DANIAH AFTAB
01-134162-007
NOOR HODA
01-134162-034

Dragon Fusion- A Decentralized Application.

Bachelor of Science in Computer Science.

Supervisor: Ms. Romana Talat.

Department of Computer Science
Bahria University, Islamabad.

July 2020.

Certificate

We accept the work contained in the report titled “Dragon Fusion - A Decentralized Application”, written by Miss. Daniah Aftab and Miss. Noor Hoda as a confirmation to the required standard for the partial fulfillment of the degree of Bachelor of Science in Computer Science.

Approved by . . . :

Supervisor: Ms. Romana Talat (Senior Lecturer)

____

Internal Examiner: Name of the Internal Examiner (Title)

____

External Examiner: Name of the External Examiner (Title)

____

Project Coordinator: Zubaria Inayat (Lecturer)

____

Head of the Department: Dr. Muhammad Muzammal (Sr. Associate Professor)

____

July 2020.

Abstract

The revolutionary technology effecting different industries widely, introduced in the market with its very first modern application “Bitcoin”. Bitcoin is a digital currency known as cryptocurrency that can be used in the place of paper money while trading online. There is a misunderstanding among the people that the concept of Blockchain technology and cryptocurrencies are the same, which is not the case. One of the applications of the Blockchain technology includes the development of cryptocurrencies other than Bitcoin. There are numerous applications developed by Blockchain technology that include Ethereum as well as Hyperledger. Ethereum and Bitcoin are categorized as public blockchains which as compared to private blockchains are nowadays less preferred because of its scalability issues, less secure network, increased chances of potential collisions, slow working and many more. However, the whole topic of public and private blockchains is not taught in daily lectures because it is an advance topic in the field of Computer Sciences and further new discoveries are being done in this particular field. People nowadays are really concentrating on investigating new ideas for the increment of their money. A number of gaming applications are developed using Blockchain technology providing the opportunity to invest their money digitally in the form of cryptocurrencies in a more entertained and amused atmosphere. Cryptokitties is a fine example of this, which is based on the concept of Ethereum. The main objective of this project is to develop a decentralized gaming application based on the concept of Blockchain using Hyperledger that would help people get familiar with Hyperledger that is one of the most emerging technology in the software industry nowadays. It would encourage the users to learn more features of the private blockchain thus solving the issues of the public blockchains and providing them with a completely interesting and joyful environment. [1]

Acknowledgments

We, as the team members of the project would really like to thank our supervisor “Ma’am Romana Talat” for investing her tireless efforts and time in this project. She always has been very helpful and supportive to us. Her experience in the field of Computer Sciences is extraordinary and the way she treats her students has been a great source of inspiration to us to work with our full determination and motivation.

DANIAH AFTAB, NOOR HODA.
Bahria University Islamabad, Pakistan.

July 2020.

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

EVM - Ethereum Virtual Machine
IBM - International Business Machines
IOT - Internet of Things
ISP - Internet Service Provider
SQL - Structured Query Language
CSS - Cascading Style Sheet
LINQ - Language Integrated Query
OOP - Object Oriented Programming
NPM - Node Packet Manager

Chapter 1

Introduction

Blockchain technology enables distributed public ledgers that hold the data in a more encrypted and secure way and it ensures that the transactions can never be altered or changed. The ledger is distributed across the whole network to make any alteration of data very difficult. Nowadays, the terms Blockchain and Bitcoin are being used interchangeably, which is not the case. Bitcoin is merely an application of Blockchain. [2] In this technology, there is no middle party or authority between the sender and receiver that would cause any type of manipulation of the data. Both the parties on either sides interact with one another directly. Another attribute of this technology is that, once the data is added in the Blockchain then it becomes almost impossible to change it , hence making the data immutable. [3] Applications of this technology includes data sharing, royalty protection, asset management, monitor supply chains, financial transactions, immutable data backup and many more.

Blockchain works with blocks where each block is a collection of data. The data is added in each block and then various blocks are linked together thus forming a Blockchain network. The first block in this network is known as genesis block. [4]

There are almost infinite number of benefits of the Blockchain technology. [5] Some of them are as follows:

1. In Blockchain technology, the transactions, which take place, are completely transparent and only the authorized users can have access to change the data. [6]

2. The Blockchain does not involve the interruption of any third party that makes the transactions speedy and fast thus increasing the processing time. [6]

3. Blockchain provides much more security than other networks as each transaction in it is completely encrypted and is linked to the previous transaction in the network. [7]

4. The data structure used in this technology is append-only. Hence, the data cannot be updated or changed easily. [7]

5. It lowers the risk of any false data or duplicate changes in the transactions as various number of consensus algorithms are required to validate any new entry in the network. [8]
6. The transactions that take place in the Blockchain network are stored on thousands and millions of computers in the form of blocks, which makes it impossible for lost data to not be recovered. [8]
7. It also results in the reduction of cost, as there is no middle person available between two parties that are participating in the transaction. [8]

A smart contract is a computer program that provide a set of rules or conditions that govern the transfer of digital currencies that takes place between the parties. A smart contract, also known as cryptocontract, not only declares the rules of the agreement but also enforces these obligations automatically. This process takes place by receiving the information as input and then assigning a value to each of the input by using the rules that are defined in the contract and then implementing the actions that are obliged by the clauses prescribed. [9]
Ethereum is a decentralized, open source, Blockchain based distributed computing platform, which is another application of Blockchain technology. It enables the development of smart contracts and decentralized applications. Ethereum uses the built-on cryptocurrency known as “Ether”. It provides the platform for generalized applications and business to consumer businesses. The smart contracts in Ethereum are usually written in Solidity. It runs on EVM i.e. Ethereum Virtual Machine which is a virtual network. [10]
Hyperledger is an open source platform created by Linux foundation in the year 2016. It has intended to create industry specific platforms and industries providing high level of confidentiality, flexibility, security, scalability and privacy. [11] It is a major project containing many sub-projects within, like the following:

1. Hyperledger Fabric.
2. Hyperledger Indy.
3. Hyperledger Burrow.
4. Hyperledger Composer.
5. Hyperledger Iroha.
6. Hyperledger Sawtooth and many more.

It is used by many large industries and organizations like banking, financing, IOT and the like. It intends to build a platform where various Blockchain or software developers can meet to develop and maintain Blockchain frameworks on a large scale.

As stated clearly above, it can be proved that Hyperledger works differently than Ethereum in many aspects. Ethereum is a public blockchain whereas Hyperledger is a private blockchain. Ethereum provides the transactions to be completely transparent and open to public in order to access or read it. However, in Hyperledger, transactions are completely confidential. Ethereum is the most preferred framework for the business-to-consumer businesses while Hyperledger is mostly used for business-to-business platforms.[10]
"Dragon Fusion" is a Blockchain game based on the concept of Hyperledger. In this game, we buy, sell and breed different types of dragons.

1.1 Problem Description

All the blockchains that we are familiar with are public blockchains like Bitcoin and Ethereum. They are completely open ecosystems where anyone can take part in it, hence this system is not preferred by many business organizations, as the transactions are completely transparent and do not provide any confidentiality. However, every business and industry is distinctive in their own way and the applications serving to their requirements must be personalized.[12]

1.2 Project Objective

"To design a decentralized application which is based on the concept of smart contracts using the framework i.e. HyperLedger."

1.3 Project Scope

The description of our project is to "build a gaming application which is based on the concept of Hyperledger Blockchain."
The main product of this project will be a decentralized gaming application that applies the ideas of smart contracts through Blockchain technology.
As Bitcoin and Ethereum, Blockchains have their own limitations that include non-confidential transactions and mining, which involves more power consumption and wastage of time.
The main assumptions for this project is the availability of required guidelines from the internet or books that would provide enough source of information. Complete support of the Hyperledger tool is also required which would be useful for the completion of our gaming application. [13]

1.4 Benefit

Our decentralized gaming application will be very beneficial to the users in the following ways:

1. It will help the people to get more familiar with one of the most developing concept of this modern world known as Blockchain technology.
2. Using our application, the users will not only be able to earn coins through the breeding process of the dragons but will also be able to explore the further features of the application in an entertained environment.
3. As the application is based on private blockchain, the user will be provided with full facility of privacy and security of his/her data. The breach of the user's data such as personal information, wallet data or transaction history is almost impossible.
4. The application allows the user to enter in the world of virtual dragons and breed them in any way desired providing aesthetically pleasing designs and graphics.

1.5 Outline:

This report consists of total seven chapters. Chapter 1 includes the section of introduction, problem description, and the objective of the project, project scope and the benefits of the project. Chapter 2 includes the literature review of the previous studies that are relevant to the topic and the remaining chapters of the report are organized as follows:

Chapter 2

Literature Review

Blockchain technology was first introduced by Satoshi Nakamoto in 2008. The first application of this technology was Bitcoin, which made it the very first digital currency thus introducing the concept of cryptocurrencies in the computer industry. With the passage of time, many applications of the technology like Bitcoin were developed thus giving rise to completely new concepts like Ethereum and Hyperledger. In the beginning, many researchers were working on public blockchains (i.e. Bitcoin and Ethereum) and as a result, many applications came into being such as CryptoKitties, and the like. There are a number of gaming applications based on the concept of Ethereum blockchain such as CryptoKitties that provides the user to play with their digital currencies in an aesthetically pleasing environment. These types of applications were well recognized in the market and became popular among people that were interested in cryptocurrencies and Blockchain technology. Since Ethereum is a public blockchain, it has its own limitations such as security issues, scalability issues and many more. In the beginning of popularity of CryptoKitties, this application began to clog the network thus making the transactions very slow and taking comparatively more time for a kitty to be born. Thus, such gaming applications have their own limitations. As a result, public blockchains were found to be less secure, slow, trustless, consumes more energy and increased risk of potential risks as compared to the private blockchains, which is now preferred by many business organizations and companies. There are almost countable developers of Hyperledger in the market. However, a large number of researches are being done in the field of private blockchains that includes Hyperledger in order to discover it further and apply the concepts of it in the practical life. [14] In this section, all the concepts, methods and techniques relevant to the Hyperledger technology will be discussed in detail.

2.1 Blockchain Technology:

Blockchain technology is a distributed database that structures the data without the need of a central server or a middle party thus satisfying many aspects of confidentiality. Blockchain is a growing list of records known as blocks that are linked together by cryptography. Each block in the blockchain consists of a cryptographic hash of the previous block, transaction data and timestamp. [15] This technology is immutable i.e. no changes can be made in the transactions once they are packed in the Blockchain. In this technology, payments can be done completely without the involvement of any bank a middle party, hence it can be used in various financial services such as online payments, digital assets and many more. It can also be used in various fields such as security services, public services, Internet of Things (IoT) and the like. The business companies and organizations that require a high level of confidentiality and reliability uses this technology in order to achieve these goals and attract more potential customers. [16]

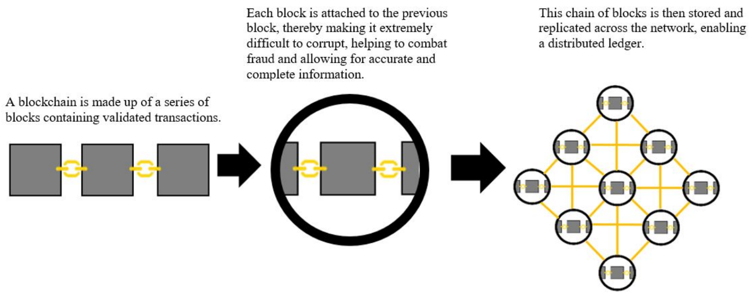

Each block is attached to the previous block, thereby making it extremely difficult to corrupt, helping to combat fraud and allowing for accurate and complete information.
This chain of blocks is then stored and replicated across the network, enabling a distributed ledger.
A blockchain is made up of a series of blocks containing validated transactions.

Figure 2.1: Distributed Ledger.

2.2 Decentralized Applications:

Decentralized applications has captured maximum media coverage due to its wide features. Decentralized applications are the type of applications that are not controlled by any owner or a single authority. They cannot be shut-off and cannot have a downtime. [17] A decentralized application must consist of all of the four characteristics which includes to be open source, decentralized, incentive and has an algorithm or protocol (i.e. it has inbuilt consensus algorithm and generates tokens). [18] If the application lacks any one of the characteristics then it is not considered a decentralized application. [19] [20]

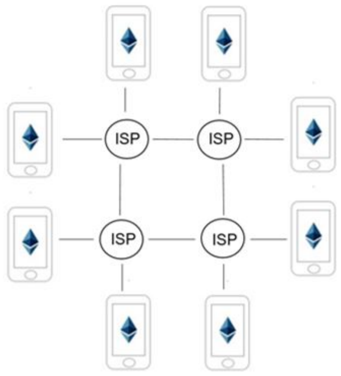


Figure 2.2: Decentralized Application.

2.3 Smart Contracts:

Before initializing any type of transaction, both businesses and organizations must define a set of contracts that would include common processes, terms, data and many more.
These set of contracts then define a layout that administers all the transactions that take place between these two businesses. Smart contracts are converted into executable program in the Blockchain network.
In Hyperledger, smart contracts are deployed to serve the purpose of carrying out all the transaction requests. It also executes business rules in order to validate and process the transactions.
The process of smart contracts in Hyperledger starts when the id of the contract is sent as a transaction request to the smart contract in order to be validated. If the smart contract does not validate or approve that specific transaction request, then the contract is not processed to the next level. On the other hand, if the smart contract validates the transaction request, the package of transaction then proceeds towards the consensus service during which it is available in the center of the processing unit. Then, the particular transaction is decided whether it will be sent and get committed to the Blockchain or not. [9]

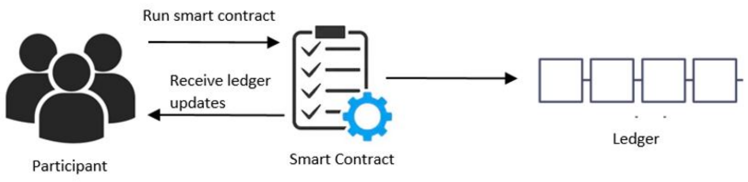


Figure 2.3: Smart Contract in Hyperledger.

2.4 Hyperledger Fabric:

Hyperledger Fabric is an open source platform that was founded by Linux foundation and is now maintained by both IBM and Linux foundation. It is one of the project of Hyperledger that is further divided into five subprojects. Hyperledger Fabric does not consist of any cryptocurrency like Ethereum and Bitcoin. In Fabric, the network is restricted to its network members only and no one else can have access to it. The transactions in Hyperledger Fabric are controlled by chain code, which is a program code that provides the ability to design and develop applications in order to interact with the network. [21] In the network, the privacy of transactions between the network members can be obtained by using channel, which is an isolated mechanism. This channel ensures that the data and transaction is available only to the members of the network. Hyperledger Fabric also allows the user to set up endorsement policies around the execution of chaincode. These policies define the user that needs to be in agreement with the results before added to the ledger. [22]

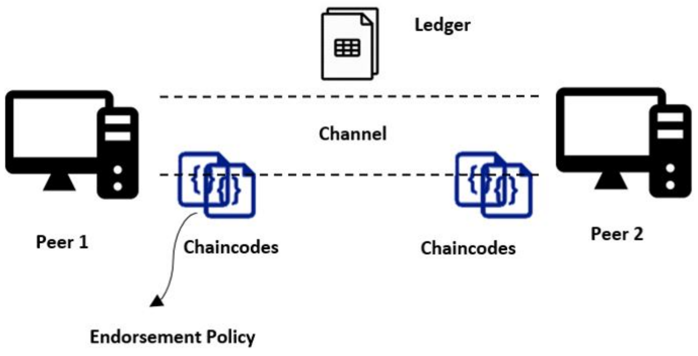


Figure 2.4: Chaincode in Hyperledger Fabric.

2.4.1 Architecture of Hyperledger Fabric:

Typical Blockchain platforms, private as well as public blockchains, follows a sequential order of execution style where transactions are ordered first and executed later. However, Hyperledger Fabric uses a different architecture, which is known as execute-order-validate architecture.[23] This architecture allows the transactions to execute before the Blockchain reaches consensus on their place in the chain. This architecture supports flexible trust assumptions, security issues such as performance attacks and scalability issues. [24]

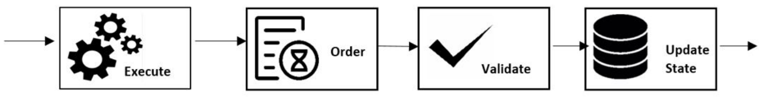


Figure 2.5: Transaction Flow in Hyperledger.

2.5 Hyperledger Composer:

Hyperledger Composer consists of different set of tools that provide the developers with a much easier approach in order to develop Blockchain network related applications which are not only used to solve a wide number of business problems but it is also used to improve the efficiencies of the network's operations.
Hyperledger Composer provides support to the infrastructure of the current Hyperledger Fabric networks, which means that it supports consensus protocols of Blockchain in order to ensure that every transaction in the network is working according to the policy designed by the participants of that specific Blockchain network. [25]
Hyperledger Composer proposes a wide range of advantages as follows:

1. It consists of a large number of tools that provides an ease to a non- technical person to work with it and build different features.
2. It makes the development of Blockchain network applications easy and time saving.
3. It also implements the need of any local or additional installations by supporting the use cases, which then can be executed through the web-based Hyperledger Composer.
4. Hyperledger Composer also provides with the facility of sharing and reusing different components among several number of business organizations.

2.5.1 Architecture of Hyperledger Composer:

The architecture of Hyperledger Composer allows the developers to acquire the full stack solution of Blockchain that means, it allows the business logic to deploy on the Blockchain application. The logic of the Blockchain is imposed by the use of REST APIs. It also integrates the Blockchain with existing enterprise of Blockchain.
Hyperledger Composer initially consists of the following components:

2.5.1.1 Execution Runtimes:

Hyperledger Composer supports multiple pluggable runtimes especially the following implementations:

1. Distributed ledger of the Composer consists of Hyperledger v1.1.
2. Second main implementation is web, which is used in developing different web pages.
3. Embedded is the last implementation which is used to for the unit testing of the business logic.

2.5.1.2 JavaScript SDK:

JavaScript SDK of Composer is fundamentally a collection of Node.js APIs that are used to create the application and used to interact with the business logic of the Blockchain. The APIs of the Blockchain are divided into two NPM modules that are composer-client and composer-admin.

2.5.1.3 Command Line Interface:

Command line interface tool of Hyperledger Composer provides the user with the facility to develop and manage different definitions of the business network.

2.5.1.4 REST Server:

REST Server tool of the Composer develops an Open API, which is known as Rest API for the business logic of the Blockchain. REST Server is initially developed using the loopback technology. It is used to convert the business logic of the Blockchain into an Open definition of API and it provides support to the functions like create, update, read and delete to the assets and participants. It also allows the transactions to be involved in the processing function of the Blockchain.

2.5.1.5 Playground Web User Interface:

Playground Web User Interface of Hyperledger Composer is used to develop, define and examine different business networks. It also provides the facility to speed up the process of sample importation and the business logic prototype that is used to be executed on runtime of Hyperledger Fabric or sometimes on the web.

2.5.1.6 Loopback Connector:

Loopback Connector is used by REST Server. It is also serves as an individual tool, which is used by different integration tools. Loopback tools combined with the Loopback connector can be used to develop more refined and customized versions of REST Server.

2.5.1.7 Yeoman Code Generator:

Hyperledger Composer uses Yeoman code generator to develop the following three projects:

1. Application of Node.js.
2. Angular Web Application.
3. Skeleton of business network.

2.5.1.8 VS Code and Atom Editor Plugins:

Hyperledger Composer provides support for very powerful extensions of VS code editor as well as for Atom Editor. As compared to VS code editor, Atom only provides the basic facility of highlighting of the syntax. On the other hand, VS code can be used to develop and text different JavaScript files of Hyperledger Composer such as permission.acl, and many more. [25]

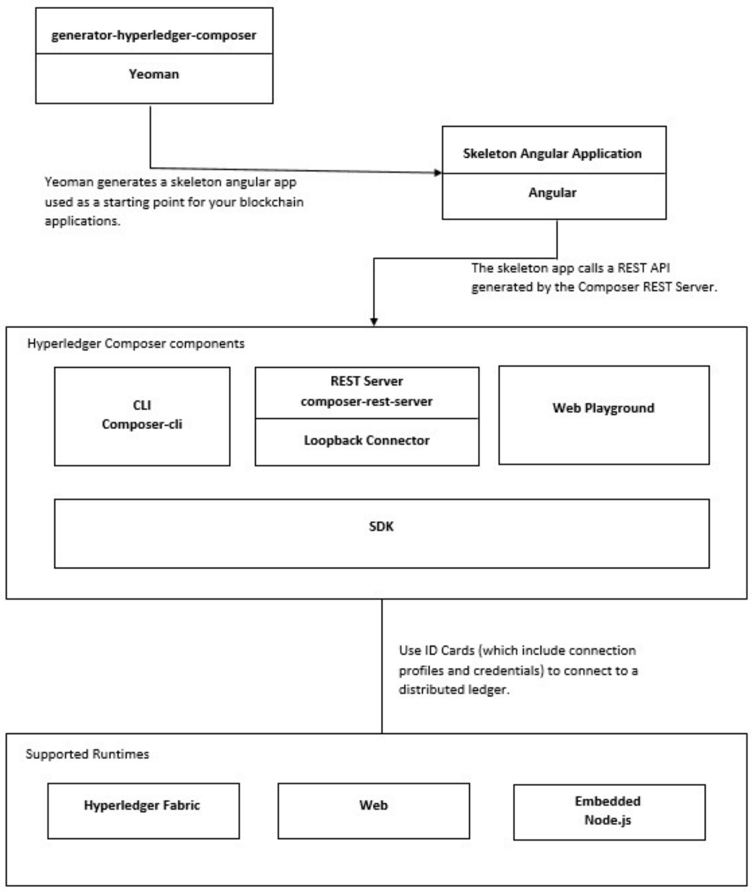

generator-hyperledger-composer
Yeoman
Yeoman generates a skeleton angular app used as a starting point for your blockchain applications.
Skeleton Angular Application
Angular
The skeleton app calls a REST API generated by the Composer REST Server.
Hyperledger Composer components
CLI Composer-cli
REST Server composer-rest-server
Loopback Connector
Web Playground
SDK
Use ID Cards (which include connection profiles and credentials) to connect to a distributed ledger.
Supported Runtimes
Hyperledger Fabric
Web
Embedded Node.js

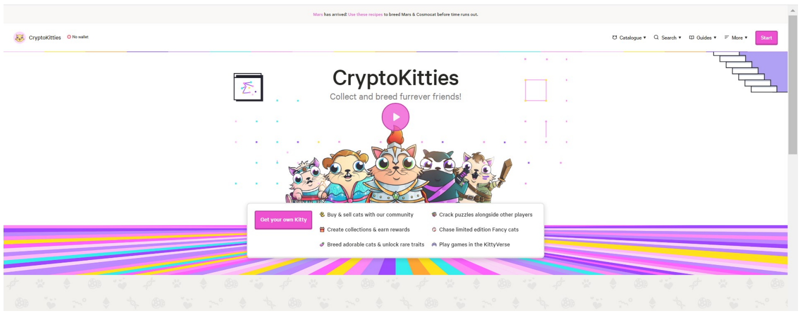
Figure 2.6: Architecture of Hyperledger Composer.

Chapter 3

Software Requirement Specifications

3.1 Existing System:

There are a number of gaming applications based on the concept of Ethereum blockchain such as CryptoKitties that provides the user to play with their digital currencies in an aesthetically pleasing environment. These types of applications were well recognized in the market and became popular among the people that were interested in cryptocurrencies and Blockchain technology. Since Ethereum is a public blockchain, it has its own limitations such as security issues, scalability issues and many more. In the beginning of the popularity of CryptoKitties, this application began to clog the network thus, making the transactions very slow and taking comparatively more time for a kitty to be born. As a result, such gaming applications created a number of limitations. [26]


Mars has arrived! Use these recipes to breed Mars & Cosmocat before time runs out.

CryptoKitties
No wallet
Catalogue
Search
Guides
More
Start
CryptoKitties
Collect and breed furrever friends!
Get your own Kitty
Buy & sell cats with our community
Create collections & earn rewards
Breed adorable cats & unlock rare traits
Crack puzzles alongside other players
Chase limited edition Fancy cats
Play games in the KittyVerse

Figure 3.1: Cryptokitties Website.

3.2 Proposed System:

The system that we have proposed is a decentralized gaming application based on the Blockchain technology using Hyperledger. The application will consist of a large number of dragons with unique features. Initially for every new user, an amount of about 200 coins in the metawallet (i.e. the wallet of application) will be allotted in order to get started with the game. After the purchase of at least two dragons, the user will be able to breed the dragons and produce a new dragon comprising of unique features that would be generated using the features of its parent dragons. Then, the user can add the price tag on the newly produced dragon of their choice and sell it to the other users of the gaming application. In this way, the user will be able to earn more coins by just investing his/her own available coins and breed the dragons in a more entertained environment consisting of eye pleasing graphics and images.

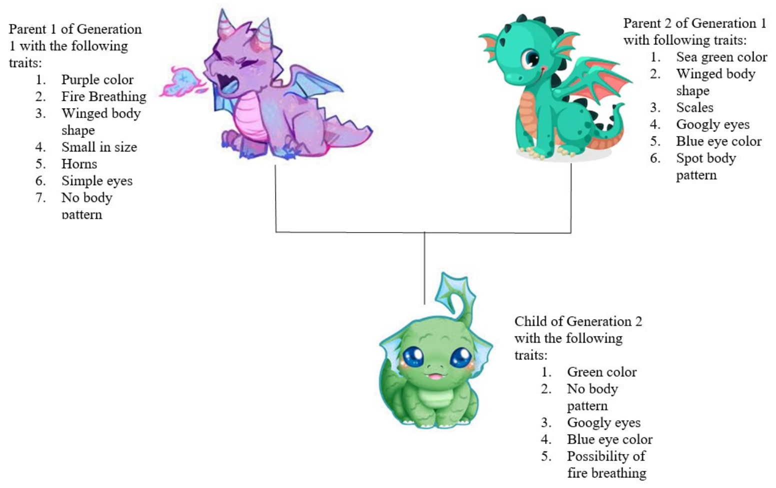


Figure 3.2: Dragon Fusion's Proposed Working.

3.3 Requirement Specifications:

The functional and non-functional requirements of our decentralized gaming application are given below in detail:

3.3.1 Functional Requirements:

This section provides an overview of the functional requirements of the proposed system. A number of functional modules that can be executed and employed by the system are as follows:

1. The Blockchain based application should be able to accept coins from metawallet (wallet of the application) of the user.
2. The application should be able to display all the available dragons to the users.
3. The application should be able to generate the offspring consisting the traits of its parents or a combination of them.

4. The application should be able to display the new dragon, which was generated by the user, to the other users when put up for sale by that specific user.
5. The application should be able to work properly executing one action per time on the request of various number of actions or clicks.
6. The application should be able to provide the user with the complete information of its transactions that were previously executed.
7. The application should provide visually appealing graphics to the user in order to maintain his/her interests.

3.3.2 Non-Functional Requirements:

The various non-functional requirements of the decentralized application are as follows:

1. The application should be able to provide complete private and confidential transactions to the user.
2. The application should be able to take the user to its account on the first successful login.
3. The user should have access to internet connection.
4. The gaming application should be user friendly providing the interface to be very simple and easy to understand.
5. The application should provide the users with sufficient storage to save its breeded or purchased dragons as much as they desire.
6. The application should be able to update the user's data efficiently on each executed action.

3.4 Use Cases:

Use Case diagram is an essential requirement of any hardware or software system design because it provides the substantial information related to that hardware or software system. Following diagrams are the use case diagrams of our application:

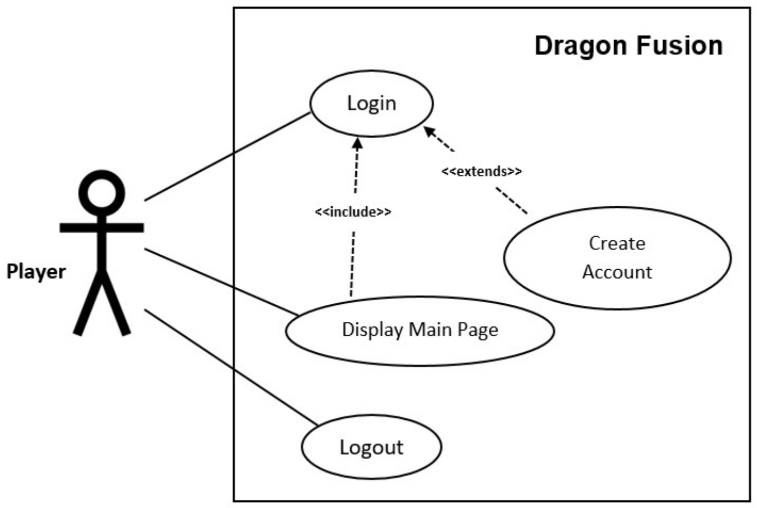


Figure 3.3: Use Case Diagram 1 (Login and Registration).

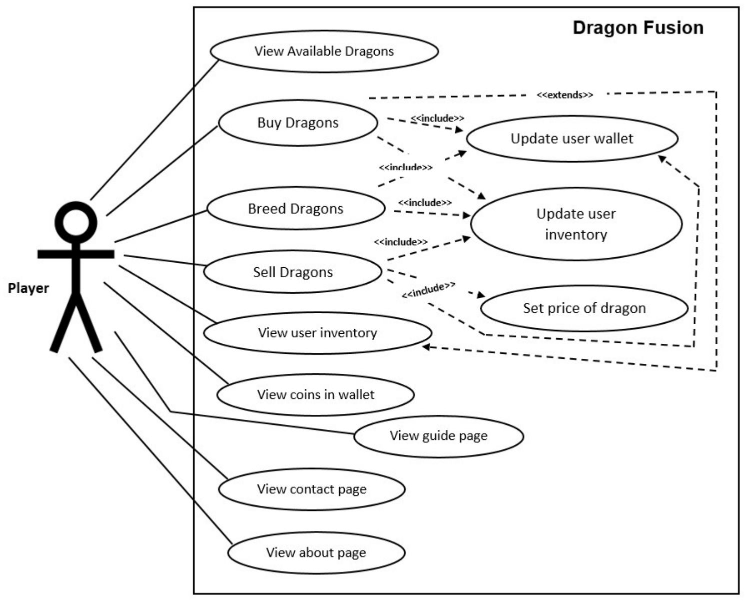


Figure 3.4: Use Case Diagram 2 (Game Functionality).

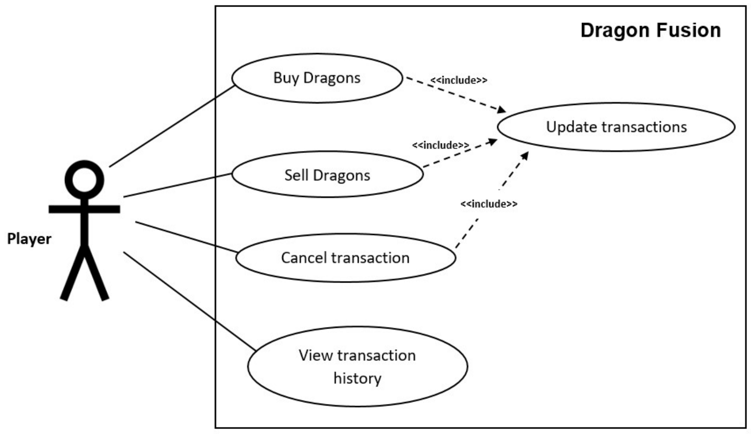


Figure 3.5: Use Case Diagram 3 (Transactions).

3.4.1 Use Case Description:

1. Use Case – 001: Create Account.
   The use case diagram and table of 'Create Account' are as follows:

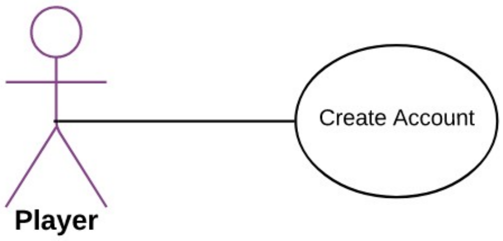


Figure 3.6: Usecase1.

Use Case id	001.
Name	Create Account.
Brief Description	The create account use case allows the player to register themselves with the application.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player accesses the application feature that enables them to create an account by entering information that is maintained in the player's account. The player enter the required information and the application will save that information in the player's account. The system validates the information given by the user and upon success; the system notifies the player that the account has been created. The use case ends.	
Alternate Flows	
Title	**Description**
Player Cancels Request.	At any time, the player can cancel the account creation. At this point, the processing stops and the player account creation is cancelled. The player is notified that the account creation is cancelled.
Player Enters Invalid Account Information.	If during the creation of account , the player enter invalid data, the system will do the following: It points out to the player which information field is incorrect and suggests the valid information to enter. Then if the user enters the valid information, the process proceeds otherwise the same process is repeated until player enters the valid data.
Pre-conditions	
Title	**Description**
Open the Application.	Player must open the application.
Post-conditions	
Title	**Description**
Success.	The player entered valid data and stored successfully in the account.
The player account was not created.	The player entered invalid data or chose to cancel the account creation request. In either case, no account will be created.

Table 3.1: Create Account.

2. Use Case – 002: Log In.
   The use case diagram and table of 'Log in' are as follows:

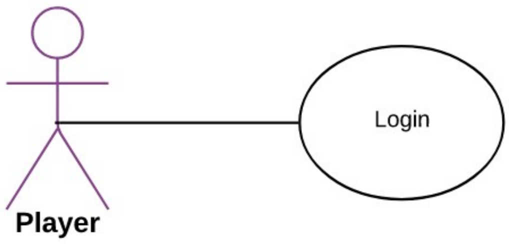


Figure 3.7: Usecase2.

Use Case id	002.
Name	Login.
Brief Description	The player provides the username and password to log into the application.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player opens the log in page and enters the username and password. The system checks if the given information is valid or the player is registered to the application. The use case ends.	
Alternate Flows	
Title	**Description**
Player Enters Invalid Account Information.	If the player enters invalid information, the system will point where the invalid information was given and ask the player to enter the information again. If the correct information is given, the player will log into the application otherwise the player will have to create an account.
Pre-conditions	
Title	**Description**
Create Account.	The player must create an account before logging into the system. However, this is only for new players.
Post-conditions	
Title	**Description**
Success.	Player logs in and is navigated to the main page.
Failed.	Either player entered invalid information or the account does not exist.

Table 3.2: Log In.

3. Use Case – 003: Log out.
   The use case diagram and table of ’Log out’ are as follows:

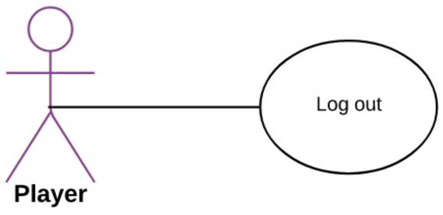


Figure 3.8: Usecase3.

Use Case id	003.
Name	Log out.
Brief Description	The player logs out of the application.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when player wants to log out of the application. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Title	**Description**
Log in the application.	The player must log into the application with the valid information.
Post-conditions	
Title	**Description**
Success.	The player's data was saved and logged out the system successfully.

Table 3.3: Log out.

4. Use Case – 004: Display Main Page.
   The use case diagram and table of 'Display Main Page' are as follows:

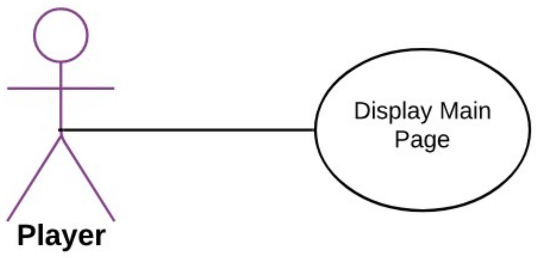


Figure 3.9: Usecase4.

Use Case id	004.
Name	Display Main Page.
Brief Description	The main page of the application will be displayed when the player will login to the application.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player enters the valid credentials in the login page and the system verifies the player by checking if the player has an account. Then the player is navigated to the main page of the application by the system. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Title	**Description**
Player must log into the application.	The player must provide valid credentials in order to log in. The system will verify the account of the player.
Post-conditions	
Title	**Description**
Success.	The main page is displayed by the system.

Table 3.4: Display Main Page.

5. Use Case – 005: View Available Dragons.
   The use case diagram and table of 'View Available Dragons' are as follows:

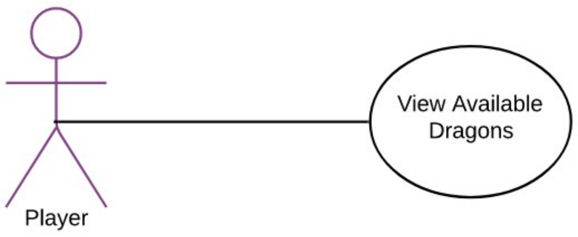


Figure 3.10: Usecase5.

Use Case id	005.
Name	View Available Dragons.
Brief Description	The player will be shown the dragons that are available for buying.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player logs into the application and to start playing, the player has to buy dragons. The system will show the player the dragons that are available for buying. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Title	**Description**
Player must have coins in the wallet.	In order for the player to buy dragons, the player must have a certain amount of coins in the wallet of the application.
Post-conditions	
Title	**Description**
Dragon Bought.	The player buys dragon successfully. The bought dragon will be available to the player for either selling or breeding.

Table 3.5: View Available Dragons.

6. Use Case – 006: Buy Dragons.
   The use case diagram and table of ’Buy Dragons’ are as follows:

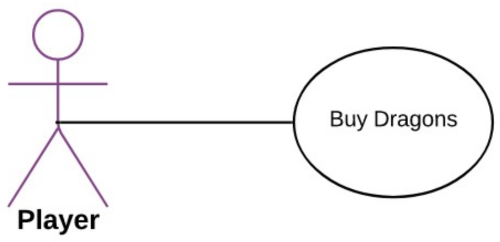


Figure 3.11: Usecase6.

Use Case id	006.
Name	Buy Dragons.
Brief Description	Player will be able to buy any type of dragons according to the coins available in the wallet.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player views the dragons up for sale and buys the specific dragon according to coins in the wallet. The system updates the user inventory and transaction history of the player. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Title	**Description**
Player's wallet must be updated.	The metawallet of the player must have some coins in it in order for the player to carry out transactions. If some transactions have been done then the wallet must update the amount of coins available for use.
Post-conditions	
Title	**Description**
Dragon is bought and available for breeding and selling.	The dragon bought can now be used by the player for breeding with another dragon or selling the dragon to other players. The system should update the user inventory.

Table 3.6: Buy Dragons.

7. Use Case – 007: Sell Dragons.
   The use case diagram and table of 'Sell Dragons' are as follows:

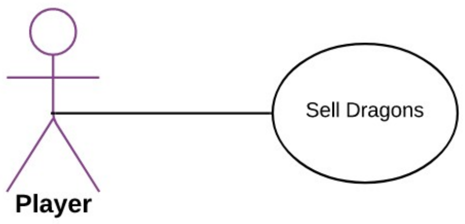


Figure 3.12: Usecase7.

Use Case id	007.
Name	Sell Dragons.
Brief Description	The player is able to sell the dragons at the price selected by the player.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player has bought or bred dragons and wants to sell them to other players. The player according to the attributes and qualities of the dragon sets the price of the dragon. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Title	**Description**
Player must have some dragons.	The player should have dragons in his inventory in order to sell them to other players at a price.
Post-conditions	
Title	**Description**
Dragon is sold.	The dragon is sold successfully and the transaction is verified. The system should update the user inventory.

Table 3.7: Sell Dragons.

8. Use Case – 008: Breed Dragons.

   The use case diagram and table of ’Breed Dragons’ are as follows:

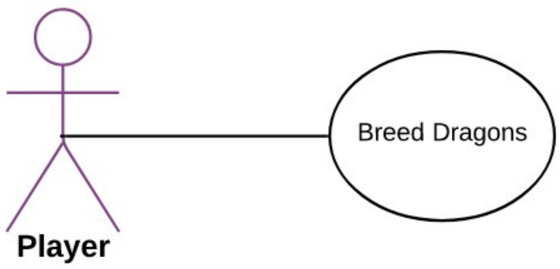


Figure 3.13: Usecase8.

Use Case id	008.
Name	Breed Dragons.
Brief Description	The player has the ability to breed a dragon of one generation with another to produce an offspring of another generation. Moreover the attributes of the offspring are determined by the parent dragons.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player has at least two dragons of some characteristics and breeds them. Breeding takes some time and then offspring is produced which will have some attributes. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Title	**Description**
Player must have at least two dragons.	In order to breed dragons, the player must possess two dragons or more in order to breed them and generate offspring.
Dragons must be of same generation.	A dragon of one generation can bred with a dragon of the same generation. The offspring will be of another generation.
Post-conditions	
Title	**Description**
Dragons are bred and offspring produced.	The offspring produced by the dragons is saved in the inventory of the player.

Table 3.8: Breed Dragons.

9. Use Case – 009: View User Inventory.
   The use case diagram and table of ’View User Inventory’ are as follows:

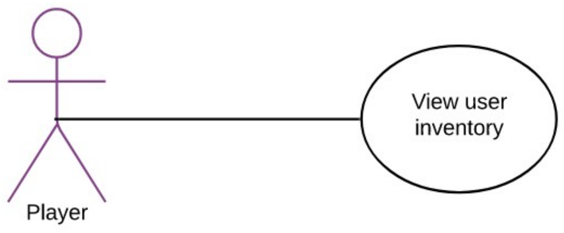


Figure 3.14: Usecase9.

Use Case id	009.
Name	View User Inventory.
Brief Description	The player is able to view the dragons that have been bought in the user inventory.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player has bought some dragons and wants to view them in the inventory. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Title	**Description**
Player must buy dragons.	In order to view dragons in the inventory the player must buy the dragons.
Post-conditions	
Title	**Description**
Success.	The dragons bought by the player will be displayed.

Table 3.9: View User Inventory.

10. Use Case – 010: View About Page.

    The use case diagram and table of ’View About Page’ are as follows:

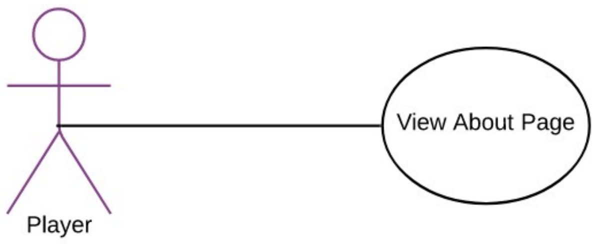


Figure 3.15: Usecase10.

Use Case id	010.
Name	View About Page.
Brief Description	Player can view the about page of the application which explains what the application is about.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player either logs in or out of the application and wants to access the about page which explains what the game is about. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
None.	None.
Post-conditions	
Title	**Description**
Success.	About page was displayed successfully.

Table 3.10: View About Page.

11. Use Case – 011: View Guide Page.
    The use case diagram and table of 'View Guide Page' are as follows:

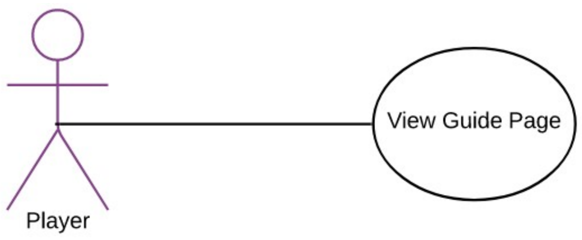


Figure 3.16: Usecase11.

Use Case id	011.
Name	View Guide Page.
Brief Description	Player can view the guide page that explains how to play the game.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player wants to know how to play the game. The guide page can be accessed for this purpose. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
None.	None.
Post-conditions	
Title	**Description**
Success.	The guide page was displayed successfully.

Table 3.11: View Guide Page.

12. Use Case – 012: View Contact Page.
    The use case diagram and table of 'View Contact Page' are as follows:

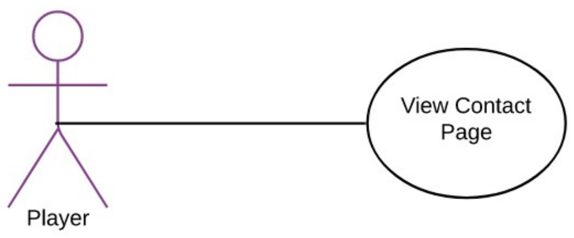


Figure 3.17: Usecase12.

Use Case id	012.
Name	View Contact Page.
Brief Description	Player can view the contact page that allows the player to send message or queries regarding the game.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player goes to the contact page, fills in the required fields and presses the submit button. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
None.	None.
Post-conditions	
Title	**Description**
Success.	The contact page was displayed successfully.

Table 3.12: View Contact Page.

13. Use Case – 013: View Coins in wallet.
    The use case diagram and table of ’View Coins in wallet’ are as follows:

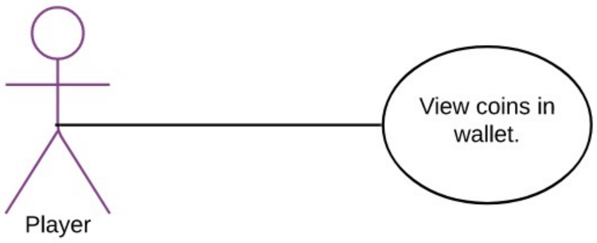


Figure 3.18: Usecase13.

Use Case id	013.
Name	View Coins in wallet.
Brief Description	Player can view the amount of coins present in their wallet.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player has registered to the game and they receive some coins in order to buy dragons. The player can view the amount of coins present in the wallet. In case of transactions, the system will update the wallet accordingly. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Player must create an account.	The player must register to the game in order to receive some coins to buy dragons initially.
Post-conditions	
Title	**Description**
Success.	Player has registered to the game and can view the amount of coins in the wallet.

Table 3.13: View coins in wallet.

14. Use Case – 014: View Transaction History.

    The use case diagram and table of ’View Transaction History’ are as follows:

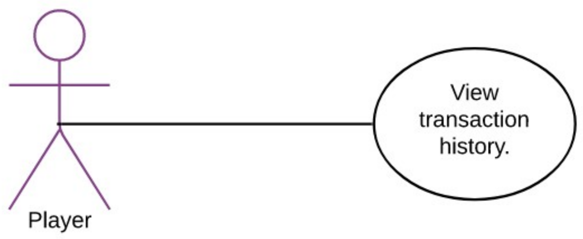


Figure 3.19: Usecase14.

Use Case id	014.
Name	View Transaction History.
Brief Description	The player can view all the transactions done.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player has done some transactions and wants to view them. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Player must have done transactions.	To view transaction history, the player must have carried out some of them.
Post-conditions	
Title	**Description**
Transactions viewed.	The system displayed the transaction history.

Table 3.14: View transaction history.

15. Use Case – 015: Cancel Transaction.
    The use case diagram and table of ’Cancel Transaction’ are as follows:

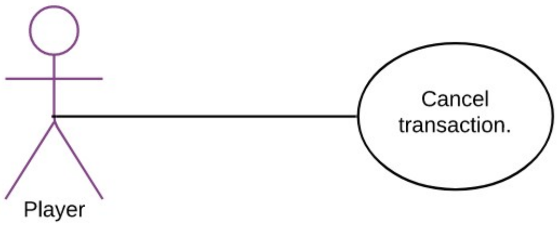


Figure 3.20: Usecase15.

Use Case id	015.
Name	Cancel Transaction.
Brief Description	The player has the ability to cancel the ongoing transaction.
Actor(s)	Player.
Flow of Events	
Basic Flow	
This use case starts when the player triggers a transaction and wants to cancel it. The system will terminate the specific transaction and update the transaction list. The use case ends.	
Alternate Flows	
Title	**Description**
None.	None.
Pre-conditions	
Transaction must be made..	The player must carry out a transaction i.e. buy a dragon.
Post-conditions	
Title	**Description**
Transactions viewed.	Player is notified by the system that the transaction is cancelled successfully.

Table 3.15: View transaction history.

Chapter 4

System Design

This chapter includes the development phase of the decentralized gaming application i.e. Dragon Fusion. The main purpose of this chapter is to bring the visual imagination of the gaming application that is discussed in earlier chapters, into reality to provide the solution to the proposed problem. This chapter includes system architecture in which conceptual model of the application is discussed in detail. The interface of the gaming application will be explained in detail to determine the interaction of the user with the software. In the design methodology section, different techniques, methods and principles will be explained in detail that will be used for the design of the Blockchain gaming application.

4.1 System Architecture:

In this section, the design of different subcomponents of our gaming application will be discussed in detail, which would be combined together into a single software i.e. decentralized gaming application. It would be used to explain the overall structure and behavior of the application. It will also be useful to provide the complete view of the system.

Our decentralized gaming application comprises of three-tier architecture that includes three layers. First is the presentation layer that contains a very interactive and rich graphical user interface with a wide variety of colorful icons and menus. Thus making it pleasant for the user to interact with the web-based application. It is the top most layer of our gaming application. Coming towards the next layer, the application layer, which consists of the processing of our entire application that includes managing different JavaScript programming language files, any function that is provoked by the user's action and many more. Finally comes our logical layer that includes the working of the database of our Blockchain application.

Our decentralized gaming applications will be divided into number of sub-components that are given as follows:

1. First, it will include a rich graphical user interface with aesthetically pleasing graphics and icons. Thus providing a user-friendly environment for the users to interact with.
2. It will consist of a metawallet, which would be considered, as the wallet of the gaming application. User will be able to use the coins that are available in their wallet to proceed with the transactions.
3. Personal account information will also be displayed before the user, which would consist of all the data about transactions that were made earlier. It will also consist of the dragons that were purchased by the user and all of the relevant data and information about it.
4. The game will also provide the user with a platform where each user will be able to place its newly breeded dragon and other users will be able to view the dragon as well as purchase it.

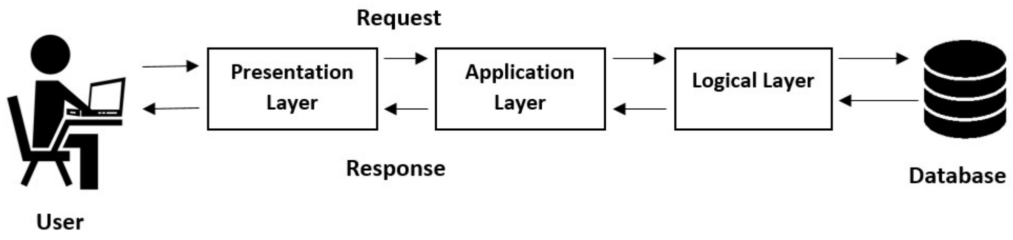


Figure 4.1: Basic Architecture Diagram.

4.1.1 Presentation Layer:

The presentation layer of an application usually consists of all the interface constraints of the application including the markup files of the application. It will be the outermost layer of our system architecture. This layer will mainly include the contents of the user interface, which is used to deliver important and useful system information to the end user. The tool that is used to develop the interface of the application is Visual Studio 2013. Asp.net web forms are used to design different web pages of our Blockchain application.

4.1.2 Application Layer:

Coming towards the next layer is the application layer that consists of the processing of our entire application. It mainly includes managing different JavaScript programming language files, any function that is provoked by the user's action and many more. While installing Hyperledger Fabric and Hyperledger Composer on Ubuntu, many script files like permission.acl, logic.js files are downloaded which are available mostly in JavaScript language. Hyperledger Composer has been used to store data and attributes of the main components of the application which are as follows:

1. Participants (i.e. Users).
2. Assets (i.e. Dragons).
3. Transactions (i.e. Data of Transactions).
4. Event (i.e. Selling of dragon).

4.1.3 Logical Layer:

The logical layer of an application usually consists of the backend processing of that particular application. It includes the working of the database of our Blockchain application. However, the logical layer of our decentralized gaming application will perform all the background processing such as data of the wallet associated with each user, information about the inventory of each user, for error handling of different textboxes of the application and many more. Hence, this layer of the application architecture is intended to handle the overall actual working, action, performance and behavior of the application.

4.2 Design Constraints:

Design constraints are a set of conditions that require being satisfied in order to execute the application or system successfully. It helps to shape the application according to the requirements of the intended user.
Given below are a number of constraints and limitations of various sub-components used in our gaming application to get a complete overview of how the application will work according to the conditions and requirements of the user:

4.2.1 Rich Graphical User Interface:

The main purpose of our Blockchain decentralized gaming application is to provide the users with a pleasing and visually appealing environment so that they can invest and play with their cryptocurrencies. This is achieved by the use of attractive and colorful dragons to get entertained with. The application used under limited access of internet will be unable to provide all available dragons at once but will be displayed one by one, which would be frustrating for the users. Even if the application is accessed in limited level of brightness of the particular device, then the use of these visually appealing dragons will be of no or little advantage.

4.2.2 MetaWallet:

MetaWallet is the wallet of the gaming application. Each user of Dragon Fusion will have their own wallet. Once the user is registered and entered in the game, they will be provided with a reward of about 200 coins in order to proceed with the transactions. Then, the user will be able to use these coins to purchase the desired dragons.

4.2.3 Account:

The gaming application will provide the user with the facility of viewing their account that would consist of the overall transactions made by the user previously as well as the dragons that are purchased by the user in order to breed and put them up for sale. It will also provide with the complete information about all the transactions made by the user such as the amount processed in the transactions, the information about the dragons, and so on. In case of any technical fault in the database, some issues can arise like deletion of the transactions that were made in a 6-month period by the user, the deletion of the all transactions or the relevant errors could be faced.

4.2.4 User's Inventory:

Dragon Fusion will provide the user with an inventory page where they would be able to view the dragons they own. Once the user has purchased any dragon from the catalogue, the required coins will be deducted from the buyer's wallet and will be added in the seller's wallet. The particular dragon will now be removed from the catalogue and will be placed in the user's inventory. After the placement of the purchased dragon in the user's inventory, the user will be able to breed the dragons.

4.2.5 Breeding of Dragons:

The user inventory of the web application will also provide the user with the facility to breed the dragons. It is obligatory for the user to select two dragons in order to start the breeding process. The user will now be provided with the option to give a desired name to the newly breeded dragon. This new dragon will now be placed in the user's inventory after 15 minutes of the breeding process. After the specified time, the user will be able to place that specific dragon up for sale in the dragon catalogue by giving a price tag of their own will.

4.2.6 Catalogue:

Dragon Fusion will also provide the user with the facility of placing their bred or any other dragon on a platform known as "Catalogue". The user will set the price of the dragon being placed up for sale. This particular dragon would be visible to all the registered users of the gaming application. Hence, any user would be able to purchase the dragon from the catalogue by investing the required coins. That dragon will be saved in the inventory of the buyer, removed from the inventory of the seller, and removed from the catalogue.

4.3 Design Methodology:

Design methodology section will include a necessary set of procedures, techniques and methods that are necessarily required to design the gaming application. In order to design the application, we will be using a systematic procedural structured approach that is as follows:

1. User will be able to create an account in the game by providing personal credentials like name, email address and password.

2. Once the user has created the account, they would be able to login into the application.

3. User will be provided with a colorful graphical user interface consisting of all the available dragons to be purchased by the user.

4. User can buy the desired dragons and breed them in order to produce a new offspring.

5. User will have wallet with about 200 coins that were allotted to him/her as a reward of being registered for Dragon Fusion.

6. The user will be able to use these coins in order to purchase any dragon from the list of the available dragons. Once the transaction has been verified, the specific dragon will then be transferred in the inventory of the user.

7. The user would be able to breed the dragons. Application would carry out the transaction process and produce a new offspring that would consist of the characteristics of either one of the parents or a combination of them.

8. The new offspring will be added into the user's inventory. The user will be able to name the newly produced dragon and put it up on sale. When placing it in the catalogue, the user will set the price of the dragon as desired.

9. On the purchase of the dragon that was placed on the catalogue, the payment that would be made by the other user will be transferred in the metawallet of the application. The dragon will be saved in the user inventory and be removed from the catalogue.

4.4 High Level Design:

The high-level design section of the gaming application will include the identification of the various subcomponents of the system and how these subcomponents will be fixed together in the end phase of the application and interact with one another. It will also include the description of the working environment in which the gaming application will run. The workflow diagram of the game is as follows:

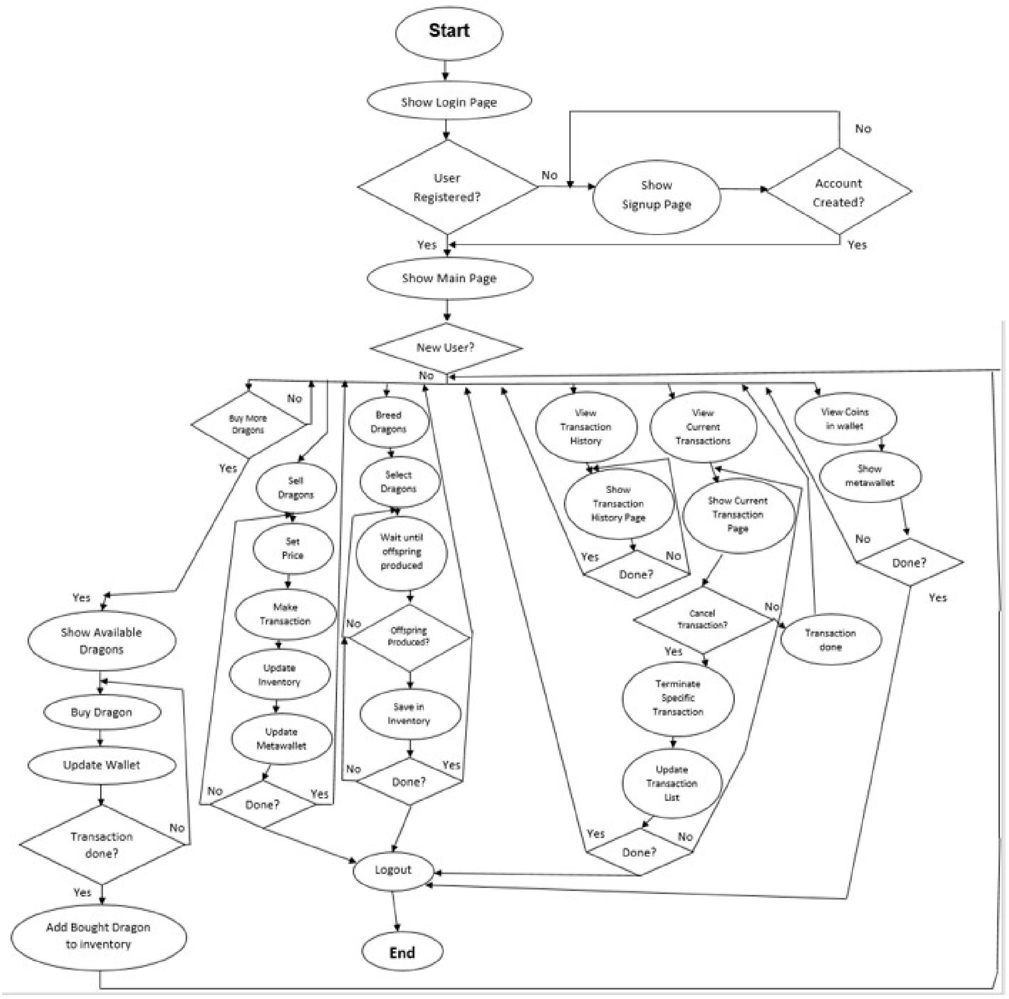


Figure 4.2: Workflow Diagram.

4.4.1 Sequence Diagram:

The sequence diagram of the gaming application will include how the different components of the application will interact with one another in order to perform a specific function and the sequence in which the use cases will be executed. It will also include the interactions of different objects that are included in a particular use case.

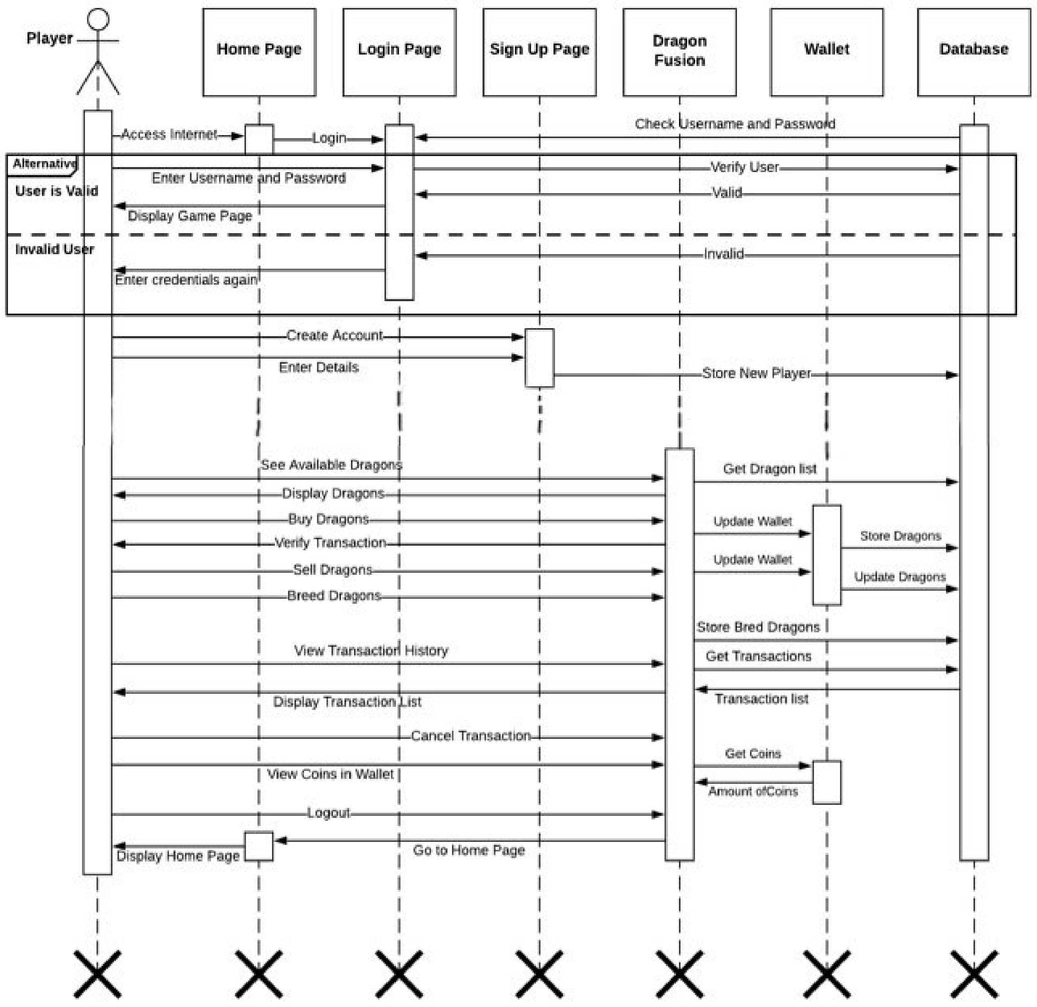


Figure 4.3: Sequence Diagram.

4.5 Software Used:

4.5.1 Hyperledger Fabric:

Hyperledger Fabric is an open source platform that was founded by Linux foundation and in now maintained by both IBM and Linux foundation. Hyperledger Fabric does not consist of any cryptocurrency like Ethereum and Bitcoin. In Fabric, the network is restricted to its network members only and no one else can have access to it. [27] In Fabric, many components like its member services, consensus can become plug can play providing various advantages. One of its various advantage is that it uses small number of nodes as compare to the public blockchains and calculates the data more concisely and precisely making it far better than the public blockchains. [28] [29]

4.5.2 Hyperledger Composer:

Hyperledger Composer initially consists of different set of tools that provides the developers with a much easier approach in order to develop Blockchain network related applications which are not only used to solve a wide number of business problems but it is also used to improve the efficiencies of the network's operations. Hyperledger Composer provides support to the infrastructure of the current Hyperledger Fabric networks, which means that it supports consensus protocols of Blockchains in order to ensure that every transaction in the network is working according to the policy designed by the participants of that specific Blockchain network. [25]

4.6 GUI Design:

The Graphical User Interface (GUI) of our game will be simple and easy to interact with. It will be user friendly, as it will include aesthetically pleasing and visually appealing icons and menus for the users to interact with. We named our decentralized Blockchain gaming application "Dragon Fusion". To provide the users with a more pleasing environment to invest and enhance their digital currencies, we came forward with the idea of using dragons, breeding them and putting them up for sale.

4.6.1 Iconography:

The iconography section of the report will include the logo of our game.

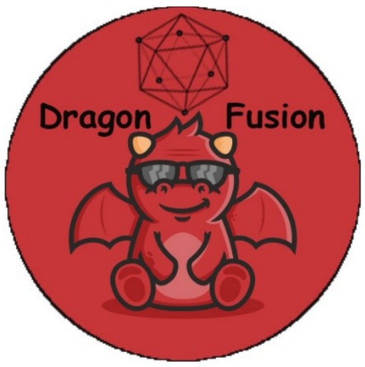
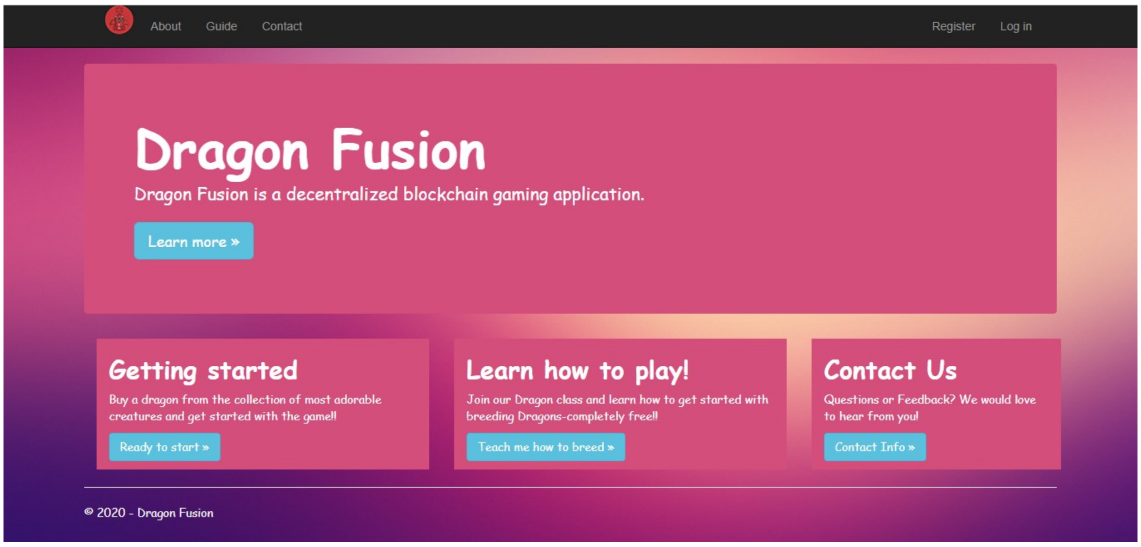


Figure 4.4: Dragon Fusion's Logo.

4.6.2 Home Page Layout:

The home page layout of the game will be the following:



Dragon Fusion

Dragon Fusion is a decentralized blockchain gaming application.

Learn more »

Getting started

Buy a dragon from the collection of most adorable creatures and get started with the game!!

Ready to start »

Learn how to play!

Join our Dragon class and learn how to get started with breeding Dragons-completely free!!

Teach me how to breed »

Contact Us

Questions or Feedback? We would love to hear from you!

Contact Info »

© 2020 - Dragon Fusion

Figure 4.5: Home Page Layout.

4.6.3 Assumptions and Dependencies:

The main assumptions and dependencies of our decentralized gaming applications are given below:

4.6.3.1 Assumptions:

The main assumptions for this project is the availability of required guidelines from the internet or books that would provide enough source of information.

4.6.3.2 Dependencies:

1. The effective working of our application is dependent on the efficient working of the transaction that is used to breed the dragons and produce an offspring that would be a combination of its parent's traits that is the core function of our gaming application.
2. Complete support of the Hyperledger tool is also required, which would be considered necessary for the completion of our gaming application.
3. The proper working environment of the gaming application requires enough availability of the internet signals to provide the quality services to the user without any interruptions.

Chapter 5

System Implementation

System implementation can be defined as a phase that involves the usage of procedures and structures that were previously developed in the architectural design stage. Then, it uses the results of the analysis phase in order to meet the requirements that are specified by the system and the stakeholder. In this chapter, system implementation will be discussed in detail that would include the implementation of the system requirements, which were stated in the early phases of the system development. The implementation of such requirements can be used to create different elements of the system.

5.1 System Architecture:

Our decentralized gaming application comprises of three-tier architecture that includes three layers. First is the presentation layer that contains a very interactive and rich graphical user interface with a wide variety of colorful icons and menus thus making it pleasant for the user to interact with web-based application. It is the top most layer of our gaming application. Coming towards the next layer, the application layer that consists of the processing of our entire application that includes managing different JavaScript programming language files, any function which is provoked by the user's action and many more. Finally comes our logical layer that includes the working of the database of our Blockchain application.

5.1.1 Interface of the Application:

The frontend or the interface of Dragon Fusion is developed by using the software that is known as Microsoft Visual Studio. Asp.net Web application was developed using C# web forms. There are four programming models that can be used to develop Asp.net web applications and Asp.net web forms is one of them. These types of forms provide a number of advantages that are as follows:

5.1.1.1 Bootstrap:

Asp.net web forms provides a complete bootstrap that is the most popular framework of CSS. It is versatile, simple and open source. Bootstrap itself was initially the product of Twitter and is a group of HTML and CSS as well as JavaScript. A significant number of web applications are developed by using bootstrap. [30] It is one of the most popular and easier way to develop web applications. Even the web developers and designers greatly recommend using this tool as it provides a large number of advantages. Firstly, it is very convenient to use and easy to understand. It provides us with almost infinite number of features such as labels, buttons, textboxes, images buttons and many more, to design the structure of the web application.

1. It also provides the facility to use multiple options in order to store and retrieve data. The data can also be displayed in various forms such as tables, drop down lists and many more.

2. It also provides a rich graphical interface that lessens the burden of designing the web pages from scratch that is very difficult task to achieve and is quite time consuming. The interface that is provided by default can also be changed according to the requirements if needed.

3. The portability issues that are faced during the development of most of the web based applications are not met in Asp.net web forms as bootstrap provides a consistent layout that is able to provide attractive layout on any type of screen i.e. desktop screen or any small screen mobile phones. The application that is developed by using these forms is compatible enough to run on any type of browser or device.

Hence, during the development of our gaming application's interface, we worked on the bootstrap that was provided to us by the Asp.net web forms. A number of changes were made in the designing and the layout of the web pages. [31]

5.1.1.2 Connectivity of interface with database using LINQ:

In order to connect the interface of our application with SQL Server, we used LINQ for storing and retrieving of the data from the database. LINQ is an extension of .net language that is used for the retrieval of data from different types of databases and data sources. [32] The purpose of using LINQ instead of any other query is that it provides many features that are difficult to achieve in other queries, which are discussed below:

1. LINQ provides with the facility of highlighting of the syntax that makes it easier to write different queries in comparatively small number of lines.

2. It also uses IntelliSense that makes it easier to notice the mistakes or errors in the queries at compile time.

3. It provides us to see and understand the relationships between different tables more clearly and even the merging of the tables becomes an easier task to accomplish.

4. LINQ also has the ability to convert a type of data into any other type of data in an easier manner. For e.g. converting SQL data into XML can be achieved straightforwardly by using LINQ. [33]

5.2 Tools and Technology:

The tools and technology that were used to develop our Blockchain based gaming web application are as follows:

5.2.1 Hyperledger:

Hyperledger is an open source platform created by Linux foundation in the year 2016. It has intended to create industry specific platform and industries providing high level of confidentiality, flexibility, security, scalability and privacy. It is a major project containing many subprojects within it like the following:

1. HyperLedger Indy.

2. HyperLedger Fabric.

3. HyperLedger Composer.

4. HyperLedger Burrow and many more.

It is used by many large organizations like banking, financing, IOT and the like. It intends to build a platform where various blockchain or software developers can meet to develop and maintain Blockchain frameworks on a large scale. The subproject that we chose for developing our decentralized gaming application is Hyperledger Composer. Hyperledger Composer consists of different set of tools that provide the developers with a much easier approach in order to develop Blockchain related applications which are not only used to solve a wide number of business problems but it is also used to improve the efficiencies of the network operations. It uses JavaScript that not only supports a variety of built-in libraries but also uses the functions, which are readily available in order to make the scripts much more functional and accessible. [34]

5.2.2 Adobe Photoshop:

Adobe Photoshop was released in the year 1990 and was developed by a company named as Adobe Systems. However, it's very first stable version named as Photoshop CC came out in 2017. It is a software program that is widely used for image editing, color shading, removal or reduction of interruptions like noise, gradient filling and many more.[35] It can also be used to create and edit any type of images in order to be used in web pages. This additional feature helps the web designers and developers to create their desired content in an easier and less time-consuming manner. One of the main advantage of using this software is that once you get to know how it works and are aware of its main features, then anyone that includes not only the professionals but also a non-technical person, can easily use it for various purposes. We used Adobe Photoshop to design our dragons that are the main assets of our gaming application. [36]

5.2.3 Microsoft SQL Server:

Microsoft SQL Server was developed by Microsoft and it is a relational database management system that is used to store and retrieve the data in the form of tables. There are a large number of versions of Microsoft SQL Server available in the market. [37] It can be linked with other software systems or applications in order to store and manage different types of data in a more organizational and accessible manner. It can be used on the same computer as well as accessed from different computers that are within the same network. Microsoft SQL Server is linked with the Asp.net web application that is used to create the interface of “Dragon Fusion”. [38]

5.2.4 Ubuntu:

Ubuntu is an open source Linux based operating system. The very first version was released in October 2004 that was Ubuntu 4.0. It was developed on the idea that everyone in this world deserves the software to be completely free and easy to use. It should be so simple and easy to understand that even a layman would be able to use it. This is the reason that almost every version of Ubuntu is readily available to be installed. All the versions of Ubuntu can be installed on the computer as well as on a virtual machine. In the development of “Dragon Fusion”, we have installed Ubuntu 16.04 on VMware 15.5 pro, which is a virtual machine. Hyperledger Fabric version 1.1.0 and Hyperleger Composer version 0.20.09 was installed on Ubuntu to carry out the backend processing of the decentralized Blockchain gaming application. [39]

5.2.5 Microsoft Visual Studio:

Microsoft Visual Studio was developed by Microsoft in the year 1997. However, it’s very first stable version, which was version 16.5.3, was released in 2019. Microsoft Visual Studio is an integrated development environment that is used to create web applications and applications for the cloud, android as well as for Mac. It is one of the popular platforms that is used for the development of different applications. Microsoft Visual Studio provides a wide range of benefits such as:

1. Writing the code can be a much faster task to achieve as it provides support of a large number of built-in libraries.
2. It is much easier to detect and remove the errors in the code because it provides the facility of debugging the code before the compiling process.
3. It provides built-in support for Git and many more.

Hence, we have used this platform to develop our web-based application. In the development of "Dragon Fusion", C# asp.net web forms were used to develop the interface of the application. The connectivity of the database of our application i.e. Microsoft SQL Server with the interface was performed by using Microsoft Visual studio. [40]

5.3 Environment Language used:

5.3.1 JavaScript:

JavaScript was originally named as "LiveScript", when it was released in the year 1995. Later on, Netscape changed the name to "JavaScript". It is fundamentally a scripting language or in simple words, it is a programming language that is used most commonly in web pages for various purposes like in developing interactive maps, displaying frequent content updates, video or image scrolling and many more. [41] A web page is just a static page that is displayed before the user. JavaScript itself manages all of the other main features like Onclick functions on a button, scrolling of the web page and many more. Majority of the files related to Hyperledger installed on Ubuntu were scripted using JavaScript.[42]

5.3.2 C#:

C# was developed and designed by Microsoft in year 2000 and it was a part of .net Framework. Its first stable version was released in September 23, 2019. It is object oriented programming (OOP) language that is an open source and multipurpose to use. It provides the web designers with a large number of flexible and attractive web applications as well as wide range of useful tools in order to develop applications using .net framework. C# is a very easy language to learn and work with because the syntax of this language is very familiar to that of C or C++. As a result, anyone who has the basic knowledge of any of these two languages can easily begin with C#. Many of the limitations that were faced in C++ language are extensively covered in C#, as it provides full support of lambda expressions, generic methods, null-able types and many more. In our gaming web-based application, Asp.net C# web forms were used that comprised of bootstrap. It consisted of a large number of default HTML and CSS files that were available in C# programming language that runs on .net Framework. [43]

Chapter 6

System Testing and Evaluation

In this chapter, we are going to discuss the performance of our decentralized gaming application. In addition to it, a detailed testing and evaluation of the application will also be illustrated by using a number of different methods and techniques. The main purpose of developing such an application using Hyperledger is to provide the customers with the facility to invest and earn their digital currency in an entertained environment providing full security of their personal and transactional data. Hence, the process of testing includes testing and evaluation of all of its functions and modules. The results from these testing phases are then compared to the requirements specified in order to determine how successful the testing has been.

6.1 Graphical User Interface Testing:

Graphical user interface is considered one of the main component of any application. GUI is a type of user interface in which the user interacts with the interface through images and other various graphical icons instead of using any text. It should be designed in a way that even a nonprofessional or a man with no knowledge would be able to interact with the application as efficiently as a technical person would do so.
Graphical user interface testing is the type of testing that involves testing of the application's user interface. The GUI of "Dragon Fusion" is very user friendly and simple to use. The application is filled with attractive and visually pleasing graphics such as color-coordinated web pages, eye-catching dragons and many more. All of these features make the interface of our gaming application aesthetically pleasing and highly interactive making it very simple and easy for the users to experience the Blockchain technology.[44]

6.2 Usability Testing:

Usability testing is the type of testing which involves the process to determine how simple and easy the application or system is to use. “Dragon Fusion” mainly targets the users that are interested in investing and playing with their digital currencies. [45] The users can interact with the system by accessing the web page. Hence, providing the users with an outstanding ease to understand and interact with the decentralized application effectively. [46]

6.3 Compatibility Testing:

Compatibility testing is the type of testing that is used to determine whether an application is capable enough of running efficiently on different browsers. The main concern in this type of testing is that different interfaces appear while running the same application on different systems.[47] However, in our gaming application, this is not the case. As “Dragon Fusion” is a web-based gaming application that is developed by using Asp.net C# web forms, it would work efficiently on all types of systems providing full web browser compatibility. The game will provide full facility to move within different environments and do not change or alter its components. [48]

6.4 Application Performance Testing:

Application performance testing is used to determine different features of the application such as its speed of response, scalability and stability under specific workload. The decentralized application satisfies all the above-mentioned criteria efficiently and effectively by taking almost no time to respond to any command given by the user. [49]

6.5 Interoperability Testing:

The database that would be used by our blockchain decentralized gaming application is Hyperledger Composer. The interface will be developed in C# i.e. C# form by using Visual studio. The data entered by the user in order to register to the game and to further log in will be saved in the database. Even the transactions that would be made by the user, the information about the user and dragons will also be recorded in the database. Thus, the connectivity of the application with these required software components would be strongly conducted and monitored. [50]

6.6 Exception Handling:

Exception handling is used to deal with the case when a function of the application is not working properly, by providing an alternative function in order to handle that particular error. The following explains how exception handling is used in our gaming application:

1. If the user has entered wrong user ID, email address or password, an error will be generated displaying that “Entered Credentials are not correct!”.
2. If the user has entered a password of less than 8 characters or numbers during the registration process, an error will be displayed before the user that “Password must consist of 8 characters!”.
3. In the registration and login form, if the user has entered an invalid email address, an error will be generated that would be “Email address is invalid!”.
4. If the user has entered the data into the login form without going through the registration form, an error message will be displayed before the user that would be “Entered data cannot be found!”.
5. If the user has requested for a dragon that requires more coins that are available in the user’s wallet, an error message will be generated displaying the message “Requested coins are not enough!” .
6. In the breed page, if the user has selected the same dragon from both of the dropdown lists, an error will be generated displaying the message “Breeding process cannot take place between the same dragons!”.
7. If the user has selected the same combination of two dragons that has already been breeded and produced a new dragon, an error will be displayed before the user that “Breeding process has already taken place between same dragons!”. [51]

6.7 Load Testing:

Load testing is used to determine how the system works under different levels of load. As we are using Blockchain technology, it has the capability to work under high load providing full accuracy and effectiveness. [52]

6.8 Test cases:

1. Test Case – 001: Login the application.

Test Case	001.
Function to be tested	Login the application.
Revision History	Created by Noor Hoda at 11:35.
Initial State	Home page is displayed before the user.
Test Setup	Main page of the web application should be displayed.
Input	Click on Login button.
Expected Output	Main page of the gaming application should be launched.
Actual Output	Main page of the gaming application is launched successfully.
Status	Pass.

Table 6.1: Login the application.

2. Test Case – 002: Register for the application.

Test Case	002.
Function to be tested	Register for the application.
Revision History	Created by Noor Hoda at 11:38.
Initial State	Home page of the gaming application is displayed before the user
Test Setup	Login page of the web application should be displayed.
Input	Click on the Register button.
Expected Output	Registration page of the application should be launched.
Actual Output	Registration page of the application is launched successfully.
Status	Pass.

Table 6.2: Register for the application.

3. Test Case – 003: A new dragon is bred successfully.

Test Case	003.
Function to be tested	New dragon is breed successfully.
Revision History	Created by Daniah Aftab at 2:10.
Initial State	Breed page of the web application should be displayed before the user
Test Setup	A new dragon should be breed.
Input	Click on Breed button.
Expected Output	A new dragon should be breed.
Actual Output	A new dragon should be breed successfully.
Status	Pass.

Table 6.3: Dragon bred successfully.

4. Test Case – 004: Name of newly bred dragon is updated.

Test Case	004.
Function to be tested	Name of newly breeded dragon is updated successfully.
Revision History	Created by Daniah Aftab at 2:13.
Initial State	Breed page of the web application should be displayed before the user
Test Setup	Name of newly breed dragon should be updated.
Input	Click on Save button.
Expected Output	The name of newly breed dragon should be updated.
Actual Output	The name of newly breed dragon is updated successfully.
Status	Pass.

Table 6.4: Name of breeded dragon is updated.

5. Test Case – 005: Price of selling dragon is set.

Test Case	005.
Function to be tested	Price of newly breed dragon is set successfully.
Revision History	Created by Noor Hoda at 12:25.
Initial State	User Inventory page of the web application should be displayed before the user
Test Setup	Price of selling dragon should be set.
Input	Click on Save button.
Expected Output	The price of selling dragon should be set.
Actual Output	The price of selling dragon should be set successfully.
Status	Pass.

Table 6.5: Price of selling dragon is set.

6. Test Case – 006: Selected dragon placed successfully on auction.

Test Case	006.
Function to be tested	Selected dragon is placed on auction successfully.
Revision History	Created by Daniah Aftab at 2:15.
Initial State	User Inventory page of the web application should be displayed before the user
Test Setup	Selected Dragon should be placed successfully on auction.
Input	Click on Yes button.
Expected Output	Selected dragon should be placed on auction.
Actual Output	Selected dragon should be placed on auction successfully.
Status	Pass.

Table 6.6: Dragon placed on auction.

Chapter 7

Conclusion

Blockchain technology has been rapidly evolving from the past few decades. People nowadays confuse the term Blockchain with Bitcoin that was the first implementation of Blockchain technology proposed in the year 2008. The further platforms that are developed under this technology like Ethereum, Hyperledger and the like, are not well known and popular among the people. Many of the benefits of Blockchain like privacy, security, transparency and many more are not highly recognized. Although there are many such web-based applications that provide the users with the facility to play with their cryptocurrencies by investing as well as earning from it. An example of such an application is Cryptokitties that is developed by using Ethereum blockchain. As Ethereum is a public blockchain, thus we came forward with the completely new idea of developing a decentralized web-based gaming application using Hyperledger, which comes under private blockchain category. [53] Thus developing a decentralized gaming application using Hyperledger technology has not been an easy task. There was literally no or very little help available on the internet. Even the research papers on this particular platform are available in a limited range. Working on such a project was really challenging and quite interesting as well. Our main objective to work on such a project was to introduce people to a totally new and emerging concept of investing their digital currencies. One of the main reason was also that there was a huge market gap of such an application developed by using Hyperledger. [54] The tool that was used mainly is our project was Hyperledger Composer and Hyperledger Fabric, which comprises of the basic skeleton of our application. Almost all of the files in the Hyperledger Composer like the logic.js etc. uses JavaScript programming language, which helped us to improve and polish our programming skills. Working on this project has also improved our teamwork skills, enhanced our research skills and also encouraged us to learn a very useful and innovative technology. While testing the application, we faced a wide range of problems and were able to overcome them. The techniques and methods that were used during the development of the application are as follows:

1. Gathering and understanding requirements of the application.
2. Application designing.
3. Implementation of the system.
4. Designing of dragons using Photoshop tool.
5. Interface.
6. Managing the discipline and teamwork throughout the project.
7. Testing of each individual module, which is known as unit testing.
8. Integrating and testing all of the modules of the application as a whole i.e. integration testing.

7.1 Improvement for the Future:

Although our decentralized gaming application named as "Dragon Fusion" has been successfully deployed but it can also be improved in the future by adding further additional features to it. Some of them can be stated as follows:

1. We have developed our application by just designing and using five dragons initially and providing all the possible combinations of these dragons. Thus, in this case, the number of dragons can be enhanced in the future, which would provide the user to select from a wide range of dragons for the breeding process.
2. In our application, we are limited to produce dragons, which are up to two generations only. This limitation can be improvised by providing the advantage to keep continuing the breeding process that will be possible by the increment of the possible generations.
3. In our originally developed gaming application, we have used the mechanism in which the breeding process of any two dragons selected by the user will result in one new offspring. Hence, the future improvement for this particular section can include the idea of producing two or more offspring because of single breeding process of the 4th or 5th generation of dragons. There can also be a possibility that instead of producing two or more offspring of higher generation of dragons, a single exclusive dragon can also be produced, which would provide the user with the advantage of earning some additional coins which would be attached with that one exclusive offspring.
4. When the user has purchased more than a specific number of dragons from the catalogue, the user can be provided with the privilege to customize its dragons

according to their own requirement, which would involve changing the color, pattern or even the body type of the dragons.

References

[1] H. Fabric. Architecture origins. Cited on p. i.

[2] I. Society. Blockchain. Cited on p. 1.

[3] AICPA. Blockchain technology and its potential impact on the audit and assurance profession. Cited on p. 1.

[4] I. Markit. Blockchain technology reports and analysis. Cited on p. 1.

[5] E. Parliament. How blockchain technology could change our lives. Cited on p. 1.

[6] ResearchGate. An overview of blockchain technology: Architecture, consensus, and future trends. Cited on p. 1.

[7] ZAGE. Real world applications of blockchain technology. Cited on p. 1.

[8] EY. Blockchain how this technology could impact the cfo. Cited on p. 2.

[9] TechTarget. Smart contract. Cited on pp. 2 and 7.

[10] Edureka. Hyperledger vs ethereum. Cited on pp. 2 and 3.

[11] Edureka. Hyperledger fabric. Cited on p. 2.

[12] B. Council. Private vs public blockchain. Cited on p. 3.

[13] CoinMonks. Public vs private blockchains in a nutshell. Cited on p. 3.

[14] Blockgeeks. Hyperledger vs ethereum training: Which one is better? Cited on p. 5.

[15] M. Digital. Blockchain beyond the hype: What is the strategic business value? Cited on p. 6.

[16] NIST. Blockchain technology overview. Cited on p. 6.

[17] O'Reilly. Chapter 1. what is a decentralized application? Cited on p. 6.

[18] HackerNoon. What are decentralized applications, dapps? Cited on p. 6.

[19] CoinSutra. What are dapps (decentralized applications)?– the beginner's guide. Cited on p. 6.

[20] B. Berlin. Decentralized applications – dapps. Cited on p. 6.

[21] Hyperledger. Hyperledger fabric documentation. Cited on p. 8.

[22] ResearchGate. The privacy protection mechanism of hyperledger fabric and its application in supply chain finance. Cited on p. 8.

[23] I. J. o. E. . Technology. A fabric architecture towards block chain application using hyper ledger. Cited on p. 8.

[24] ClairVoyant. Hyperledger fabric part 1 components and architecture. Cited on p. 8.

[25] Github. Hyperledger composer. Cited on pp. 9, 11, and 40.

[26] A. Zen. Cryptokitties. Cited on p. 13.

[27] IBM. Hyperledger fabric: A distributed operating system for permissioned blockchains. Cited on p. 39.

[28] QuillHash. What is hyperledger fabric? everything you need to know about hyperledger fabric. Cited on p. 39.

[29] DevTeam.SpaceTM. Pros and cons of hyperledger fabric for blockchain networks. Cited on p. 39.

[30] Toptal. What is bootstrap? a short bootstrap tutorial on the what, why, and how. Cited on p. 44.

[31] TutorialsPoint. Asp.net mvc – bootstrap. Cited on p. 44.

[32] CCorner. Using linq in .net. Cited on p. 44.

[33] CCorner. Advantage of linq? Cited on p. 45.

[34] Blockgeeks. What is hyperledger. Cited on p. 45.

[35] A. G. Institute. What is photoshop. Cited on p. 46.

[36] Techopedia. Adobe photoshop. Cited on p. 46.

[37] TechTarget. Microsoft sql server. Cited on p. 46.

[38] Microsoft. Sql server. Cited on p. 46.

[39] H. Ubuntu. What is ubuntu? Cited on p. 46.

[40] Microsoft. Visual studio. Cited on p. 47.

[41] W3Schools. What is javascript? Cited on p. 47.

[42] M. w. docs. What is javascript? Cited on p. 47.

[43] Wikipedia. C sharp (programming language). Cited on p. 48.

[44] Guru99. Gui testing tutorial: User interface (ui) testcases with examples. Cited on p. 49.

[45] ExperienceUX. What is usability testing? Cited on p. 50.

[46] Guru99. What is usability testing? ux(user experience) testing example. Cited on p. 50.

[47] S. T. Help. Cited on p. 50.

[48] Guru99. What is compatibility testing? forward backward testing (example). Cited on p. 50.

[49] InfoQ. 12 quick tips about application level performance testing and more. Cited on p. 50.

[50] Guru99. What is interoperability testing in software testing? (with examples). Cited on p. 50.

[51] Microsoft. Using exception handling in web applications. Cited on p. 51.

[52] Guru99. Load testing tutorial: What is? how to? (with examples). Cited on p. 51.

[53] Hindawi. Decentralized applications – dapps. Cited on p. 55.

[54] I. Developer. Top 6 technical advantages of hyperledger fabric for blockchain networks. Cited on p. 55.