**ATTAINING LIABILITY USING**

**CIPHER CURRENCY**

A REPORT

Submitted by

JAVERI CHOWDARY.E(17MIS1081)

*In partial fulfilment for the award*

Of

**M. Tech. Software Engineering (Integrated)**

**School of Computer Science and Engineering**

**SEPTEMBER, 2020**





**School of Computer Science and Engineering**

**DECLARATION**

I hereby declare that the project entitled **“Attaining Liability using Cipher Currency”** submitted by me to the School of Computer Science and Engineering, Vellore Institute of Technology, Chennai Campus, Chennai 600127 in partial fulfilment of the requirements for the award of the degree of **Master of Technology - Software Engineering (Integrated)** is a record of bonafide work carried out by me**.** I further declare that the work reported in this report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or university.

Signature

**Javeri Chowdary. E**

**(17MIS1081)**



**School of Computer Science and Engineering**

**CERTIFICATE**

The project report entitled “**Attaining Liability using Cipher Currency**” is prepared and submitted by **Javeri Chowdary. E (Register No: 17MIS1081)**.Ithas been found satisfactory in terms of scope, quality and presentation as partial fulfilment of the requirements for the award of the degree of **Master of Technology – Software Engineering (Integrated)** in Vellore Institute of Technology, Chennai, India.

**Examined by**:

**Examiner I Examiner II**

**ACKNOWLEDGEMENT**

* Dr. Asnath Victy Phamila Y, Head of the Department (HoD), M.Tech Software Engineering (5 year integrated), SCSE, VIT Chennai
* Dr. Jagadeesh Kannan R, Dean of the School of Computer Science & Engineering, VIT Chennai
* Dr. Geetha S, Associate Dean of the School of Computer Science & Engineering, VIT Chennai

**CONTENTS**

**CHAPTER TITLE PAGE**

TITLE PAGE i

DECLARATION ii

CERTIFICATE iii

INDUSTRY CERTIFICATE iv

ACKNOWLEDGEMENT v

TABLE OF CONTENTS vi

LIST OF TABLES vii

LIST OF FIGURES viii

LIST OF ABBREVIATION ix

ABSTRACT x

1. INTRODUCTION 01

1. SYSTEM ANALYSIS 05
2. FEASIBILITY REPORT 07
3. SOFTWARE REQUIREMENT SPECIFICATION 09
4. SYSTEM DEVELOPMENT ENVIRONMENT 14
5. SYSTEM DESIGN 31
6. SYSTEM SECURITY 34
7. CONCLUSION 35

REFERENCES 36

APPENDIX I 42

APPENDIX II 50

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**LIST OF TABLES**

|  |  |
| --- | --- |
| **Title** | **Page** |
| 1. ADMIN | 29 |
| 2) SELLER | 30 |
| 3) BUYER | 30 |
|  |  |

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **Title** | **Page** |
| 1. SPIRAL MODEL | 03 |
| 1. SQL DATABASE | 13 |
| 3) .NET FRAMEWORK | 17 |
| 4) ER DIAGRAM | 21 |
| 5) DFD DIAGRAMS  LEVEL 0 &LEVEL 1 | 25 |
| 6) DFD DIAGRAMS  LEVEL 2  7) USECASE DIAGRAM  8) SEQUENCE DIAGRAM  9) CLASS DIAGRAM    10) ACTIVITY DIAGRAM | 26  27  28  29  31 |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **Abbreviation** | **Expansion** |
| BPaas | Business process as a Service |
| SRS | Software Requirement Specification |
| UI | User Interface |
| FCL | Framework Class Library |
| CLR | Common Language Runtime |
| BCL  CTS  CLI  RDBMS  ANSI  BLOB  UWP  ERD  DFD | Base Class Library  Common Type System  Common Language Infrastructure  Relational Database Management System  American National Standards Institute  Binary Large Objects  Universal Windows Platform  Entity Relationship Diagram  Data Flow Diagram |

**ABSTRACT**

In this crypto currency transaction many anonymous clients may get the transaction profile easily by computerized technique. The privacy of crypto currency was less secure the data which have been stored during the transaction process the leakage of data were handle by the using the hash key. Each and every transaction process is stored in secured manner with help of hash key. In our project, we are getting the rare coins/notes equivalent Crypto currency from the seller. The user privacy is the more secure. In this process the rare coin/notes has sold by the seller and the users will purchase the coin’s equivalent crypto currency and the crypto currency will be credited to their wallet. In this the transaction the details have been stored with hash key.

**1. INTRODUCTION**

The scope of this application is to share the data in secured way when multiple transaction has done it will be transacted in encrypted way of data while changing the coins. To better illustrate our approach, we consider the scenario of a configurable Web store checkout BPaaS. The data of this process will be small and medium sized Web stores, while the users will be customers. This targets businesses selling physical or digital goods, as standard orders or pre-orders. The process places shipping orders for physical goods and re-thieves download links for digital goods. Specific tasks in the process include validating customers, obtaining payment details, updating inventory and accounting systems, and processing customer payment amongst others

**2. SYSTEM ANALYSIS**

**2.1 INTRODUCTION**

After analyzing the requirements of the task to be performed, the next step is to analyse the problem and understand its context. The first activity in the phase is studying the existing system and another is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then the successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of the existing running system is also difficult, improper understanding of the present system can lead to diversion from solution.

**2.2 ANALYSIS MODEL**

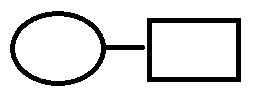
The model that is basically being followed is the SPIRAL MODEL, which states that the phases are organized in a linear order. First of all the feasibility study is done. Once that part is over the requirement analysis and project planning begins. If a system exists one and modification and addition of the new module is needed, analysis of the present system can be used as the basic model.

The design starts after the requirement analysis is complete and the coding begins after the design is complete. Once the programming is completed, the testing is done. In this model, the sequence of activities performed in a software development project is: -

* Requirement Analysis
* Project Planning
* System design
* Detail design
* Coding
* Unit System integration & testing

Here the linear ordering of these activities is critical. End of the phase and the output of testing

One phase is the input of another phase. The output of each phase is to be consistent with the overall requirement of the system.

****SPIRAL MODEL was defined by Barry Boehm in his 1988 article, “A spiral Model of Software Development and Enhancement. This model was not the first model to discuss iterative development, but it was the first model to explain why the iteration models.

**Fig 2.2.1 SPIRAL MODEL**

**2.3 SYSTEM REQUIREMENTS**

**HARDWARE AND SOFTWARE SPECIFICATIONS**

* HARDWARE REQUIREMENTS

RAM 8GB

Dual-Core 2.8 GHz Processor and Above

HDD 80 GB Hard Disk Space and Above

* SOFTWARE REQUIREMENTS

WINDOWS OS (7 /XP)

Visual Studio .Net 2015

Visual Studio .Net Framework 4.5

SQL Server 2014.

**2.4 EXISTING SYSTEM**

The transaction is done by encrypted manner and it is a crypto currency transactions. The crypto currency is more efficient but during the transaction there is purpose of hacking or thefting the profile of a person transaction during the transaction the amount may or may not sent to the proper trustworthy person. Further details and crypto amount where theft by the normal trustworthy user. There would be no intermediate person to store the data secure. The value of the crypto currency is more benefit.

**DISADVANTAGES**

* During transaction there is no secure the encrypted data stored in database can be easily theft by the third party.
* It has less security for the transaction details.
* There is leakage of data, due to the untrustworthy party.
* Malicious attack will attack the further transaction details.
* There is less secure in crypto currency money transaction.

**2.5 PROPOSED SYSTEM**

In this proposed system, there is more secure for crypto currency because the further transaction details are stored with hash key. Hash key it is unique for every transaction. Each transaction details are stored with current hash key and with previous. When the unauthorized person or third party involve they cannot get the required detail. Crypto currency is more secure and the data are secured while transaction is performed in this process .the data which has fixed range that can be shown in form of crypto value by handling these hidden data the value are in saver way to show the third party in the application. Once the amount has been transacted those data will be show to the admin so the receiver will encrypt the data store those data in the application.

**ADVANTAGES**

* Data transmitting is advantage and in a secured way
* Handling encrypted value which is the source of traceable monero.
* Crypto currency is the process of charging data and fixed form of data is performed in this process.
* It implies the required form of the leveraged public key encryption scheme guarantees could be uniquely recovered here.

**2.6 STUDY OF THE PROJECT**

**MODULE**

In the application, the following modules are given below,

**2.6.1 UPLOAD PROCESS:**

The seller will be uploading the ancient coins/notes equivalent crypto currency in the database securely. The seller process was viewed to the higher official who acts as the Admin of the application. Admin will be collecting the history of the buyer and the user. It will check the buyer is authorized or not. The whole seller process and the buyer process were handled by the admin. The feedback of the buyer was shown to the admin as the notification.

**2.6.2 TRANSACTION PROCESS:**

The seller will get the rare coins/notes equivalent crypto currency by viewing the certain product uploaded by the seller. Once the user buys the selected coin/notes the transaction process starts. The user with the required amount will buy the coin’s/notes equivalent to the product. Those transaction processes were done in a secure manner during the transaction the details are stored in the hash value.

**2.7 Domain Knowledge**

A blockchain is a decentralized, distributed and public digital ledger that is used to record transactions across many computers so that any involved record cannot be altered retroactively, without the alteration of all subsequent blocks. This allows the participants to verify and audit transactions independently and relatively inexpensively. A blockchain database is managed autonomously using a peer-to-peer network and a distributed time stamping server. They are authenticated by mass collaboration powered by collective self-interests. Such a design facilitates robust workflow where participants' uncertainty regarding data security is marginal.

The use of a blockchain removes the characteristic of infinite reproducibility from a digital asset. It confirms that each unit of value was transferred only once, solving the long-standing problem of double spending. A blockchain has been described as a value-exchange protocol. This blockchain-based exchange of value can be completed quicker, safer and cheaper than with traditional systems. A blockchain can maintain title rights because, when properly set up to detail the exchange agreement, it provides a record that compels offer and acceptance.

**3. FEASIBILITY REPORT**

The preliminary investigation examines project feasibility; the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging the oldest running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

* Technical Feasibility
* Operational Feasibility
* Economic Feasibility

**3.1. Technical Feasibility**

The technical issue usually raised during the feasibility stage of the investigation includes the following:

* Does the necessary technology exist to do what is suggested?
* Does the proposed equipment have the technical capacity to hold the data required to use the new system?
* Will the proposed system provide an adequate response to inquiries, regardless of the number or location of users?
* Can the system be upgraded if developed?
* Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of the ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a browser-based user interface for audit workflow. Thus, it provides easy access to users.

The purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the rules specified. Therefore, it provides a technical guarantee of accuracy, reliability, and security. The software and hard requirements for the development of this project are not many and are already available or are available as free as open source.

The work for the project is done with the current equipment and existing software technology. Necessary bandwidth exists for providing fast feedback to the users irrespective of the number of users using the system.

**3.2. Operational Feasibility**

The analyst considers the extent the proposed system will fulfill his departments. That is, whether the proposed system covers all aspects of the working system and whether it has considerable improvements. We have found that the proposed “Secure transaction” will certainly have considerable improvements over the existing system.

**3.3. Economic Feasibility**

The proposed system is economically feasible because the cost involved in purchasing the hardware and the software is approachable. Working in this system need not require a highly qualified professional. The operating-environment costs are marginal. The less time involved also helped in its economic feasibility.

**4. SOFTWARE REQUIREMENT SPECIFICATION**

**4.1 INTRODUCTION**

The purpose of this document is to present a detailed description of the Web application system. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for both the stakeholders and the developers of the system and will be proposed to the Regional Historical Society for its approval.

Purpose

The purpose of this Software Requirement Specification (SRS) is to help the project. It is provided with some requirements which are used in the Transaction Mercator System. All parts design, coding, and testing will be prepared with the helping of SRS. The purpose of this document is to detail the requirements placed on the Transaction Mercator System and serves as a contract between the customer and the developers as to what is to be expected of the stock exchange, and how the components of the system are working with each other with external systems.

**DEVELOPERS RESPONSIBILITIES OVERVIEW:**

The developer is responsible for:

Developing the system, which meets the SRS and solving all the requirements of the system?

* Demonstrating the system and installing the system at the client's location after the acceptance testing is successful.

Submitting the required user manual describing the system interfaces to work on it and also the documents of the system

* Conducting any user training that might be needed for using the system.
* Maintaining the system for a period of one year after installation.

**4.2 FUNCTIONAL REQUIREMENTS:**

Following is a list of functionalities of the browsing enabled system.

* An Activity with a UI that allows you to browser settings. Provide a second Activity that allows users to access the share with permission from the administrator. Handle the activity lifecycle appropriately. A precondition for any points in this part of the grade is code that compiles and runs.
* Your application should allow a user to browse the shares, buy and sell the shares with specific metadata. The assignment requires you to create a UI for browsing and a UI for integrating the two.
  1. **NON-FUNCTIONAL REQUIREMENTS:**
* Each member should have a separate system. The system should ask the username and password to open the application. It doesn’t permit to unregistered user to access the System. The system should have Role-based System functions access. Approval Process has to be defined. The system should have Modular customization components so that they can be reused across the implementation.
* These are mainly the following:
* Secure access to confidential data (employee’s details). 24 X 7 availability
* Better component design to get better performance at peak time
* Flexible service based architecture will be highly desirable for future extension

**4.4 PERFORMANCE REQUIREMENTS**

Performance is measured in terms of the output provided by the application. Requirement specification plays an important part in the analysis of a system. Only when the required specifications are properly given, it is possible to design a system, which will fit into the required environment. It rests largely on the part of the users of the existing system to give the required specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

**5. SYSTEM DEVELOPMENT ENVIRONMENT**

**5.1 INTRODUCTION TO .NET**

The .NET Framework Architecture.NET Framework (pronounced dot net) is a software framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large class library known as Framework Class Library (FCL) and provides language interoperability (each language can use code written in other languages) across several programming languages. Programs are written for the .NET Framework execute in a software environment (as contrasted to a hardware environment), known as the Common Language Runtime (CLR), an application virtual machine that provides services such as security, memory management, and exception handling. FCL and the CLR together constitute .NET Framework.

FCL provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. Programmers produce software by combining their own source code with the .NET Framework and other libraries. .NET Framework is intended to be used by most new applications created for the Windows platform. Microsoft also produces an integrated development environment largely for .NET software called Visual Studio.

The first level of the representation is the operating system; the .NET layer is located between the system and applications. The second level is the Common Language Runtime (CLR), which provides the part of the .NET Framework doing the most work. We discuss the CLR later in this chapter. The next level is the Base Class Library (BCL), which provides all .NET objects that can be used both in your code and by Visual Basic when creating applications. The BCL also provides the infrastructure of several .NET technologies that you use in building applications, such as WPF, Windows Forms, ASP.NET, WCF, and so on. The last level is represented by applications that rely on the previous layers.

**Common Language Runtime engine**

Common Language Runtime (CLR) serves as the execution engine of the .NET Framework. All .NET programs execute under the supervision of the CLR, guaranteeing certain properties and behaviors in the areas of memory management, security, and exception handling.

**Language independence**

.NET Framework introduces a Common Type System, or CTS. The CTS specification defines all possible data types and programming constructs supported by the CLR and how they may or may not interact with each other conforming to the Common Language Infrastructure (CLI) specification. Because of this feature, the .NET Framework supports the exchange of types and object instances between libraries and applications written using any conforming.NET language.

**Framework Class Library**

Framework Class Library (FCL) is a library of functionality available to all languages using the .NET Framework. FCL provides classes that encapsulate a number of common functions, including file reading and writing, graphic rendering, database interaction, XML document manipulation, and so on. It consists of classes, interfaces of reusable types that integrate CLR.

**Simplified deployment**

The .NET Framework includes design features and tools which help manage the installation of computer software to ensure that it does not interfere with previously installed software and that it conforms to security requirements.

**Security**

The design addresses some of the vulnerabilities, such as buffer overflows, which have been exploited by malicious software. Additionally, .NET provides a common security model for all applications.

**Portability**

While Microsoft has never implemented the full framework on any system except Microsoft Windows, it has engineered the framework to be platform-agnostic, and cross-platform implementations are available for other operating systems (see the Silver light and § Alternative implementations). Microsoft submitted the specifications for CLI (which includes the core class libraries, CTS, and the Common Intermediate Language), C#, and C++/CLI to both ECMA and ISO, making them available as official standards. This makes it possible for third parties to create compatible implementations of the framework and its languages on other platforms.

**Common Language Specification**

• CLS is a set of rules that specifies the features that all languages should support.

* **Goal**: have the .NET framework support multiple languages.
* **CLS** is an agreement among language designers and class library designers about the features and usage conventions that can be relied upon.

• Example: public names should not rely on the case for the uniqueness

Since some languages are not case sensitive.

• This does not mean all languages are not case sensitive above the CLR.

**The Common Type System**

• At the core of the Framework is a universal type system called the .NET Common Type System (CTS)?

• Everything is an object - but efficient

– Boxing and Unboxing

• All types fall into two categories - Value types and Reference types.

– Value types contain actual data (cannot be null).

– Stored on the stack.

– Three kinds of value types: Primitives, structures, and enumerations.

• Language compilers map keywords to the primitive types. For

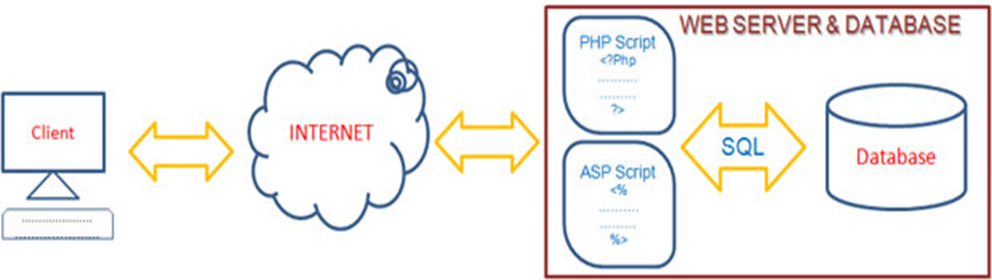
For example, a C# “int” is mapped to System.Int32.

**5.2 SQL SERVER 2014**

SQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. It is a popular choice of database for use in web applications and is an open-source product.

The process of setting up a SQL database varies from host to host, however, we will end up with a database name, a user name, and a password. Before using our database, we must create a table. A table is a section of the database for storing related information. In a table, we will set up the different fields that will be used. Creating a table is simple; we just have to type the name, select the number of fields and click the ‘go’ button.

We will then be taken to a setup screen where you must create the fields in the database.

****

**Fig 5.2.1 SQL DATABASE**

SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems.

SQL statements are used to perform tasks such as update data on a database or retrieve data from a database. Some common relational database management systems that use SQL are Oracle, Sybase, Microsoft SQL Server, Access, Ingres, etc. Although most database systems use SQL, most of them also have their own additional proprietary extensions that are usually only used on their system. However, the standard SQL commands such as "Select", "Insert", "Update", "Delete", "Create", and "Drop" can be used to accomplish almost everything that one needs to do with a database.

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet).

SQL Server 2014 (formerly codenamed "Katmai") was released on August 6, 2008, and aims to make data management, self-tuning, self-organizing, and self-maintaining with the development of SQL Server Always On technologies, to provide near-zero downtime. SQL Server 2008 also includes support for structured and semi-structured data, including digital media formats for pictures, audio, video, and other multimedia data. In current versions, such multimedia data can be stored as BLOBs (binary large objects), but they are generic bitstreams. Intrinsic awareness of multimedia data will allow specialized functions to be performed on them. According to Paul Flessner, senior Vice President, Server, Applications, Microsoft Corp., SQL Server 2008 can be a data storage backend for different varieties of data: XML, email, time/calendar, file, document, spatial, etc as well as perform search, query, analysis, sharing, and synchronization across all data types.

**6. SYSTEM DESIGN**

**6.1 INTRODUCTION**

Software design sits in the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the

development phase for any engineered product or system. The designer’s goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement has been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word “Quality”. Design is the place where quality is fostered in software development. The design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer’s view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design, we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either a technical or project management perspective. From the technical point of view, the design is

comprised of four activities – architectural design, data structure design, interface design, and procedural design.

**6.2 NORMALIZATION**

It is a process of converting a relation to a standard form. The process is used to handle the problems that can arise due to data redundancy, i.e. repetition of data in the database, maintainata integrity as well as handling problems that can arise due to insert, Update, deletes anomalies.

Decomposing is the process of splitting relations into multiple relations to eliminate anomalies and maintain anomalies and maintain data integrity. To do this we use normal forms or rules for structuring relations.

Insertion anomaly: Inability to add data to the database due to the absence of other data.

Deletion anomaly: Unintended loss of data due to the deletion of other data.

Update anomaly: Data inconsistency resulting from data redundancy and partial update

Normal Forms: These are the rules for structuring relations that eliminate anomalies.

**FIRST NORMAL FORM:**

A relation is said to be in first normal form if the values in the relation are atomic for every attribute in the relation. By this, we mean simply that no attribute value can be a set of values or, as it is sometimes expressed, a repeating group.

**SECOND NORMAL FORM:**

A relation is said to be in the second normal form is it is in first normal form and it should satisfy any one of the following rules.

* The primary key is not a composite primary key
* No, no key attributes are present
* Every now key attribute is fully functionally dependent on a full set of the primary key.

**THIRD NORMAL FORM:**

A relation is said to be in third normal form if their exits no transitive dependencies.

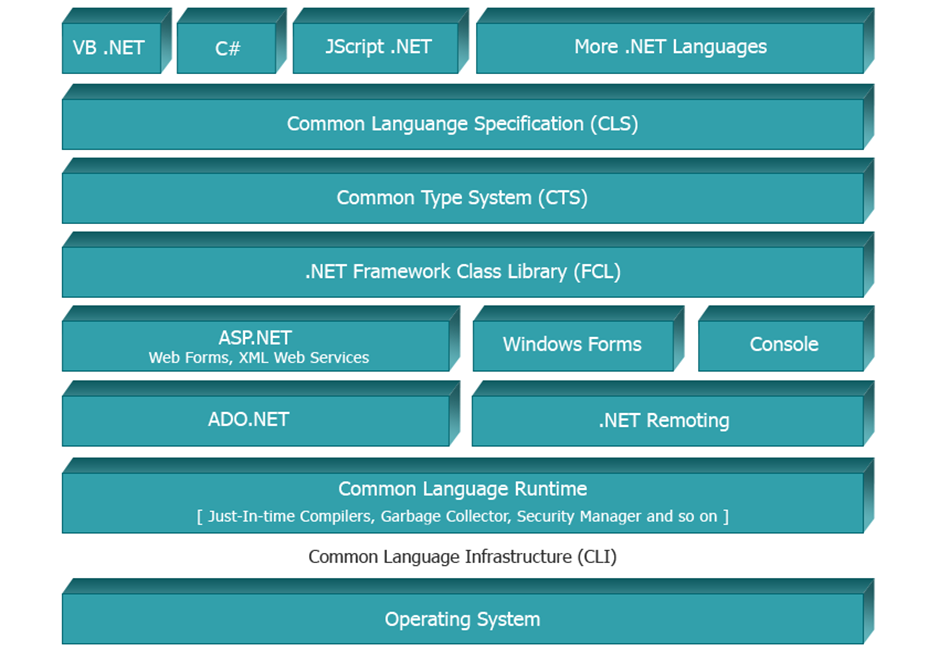
Transitive Dependency:

If two on key attributes depend on each other as well as on the primary key, then they are said to be transitively dependent.

The above normalization principles were applied to decompose the data in multiple tables, thereby making the data to be maintained in a consistent state.

**6.3 SYSTEM ARCHITECTURE**

**ARCHITECTURE OF .NET FRAMEWORK**

****

**Fig 6.3.1 .NET FRAMEWORK**

.NET Framework is a software framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large class library named Framework Class Library (FCL) and provides language interoperability (each language can use code written in other languages) across several programming languages. Programs are written for .NET Framework execute in a software environment (in contrast to a hardware environment) named Common Language Runtime(CLR), an application virtual machine that provides services such as security, memory management, and exception handling. As such, computer code written using .NET Framework is called "managed code". FCL and CLR together constitute the .NET Framework.

FCL provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. Programmers produce software by combining their source code with .NET Framework and other libraries. The framework is intended to be used by most new applications created for the Windows platform. Microsoft also produces an integrated development environment largely for .NET software called Visual Studio.

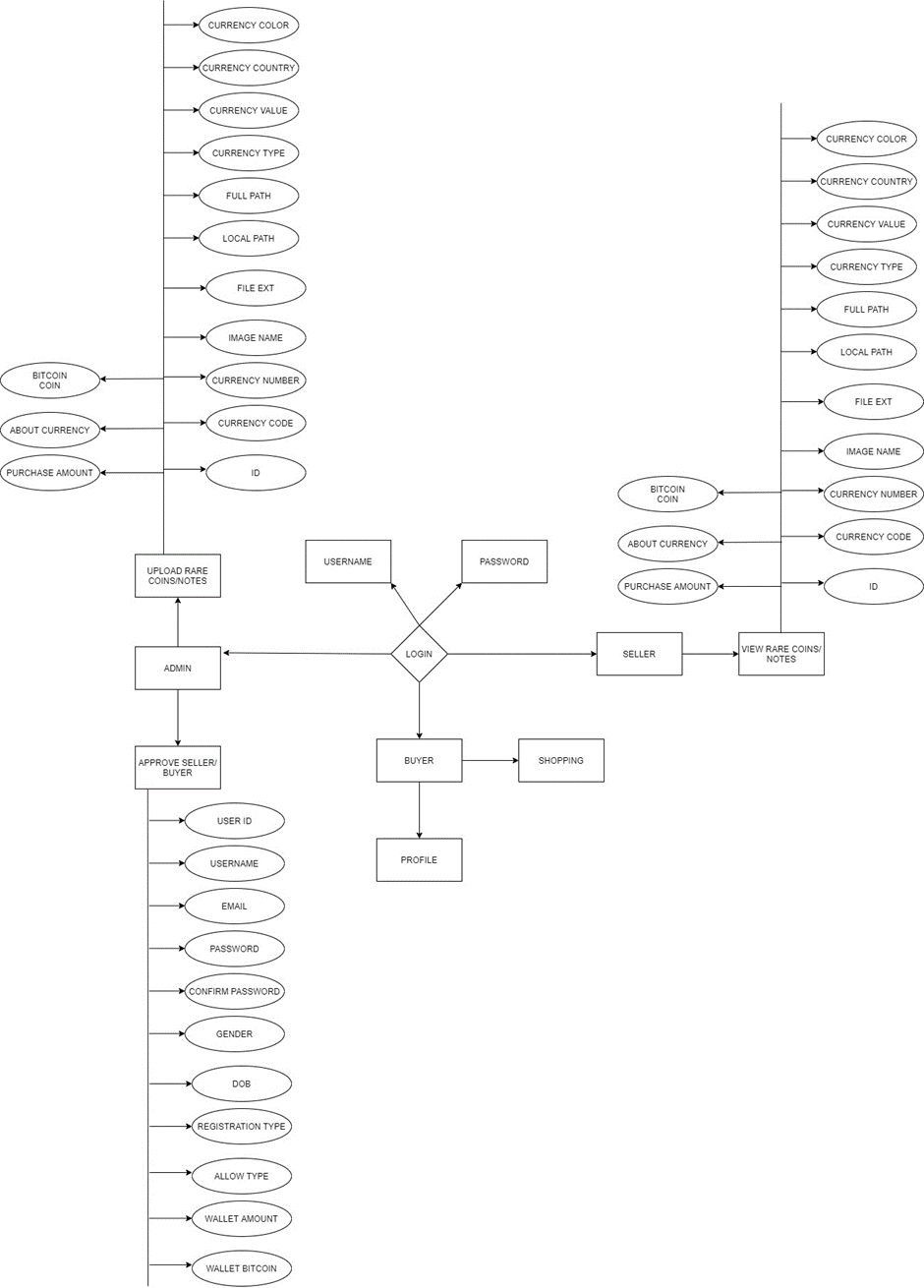
.NET Framework began as proprietary software, although the firm worked to standardize the software stack almost immediately, even before its first release. Despite the standardization efforts, developers, mainly those in the free and open-source software communities, expressed their unease with the selected terms and the prospects of any free and open-source implementation, especially regarding software patents. Since then, Microsoft has changed .NET development to more closely follow a contemporary model of a community-developed software project, including issuing an update to its patent promising to address the concerns.

.NET Framework led to a family of .NET platforms targeting mobile computing, embedded devices, alternative operating systems, and web browser plug-ins. A reduced version of the framework, .NET Compact Framework, is available on Windows CE platforms, including Windows Mobile devices such as smartphones. .NET Micro Framework is targeted at very resource-constrained embedded devices. The silver light was available as a web browser plug-in. Mono is available for many operating systems and is customized into popular smartphone operating systems (Android and iOS) and game engines. .NET Core targets the Universal Windows Platform (UWP), and cross-platform and workloads.

**6.4 E – R DIAGRAMS**

* The relation upon the system is structured through a conceptual ER-Diagram, which not only specifics the existing entities, but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.
* The Entity-Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct, the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object description.
* The set of primary components that are identified by the ERD are
* Data object
* Relationships
* Attributes
* Various types of indicators.

The primary purpose of the ERD is to represent data objects and their relationships.



**6.5 DATA FLOW DIAGRAM**

Definition:

A Data Flow Diagram (DFD) is a graphical tool used to describe and analyze the movement of data through the system. It is a graphical representation of the “flow” of data through a computer system or a data or it looks at how data flows through a system. These are a central tool and basic from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. The development of DFD is done at several levels. The flow diagram describes the boxes that describe computations, decisions, interactions & loops. It is important to keep in mind that the flow diagrams are not flowcharts and should not include control elements.

Characteristics

* Information and/or data flow is represented by a labeled arrow
* Processes (transformations) are represented by labeled circles (bubbles)
* Information sources and sinks are represented by boxes
* Files and depositories are represented by a rounded rectangle or a double line.

Types

* Logical data flow diagram
* Physical data flow diagram

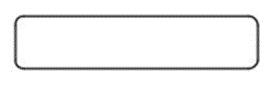
Features

* The DFD shows data, not the control loops and decisions are controlled considerations do not appear on a DFD
* The DFD does not indicate the time factor involved in any process whether the dataflow takes place easily daily, weekly, monthly or yearly
* The sequence of events is to bring out on DFD

DFD Symbols

Process

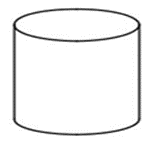
A process transforms incoming data flow into outsourcing data flow.



**Fig 6.5.1 DFD Process**

Data store

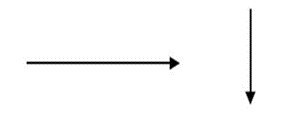
Data sources are repositories of data in the system. They are sometimes also referred to as files.



**Fig 6.5.2 DFD Datastore**

Data flow

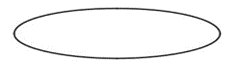
Data flows are pipelines through which packets of information flow. Label the arrows with the name of the data that moves through it.



**Fig 6.5.3 DFD Dataflow**

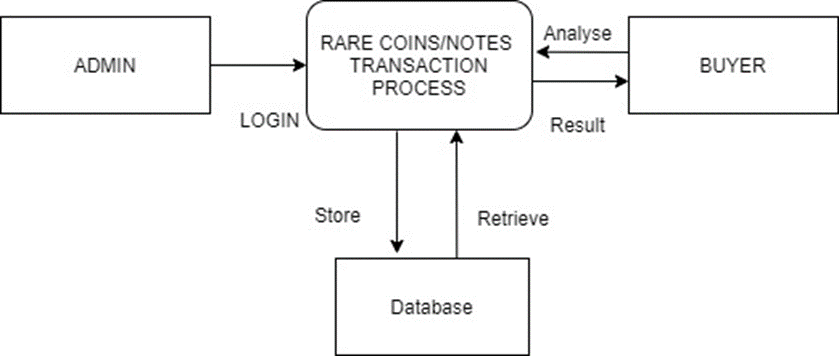
External Entity

External entities are objects outside the system, with which the system communicates.

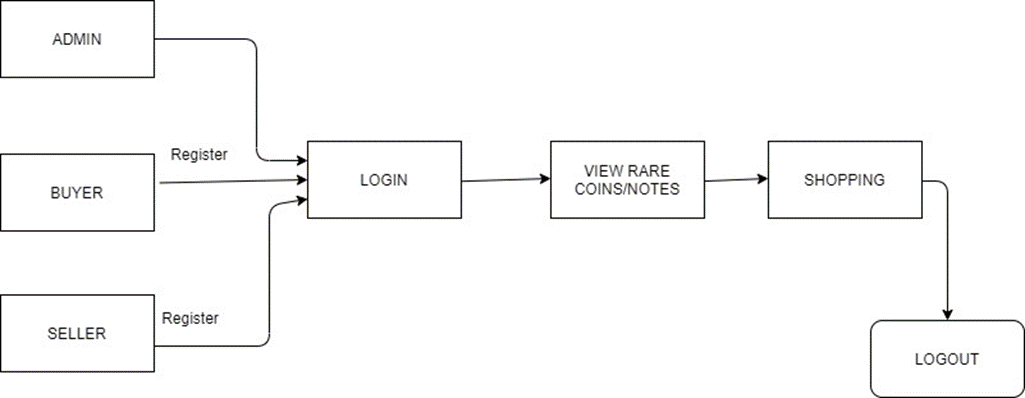


**Fig 6.5.4 DFD External Entity**

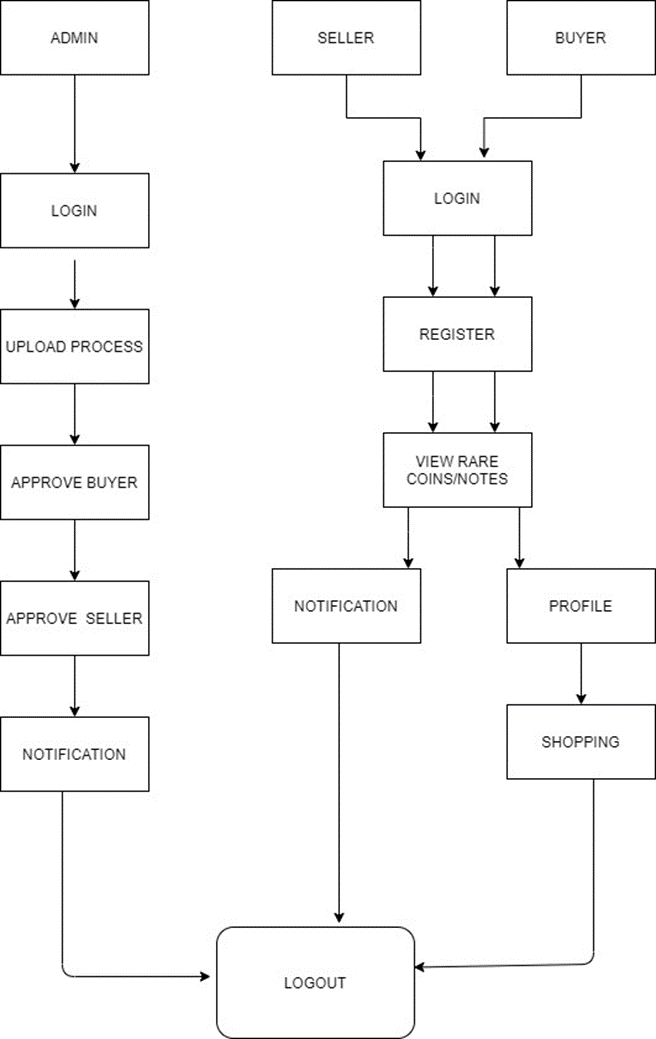
**DFD LEVEL-0**



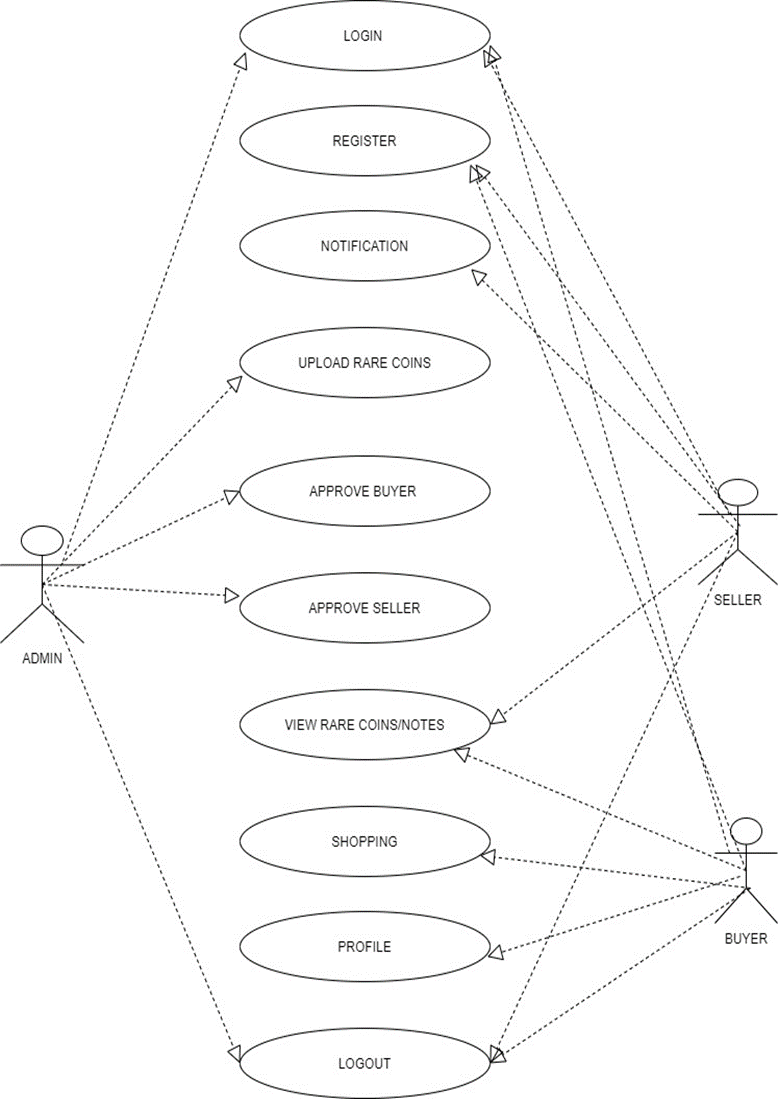
**DFD LEVEL-1**



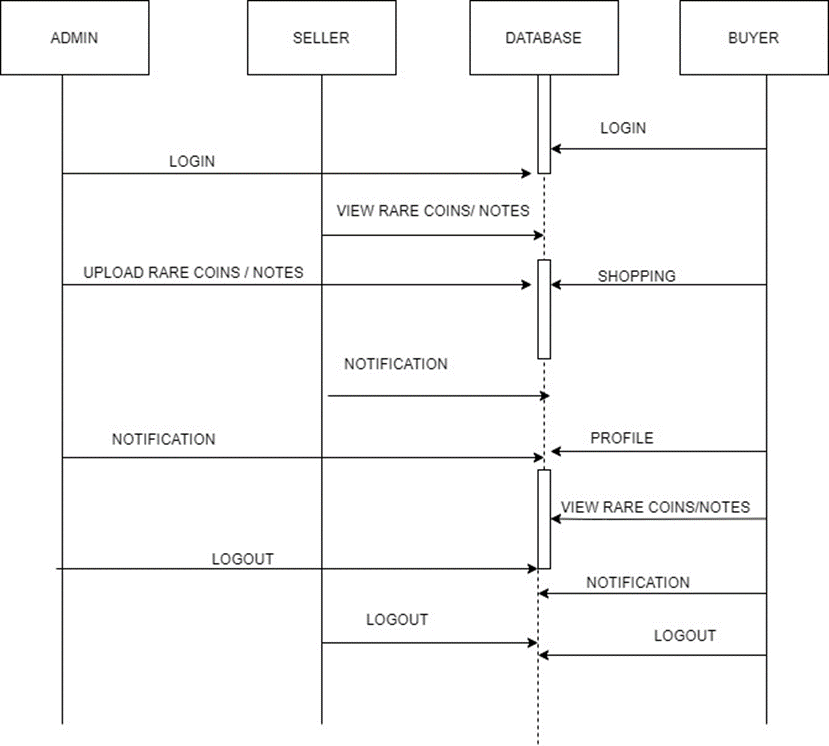
**DFD LEVEL2**



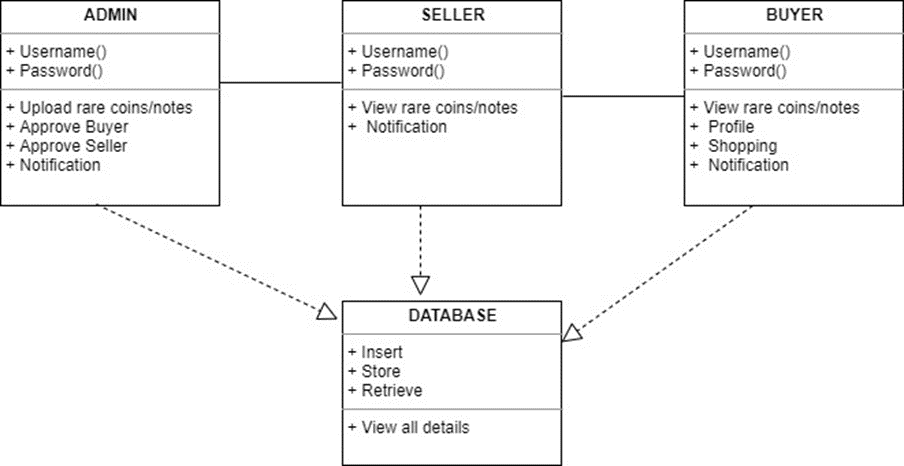
**6.6 USE CASE DIAGRAM**



**6.7 SEQUENCE DIAGRAM**



**6.8 CLASS DIAGRAM**



**ADMIN**

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN NAME | DATATYPE | SIZE | CONSTRAINTS |
| USER NAME | **VARCHAR** | **25** | **NULL** |
| PASSWORD | **VARCHAR** | **15** | **NULL** |
| UPLOAD RARE COINS/NOTES | **INT** | **15** | **NULL** |
| APPROVE BUYER | **VARCHAR** | **25** | **NULL** |
| APPROVE SELLER | **VARCHAR** | **25** | **NULL** |

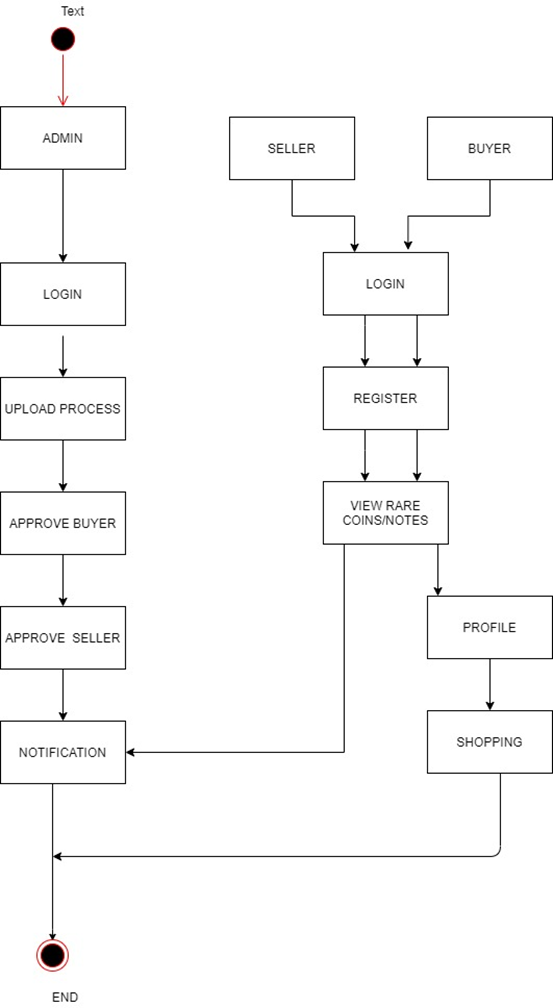
**SELLER**

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN NAME | DATATYPE | SIZE | CONSTRAINTS |
| USERNAME | **VARCHAR** | **25** | **NULL** |
| PASSWORD | **VARCHAR** | **15** | **NULL** |
| VIEW RARE COINS/NOTES | **INT** | **25** | **READ ONLY** |

**BUYER**

|  |  |  |  |
| --- | --- | --- | --- |
| COLUMN NAME | DATATYPE | SIZE | CONSTRAINTS |
| USERNAME | **VARCHAR** | **25** | **NULL** |
| PASSWORD | **VARCHAR** | **15** | **NULL** |
| VIEW RARE COINS/NOTES | **INT** | **15** | **READ ONLY** |
| PROFILE | **VARCHAR** | **25** | **NULL** |
| UPDATE SHOPPING | **VARCHAR** | **25** | **NULL** |

**6.9 ACTIVITY DIAGRAM**



**7. SYSTEM SECURITY**

**10.1 INTRODUCTION**

The protection of computer-based resources that includes hardware, software, data, procedures and people against unauthorized use or natural

Disaster is known as System Security.

A security system can be divided into four related issues:

* Security
* Integrity
* Privacy
* Confidentiality

SYSTEM SECURITY refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

DATA SECURITY is the protection of data from loss, disclosure, modification, and destruction.

SYSTEM INTEGRITY refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

PRIVACY defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair or excessive dissemination of information about it.

CONFIDENTIALITY is a special status given to sensitive information in a database to minimize the possible invasion of privacy. It is an attribute of information that characterizes its need for protection.

**10.2 SECURITY IN SOFTWARE**

System security refers to various validations on data in the form of checks and controls to avoid the system from failing. It is always important to ensure that only valid data is entered and only valid operations are performed on the system. The system employs two types of checks and controls:

CLIENT SIDE VALIDATION

Various client-side validations are used to ensure on the client-side that only valid data is entered. Client-side validation saves server time and loads to handle invalid data. Some checks are imposed:

* JavaScript is used to ensure those required fields are filled with suitable data only. The maximum lengths of the fields of the forms are appropriately defined.
* Forms cannot be submitted without filling up the mandatory data so that manual mistakes of submitting empty fields that are mandatory can be sorted out at the client-side to save the server time and load.
* Tab-indexes are set according to the need and taking into account the ease of use while working with the system.

SERVER SIDE VALIDATION

Some checks cannot be applied on the client-side. Server-side checks are necessary to save the system from failing and intimating the user that some invalid operation has been performed or the performed operation is restricted. Some of the server-side checks imposed are:

* A server-side constraint has been imposed to check for the validity of the primary key and foreign key. A primary key value cannot be duplicated. Any attempt to duplicate the primary value results in a message intimating the user about those values through the forms using the foreign key can be updated only of the existing foreign key values.
* The user is intimated through appropriate messages about the successful operations or exceptions occurring at the server-side.

**8. CONCLUSION**

We introduced the concept of Traceable Monero and proposed a concrete construction of this new crypto currency. The proposed framework achieves the properties of correctness, balance, anonymity and traceability. Both the efﬁciency analysis and the implementation results show that the proposed system is comparable to the underlying Monero in efﬁciency. We leave the following problems as future works. Finding more methods to reduce the trust of the tracing authority except threshold mechanisms. Designing new tracing approaches for other anonymous crypto currencies.

**REFERENCES**

[1] Y. Yu, Y. Li, J. Tian, J. Liu. “Blockchain-Based Solutions to Security and Privacy Issues in the Internet of Things”, IEEE Wireless Commun., vol. 25, no. 6, pp. 12-18, 2018.

[2] N. van Saberhagen. “Cryptonote v 2.0”. https://cryptonote.org/whitepaper. pdf, 2013.

[3] J. K. Liu, V. K. Wei, D. S. Wong. “Linkable spontaneous anonymous group signature for ad hoc groups”, Australasian Conference on Information Security and Privacy.Springer, Berlin, Heidelberg, pp. 325-335, 2004.

[4] M. Jakobsson, M. Yung. “Revokable and versatile electronic money”, In Proceedings of the 3rd ACM conference on Computer and communications security, pp. 76-87, 1996.

[5] J. Camenisch, J.-M. Piveteau, M. Stadler. “An Efficient Fair Payment System”, ACM Conference on Computer and Communications Security, pp. 88-94, 1966.

[6] J. Camenisch, U. Maurer, M. Stadler. “Digital payment systems with passive anonymity-revoking trustees”. Journal of Computer Security, vol. 5, no. 1, pp. 69-89, 1997.

[7] D. Kulger, H. Vogt. “Off-line payment with auditable tracing”, Financial cryptography-FC’ 2002, Berlin: SpringerVerlag, pp.42-55, 2002.

[8] J. Barcelo. “User privacy in the public bitcoin blockchain”. http://www. dtic. upf.edu/˜jbarcelo/papers/20140704 User Privacy in the Public Bitcoin Blockc hain/paper. pdf, 2014.

[9] M. Mo¨ser, K. Soska, E. Heilman, K. Lee, H. Heffan, S. Srivastava, N. Christin. “An Empirical Analysis of Traceability in the Monero Blockchain”, Proceedings on Privacy Enhancing Technologies, vol. 2018, no. 3, pp. 143-163, 2018.

[10] E. Cecchetti, F. Zhang, Y. Ji, A. Kosba, A. Juels, E. Shi. “Solidus: Confidential distributed ledger transactions via PVORM”, Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. ACM, pp. 701- 717, 2017.

[11] C. Garman, M. Green and I. Miers. “Accountable privacy for decentralized anonymous payments”, International Conference on Financial Cryptography and Data Security. Springer, Berlin, Heidelberg, pp. 81-98, 2016.

[12] J. Katz. “Digital signatures”, Springer Science and Business Media, 2010

**APPENDIX I**

**CODE**

**APP CREATION**

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

using System.Data.SqlClient;

public class Class1

{

SqlConnectio con = new SqlConnection(ConfigurationManager.AppSettings["socialcln"]);

string id, id1,id2,id3,id4,id5,id6,id7;

int eid, eid1,eid2,eid3,eid4,eid5,eid6,eid7;

public Class1()

{

}

public int addproductidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(proid) from addproduct", con);

id = Convert.ToString(c1.ExecuteScalar());

if (id == "")

{

eid = 1;

}

else

{

eid = Convert.ToInt16(id);

eid = eid + 1;

}

con.Close();

return eid;

}

public int registrationidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(userid) from registration", con);

id1 = Convert.ToString(c1.ExecuteScalar());

if (id1 == "")

{

eid1 = 1;

}

else

{

eid1 = Convert.ToInt16(id1);

eid1 = eid1 + 1;

}

con.Close();

return eid1;

}

public int admincomposeidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(adminuserid) from admincompose", con);

id2 = Convert.ToString(c1.ExecuteScalar());

if (id2 == "")

{

eid2 = 1;

}

else

{

eid2 = Convert.ToInt16(id2);

eid2 = eid2 + 1;

}

con.Close();

return eid2;

}

public int usercomposeidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(usercomposeid) from usercompose", con);

id3 = Convert.ToString(c1.ExecuteScalar());

if (id3 == "")

{

eid3 = 1;

}

else

{

eid3 = Convert.ToInt16(id3);

eid3 = eid3 + 1;

}

con.Close();

return eid3;

}

public int postentryidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(entryid) from postentry", con);

id4 = Convert.ToString(c1.ExecuteScalar());

if (id4 == "")

{

eid4 = 1;

}

else

{

eid4 = Convert.ToInt16(id4);

eid4 = eid4 + 1;

}

con.Close();

return eid4;

}

public int adminupdateidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(updateid) from adminupdate", con);

id5 = Convert.ToString(c1.ExecuteScalar());

if (id5 == "")

{

eid5 = 1;

}

else

{

eid5 = Convert.ToInt16(id5);

eid5 = eid5 + 1;

}

con.Close();

return eid5;

}

public int commentidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(cmtid) from addcomment", con);

id6 = Convert.ToString(c1.ExecuteScalar());

if (id6 == "")

{

eid6 = 1;

}

else

{

eid6 = Convert.ToInt16(id6);

eid6 = eid6 + 1;

}

con.Close();

return eid6;

}

public int custidgeneration()

{

con.Open();

SqlCommand c1 = new SqlCommand("select max(custid) from ordercust", con);

id7 = Convert.ToString(c1.ExecuteScalar());

if (id7 == "")

{

eid7 = 1;

}

else

{

eid7 = Convert.ToInt16(id7);

eid7 = eid7 + 1;

}

con.Close();

return eid7;

}

}

**CRYPTOGRAPHY FOR APP**

using System;

using System.Data;

using System.Configuration;

using System.Linq;

using System.Web;

using System.Web.Security;

using System.Web.UI;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

using System.Web.UI.WebControls.WebParts;

using System.Xml.Linq;

using System.Text;

using System.Security.Cryptography;

using System.IO;

public class Cryptography

{

//public static RSACryptoServiceProvider rsa;

Private static string sKey = "UJYHCX783her\*&5@$%#(MJCX\*\*38n\*#6835ncv56tvbry(&#MX98cn342cn4\*&X#&";

public Cryptography()

{

}

public string Encrypt(string sPainText)

{

if (sPainText.Length == 0)

return (sPainText);

return (EncryptString(sPainText, sKey));

}

public string Decrypt(string sEncryptText)

{

if (sEncryptText.Length == 0)

return (sEncryptText);

return (DecryptString(sEncryptText, sKey));

}

protected string EncryptString(string InputText, string Password)

{

RijndaelManaged RijndaelCipher = new RijndaelManaged();

byte[] PlainText = System.Text.Encoding.Unicode.GetBytes(InputText);

byte[] Salt = Encoding.ASCII.GetBytes(Password.Length.ToString());

PasswordDeriveBytes SecretKey = new PasswordDeriveBytes(Password, Salt);

ICryptoTransform Encryptor = RijndaelCipher.CreateEncryptor(SecretKey.GetBytes(16), SecretKey.GetBytes(16));

MemoryStream memoryStream = new MemoryStream();

CryptoStream cryptoStream = new CryptoStream(memoryStream, Encryptor, CryptoStreamMode.Write);

cryptoStream.FlushFinalBlock();

byte[] CipherBytes = memoryStream.ToArray();

memoryStream.Close();

cryptoStream.Close();

string EncryptedData = Convert.ToBase64String(CipherBytes);

return EncryptedData;

}

protected string DecryptString(string InputText, string Password)

{

try

{

RijndaelManaged RijndaelCipher = new RijndaelManaged();

byte[] EncryptedData = Convert.FromBase64String(InputText);

byte[] Salt = Encoding.ASCII.GetBytes(Password.Length.ToString());

PasswordDeriveBytes SecretKey = new PasswordDeriveBytes(Password, Salt);

ICryptoTransform Decryptor = RijndaelCipher.CreateDecryptor(SecretKey.GetBytes(16), SecretKey.GetBytes(16));

MemoryStream memoryStream = new MemoryStream(EncryptedData);

CryptoStream cryptoStream = new CryptoStream(memoryStream, Decryptor, CryptoStreamMode.Read);

byte[] PlainText = new byte[EncryptedData.Length];

int DecryptedCount = cryptoStream.Read(PlainText, 0, PlainText.Length);

memoryStream.Close();

cryptoStream.Close();

string DecryptedData = Encoding.Unicode.GetString(PlainText, 0, DecryptedCount);

return DecryptedData;

}

catch (Exception exception)

{

return (exception.Message);

}

}

public string pwdgeneration()

{

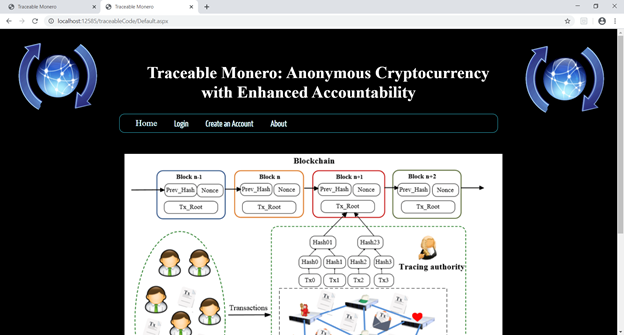
string pwd = "5jqeBRFP6nA03b0sSXBg+pNi1kli22kVwnOjwKVtBXGC5+k/nNDwoFDvHmJgYKUP";

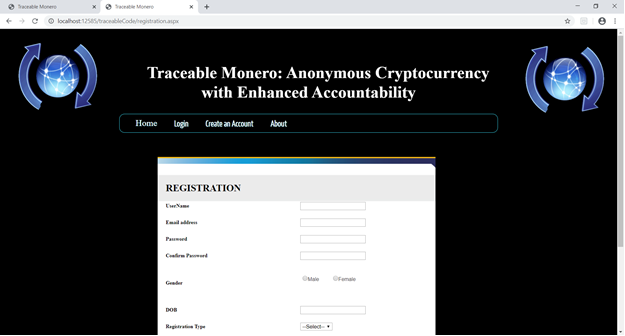
return pwd;

}

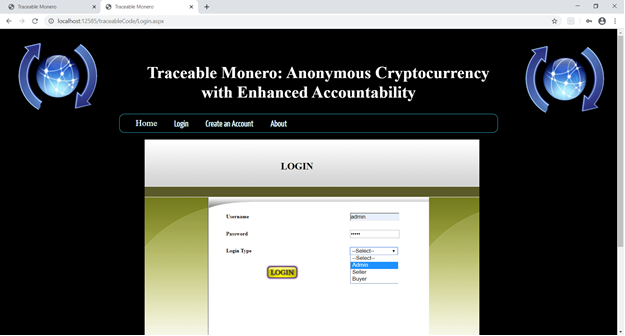
}

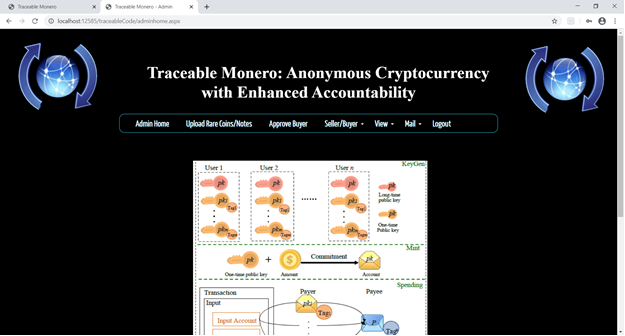
**APPENDIX II**

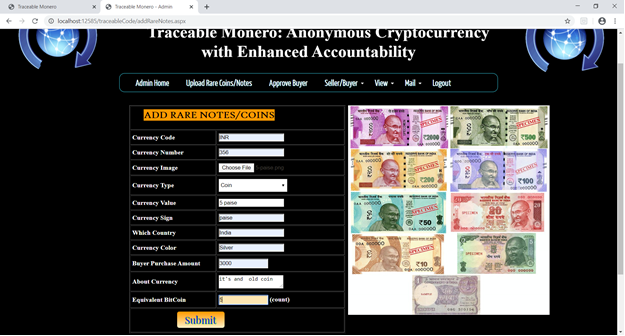
****

****

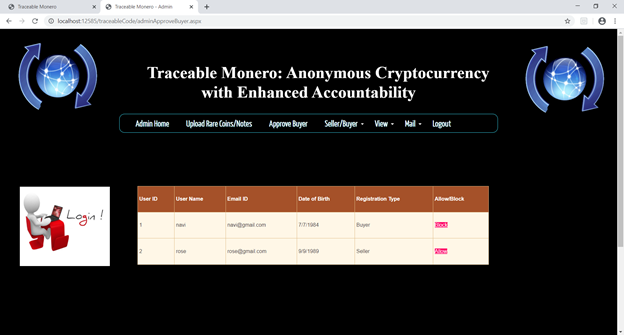
**ADMIN**

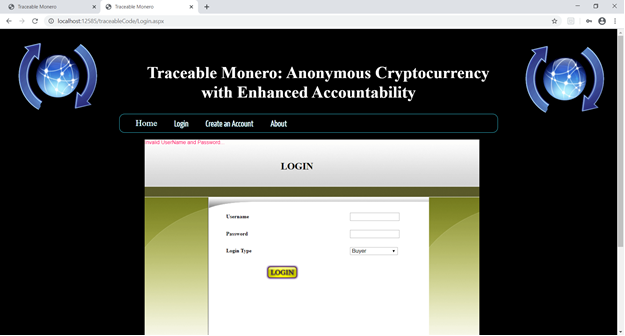
****

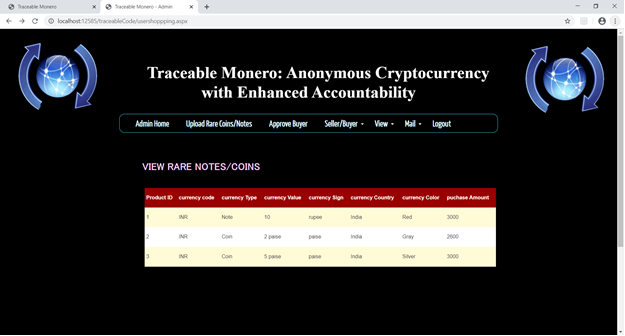
****

****

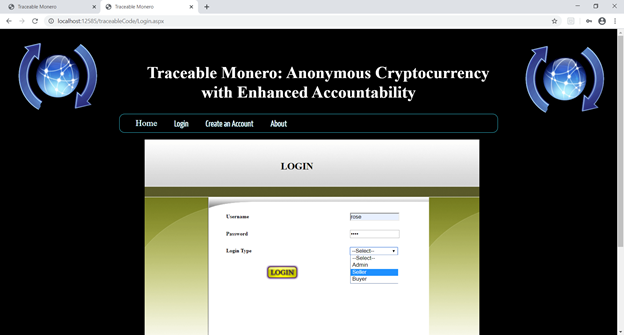
**APPROVE BUYER**

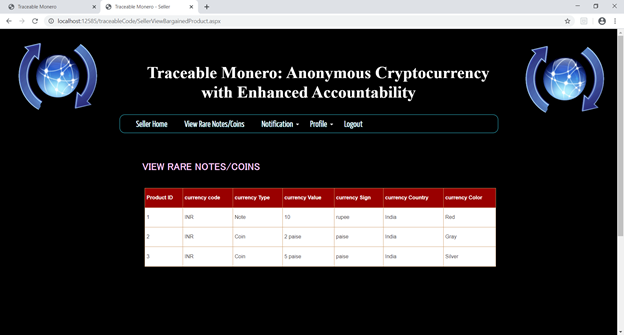
****

****

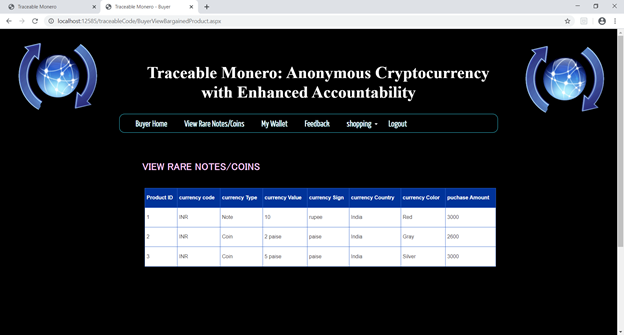
****

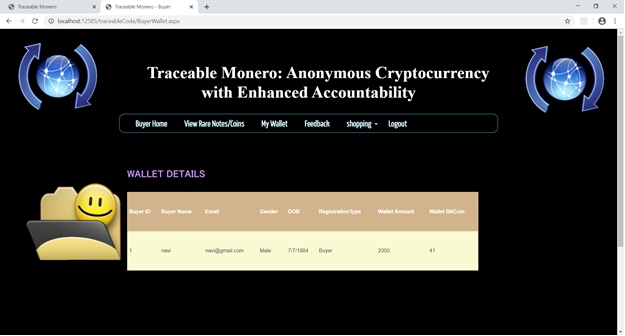
**SELLER**

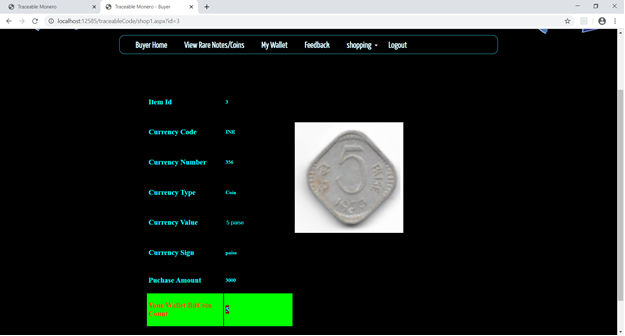
****

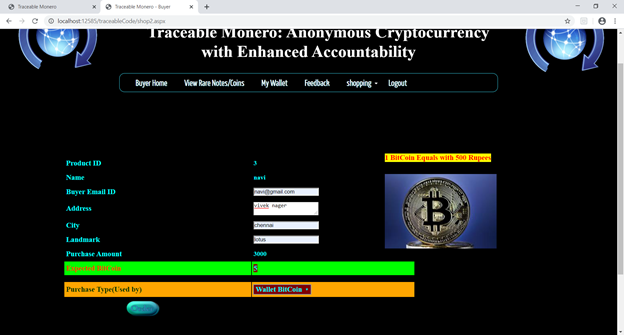
****

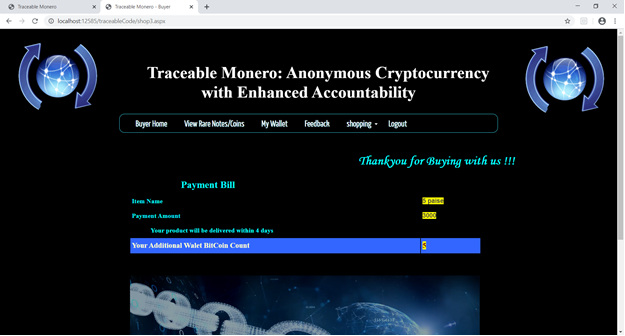
**BUYER**

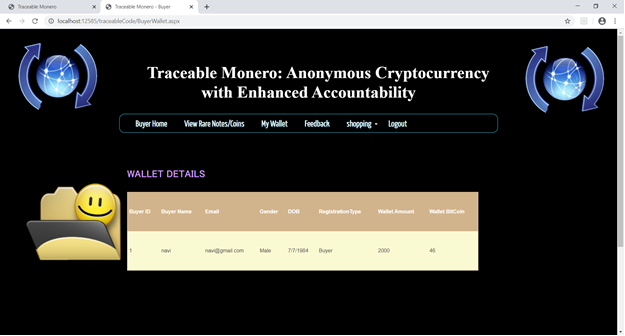
****

****

****

****

****

****