U8	As a user, I want to be able to agree to negotiation terms so that the other party knows that I have agreed to his/her terms.
U9	As a user, I want to be able to negotiate terms so that I can ask for a better offer.
U10	As a user, I want to be able to search listings so that I can see what I specifically want.
U11	As a seller, I want to be able to list items so that potential buyers can see what I want to sell.

5.2 Use Case Diagram



5.3 Use Case Description

#U1	
Use case name	Pay for transactions
Scenario	User is able to pay for a transaction
Triggering event	User clicks on 'buy' button
Overview	User can pay the collateral of the listing, and also the price if user is a buyer
Actors	User
Pre-conditions	User needs to have created a valid account
Post-conditions	User's commitment is updated
Basic flow of activities:	
<ol style="list-style-type: none">1. User is at Smart Contracts or Marketplace page2. User clicks on 'pay' button3. Metamask extension pops up4. Sign in to Metamask extension if not already signed in5. Click on 'pay' button on Metamask6. End.	
Exception Flow	User does not have enough balance in Metamask

#U2	
Use case name	Verify received items
Scenario	To verify that buyer has received item
Triggering event	Buyer has received item and wants to get back collateral
Overview	Buyer verify item has been received
Actors	Buyer
Pre-conditions	Both buyer and seller has committed their ether to a transaction
Post-conditions	Item is verified and collateral is returned to buyer, while collateral + payment is transferred to seller
Basic flow of activities:	
<ol style="list-style-type: none"> 1. Buyer clicks on 'verify' button 2. Collateral is returned to buyer 3. Collateral + payment is transferred to seller 4. End 	
Exception Flow	None

#U3	
Use case name	View all listings
Scenario	To view all listings on the platform
Triggering event	User arrives at site
Overview	User is shown a table of all listings available upon arrival at the site
Actors	User
Pre-conditions	None
Post-conditions	None
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User arrives at the site 2. Table of all listings is displayed 3. End 	
Exception Flow	None

#U4	
Use case name	View response during negotiation
Scenario	To view the other party's negotiation response
Triggering event	User wants to view other party's negotiation term
Overview	User sees what the other party's response to the negotiated terms is
Actors	User
Pre-conditions	Other party has accepted prior negotiated terms or has responded with a new price
Post-conditions	Negotiation list is displayed
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User clicks on 'negotiation' button 2. User arrives at list of negotiation page 3. A table will display the latest negotiation price for items in negotiation 4. End 	
Exception Flow	None

#U5	
Use case name	Sign up
Scenario	User wants to sign up for an account
Triggering event	User clicks on the sign up button
Overview	User clicks on the sign up button and gets signed up to the marketplace
Actors	User
Pre-conditions	User must not have an existing account
Post-conditions	An account has been added
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User enters input in username field 2. User enters input in password field 3. User clicks on 'sign up' button 4. End 	
Exception Flow	<ol style="list-style-type: none"> 1. Username is already taken 2. Password is not strong enough

#U6	
Use case name	Log in
Scenario	User wants to log in
Triggering event	User clicks on 'log in' button
Overview	User logs in to site and can begin using site features
Actors	User
Pre-conditions	User needs to have a valid account
Post-conditions	User is logged in
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User enters input in username field 2. User enters input in password field 3. User clicks on 'log in' button 4. End 	
Exception conditions	<ol style="list-style-type: none"> 1. Username does not exist 2. Password entered is incorrect for username provided

#U7	
Use case name	Log out
Scenario	User wants to log out
Triggering event	User clicks on 'log out' button
Overview	User logs out and ends the current session
Actors	User
Pre-conditions	User is logged in
Post-conditions	User is logged out
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User clicks on 'log out' button 2. End 	
Exception Flow	None

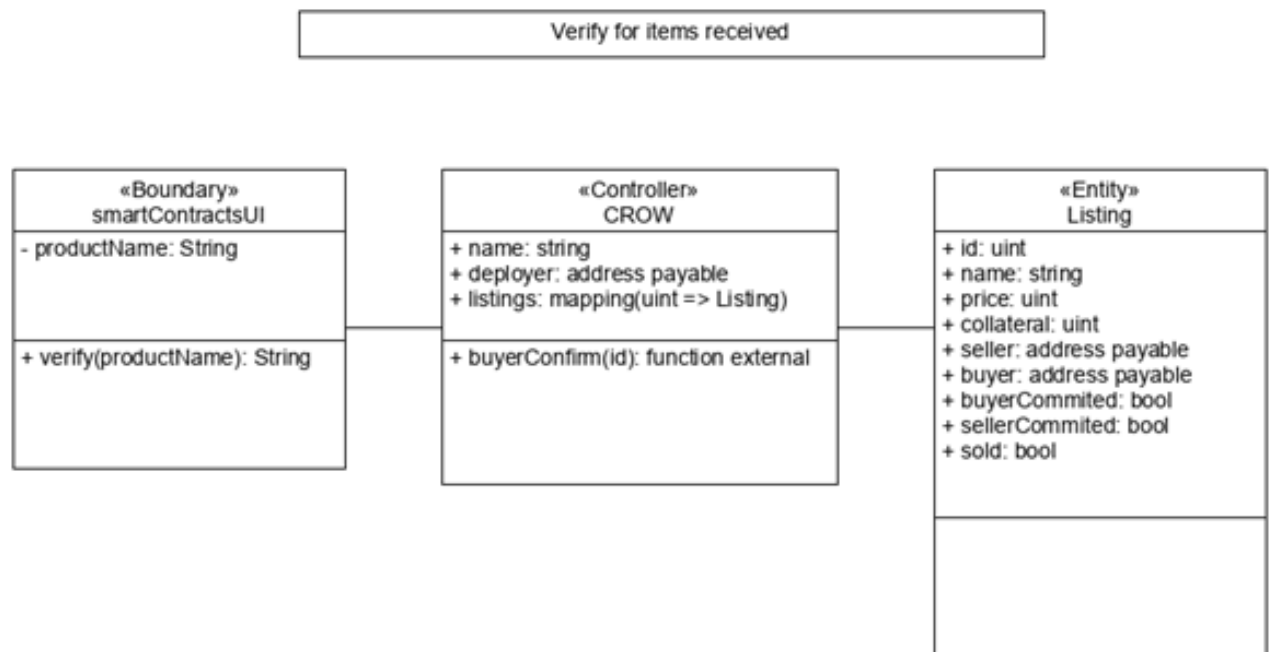
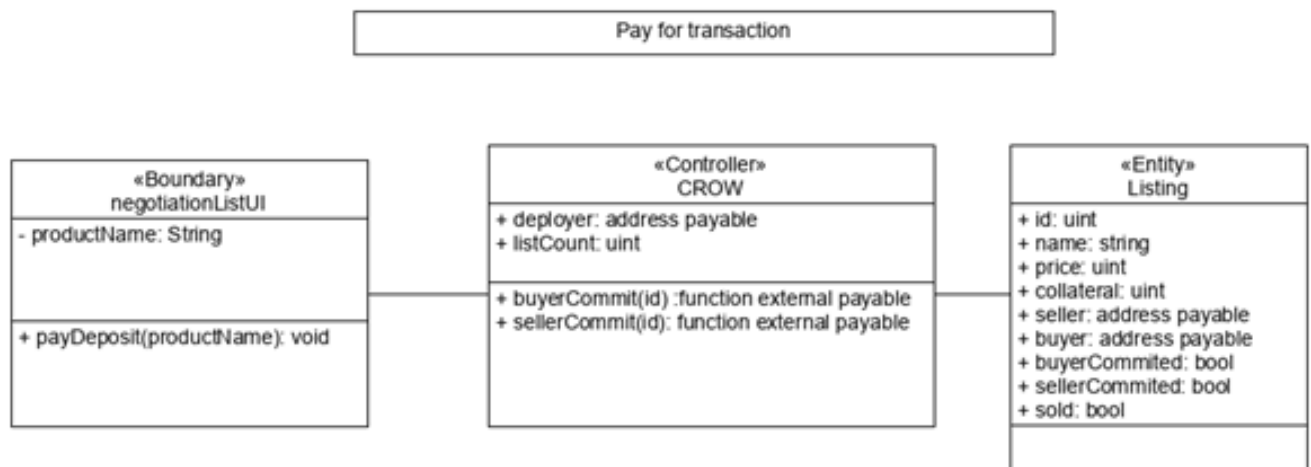
#U8	
Use case name	Agree to negotiation terms
Scenario	User wants to agree to a negotiation term
Triggering event	User clicks on 'accept' button
Overview	User agrees to negotiation term and can proceed with transaction
Actors	User
Pre-conditions	Other party has made an offer price for negotiation
Post-conditions	Terms will be accepted and negotiation will be transferred to the next process under Smart Contracts
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User clicks on 'negotiation' button 2. User arrives at list of negotiation page 3. A table will display the latest negotiation price for items in negotiation 4. Click on 'accept' button 5. End 	
Exception Flow	None

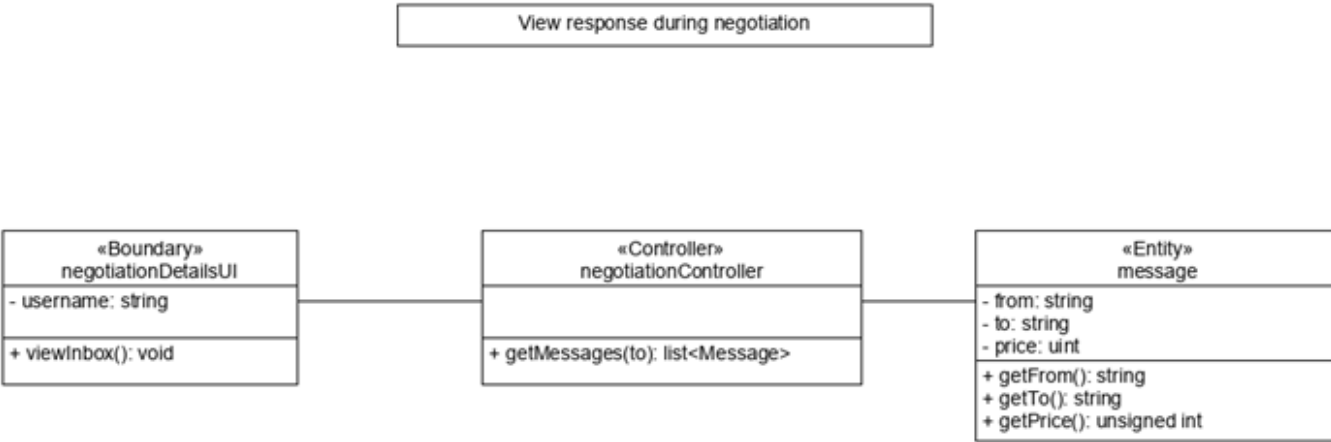
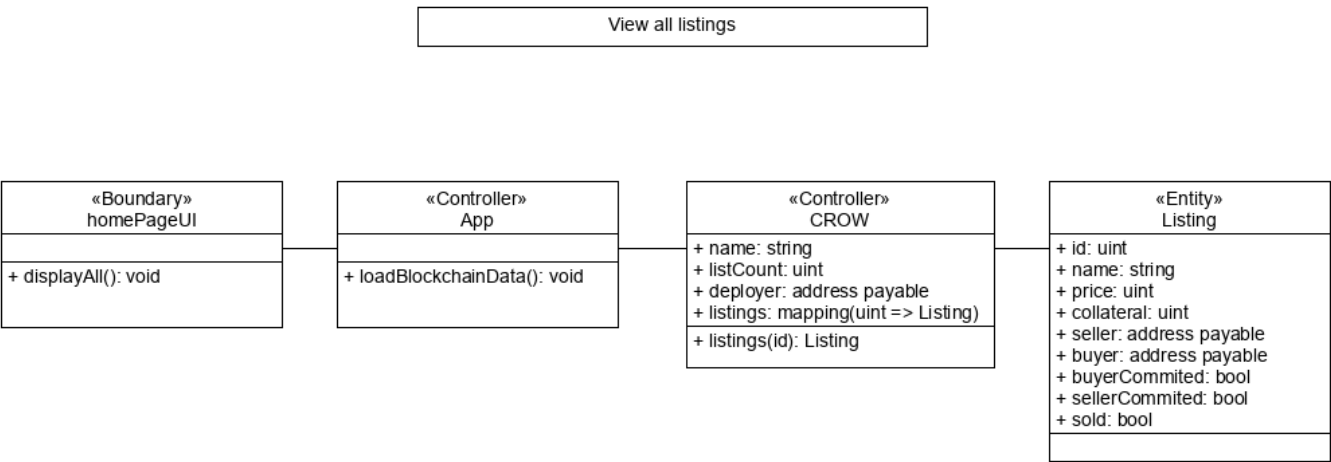
#U9	
Use case name	Negotiate terms
Scenario	User wants to negotiate on the price of an item
Triggering event	User clicks on 'negotiate' button
Overview	User sends a new price for negotiation
Actors	User
Pre-conditions	User must be on negotiation page and click on negotiate price button
Post-conditions	Updated negotiation price value will be sent to the relevant party
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User is at 'view all listings' page 2. User clicks on 'negotiate' button next to the listing he wants 3. User enter price he wants 4. User clicks on 'Negotiate' button 5. End 	
Exception Flow	None

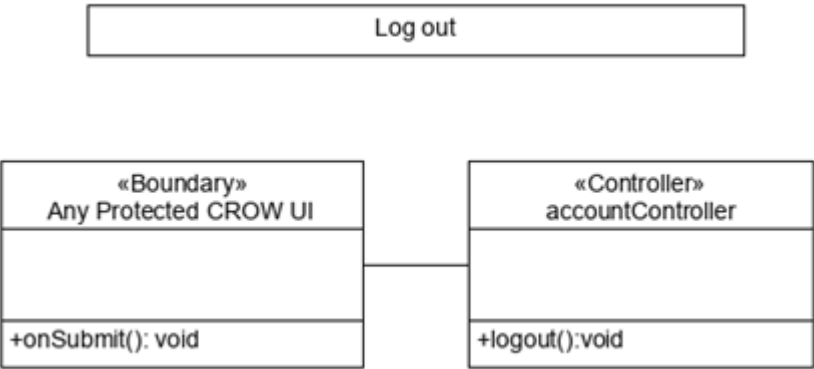
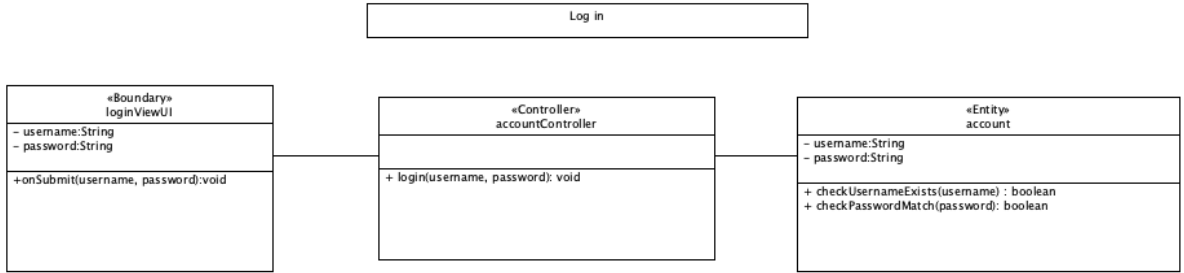
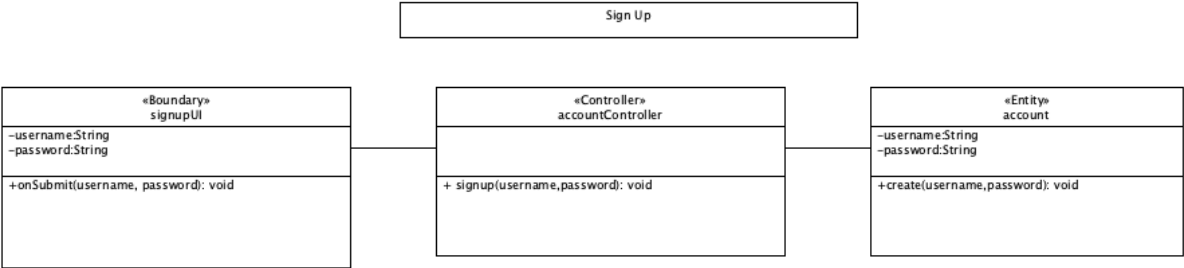
#U10	
Use case name	Search listings
Scenario	User wants to search for a listing that matches what he wants
Triggering event	User clicks on search bar and enters desired search value
Overview	User inputs a string, only listings with names that contain that string will appear
Actors	User
Pre-conditions	User must insert input into the search text field
Post-conditions	Matched results would be reflected
Basic flow of activities:	
<ol style="list-style-type: none"> 1. User is at 'view all listings' page 2. User enters what he wants to search in the search bar 3. User clicks on 'search' button 4. Table updates to results that contains user's input in product name 5. End 	
Exception conditions	None

#U11	
Use case name	List items
Scenario	Seller wants to list an item for sale
Triggering event	Seller clicks on 'Add listing' button
Overview	Seller can add an item he wants to sell as a listing
Actors	Seller
Pre-conditions	Seller must be on the Listing page
Post-conditions	An item would be listed onto the marketplace
Basic flow of activities:	
<ol style="list-style-type: none"> 1. Seller is at 'view all listings' page 2. Seller clicks on 'Add Listing' button 3. Seller enters input in 'Product Name', 'Price (Ether)', and 'Contact Info' text fields 4. Seller clicks on 'Add Listing' button 5. End 	
Exception Flow	None

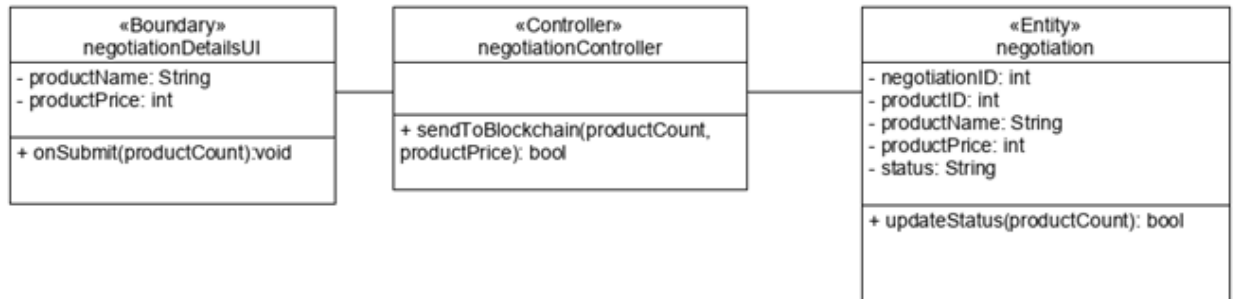
5.4 BCE Class Diagrams



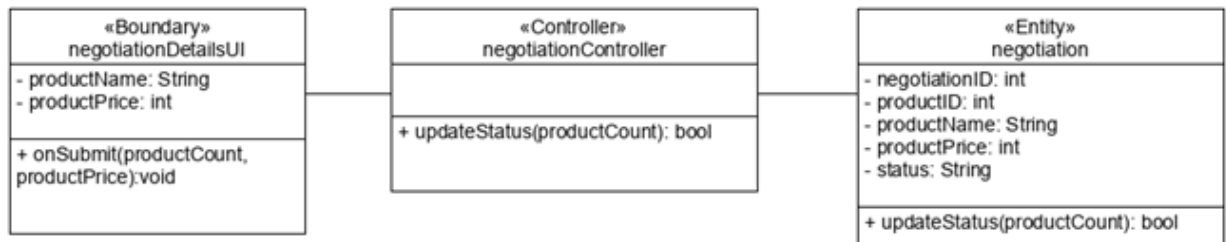




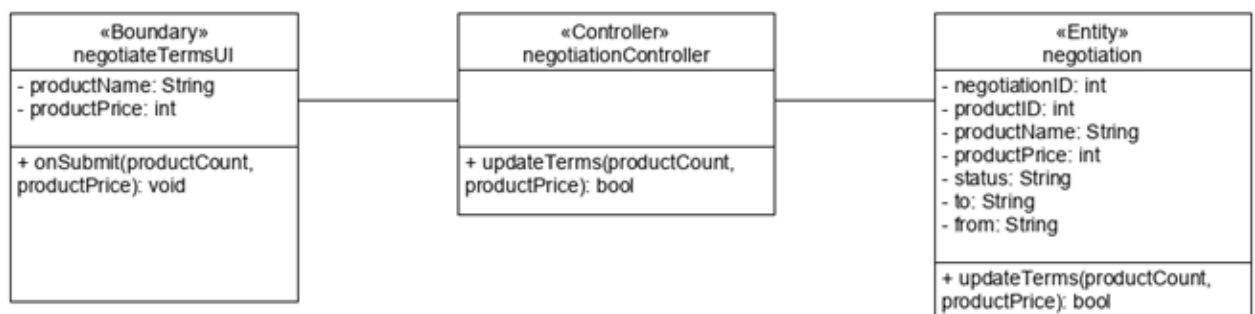
Agree to negotiation terms SELLER POV

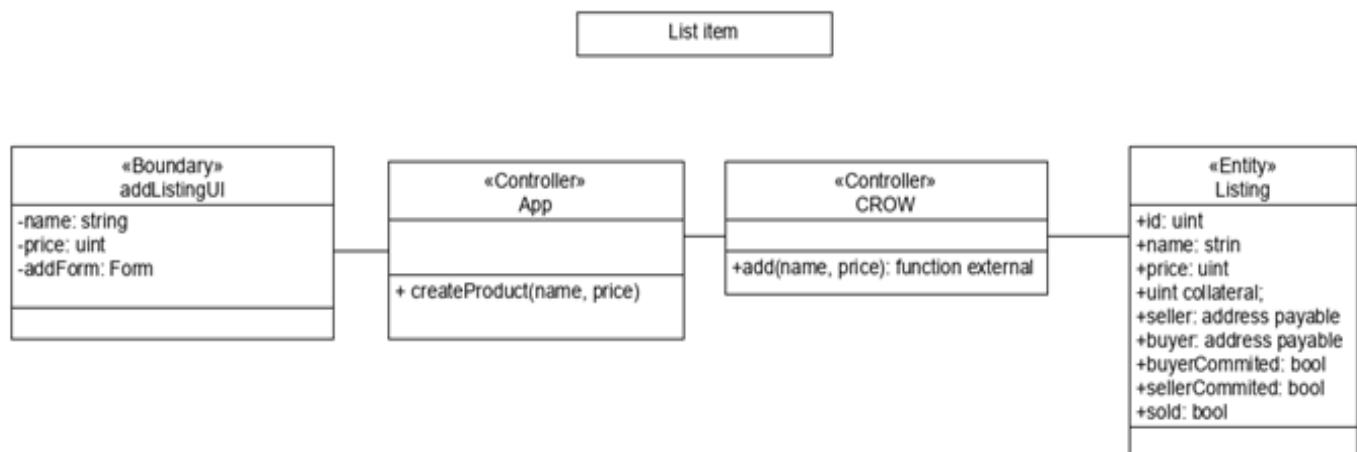
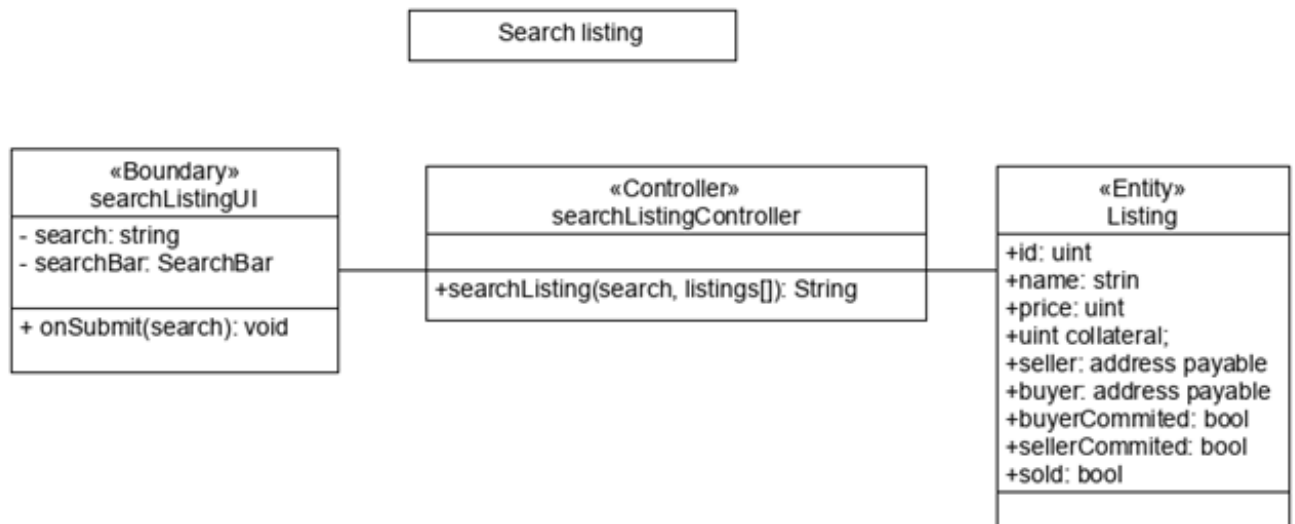


Agree to negotiation terms BUYER POV

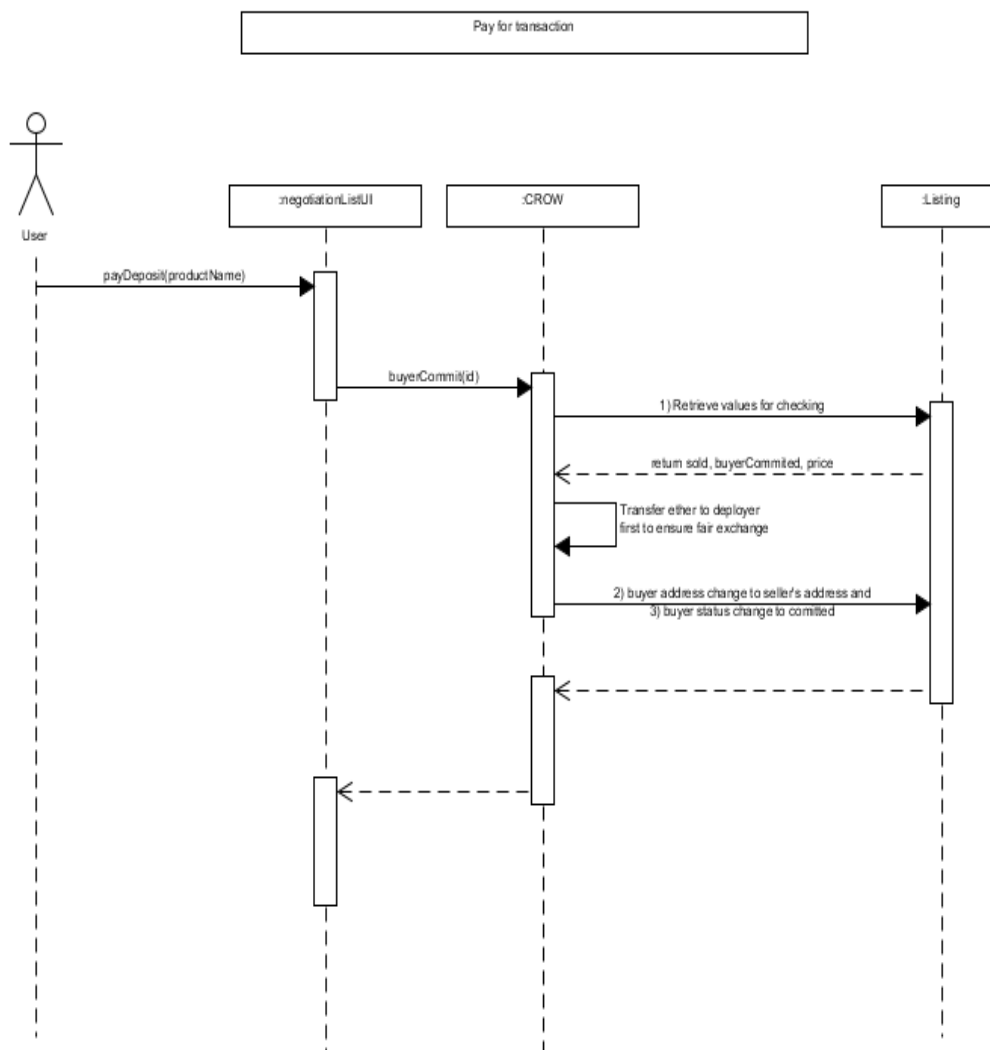


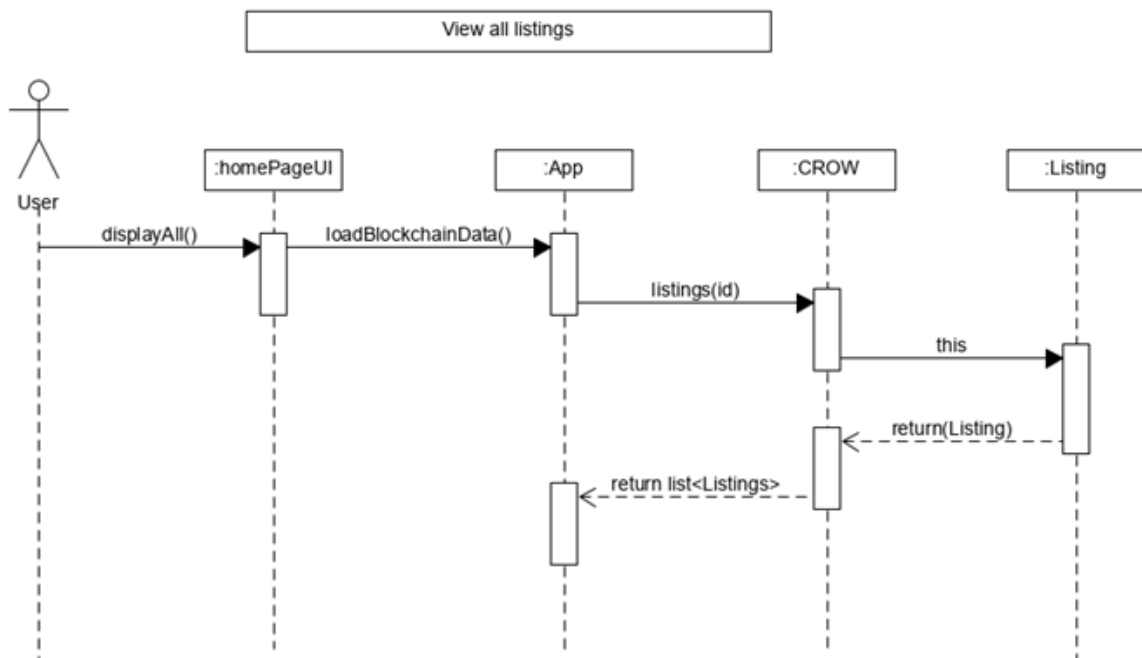
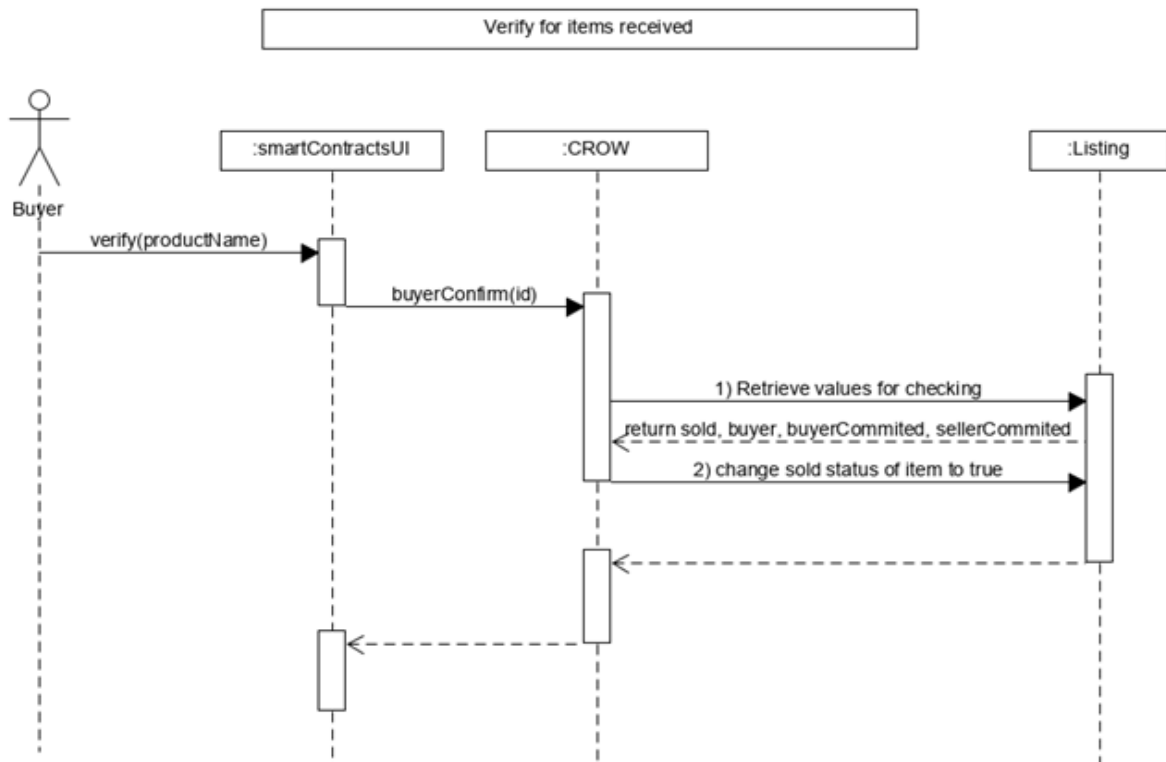
Negotiate terms

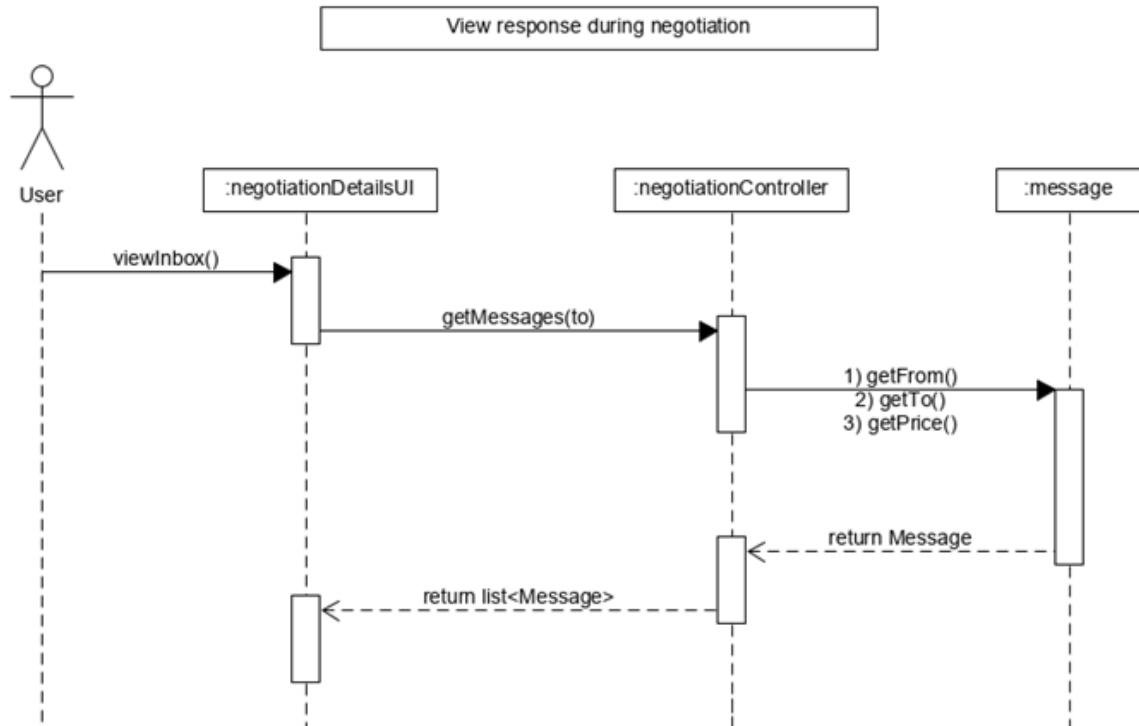


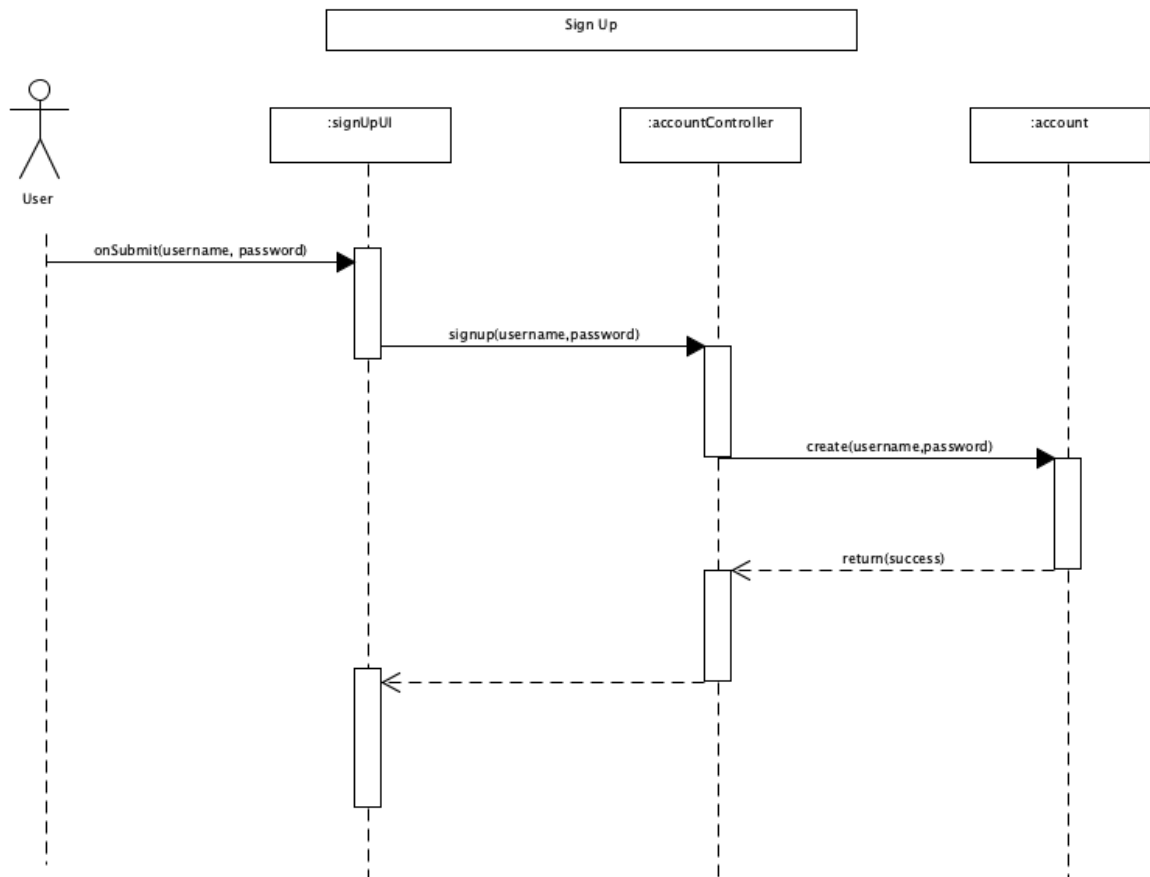


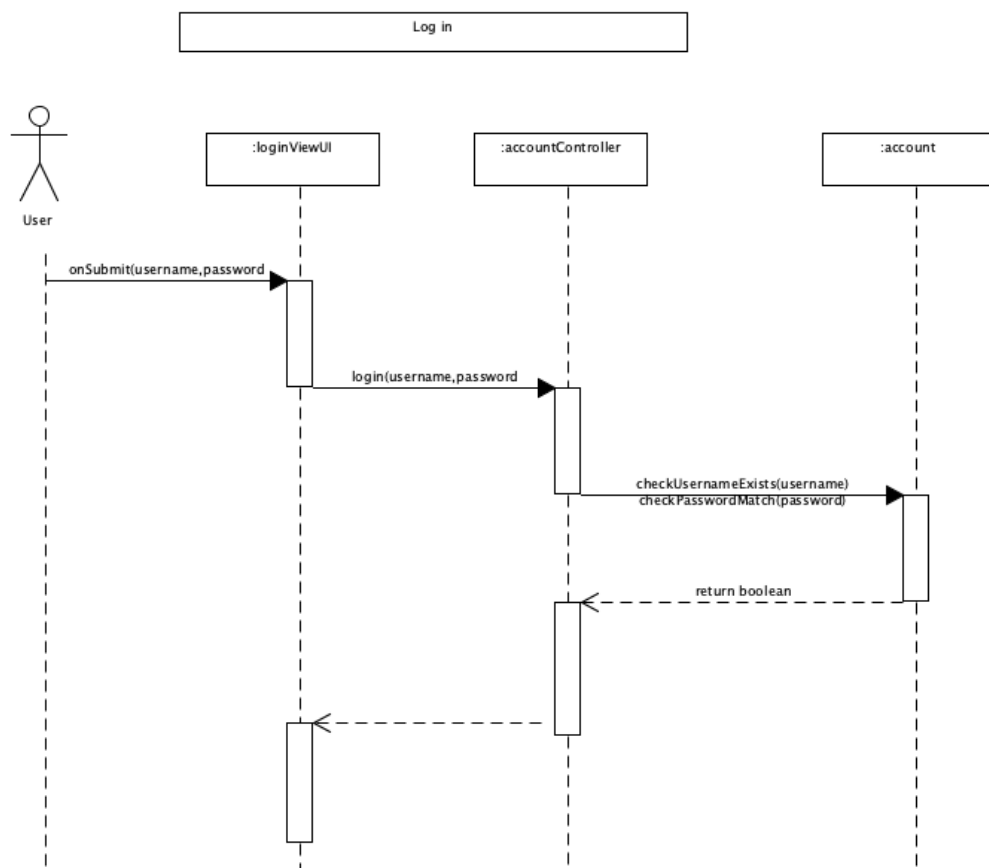
5.5 BCE Sequence Diagrams



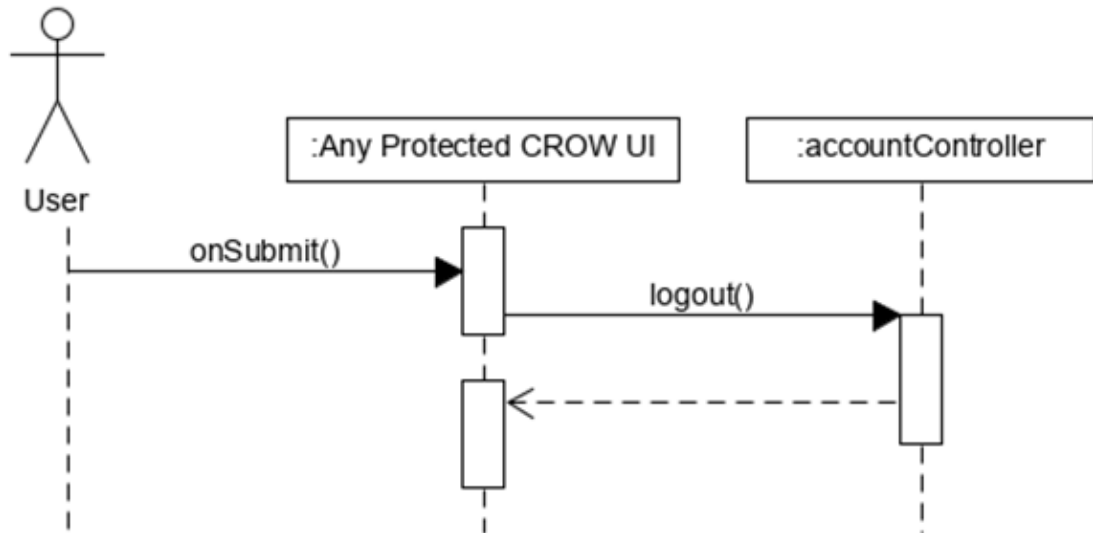




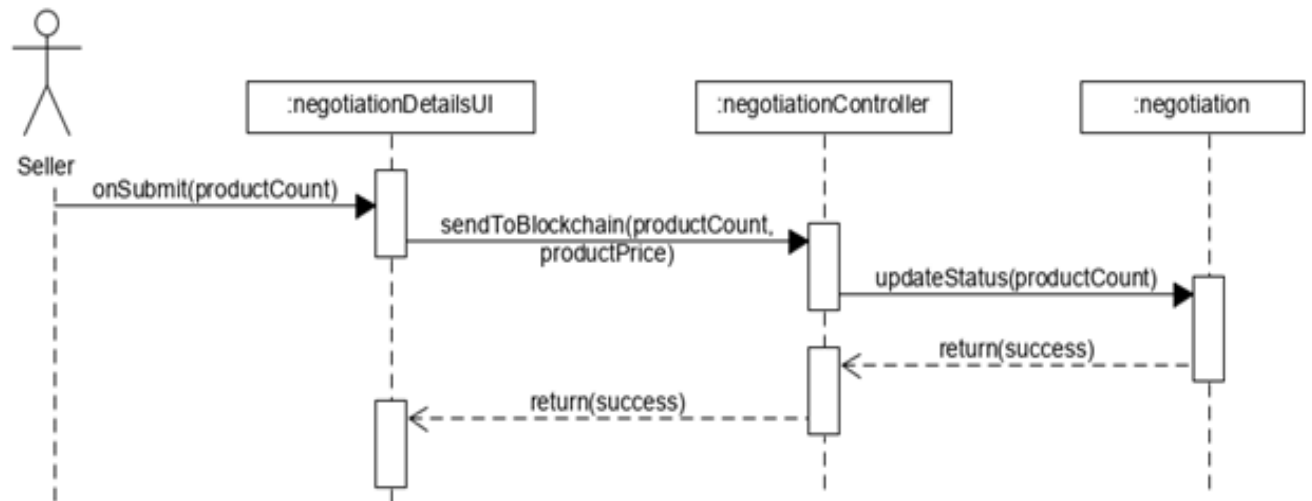




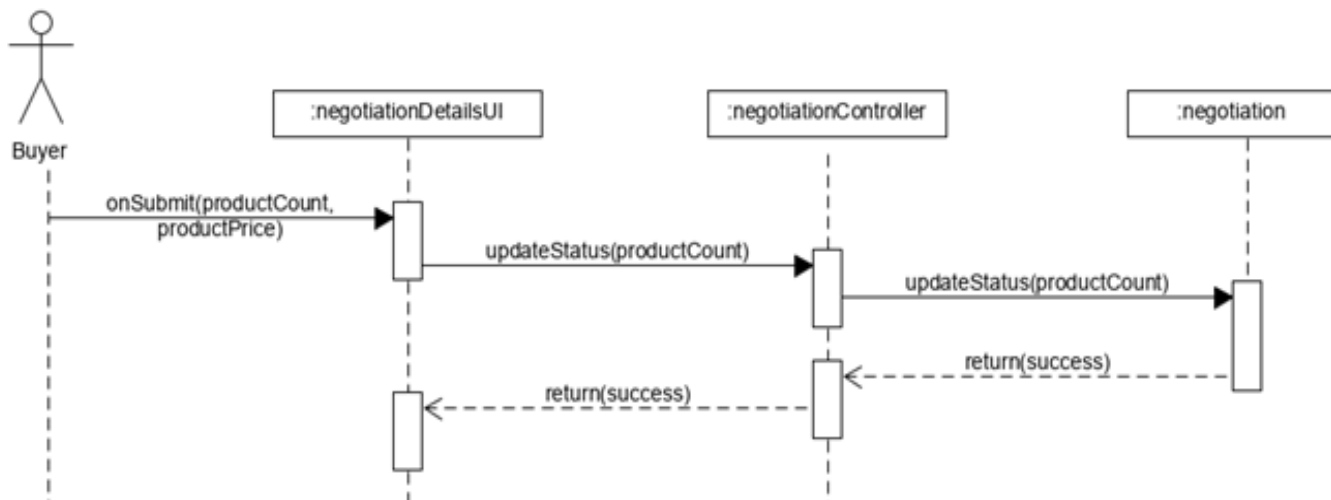
Log out

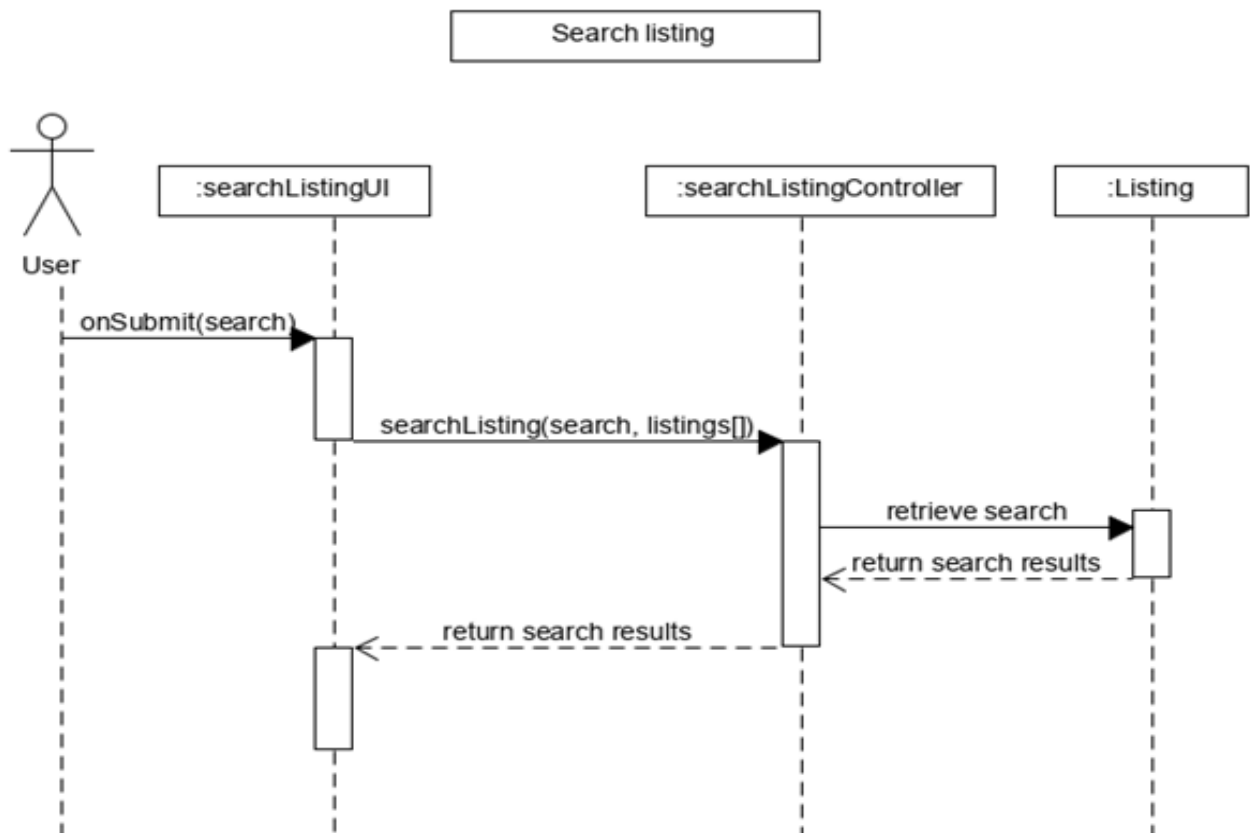
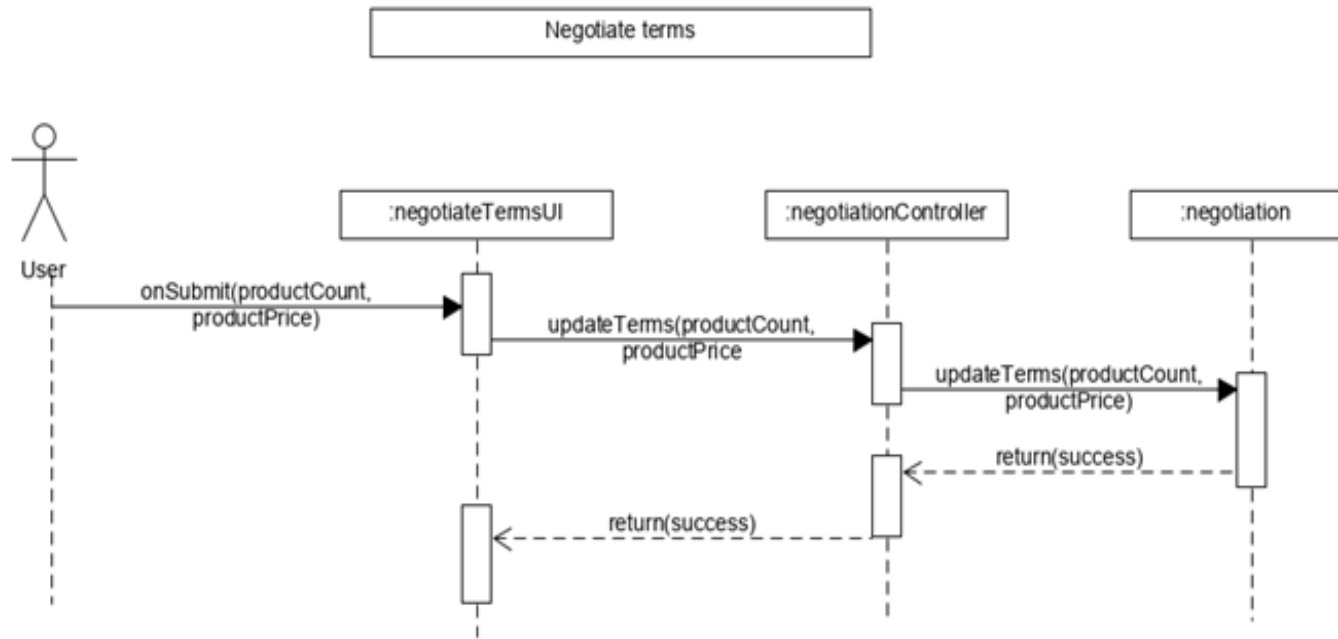


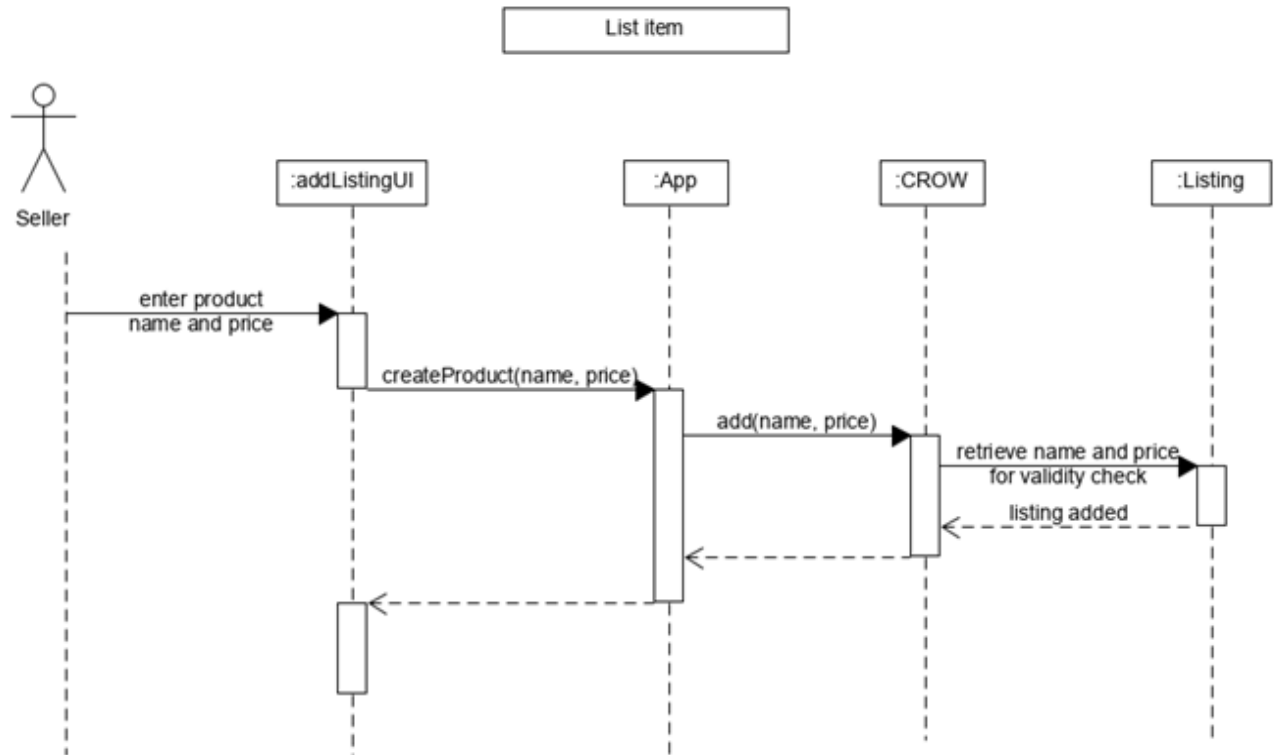
Agree to negotiation terms SELLER POV



Agree to negotiation terms BUYER POV



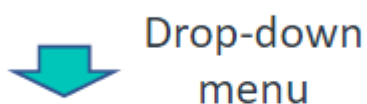




5.6 Wireframes

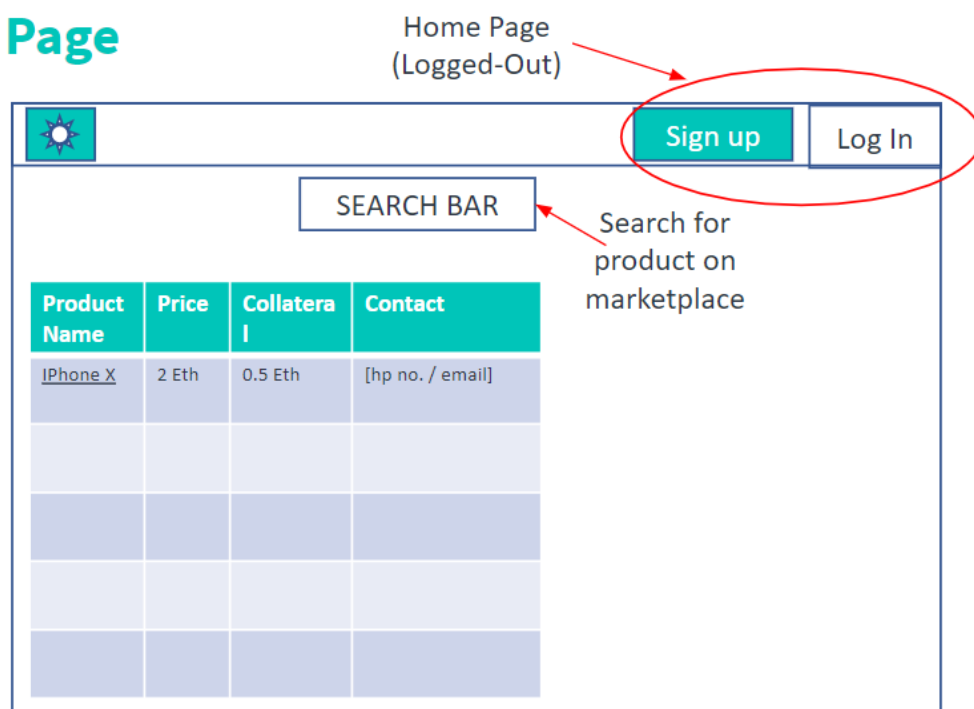
The team has drawn wireframes of some of the product's features. These wireframes are used as visual aids to better understand and align our perspectives and objectives towards our team's end product. The team has yet to include some feature's wireframe and is thus not displayed in this report.

LEGENDS



Home button

Home Page



Sign up

Home Page
(Logged-Out)

Sign up

Log In

Username

Password

Sign up

View All Listings

Upon clicking, will redirect to Add Listing page

Product Name	Price	Collateral	Contact
<u>iPhone X</u>	2 Eth	0.5 Eth	[hp no. / email]

Negotiate Buy

Negotiate Buy

Negotiate Buy

Negotiate Buy

Negotiate Buy

Add Listing

Product Name

Price (Ether)


Contact Info

Add Listing

Add listing to marketplace

User can enter hp. Number or email

Negotiate terms

 Listing Page Negotiations Smart Contracts **Log out**

Price

"Please input desired price"

Negotiate

Home Page (Logged-In)

View response during negotiation

Product Name	Product Price	Collateral	Contact Info	Action Required By
List of negotiations:				
Iphone X	3 Eth	0.5 Eth	peter@parker.com.sg	Buyer Negotiate 
				Seller Negotiate 
				Seller Negotiate 
				Buyer Negotiate 
				Buyer Negotiate 
				Seller Negotiate 

Redirect to negotiation page

Accept Offer

Reject Offer

Pay for Transactions and Verify items received

Listing Page

Negotiations

Smart Contracts

Log out

Smart Contracts

ID	Product Name	Product Price	Collateral	Status	Action Required By
bcs135	Iphone X	3 Eth	0.5 Eth		Buyer <div>Pay</div>
abs123					Seller
					Seller <div>Pay</div>
				In transit	Buyer <div>Verify Item Received</div>
					Buyer
					Seller

Pay for collateral

Buyer to Verify that the item has been received by the buyer

5.7 Functional Requirements

5.7.1 Description

No.	Functional Requirement Description
F1	The application must allow users to sign up for an account.
F2	The application must authenticate users through a username and password and allow users to terminate their sessions.
F3	The application must allow sellers to list items for sale and enter information such as name and price.
F4	The application must allow users to view a listing and browse between different pages.
F5	The application must allow buyers to initiate a negotiation and for both buyers and sellers to discuss and view the terms of the smart contract.
F6	The application must allow users to agree to the terms and establish a smart contract.
F7	The application must allow users to search for listings.
F8	The application must allow buyers to verify that they have received their item(s).
F9	The application must have a summary page to show all established smart contracts and ongoing negotiations.

5.7.2 Elaboration

No.	Related Functional Requirements	Description
1	F1	A sign-up button will appear on the login page. The button will direct the user to a form, which will have text boxes prompting for username and password. Password textbox will be masking the input values.
2	F1	The form submitted by the user will check for validity of username (if it already exists), and same values for password and confirm password. Password format will require at least 8 characters. No special characters will be allowed and maximum length will be 12 characters for backend operations purposes.
3	F1	If the values input by the user is valid, the information will be recorded in a database.
4	F2	On the login page, there will be a textbox prompting for username, another textbox prompting for password, and a "Login" button to submit the values. Password input values are masked.
5	F2	Upon submission, the system will compare the records to its database. If there exists a record with the submitted username, the system will then check for a match for its password. It will reject the user if either the username does not exist, or when there is a password mismatch. The system will remain in the login page if the user is rejected. If the user is authenticated, the webpage will direct the user to its homepage.

6	F3	User will click on the “List Item” button on the homepage. This will direct the user to a page with a form. Form will contain two textboxes to prompt for product name and price. Product name and price are both mandatory fields. If the price is negative or cannot be read as a number, or contains more than 2 decimal numbers, the listing will be rejected. A button to submit will be at the bottom of the form. If the mandatory fields are filled and the listing is not rejected, the information from the form will pass onto the system and redirect the user back to the home page. Otherwise, the page will just remain as it is.
7	F3	Once the system has received information of a new listing, it will upload that information onto the home page in a predetermined format.
8	F4	The user will click on the home button which will direct the user to the listings page. The listings page displays all available listings in chronological order, with 20 different listings in one page. A new page will be created for every new 20 listings, and users can navigate through these pages through a left arrow button to view newer listings, and a right arrow button to view older listings.
9	F5	In the listings page, there will be a “negotiate” button. This will direct the user to a form, to input his / her terms. The form will have a textbox to input a new price. A check on the price will be done, price cannot be negative and can only have 2 decimal places at most. Upon valid input, the new terms will be sent to the seller’s inbox.

10	F6	The user will be able to input their terms in the website before allowing them to click a button that will establish a smart contract and will direct to a page that has their terms and other information regarding the smart contract the user has established
11	F7	The user will be able to input their type of listings they would like to see in a search bar function. The user would simply have to input certain keywords that match of the listings in the blockchain
12	F8	The user will be able to go to the website page that displays their ongoing smart contract transaction and in that page the user will be able to click a button that indicates “verify” on the specific smart contract transaction. The verify button enables the user to acknowledge they have received their item(s) and that the smart contract is closed and settled.
13	F9	The website will have a smart contracts and negotiations tab. Through the smart contracts tab, the user will be able to view their established smart contracts and display information such as smart contract terms (product name, price and collateral) and actions to be taken.. The negotiations tab that the user can access will display all the ongoing negotiations the user has made and will display information such as product name, price , collateral, negotiation response from the other party and action the user needs to make.

5.8 Non-Functional Requirements

5.8.1 Availability

The product should be portable. The product should pose no compatibility issues when using different types of devices or browsers.

5.8.2 Data Integrity

Users will not be able to update or key in the wrong type of data in certain fields when using the website. For example, keying in letters into an input field that requires only numbers or date as the input for it to update into the system.

5.8.3 Integrity

The smart contract can only be deployed on the blockchain only when both buyer and seller mutually agree to the terms of the contract. Contracts that have been deployed are immutable which means they can't be changed and no one can tamper with or break a contract. The contracts are also distributed to allow the outcome of the contract to be validated by others in the network. Distribution deems it impractical for a cyberattack to seize command over the handling of the funds, as all other users would detect such an attempt and mark it as invalid.

5.8.4 Performance

The website should be able to load in 3 seconds when the number of simultaneous users are less than or equal to 10000. It should also be capable enough to handle 100,000 users without affecting its performance during runtime.

5.8.5 Recoverability

The website will be able to recover crucial data directly affected in the event of a system failure and the time required to get the system back up to working conditions will not exceed one hour.

5.8.6 Reliability

The product is to maintain an acceptable level of performance while the system is under harsh conditions such as overloaded amount of users, simultaneous uploads and heavy traffic.

5.8.7 Security

Passwords of our users are stored using a cryptographic hash function instead of storing in plaintext.

6. Project Progression

6.1 Week 1 - 15 Oct to 21 Oct

- Researched on what is a blockchain and how we could utilize it with smart contracts.
- Tied down roles and responsibilities between group members.
- Run through list of compatible languages and decide as a group which to use
- Narrow down an industry to target for our project by getting each member to present a use case at the end of the week. Group then voted for the best use case.
- Decided on project team software development methodology (agile, waterfall & etc)
- Chose project tracking tool to manage team progress
- Created repository in github for group project to store the source code

6.2 Week 2 - 22 Oct to 28 Oct

- Change of the type of platform from payment gateway to a carousell like system
- Discussion on to whether create a website or a mobile application for users to use
- Researched on how the payment would be done on the website.

6.3 Week 3 - 29 Oct to 4 Nov

- Modified and finalised our user case stories
- Discussion on how the final product and creating wireframes to illustrate it

6.4 Week 4 - 5 Nov to 11 Nov

- Group reviewed user stories
- Completed use case diagram, use case descriptions, functional and non-functional requirements
- Collated the first draft of requirements specifications
- Discussed on the proposed technology stack required for the project
- Met up with project supervisor (Kheng Teck) to review requirements specification document
- Team worked on identifying risks related to project

6.5 Week 5 - 12 Nov to 18 Nov

- Created first draft of wireframes
- Reviewed and finalised requirements specification document

6.6 Week 6 - 19 Nov to 25 Nov

- Discussed about how to move forward with regards to submission on 2nd Jan and agreed to take a break next week to focus on our final's
- Team decided that the focus for the next weeks should be:
 - Work on learning programming languages that is required for the project - Solidity, HTML, CSS, Javascript
 - Research on where the web application can be hosted
 - Research on which database to use to store data for web application

6.7 Week 7 - 26 Nov to 2 Dec

- Team took a break to focus on finals

6.8 Week 8 - 3 Dec to 9 Dec

- Team looked into self-learning tutorials for Solidity (Pet-Shop tutorial from TruffleSuite)
- Explored different hosting methods for marketing site, including using Github pages

6.9 Week 9 - 10 Dec to 16 Dec

- Decided on Github pages for hosting our live site
- Performed peer-learning through physical meetup, helped each other with setting up the development environment and tried to the best of our abilities to clear each other's doubts about Solidity source code and how it behaves.
- Team set up a remote database (postgresql) on heroku to store account and negotiations related data

6.10 Week 10 - 17 Dec to 23 Dec

- Working on the project source codes

6.11 Week 11 - 24 Dec to 30 Dec

- Work on updating the existing wireframe
- Updating the use case diagrams
- Updating the use case descriptions
- Updating on the bce diagrams
- Updating on the sequence diagrams
- Finished frontend prototype

6.12 Week 12 - 31 Dec to 6 Jan

- Preparing slides for the coming prototype presentation

- Presented our mid-point prototype demo

7. References

NG, C. (2020). \$1.3m lost to e-commerce scams in first quarter of this year. Retrieved 13 November 2020, from <https://www.straitstimes.com/singapore/courts-crime/13m-lost-to-e-commerce-scams-in-first-quarter-of-this-year>

LOW, D. (2020). At least \$380k lost to e-commerce scams for electronic products in first 3 months of year. Retrieved 13 November 2020, from <https://www.straitstimes.com/singapore/courts-crime/at-least-380k-lost-to-e-commerce-scams-for-electronic-products-in-first-3>

Higher profits, lower costs when you sell online on QuuBe. (2019). Retrieved 13 November 2020, from <https://www.straitstimes.com/business/higher-profits-lower-costs-when-you-sell-online-on-quube>