# PRODUCT TRACKING AND TRACING WITH DECENTRALISED BLOCKCHAIN

## A PROJECT REPORT

***Submitted by***

**AISHWARYA SURESH (REGISTER NO:211417104007) DEEPIKA P (REGISTER NO:211417104047) KIRUTHIKA S (REGISTER NO:211417104121)**

***in partial fulfillment for the award the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

## COMPUTER SCIENCE AND ENGINEERING

**PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.**

**ANNA UNIVERSITY: CHENNAI-600 025**

## AUGUST 2021

**BONAFIDE CERTIFICATE**

Certified that this project report **“PRODUCT TRACKING AND TRACING WITH DECENTRALISED BLOCKCHAIN”** is the bonafide work of “**AISHWARYA SURESH (211417104007), DEEPIKA P (211417104047)** and

**KIRUTHIKA S (211417104121)”** who carried out the project work under my supervision.

## SIGNATURE Dr.S.MURUGAVALLI,M.E.,Ph.D., HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, NAZARATHPETTAI,

POONAMALLEE, CHENNAI-600 123.

## SIGNATURE Mrs.K.KIRUTHIKA, M.Tech SUPERVISOR

**ASSISTANT PROFESSOR**

DEPARTMENT OF CSE,

PANIMALAR ENGINEERING COLLEGE, NAZARATHPETTAI,

POONAMALLEE, CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on **05.08.2021.**

## INTERNAL EXAMINER EXTERNAL EXAMINER

**ACKNOWLEDGEMENT**

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors

***Tmt.C.VIJAYARAJESWARI****,* ***Dr.C.SAKTHIKUMAR, M.E., Ph.D.,*** *and*

**Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,**

for the support extended throughout the project.

We would like to thank my **Project Guide Mrs.K.KIRUTHIKA, M.Tech** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

AISHWARYA SURESH

DEEPIKA P KIRUTHIKA S

## ABSTRACT

Supply chain management enhanced by the Internet of Things (IoT) solutions integrate special tags (e.g., RFID, NFC, and QR-codes) with products to create Smart Tags, in addition to storing supplemental information about a product, which is also used to track products during their lifecycle. However, a product consumer has to implicitly trust the Smart Tag creator and other stakeholders within the supply chain that they are providing authentic data within a product's tag. The DL-Tags solution steps into this environment to offer a decentralized, privacy-preserving, and variable management of Smart Tags during a product's lifecycle. The solution is based on distributed ledger technology (DLT) and uses the Ethereum blockchain to mediate interactions between the stakeholders during a product's exchange process. By reaching a consensus on the product's description and state logged on the blockchain, all involved stakeholders and product consumers can verify the product's authenticity without revealing their identity. The paper describes the DL-Tags solution and includes a cost analysis of all implemented transactions on the Ethereum blockchain. The proposed solution provides evidence of the product's origin and its journey across the supply chain while preventing tag duplication and manipulation. It is among the rest documented practical solutions using DLT and IoT for supply chain management, which is designed to be distributed ledger

agnostic.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **ABSTRACT**  **LIST OF TABLES** | iii  vii |
|  | **LIST OF FIGURES** | viii |
|  | **LIST OF ABBREVIATIONS** | x |
| **1.** | **INTRODUCTION** | 1 |
|  | 1.1 Overview | 1 |
|  | 1.2 Problem Definition | 1 |
| **2.** | **LITERATURE SURVEY** | 3 |
| **3.** | **SYSTEM ANALYSIS** | 6 |
|  | 3.1 Existing System | 6 |
|  | 3.2 Proposed system | 6 |
|  | 3.3 Purpose | 7 |
|  | 3.4 Requirement Analysis and Specification | 7 |
|  | 3.4.1 Performance Requirements | 9 |
|  | 3.4.2 Safety Requirements | 9 |
|  | 3.4.3 Hardware Requirements | 10 |
|  | 3.4.4 Software Requirements | 10 |
|  | 3.5 Technology Stack | 10 |
| **4.** | **DESIGN AND IMPLEMENTATION CONSTRAINTS** | 11 |
|  | 4.1 Constraints in Analysis | 11 |
|  | 4.2 Constraints in Design | 11 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | 4.3 Constraints in Implementation | 11 |
| **5.** | **SYSTEM DESIGN** | 12 |
|  | 5.1 UML Diagram | 12 |
|  | 5.1.1. Use case Diagram | 12 |
|  | 5.1.2. Sequence Diagram | 13 |
|  | 5.1.3. Activity Diagram | 14 |
|  | 5.1.4. Collaboration Diagram | 15 |
|  | 5.1.5. Class Diagram | 15 |
|  | 5.1.6. Data Flow Diagram | 16 |
| **6.** | **SYSTEM ARCHITECTURE** | 19 |
|  | 6.1 Architecture Overview | 19 |
|  | 6.2 Module Design Specification | 21 |
|  | 6.2.1. Module Explanation | 21 |
|  | 6.2.1.1. Creating Suppliers | 21 |
|  | 6.2.1.2. Manufacturer Process | 21 |
|  | 6.2.1.3. Distributors Transactions | 21 |
|  | 6.2.1.4. Product Verification | 21 |
|  | 6.3 Program Design Language | 22 |
|  | 6.3.1. Java | 22 |
|  | 6.3.2. Apache Tomcat Server | 26 |
|  | 6.3.3. Blockchain | 27 |
| **7.** | **SYSTEM IMPLEMENTATION** | 31 |

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | 7.1 Client-side coding | 31 |
|  | 7.2 Server-side coding | 33 |
| **8.** | **SYSTEM TESTING** | 40 |
|  | 8.1 Unit Testing | 40 |
|  | 8.2 Integration Testing | 40 |
|  | 8.3 Functional Tests | 41 |
|  | 8.4 Performance Test | 41 |
|  | 8.5 Stress Test | 41 |
|  | 8.6.Structured Test | 41 |
|  | 8.7. Testing Techniques / Testing Strategies | 42 |
|  | 8.8. Test Cases & Reports / Performance Analysis | 46 |
|  | 8.8.1. Supplier Login Analysis | 46 |
|  | 8.8.2. Manufacturer Login Analysis | 47 |
|  | 8.8.3. Distributor Login Analysis | 48 |
|  | 8.8.4. Bank Transaction Analysis | 49 |
| **9.** | **CONCLUSION** | 51 |
|  | 9.1 Conclusion and Future Enhancements | 51 |
|  | **APPENDICES** | 52 |
|  | A.1 Sample Screens | 52 |
|  | A.2 Publications | 57 |
|  | **REFERENCES** | 58 |

**LIST OF TABLES:**

|  |  |  |
| --- | --- | --- |
| **TABLE NO.** | **TITLE** | **PAGE NO.** |
| 8.8.1 | SUPPLIER LOGIN ANALYSIS | 62 |
| 8.8.2 | MANUFACTURER LOGIN ANALYSIS | 63 |
| 8.8.3 | DISTRIBUTER LOGIN ANALYSIS | 64 |
| 8.8.4 | BANK TRANSACTION ANALYSIS | 65 |

**LIST OF FIGURES:**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **TITLE** | **PAGE NO.** |
| 5.1 | Use case Diagram | 12 |
| 5.2 | Sequence Diagram | 13 |
| 5.3 | Activity Diagram: | 14 |
| 5.4 | Collaboration Diagram: | 15 |
| 5.5 | Class Diagram | 15 |
| 5.6 | 5.6.1. Data Flow Diagram Level 0 | 16 |
|  | 5.6.2. Data Flow Diagram Level 1 | 17 |
|  | 5.6.3. Data Flow Diagram Level 2 | 17 |
|  | 5.6.4. Data Flow Diagram Level 3 | 18 |
| 6.1 | System Architecture | 19 |
| 6.2 | Working of Java | 23 |
| 6.3 | The Java Platform | 24 |
| 6.4 | SHA256 Hash | 28 |
| 6.5 | Block | 28 |
| 6.6 | Blockchain | 29 |

|  |  |  |
| --- | --- | --- |
| 9.1 | Initializing Blockchain | 67 |
| 9.2 | Supplier Page | 67 |
| 9.3 | Supplier Product Page | 68 |
| 9.4 | Manufacturer Registration and Login Page | 68 |
| 9.5 | Blockchain | 69 |
| 9.6 | Getting Blockchain Data Page | 69 |
| 9.7 | Customer Page | 70 |
| 9.8 | QR Page | 70 |
| 9.9 | Bank Payment Page | 71 |
| 9.10 | Order Status Page | 71 |

# LIST OF ABBREVIATIONS

|  |  |
| --- | --- |
| **JDK** | Java Development Toolkit |
| **DEX** | Dalvik Executables |
| **TCP** | Transmission Control Protocol |
| **IP** | Internet Protocol |
| **HTTP** | Hyper Text Transfer Protocol |
| **ADT** | Android Development Tool |

## INTRODUCTION

* 1. **Overview**

In a supply chain management system, the smart tags (like QR, NFC) are used in order to store the product details. By that the consumers can also track the products during entire life cycle. Product consumers have no knowledge about the data’s that is provided in the smart tags by the smart tag creators and stack holders are true or not. The DL-Tags solution steps into this environment to offer a decentralized, privacy-preserving, and verifiable management of Smart Tags during a product’s lifecycle. The solution is based on distributed ledger technology (DLT) and uses the Block chain to mediate interactions between the stakeholders during a product’s exchange process. The paper describes the DL-Tags solution and includes a cost analysis of all implemented transactions on the Blockchain. The proposed solution provides evidence of the product’s origin and its journey across the supply chain while preventing tag duplication and manipulation. It is among the first documented practical solutions using DLT and IoT for supply chain management, which is designed to be distributed ledger agnostic.

## Problem Definition

As TIS infrastructure is a centralized setup, Smart Tag duplication which might arise during a product lifecycle is a situation when a fraudulent retailer duplicates the Smart Tags. Smart Tags are typically provided in the form of dynamic QR codes printed with special ink. Dynamic QR codes change due to specific environmental conditions (e.g., temperature, humidity, pressure, rain, sun etc.) Blockchain is one of the developing technologies and has been attracting a lot of attention among the industry. The blockchain is a technology which can perform tremendous things when combined with Internet of Things. Blockchain provides the key features that has been lacking in the previous developments such as, send large amount of encrypted data over the internet and provides decentralized access to those data with no need for a third party. In this paper, we use this technology and provide

a solution for the supply chain management. The Distributed Ledger (DL) Tag and Ethereum blockchain is used to provide customers a trustable and verifiable platform. The customers will be provided with smart tags such as QR, NFC for tracking and tracing the product across the blockchain. The transactions across the blockchain will be stored in multiple blockchain. Then the customers and the stakeholders can verify the products origin and authenticity. This method is secure, cost efficient and decentralized, so it will completely prevent product duplication and manipulation across the supply chain.

## LITERATURE SUTRVEY

The title is a privacy-preserving Internet of Things device management scheme based on blockchain published by the authors Qingsu He, Yu Xu, Zhoubin Liu Jinhong He,You Sun, and Rui Zhang in the year 2018. The policies are posted on the block chain, so it is visible to all the users, and it can avoid the third-party ultra vires. It can easily realize the transfer of access rights without the need for the involvement of resource owners and make the authority management more flexible. No block chain system can support time bound and attribute-based access with high efficiency. Very difficult to directly establish trust between two strange entities without a third-party center.

The title is Applications of Blockchains in the Internet of Things: Comprehensive Survey published by Ali Dorri, Salil S. Kanhere, and Raja Jordan in the year 2017. Achieve decentralization, security and auditablility. The trustless network environment of block chains allow secure micro- transactions for IOT services and data. High server maintenance costs. Centralized server has the risk of being deleted or tampered with.

Blockchain Technology Beyond Bitcoin: An Overview. Provides anonymity, data integrity without intermediary organization. Supports digital voting, smart contracts, decentralized exchanges, distributed cloud storage etc. Provides anonymity, data integrity without intermediary organization. Supports digital voting, smart contracts, decentralized exchanges, distributed cloud storage etc., Scaling issue is faced by bitcoin (it processes 7 transactions per seconds). All the places are not updated to use bitcoins. Regularization of bitcoin by the government.

The title is Block chain and Smart Contracts for Internet of Things: A Systematic Literature Review published by Anup Dhakal, Xiahui cuis in the year 2018. It is very advantageous to track the goods and find out whether the products are counterfeit or real. With Block chain and IOT, empty-apartments, spare bedrooms, houses, cars, or vacant conference rooms will rent themselves out. Patents will license themselves. Our e-mail will be able to charge spammers for each item received. Some challenges that cannot be solved by using technologies designed for traditional internet. Open-source system may be susceptible for exploitations and accidents.

First purpose-built protocol for supply chains based on blockchain published by the author Branimir Rakic in the year 2017. Blockchain technology has huge potential to decentralize trust in supply chains and bring measurable benefits and value to the public and private sectors. To unlock this potential, the Origin Trail protocol was designed purposefully to tackle the main challenges which limit the fluent exchange of, and integrity of data in product supply chains. With supply chain data becoming increasingly fragmented, scalability and cost concerns of current decentralized solutions become evident.

Origin Trail is a unique solution allowing IT providers in supply chains to set up blockchain supported data sharing in multi- organizational environment. It helps them build transparency beyond the “one step down, one step up” traceability principle. Furthermore, it improves the integrity of product data and drives efficiencies for stakeholders. The first version of the OriginTrail solution is proven and currently deployed in the food industry. The upcoming open-source version will be suitable to any product supply chain such as automotive, consumer goods, pharmaceutical etc.

Ethereum: a secure decentralised generalised transaction ledger eip-150 revision (1e18248 - 2017-04-12) published by the author Dr. Gavin Wood in the year 2017. It is possible to use the internet to make a decentralized value-transfer system, shared across the world and virtually free to use. Ethereum is a specialized version of a cryptographically secure, transaction- based state machine. Ethereum is a specialized version of a cryptographically secure, transaction- based state machine. It has too many platforms. It is the biggest minus of this crypto money. There can be some breakdowns in the system. In order to upgrade the project to a new level, this block chain will switch from being Proof of Work to Proof of Stake.

Public versus Private Blockchain s was published by BitFury Group in collaboration with Jeff Garzik in the year 2015. Private block chains can process much higher transactions per second (TPS) as compared to public blockchains. Private blockchains have a different consensus mechanism compared to public blockchains and a more suited to a consortium of organizations, such as the banking industry. Block chains use excessive energy. Mining does not provide network security.

## SYSTEM ANALYSIS:

* 1. **EXISTING SYSTEM:**

The (TIS) system is an IoT-based solution for supply chain management which issues product’s Smart Tags and supports sharing of product- related information between stakeholders involved in a product lifecycle using such tags. Smart Tags are typically provided in the form of dynamic QR codes printed with special ink. Dynamic QR codes change due to specific environmental conditions (e.g., temperature, humidity, light intensity), and are adequate for tracking of fast- moving consumer goods (FMCG) and their surrounding conditions. However, by using the TIS infrastructure in a centralized setup, a product consumer has to implicitly trust the TIS platform as the Smart Tag creator, as well as other stakeholders in the supply chain, that they are providing authentic data about a product without tempering with its Smart Tag.

## PROPOSED SYSTEM:

Our proposed system uses the DL-Tags (distributed ledger technology) to offer a decentralized, privacy-preserving, and verifiable management of Smart Tags during a product’s lifecycle. The solution relies on IoT technologies for data collection, and stores product related data in the Blockchain, which is a private decentralized data storage layer. DLT enables the maintenance of a global, append only, data structure by a set of mutually untrusted participants in a distributed environment. i.e., if a consumer wants to buy a product which is duplicated by a fraudulent retailer by changing the data in the smart tags. In case if a distributed ledger technology is used change is detected and products with duplicated smart tags will not be sold regularly. Here whenever a transaction is made between different participants, the legal agreement will be signed in the smart contracts. And the transaction details will be stored inside the Decentralized Blockchain Network.

## PURPOSE

The main aim of this project is to provide evidences of products origin and its journey across the supply chain by preventing tag duplication and manipulation.

## REQUIREMENT ANALYSIS AND SPECIFICATION

Supply chain management relates to complex business to- business and business-to- customer networks, and is traditionally used to operate and maintain producer's relationships with suppliers, logistics, and customers with a goal to deliver superior customer value at reduced cost. Internet of Things (IoT), as a concept which enables the usage of high volumes of smart devices and actuators connected to the Internet, offers innovative solutions for tracking the products and materials relevant to supply

chains. In particular, it addresses the main requirement of supply chain management seamless sharing of product-related information between stakeholders involved in a product lifecycle. IoT-based solutions enhanced by Distributed Ledger Technology (DLT) go even a step further to facilitate a sustainable, the associate editor coordinating the review of this manuscript and approving it for publication was Giacomo Verticale. decentralized and privacy-preserving information sharing model without a trusted intermediary. Supply chain management has indeed been identified as one of the main applications combining blockchain technology with IoT, since involved stakeholders share/exchange product-related data within a trustless environment, but have the possibility to verify product authenticity and agree on its current state. The TagItSmart (TIS) system is an IoT-based solution for supply chain management which issues product's Smart Tags and supports sharing of product- related information between stakeholders involved in a product lifecycle using such tags. Smart Tags are typically provided in the form of dynamic QR codes printed with special ink. Dynamic QR codes change due to specific environmental conditions (e.g., temperature, humidity, light intensity), and are adequate for tracking of fast-moving consumer goods (FMCG) and their surrounding conditions. A product enhanced with its Smart Tag becomes a *digital product* enabling innovative services across the entire supply chain, from manufacturer, logistics, and retail, to consumer and recycling. However, by using the TIS infrastructure in a centralized setup, a product consumer has to implicitly trust the TIS platform as the Smart Tag creator, as well as other stakeholders in the supply chain, that they are providing authentic data about a product without tempering with its Smart Tag. DL- Tags steps into this environment as a solution for verifying shared product information in a privacy-preserving and decentralized way, under the assumption that a single centralized authority should be avoided to orchestrate the relevant processes. Since data authenticity and integrity are vital to ensure sustainable management of supply chains, DL-Tags uses DLT, in particular the *Ethereum blockchain*, as a decentralized intermediary adequate for environments involving many stakeholders which do not trust each other. By using the DL-Tags solution, stakeholders are required to reach consensus about the information stored in Smart Tags that is logged on the blockchain. This prevents the possibility to blame one of the former stakeholders in the product lifecycle for unacceptable facts being stored in a Smart Tag. Furthermore, the proposed solution enables *brand protection*: End

consumers can verify whether a product belongs to a certain brand as advertised, and to verify that the changes of product ownership along the supply chain is acknowledged by all stakeholders. For example, the proposed solution can be applied in online shopping use cases where e-commerce stores serve as retailers of physical products. One of the reasons why consumers are reluctant to shop online is because they are concerned whether declared information about a product is indeed genuine or not. The main benefit of DL-Tags for end consumers is in the following: Consumers can simply verify product provenance and its overall journey through the supply chain before completing product purchase by using a DL-Tags enabled mobile application to scan a Smart Tag. The paper contributes an original and practical solution for verifiable supply chain management based on IoT and DLT which prevents counterfeit goods to be sold as originals, while the involved stakeholders have no need to expose their identity and business-related data to third parties. Stakeholders share product-related data solely directly, as product exchanges occur on the supply chain, and control the data without the need for a trusted third party. A public ledger serves here as an intermediary providing *proof of existence* for significant events occurring on the supply chain, meaning that only data hashes based on actual product-related data exchanges are stored on the ledger. By using proofs of existence, the integrity of the actual data can easily be verified. The implemented solution is designed to be *agnostic of a DLT platform* used in the actual implementation, while the shared data format can be adapted to a specific use case. The solution is tested in the *TagItSmart* use case, where wine bottles labeled by TIS Smart Tags are physically transferred between stakeholders, while the Ethereum blockchain serves as a trustless intermediary between them. Finally, we report a cost analysis of the implemented solution on the Ethereum blockchain that would need to be paid by the stakeholders. The largest expense relates to the product creation function which is paid by a producer, while all functions require much less gas. In December 2018, a producer would pay around 5 USD for logging a product creation event on the Ethereum chain, which is acceptable for high end products, while other functions cost in the range from 0:001 to 0:3 USD, which should be acceptable for other involved stakeholders. Note that all readings from the blockchain do not incur additional cost, and thus product verification by consumers is free of charge. The paper is organized in the following way: Section II provides an overview of DLT, while relevant solutions combining IoT and DLT for supply chain management are

analyzed in Section III. The benefits and requirements of the DL-Tags solution are introduced in Section IV. Technical details regarding the DL-Tags architecture and design are given in Section V. Our specific use case depicting a concrete application of DL-Tags is presented in Section VI, while the cost incurred by the solution is analyzed in Section VII. Section VIII concludes the paper and identifies directions for further work.

## PERFORMANCE REQUIREMENTS

The application at this side controls and communicates with the following three main general components.

* + - * embedded browser in charge of the navigation and accessing to the web service;
      * Server Tier: The server side contains the main parts of the functionality of the proposed architecture. The components at this tier are the following.

Web Server, Security Module, Server-Side Capturing Engine, Preprocessing Engine, Database System, Verification Engine, Output Module.

## SAFETY REQUIREMENTS

1. The software may be safety-critical. If so, there are issues associated with its integrity level
2. The software may not be safety-critical although it forms part of a safety- critical system. For example, software may simply log transactions.
3. If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.
4. There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable.
5. If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level.
6. Systems with different requirements for safety levels must be separated.
7. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.

## HARDWARE REQUIREMENTS

* Hard Disk : 80GB and Above
* RAM : 4GB and Above
* Processor : P IV and Above

## SOFTWARE REQUIREMENTS

* Windows 7 and above
* JDK 1.7
* JDK 1.8
* J2EE
* Tomcat 7.0
* MySQL

## TECHNOLOGY STACK

* J2EE (JSP, Servlet), JavaScript, HTML, CSS, AJAX.
* Hibernate Framework
* MVC Pattern
* Design Pattern
* Spring

## Design and Implementation Constraints

* 1. **Constraints in Analysis**
* Constraints as Informal Text
* Constraints as Operational Restrictions
* Constraints Integrated in Existing Model Concepts
* Constraints as a Separate Concept
* Constraints Implied by the Model Structure

## Constraints in Design

* Determination of the Involved Classes
* Determination of the Involved Objects
* Determination of the Involved Actions
* Determination of the Require Clauses
* Global actions and Constraint Realization

## Constraints in Implementation

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore, it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object-oriented methods.

## SYSTEM DESIGN:

* 1. **UML DIAGRAM:**

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object- oriented software intensive system under development

## USECASE DIAGRAM:

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

* Use case: A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.
* Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

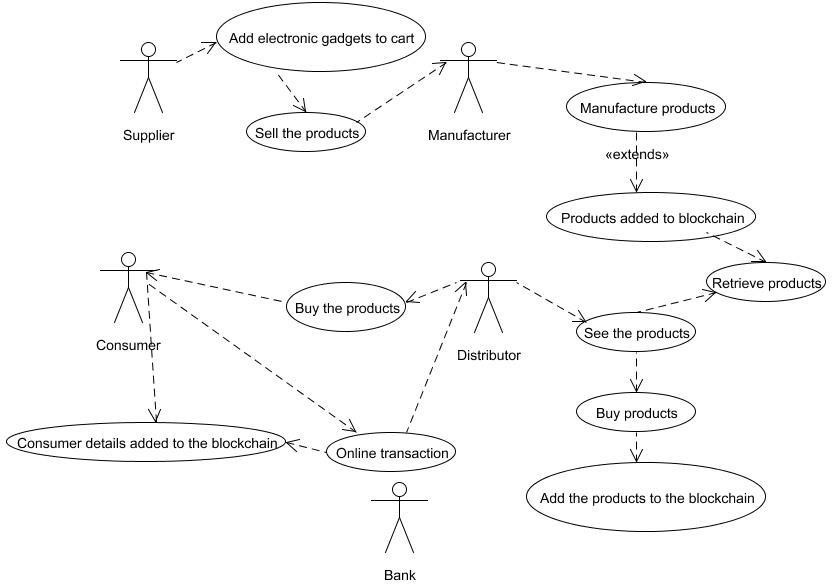


Fig.5.1.Use Case Diagram

## SEQUENCE DIAGRAM:

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

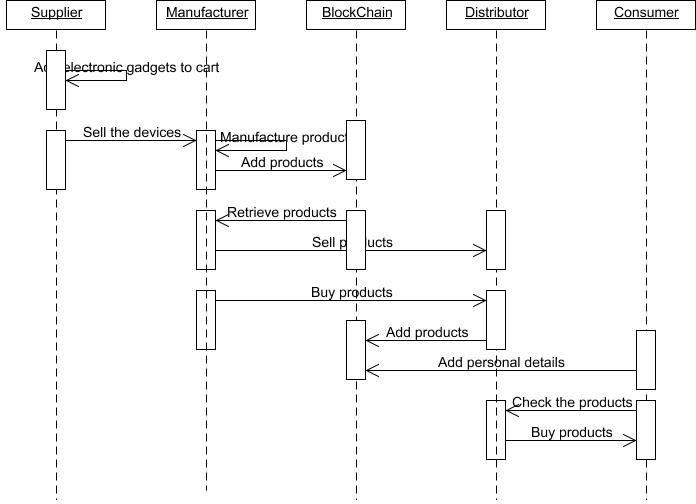


Fig.5.2.Sequence Diagram

* + 1. **ACTIVITY DIAGRAM:**

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

* + - * Rounded rectangles represent activities.
      * Diamonds represent decisions.
      * Bars represent the start or end of concurrent activities.
      * A black circle represents the start of the workflow.
      * An encircled circle represents the end of the workflow.

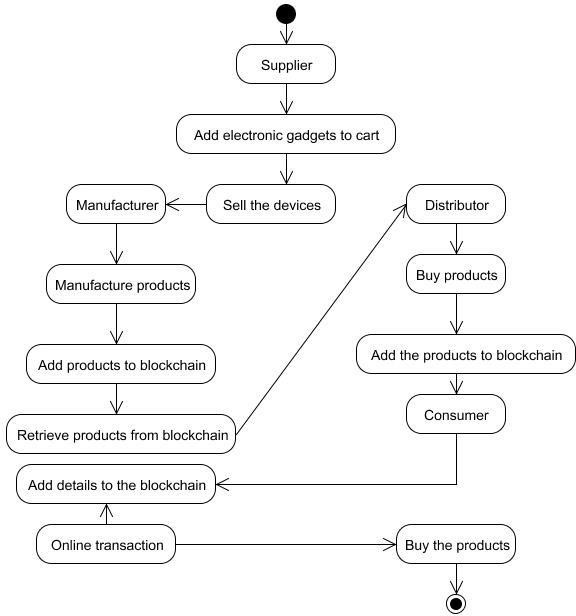


Fig.5.3.Activity Diagram

## COLLABORATION DIAGRAM:

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects.

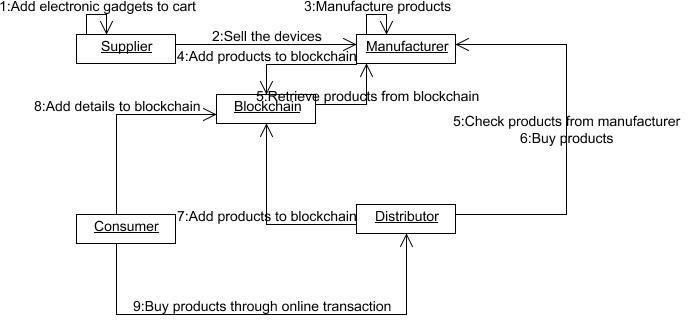


Fig.5.4. Collaboration Diagram

## CLASS DIAGRAM

A Class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.

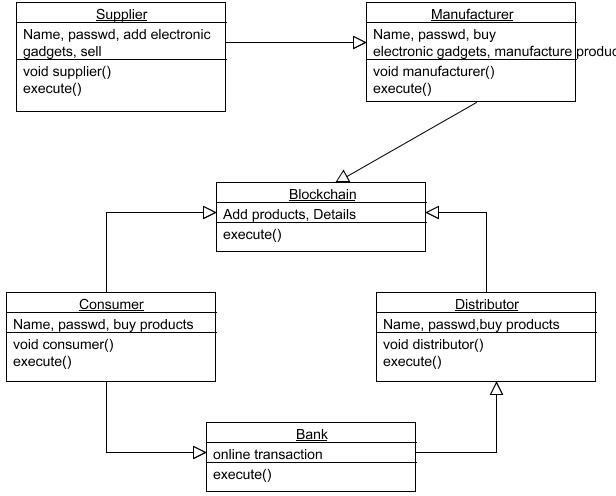


Fig.5.5.Class Diagram

## DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its aspects. It is a preliminary step used to create an overview of the system which can later be elaborated DFDs can also be used for visualization of data processing.

## LEVEL 0:

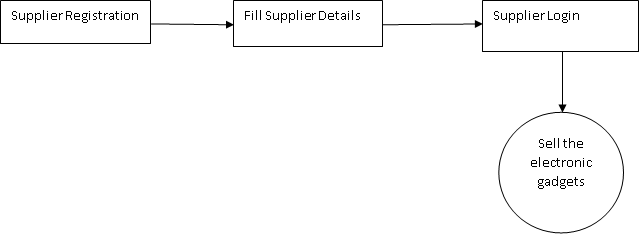


Fig.5.6.1. DFD Level 0

## LEVEL 1:

Fig.5.6.2. DFD Level 1

## LEVEL 2:

Fig.5.6.3.DFD Level 2

## LEVEL 3:

Fig.5.6.4. DFD Level 3

## SYSTEM ARCHITECTURE:

* 1. **ARCHITECTURE OVERVIEW**

Supplier first register in the application and login. Then they sell the products to the manufacturers. An account is created by the manufacturer. The raw materials are analyzed by the manufacturer. Then the manufacturer will request quantity of raw materials to supplier.

Then the request was accepted by the suppliers and raw material will be added to the manufacturer inventory. The product ID will be sent to block chain by the manufacturer and product will be added to manufacturer shipment. The product will be retrieved by the manufacturer from block chain. The distributer and the login details are present in the registration part.

The distributer will see the product in the cart. If distributer buys the product, then that will be added to blockchain. The KYC form is maintained by the distributers. Consumers are of two types.

Without knowing the product details, one will order. Therefore, it is difficult for them to identify the product is duplicate or original. The second type of customer is the one who views full details of the product what they are willing to buy, so these types of consumers view the overall blockchain content.

Fig.6.1. System Architecture

Seller adds the Electronic Gadgets

All information are stored into blockchain

Buying Electronics devices



Collect products

Assemble products



Transport

details

BlockChain

Banking



Storage



Market

All Information are Stored into Blockchain

Scan and View product details

RFID or qr code scanner

User



## MODULE DESIGN SPECIFICATION:

* Creating Suppliers
* Manufacturer Process
* Distributors Transactions
* Product Verification

## MODULE EXPLANATION:

* + - 1. **CREATING SUPPLIERS**

First registration. The registration form contains supplier details. Then login. Supplier sells the products to all manufactures what the produce.

## MANUFACTURER PROCESS

The manufacturer initially creates the account. They will analyze the raw materials and the manufacturer will request the quantity of raw materials to the supplier. Then suppliers will accept the request from manufacturer and raw material will be added to the manufacturer inventory. The manufacture will send the product ID to the block chain and then the created product will be added to manufacturer shipment. From the block chain the manufacturer will retrieve the product.

## DISTRIBUTORS TRANSACTIONS

The registration part contains distributer details. And login. The distributer will be seeing the product in the manufacturer cart and then buying product by the distributer will be added to the block chain. The distributers maintain the KYC form for adding duplicate products, it cannot be stored in blockchain.

## PRODUCT VERIFICATION

There are two types of consumers. One is ordering the product without knowing the product details. So, they cannot identify the product is duplicate or original. The

second type of customer i8s view the full details of the product what they are buying so they view the blockchain content.

## PROGRAM DESIGN LANGUAGE

* + 1. **JAVA**

Java is an object-oriented programming language developed initially by James Gosling and colleagues at Sun Microsystems. The language, initially called Oak (named after the oak trees outside Gosling's office), was intended to replace C++, although the feature set better resembles that of Objective C.

## INTRODUCTION TO JAVA

Java has been around since 1991, developed by a small team of Sun Microsystems developers in a project originally called the Green project. The intent of the project was to develop a platform-independent software technology that would be used in the consumer electronics industry. The language that the team created was originally called Oak.

The first implementation of Oak was in a PDA-type device called Star Seven (\*7) that consisted of the Oak language, an operating system called GreenOS, a user interface, and hardware. The name \*7 was derived from the telephone sequence that was used in the team's office and that was dialed in order to answer any ringing telephone from any other phone in the office.

Around the time the First-Person project was floundering in consumer electronics, a new craze was gaining momentum in America; the craze was called "Websurfing." The World Wide Web, a name applied to the Internet's millions of linked HTML documents was suddenly becoming popular for use by the masses. The reason for this was the introduction of a graphical Web browser called Mosaic, developed by ncSA. The browser simplified Web browsing by combining text and graphics into a single interface to eliminate the need for users to learn many confusing UNIX and DOS commands. Navigating around the Web was much easier using Mosaic.

It has only been since 1994 that Oak technology has been applied to the Web. In 1994, two Sun developers created the first version of Hot Java, and then called Web

Runner, which is a graphical browser for the Web that exists today. The browser was coded entirely in the Oak language, by this time called Java. Soon after, the Java compiler was rewritten in the Java language from its original C code, thus proving that Java could be used effectively as an application language. Sun introduced Java in May 1995 at the Sun World 95 convention.

Web surfing has become an enormously popular practice among millions of computer users. Until Java, however, the content of information on the Internet has been a bland series of HTML documents. Web users are hungry for applications that are interactive, that users can execute no matter what hardware or software platform they are using, and that travel across heterogeneous networks and do not spread viruses to their computers. Java can create such applications.

## WORKING OF JAVA

For those who are new to object-oriented programming, the concept of a class will be new to you. Simplistically, a class is the definition for a segment of code that can contain both data (called attributes) and functions (called methods).

When the interpreter executes a class, it looks for a particular method by the name of main, which will sound familiar to C programmers. The main method is passed as a parameter an array of strings (similar to the argv [] of C), and is declared as a static method.

To output text from the program, we execute the println method of System.out, which is java’s output stream. UNIX users will appreciate the theory behind such a stream, as it is actually standard output. For those who are instead used to the Wintel platform, it will write the string passed to it to the user’s program.

Java consists of two things:

* + - * + Programming language
        + Platform

## THE JAVA PROGRAMMING LANGUAGE:

Java is a high-level programming language that is all of the following:

* + - * + Simple
        + Object-oriented
        + Distributed
        + Interpreted
        + Robust
        + Secure
        + Architecture-neutral
        + Portable
        + High-performance
        + Multithreaded
        + Dynamic

The code and can bring about changes whenever felt necessary. Some of the standard needed to achieve the above-mentioned objectives are as follows:

Java is unusual in that each Java program is both co implied and interpreted. With a compiler, you translate a Java program into an intermediate language called Java byte codes – the platform independent codes interpreted by the Java interpreter. With an interpreter, each Java byte code instruction is parsed and run on the computer. Compilation happens just once; interpretation occurs each time the program is executed. This figure illustrates how it works:

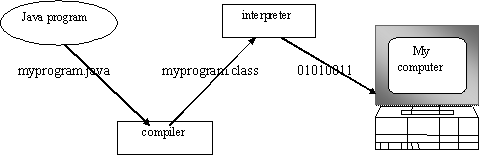


Fig.6.2. Java Interpreter

You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (JVM). Every Java interpreter, whether it’s a Java development tool or a Web browser that can run Java applets, is an implementation of JVM. That JVM can also be implemented in hardware. Java byte codes help make “write once, run anywhere” possible.

You can compile your Java program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the JVM. For example, that same Java program can be run on Windows NT, Solaris and MacIntosh

## THE JAVA PLATFORM

A platform is the hardware or software environment in which a program runs. The Java platform differs from most other platforms in that it’s a software-only platform that runs on top of other, hardware-based platforms. Most other platforms are described as a combination of hardware and operating system.

The Java platform has two components:

* + - * + The Java Virtual Machine (JVM)
        + The Java Application Programming Interface (Java API)

You’ve already been introduced to the JVM. It’s the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped into libraries (packages) of related components. The following figure depicts a Java program, such as an application or applet, that’s running on the Java platform. As the figure shows, the Java API and Virtual Machine insulates the Java program from hardware dependencies.

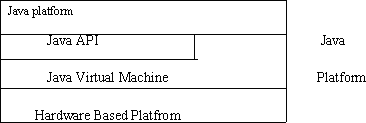


Fig.6.3. Java Platform

As a platform-independent environment, Java can be a bit slower than native code. However, smart compliers, well-tuned interpreters, and just-in-time byte compilers can bring Java’s performance close to that of native code without threatening portability.

## APACHE TOMCAT SERVER

Apache Tomcat (formerly under the Apache Jakarta Project; Tomcat is now a top level project) is a web container developed at the Apache Software Foundation. Tomcat implements the servlet and the JavaServer Pages (JSP) specifications from Sun Microsystems, providing an environment for Java code to run in cooperation with a web server. It adds tools for configuration and management but can also be configured by editing configuration files that are normally XML-formatted. Because Tomcat includes its own HTTP server internally, it is also considered a standalone web server.

## ENVIRONMENT

Tomcat is a web server that supports servlets and JSPs. Tomcat comes with the Jasper compiler that compiles JSPs into servlets.

The Tomcat servlet engine is often used in combination with an Apache web server or other web servers. Tomcat can also function as an independent web server. Earlier in its development, the perception existed that standalone Tomcat was only suitable for development environments and other environments with minimal requirements for speed and transaction handling. However, that perception no longer exists; Tomcat is increasingly used as a standalone web server in high-traffic, high- availability environments.

Since its developers wrote Tomcat in Java, it runs on any operating system that has a JVM.

## PRODUCT FEATURES

Tomcat 3.x (initial release)

* + - * + implements the Servlet 2.2 and JSP 1.1 specifications
        + servlet reloading
        + basic HTTP functionality Tomcat 4.x
        + implements the Servlet 2.3 and JSP 1.2 specifications
        + servlet container redesigned as Catalina
        + JSP engine redesigned as Jasper
        + Coyote connector
        + Java Management Extensions (JMX), JSP and Struts-based administration
        + Tomcat 5.x
        + implements the Servlet 2.4 and JSP 2.0 specifications
        + reduced garbage collection, improved performance and scalability
        + native Windows and Unix wrappers for platform integration
        + faster JSP paring

## HISTORY

Tomcat started off as a servlet specification implementation by James Duncan Davidson, a software architect at Sun. He later helped make the project open source and played a key role in its donation by Sun to the Apache Software Foundation.

Davidson had initially hoped that the project would become open-sourced and, since most open-source projects had O'Reilly books associated with them featuring an animal on the cover, he wanted to name the project after an animal. He came up with Tomcat since he reasoned the animal represented something that could take care of and fend for itself. His wish to see an animal cover eventually came true when O'Reilly published their Tomcat book with a tomcat on the cover.

## Blockchain:

* + - 1. **Introduction to Blockchain**

With the emergence of Digital Currency (aka Crypto currency), several enterprises or financial institutions are experimenting with the Distributed Ledger system as a trusted way to track the ownership of the assets without any central authority.

The core system behind the new currency system is Blockchain technology. A walkthrough of the basic building blocks of the Blockchain technology is described below.

A Blockchain is basically a chain of Blocks. Blocks are hashed using SHA-256 hashing algorithm to generate the signature of the data associated with it.

Imagine a Blockchain as a linked-list whose node contains below attributes:

* + - * 1. Block number – a sequence number (monotonically increasing) assigned to the block
        2. Nonce – a random number which is used to generate Hash (as in #5) value which starts with 4 zeroes (0000). The process of generating this Nonce is called Mining.
        3. Data – the actual user data associated with the block
        4. Prev – contains the Hash of the previous block (e.g. current block # -1). The value for the first block in the chain is 64 zeroes (0000000000000000000000000000000000000000000000000000000000000 000).
        5. Hash – current block’s Hash value (generated using SHA-256). All of the above attributes excluding Hash e.g. Block #, Nonce, data, Prev are used to calculate the Hash of this block.

[#=1, Nonce=3409, Data=x, Prev=00..0, Hash=0000ffgr5rg67j] <- [#=2, Nonce=4986, Data=x, Prev=0000ffgr5rg67j, Hash=000045tggr5rg..77yh] <-

……and the chain goes on… e.g. in above block #1, the value for Hash=0000ffgr5rg67j is generated using the values 1,3409,x,00..0. In case value for any of these 4 attributes changes, it will change the Hash value of this block. Once the Hash value of this Block changes (e.g., from 0000ffgr5rg67j to 34sdffgr5rg67j), it will break the next Block (#2) as its Prev field will point to invalid Hash (0000ffgr5rg67j doesn’t exist anymore). This leads to a ripple effect and turns whole chain as invalid/tampered.

One way to fix it is to run mining and recalculate the Hash value of Block #1 which basically will generate new value for Nonce and hence leading to a valid Hash value which starts with 4 zeroes. Copying this to next Block #2’s Prev field will fix these 2 Blocks. However, in order to fix the whole Blockchain, we need to continue with this process for all the Blocks in the chain so that all Blocks point to new & valid Hash codes of their previous blocks.

The cost of fixing the tampered Blockchain as described in above process is very high. Because we have to go and fix the Chain from the starting Block to the last one. In case the Chain is large, it becomes costly operation. In case of Distributed Blockchain where several Peers are involved in the process and keeping the copy of the Blockchain, the repairing the Blocks becomes even more costly operation.

The other and more efficient process is to come up with the compensating data and add this Block at the end of the Chain. E.g., In case your Chain contains the financial transaction (money movement) in Data field of the Block, then instead of fixing each of the Block’s Data with corrected financial transaction, come up with the adjusted financial transaction (aka compensating transaction) and create a Block (with Data=adjusted transaction record) and add this Block to the Blockchain (adds to the end of the Chain).



Fig.6.4. SHA 256 Hash

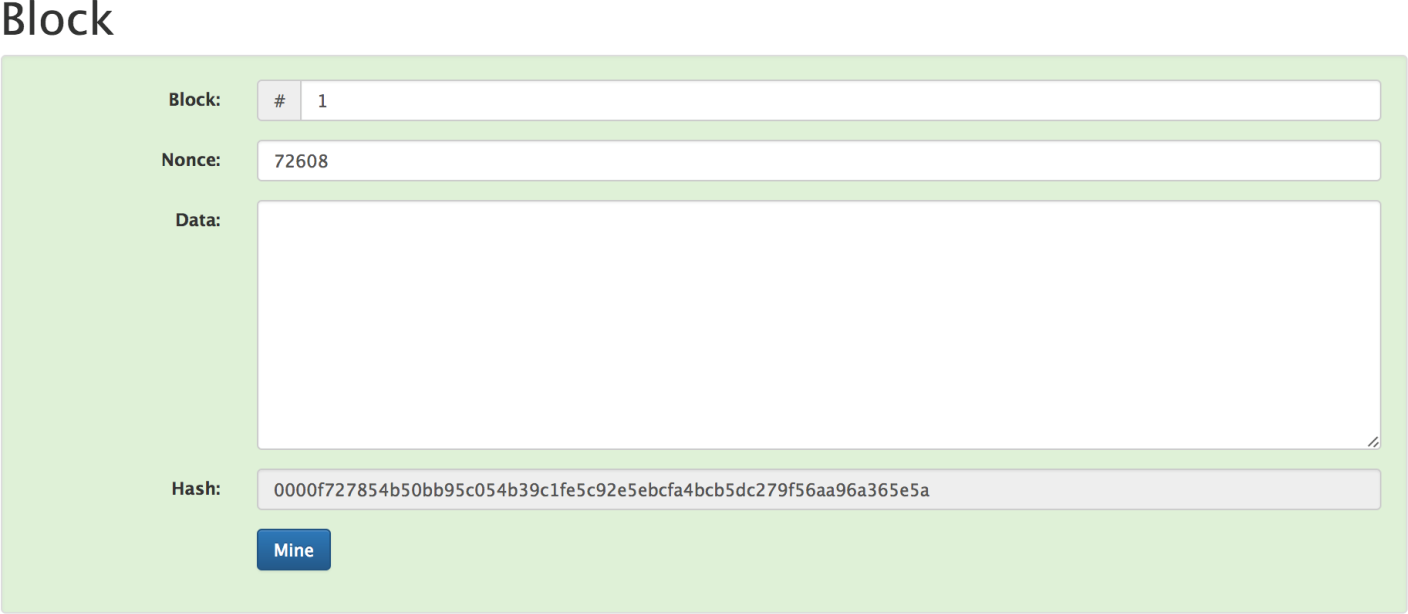


Fig.6.5.Block

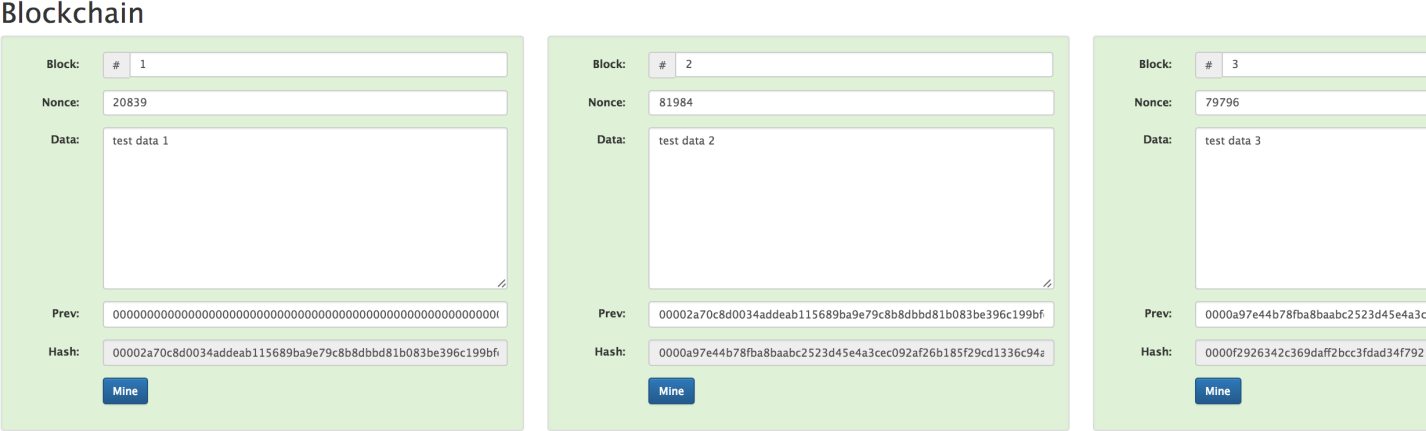


Fig.6.6. Blockchain

## SYSTEM IMPLEMENTATION:

* 1. **CLIENT SIDE CODING:**

**index.html**

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<base href="<%=basePath%>">

<title>Bank Server</title>

<meta http-equiv="pragma" content="no-cache">

<meta http-equiv="cache-control" content="no-cache">

<meta http-equiv="expires" content="0">

<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">

<meta http-equiv="description" content="This is my page">

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

<link href="style.css" rel="stylesheet" type="text/css" />

<link rel="stylesheet" href="jquery/jqueryui.css">

<script src="jquery/jquery10.js"></script>

<script src="jquery/jqueryuijq.js"></script>

<script>

$(document).ready(function()

{

$("#errorreportlab").hide();

});

</script>

<script>

function adminformfunc()

{

var username=document.adminlogform.adminusername.value; var pass=document.adminlogform.adminpassword.value; if(username=="" || pass=="")

{

$("#errorreportlab").fadeIn(700); document.getElementById('errorreportlab').innerHTML="Fields should not be empty."; return false;

}

}

</script>

</head>

<body>

<div id="topPan">

<center><img src="images/banklogo3.png" alt="Education Zone" border="0" width="350" height="130" style="margin-top:10px;" title="Education Zone"/></center>

<div id="topContactPan">

</div>

<div id="topMenuPan">

<div id="topMenuLeftPan"></div>

<div id="topMenuMiddlePan">

<ul>

<li class="home">Home</li>

<li><a href="#"></a></li>

<li><a href="#">NewUser</a></li>

<li><a href="#"></a></li>

<li><a href="#">History</a></li>

<li><a href="#"></a></li>

<li><a href="#"></a></li>

<li class="contact"><a href="#" class="contact">Contact</a></li>

</ul>

</div>

<div id="topMenuRightPan"></div>

</div>

</div>

<div id="bodyPan">

<div id="bodyLeftPan">

<div style="border:1px solid #CFCFCF;padding:15px;">

<center><label style="font-weight:bold;color:black;font:28px/28px Georgia,Times New Roman,Times,serif;">Admin Login</label></center>

</div>

<br/>

<center><label id="errorreportlab" style="font-weight:bold;color:red;text-align:center;

letter- spacing:2px;">

</label></center>

<br/>

<div style="border:1px solid #CFCFCF;padding:15px;">

<center>

<s:form action="banklogin" name="adminlogform" onsubmit="return adminformfunc()">

<s:textfield name="adminusername" label="Username"></s:textfield>

<s:password name="adminpassword" label="Password"></s:password>

<s:submit></s:submit>

</s:form>

<s:actionmessage/>

</center>

</div>

</div>

</div>

<div>

<div id="footerPan" style="padding:10px;">

<center><p class="copyright">©Google zone. All right reserved.</p></center>

</div>

</div>

</body>

</html>

## SERVER SIDE CODING:

**BankLogin.java**

package com.banklogic;

import

java.util.ArrayList; import java.util.Collections; import java.util.Random;

import com.opensymphony.xwork2.ActionSupport;

public class BankLogin extends ActionSupport

{

String adminusername="",adminpassword=""; Random rand=new Random();

ArrayList<String> bankaccntholdnames=new ArrayList<String>();

String usernametext,phonetext,emailtext,addresstext,citytext,pincodetext,statetext;

String firstselectaccountnotext, firstbankholdernames, firstamounttext, firstphonetext,firstmai ltext,firstaddresstext,firstcitytext,firstpincodetext,firststatetext;

public String getFirstselectaccountnotext() { return firstselectaccountnotext;

}

public void setFirstselectaccountnotext(String firstselectaccountnotext) { this.firstselectaccountnotext = firstselectaccountnotext;

}

public String getFirstbankholdernames() { return firstbankholdernames;

}

public void setFirstbankholdernames(String firstbankholdernames) { this.firstbankholdernames = firstbankholdernames;

}

public String getFirstamounttext() { return firstamounttext;

}

public void setFirstamounttext(String firstamounttext) { this.firstamounttext = firstamounttext;

}

public String getFirstphonetext() { return firstphonetext;

}

public void setFirstphonetext(String firstphonetext) { this.firstphonetext = firstphonetext;

}

public String getFirstmailtext() { return firstmailtext;

}

public void setFirstmailtext(String firstmailtext) { this.firstmailtext = firstmailtext;

}

public String getFirstaddresstext() { return firstaddresstext;

}

public void setFirstaddresstext(String firstaddresstext) { this.firstaddresstext = firstaddresstext;

}

public String getFirstcitytext() { return firstcitytext;

}

public void setFirstcitytext(String firstcitytext) { this.firstcitytext = firstcitytext;

}

public String getFirstpincodetext() {

return firstpincodetext;

}

public void setFirstpincodetext(String firstpincodetext) { this.firstpincodetext = firstpincodetext;

}

public String getFirststatetext() { return firststatetext;

}

public void setFirststatetext(String firststatetext) { this.firststatetext = firststatetext;

}

public String getEmailtext() { return emailtext;

}

public void setEmailtext(String emailtext) { this.emailtext = emailtext;

}

public String getAdminusername() { return adminusername;

}

public void setAdminusername(String adminusername) { this.adminusername = adminusername;

}

public String getAdminpassword() { return adminpassword;

}

public void setAdminpassword(String adminpassword) { this.adminpassword = adminpassword;

}

public String getUsernametext() { return usernametext;

}

public void setUsernametext(String usernametext) { this.usernametext = usernametext;

}

public String getPhonetext() { return phonetext;

}

public void setPhonetext(String phonetext) { this.phonetext = phonetext;

}

public String getAddresstext() { return addresstext;

}

public void setAddresstext(String addresstext) { this.addresstext = addresstext;

}

public String getCitytext() { return citytext;

}

public void setCitytext(String citytext) { this.citytext = citytext;

}

public String getPincodetext() { return pincodetext;

}

public void setPincodetext(String pincodetext) { this.pincodetext = pincodetext;

}

public String getStatetext() { return statetext;

}

public void setStatetext(String statetext) { this.statetext = statetext;

}

public ArrayList<String> getBankaccntholdnames() { return bankaccntholdnames;

}

public void setBankaccntholdnames(ArrayList<String> bankaccntholdnames)

{

this.bankaccntholdnames = bankaccntholdnames;

}

public String execute()

{

String status="error"; try

{

if(adminusername.equals("admin")&&adminpassword.equals("admin"))

{

XlApi xlap=new XlApi("webapps/BankAdmin/Pincode.csv",this);

commonMetDb(); status="success";

}

els

{

}

}

addActionMessage("Login Failed!");

## SYSTEM TESTING:

Testing is performed to identify errors. It is used for quality assurance. Testing is an integral part of the entire development and maintenance process. The goal of the testing during phase is to verify that the specification has been accurately and completely incorporated into the design, as well as to ensure the correctness of the design itself. For example, the design must not have any logic faults in the design is detected before coding commences, otherwise the cost of fixing the faults will be considerably higher as reflected. Detection of design faults can be achieved by means of inspection as well as walkthrough.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

* Static analysis is used to investigate the structural properties of the Source code.
* Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.

## UNIT TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of thesoftware design (i.e.), the module. The white-box testing techniques were heavily employed for unit testing.

## INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing

of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

The major error that was faced during the project is linking error. When all the modules are combined the link is not set properly with all support files. Then we checked out for interconnection and the links. Errors are localized to the new module and its intercommunications. The product development can be staged, and modules integrated in as they complete unit testing. Testing is completed when the last module is integrated and tested.

## FUNCTIONAL TESTS

Functional test cases involved exercising the code with nominal input values for which the expected results are known, as well as boundary values and special values, such as logically related inputs, files of identical elements, and empty files.

Three types of tests in Functional test:

* + - Performance Test
    - Stress Test
    - Structure Test

## PERFORMANCE TEST

It determines the amount of execution time spent in various parts of the unit, program throughput, and response time and device utilization by the program unit.

## STRESS TEST

Stress Test is that test designed to intentionally break the unit. A Great deal can be learned about the strength and limitations of a program by examining the manner in which a programmer in which a program unit breaks.

## STRUCTURED TEST

Structure Tests are concerned with exercising the internal logic of a program and traversing particular execution paths. The way in which White-Box test strategy

was employed to ensure that the test cases could Guarantee that all independent paths within a module have been have been exercised at least once.

* + - Exercise all logical decisions on their true or false sides.
    - Execute all loops at their boundaries and within their operational bounds.
    - Exercise internal data structures to assure their validity.
    - Checking attributes for their correctness.
    - Handling end of file condition, I/O errors, buffer problems and textual errors in output information

## TESTING TECHNIQUES / TESTING STRATERGIES

* + 1. **TESTING**

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an as-yet –undiscovered error. A successful test is one that uncovers an as-yet- undiscovered error. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently as expected before live operation commences. It verifies that the whole set of programs hang together. System testing requires a test consists of several key activities and steps for run program, string, system and is important in adopting a successful new system. This is the last chance to detect and correct errors before the system is installed for user acceptance testing.

The software testing process commences once the program is created and the documentation and related data structures are designed. Software testing is essential for correcting errors. Otherwise, the program or the project is not said to be complete. Software testing is the critical element of software quality assurance and represents the ultimate the review of specification design and coding. Testing is the process of executing the program with the intent of finding the error. A good test case design is one that as a probability of finding a yet undiscovered error. A successful test is one that uncovers a yet undiscovered error. Any engineering product can be tested in one of the two ways:

## WHITE BOX TESTING

This testing is also called as Glass box testing. In this testing, by knowing the specific functions that a product has been design to perform test can be conducted that demonstrate each function is fully operational at the same time searching for errors in each function. It is a test case design method that uses the control structure of the procedural design to derive test cases. Basis path testing is a white box testing.

Basis path testing:

* + - * + Flow graph notation
        + Cyclometric complexity
        + Deriving test cases
        + Graph matrices Control

## BLACK BOX TESTING

In this testing by knowing the internal operation of a product, test can be conducted to ensure that “all gears mesh”, that is the internal operation performs according to specification and all internal components have been adequately exercised. It fundamentally focuses on the functional requirements of the software.

The steps involved in black box test case design are:

* + - * + Graph based testing methods
        + Equivalence partitioning
        + Boundary value analysis
        + Comparison testing

## SOFTWARE TESTING STRATEGIES:

A software testing strategy provides a road map for the software developer. Testing is a set activity that can be planned in advance and conducted systematically. For this reason, a template for software testing a set of steps into which we can place specific test case design methods should be strategy should have the following characteristics:

* Testing begins at the module level and works “outward” toward the integration of the entire computer-based system.
* Different testing techniques are appropriate at different points in time.
* The developer of the software and an independent test group conducts testing.
* Testing and Debugging are different activities but debugging must be accommodated in any testing strategy.

## INTEGRATION TESTING:

Integration testing is a systematic technique for constructing the program structure while at the same time conducting tests to uncover errors associated with. Individual modules, which are highly prone to interface errors, should not be assumed to work instantly when we put them together. The problem of course, is “putting them together”- interfacing. There may be the chances of data lost across on another’s sub functions, when combined may not produce the desired major function; individually acceptable impression may be magnified to unacceptable levels; global data structures can present problems.

## PROGRAM TESTING:

The logical and syntax errors have been pointed out by program testing. A syntax error is an error in a program statement that in violates one or more rules of the language in which it is written. An improperly defined field dimension or omitted keywords are common syntax error. These errors are shown through error messages generated by the computer. A logic error on the other hand deals with the incorrect data fields, out-off-range items and invalid combinations. Since the compiler s will not deduct logical error, the programmer must examine the output. Condition testing exercises the logical conditions contained in a module. The possible types of elements in a condition include a Boolean operator, Boolean variable, a pair of Boolean parentheses A relational operator or on arithmetic expression. Condition testing method focuses on testing each condition in the program the purpose of condition test is to deduct not only errors in the condition of a program but also other an error in the program.

## SECURITY TESTING:

Security testing attempts to verify the protection mechanisms built in to a system well, in fact, protect it from improper penetration. The system security must be tested for invulnerability from frontal attack must also be tested for invulnerability from rear attack. During security, the tester places the role of individual who desires to penetrate system.

## VALIDATION TESTING

At the culmination of integration testing, software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of software test-validation testing begins. Validation testing can be defined in many ways, but a simple definition is that validation succeeds when the software functions in manner that is reasonably expected by the customer. Software validation is achieved through a series of black box tests that demonstrate conformity with requirement. After validation test has been conducted, one of two conditionsexists.

* The function or performance characteristics confirm to specifications and are accepted.
* A validation from specification is uncovered and a deficiency created.

Deviation or errors discovered at this step in this project is corrected prior to completion of the project with the help of the user by negotiating to establish a method for resolving deficiencies. Thus, the proposed system under consideration has been tested by using validation testing and found to be working satisfactorily. Though there were deficiencies in the system they were not catastrophic.

## USER ACCEPTANCE TESTING

User acceptance of the system is key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system and user at the time of developing and making changes whenever required. This is done in regarding to the following points.

* Input screen design.
* Output screen design.

## TEST CASES & REPORTS / PERFORMANCE ANALYSIS

* + 1. **SUPPLIER LOGIN ANALYSIS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED OUTPUT** | **ACTUAL OUTPUT** | **TEST RESULT** |
| 1. | Enter the Name, Product Type, Email, Mobile Number, Address and Password for registration of supplier | Name: XXX Product Type: RAM  Email: [abc@gmail.com](mailto:abc@gmail.com) Mobile Number: 9876543210  Address: Gandhi Nagar, Chennai Password: \*\*\* | XXX RAM  [abc@gmail.com](mailto:abc@gmail.com) 9876543210  Gandhi Nagar, Chennai  \*\*\* | XXX RAM  [abc@gmail.com](mailto:abc@gmail.com) 9876543210  Gandhi Nagar, Chennai  \*\*\* | Passed |
| 2. | Compare username and password with registered field while login | Email: [abc@gmail.com](mailto:abc@gmail.com) Password: \*\*\* | It Redirects to **SupplierAddProduct** web page | It Redirects to **SupplierAddProduct** web page | Passed |
| 3. | Compare username and password with registered field while login | Email: [abv@gmail.com](mailto:abc@gmail.com) Password: \*\*\* | Invalid Email Or Password | Invalid Email Or Password | Passed |

* + 1. **MANUFACTURER LOGIN ANALYSIS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED OUTPUT** | **ACTUAL OUTPUT** | **TEST RESULT** |
| 1. | Enter the Name, Email, Phone Number, and Password for registration of Manufactur er | Name: YYY Email: yyy[@gmail.com](mailto:abc@gmail.com) Phone Number: 8888888888  Password: \*\*\* | YYY  yyy[@gmail.com](mailto:abc@gmail.com) 8888888888  \*\*\* | YYY  yyy[@gmail.com](mailto:abc@gmail.com) 8888888888  \*\*\* | Passed |
| 2. | Compare username and password with registered field while login | Email: [yyy@gmail.com](mailto:yyy@gmail.com) Password: \*\*\* | It Redirects to **ManufactureLogin** web page | It Redirects to **ManufactureLogin** web page | Passed |
| 3. | Compare username and password with registered field while login | Email: [yye@gmail.com](mailto:yye@gmail.com) Password: \*\*\* | Invalid Email Or Password | Invalid Email Or Password | Passed |

* + 1. **DISTRIBUTOR LOGIN ANALYSIS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED OUTPUT** | **ACTUAL OUTPUT** | **TEST RESULT** |
| 1. | Enter the Name, Email, Phone Number, and Password for registration of Distributor | Name: WWW Email: www[@gmail.co](mailto:abc@gmail.com) [m](mailto:abc@gmail.com)  Phone Number: 8888877777  Password: \*\*\* | WWW  www[@gmail.com](mailto:abc@gmail.com) 8888877777  \*\*\* | WWW  www[@gmail.com](mailto:abc@gmail.com) 8888877777  \*\*\* | Passed |
| 2. | Compare username and password with registered field while login | Email: [www@gmail.co](mailto:www@gmail.com) [m](mailto:www@gmail.com)  Password: \*\*\* | It Redirects to **Login**  web page | It Redirects to **Login**  web page | Passed |
| 3. | Compare username and password with registered field while login | Email: [wew@gmail.com](mailto:wew@gmail.com) Password: \*\*\* | Invalid Email Or Password | Invalid Email Or Password | Passed |

* + 1. **BANK TRANSACTION ANALYSIS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **ACTION** | **INPUT** | **EXPECTED OUTPUT** | **ACTUAL OUTPUT** | **TEST RESULT** |
| 1. | Give Account Number, PIN and OTP in the setuserpay ment.jsp page to buy the particular product | Account Number: 123456 PIN: \*\*\*\*\*\* OTP: \*\*\*\*\*\* | Verifies whether the Account Number is present in the Bank Database and verifies the PIN and OTP  **If** Account Number, PIN and OTP matches with the Details in Bank Database  **Then** Proceeds to the payment of product | Proceeds to the payment of product | Passed |
| 2. | Give Account Number, PIN and OTP in the setuserpay ment.jsp page to buy the particular product | Account Number: 123456 PIN: \*\*\*\*\*\* OTP: \*\*\*\*\*\* | **If** Account Number, PIN and OTP does not matches with the Details in Bank Database  **Then “**Transaction Failed” is displayed | Transaction Failed | Passed |
| 3. | Compare the amount of the product with the amount the customer have in their account in Bank Database | Account Number: 123456 PIN: \*\*\*\*\*\* OTP: \*\*\*\*\*\* | **If** amount(product) <= amount (customer have in account)  **Then “**Amount debited from your account” | Amount debited from your account | Passed |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 4. | Compare the amount of the product with the amount the customer have in their account in Bank Database | Account Number: 123456 PIN: \*\*\*\*\*\* OTP: \*\*\*\*\*\* | **If** amount(product) > amount (customer have in account)  **Then “**Transaction Failed” is displayed | Transaction Failed | Passed |

1. **CONCLUSION:**
   1. **CONCLUSION AND FUTURE ENHANCEMENTS**

The application of blockchain in supply chain management is a boom now a days. Blockchain has a great feature like traceability, decentralization, transparency and immutability. These features are implemented in supply chain management. With the help of this distributed ledger technology, the duplication of smart tag will be prevented. Several interactions occur between Stakeholders during product exchange. These interactions between the participants are stored on blockchain. Since blockchain is a decentralized network, all the transaction details are stored in blockchain. The Online Banking System is provided only for the Customer side in our project. Our project’s future enhancement is to provide the Online Banking System to the Manufacturer and Distributor.

## APPENDICES

* 1. SAMPLE SCREENS

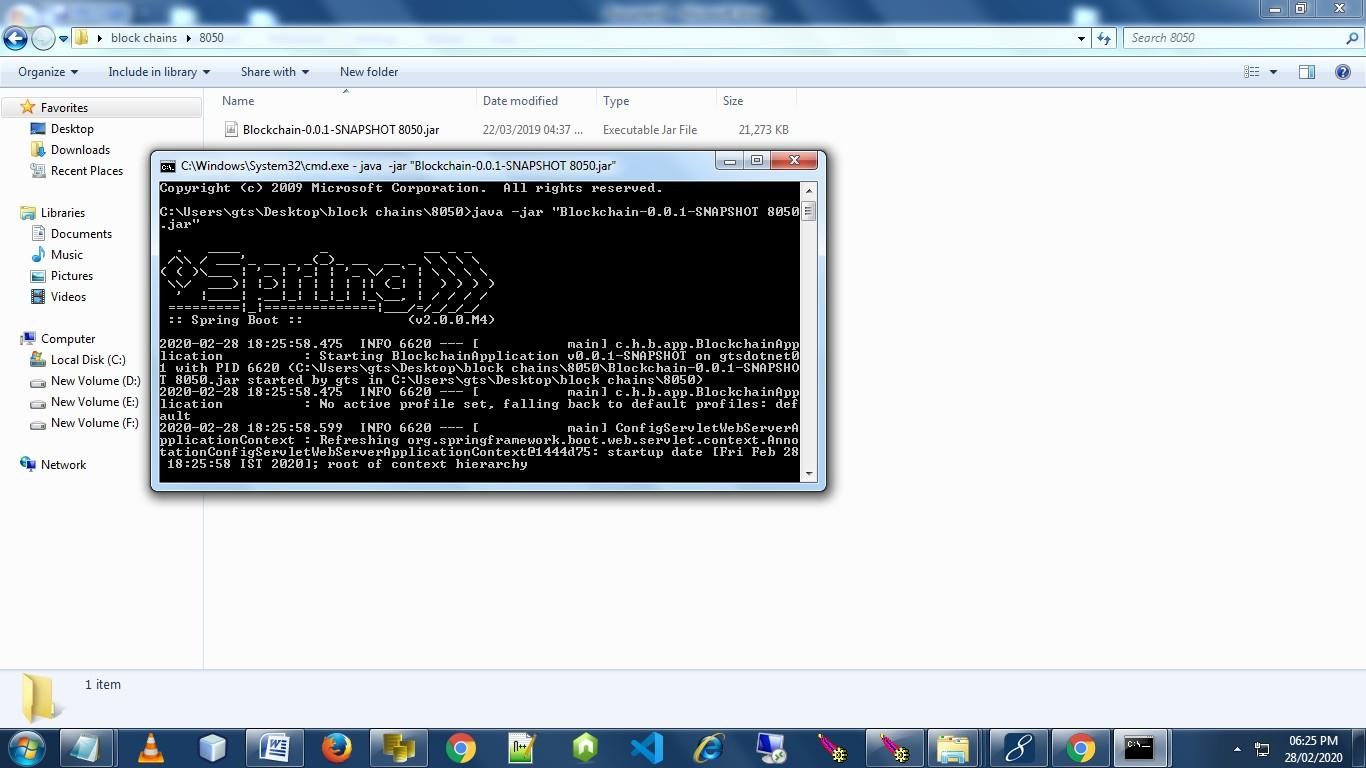


Fig.9.1. Spring Boot

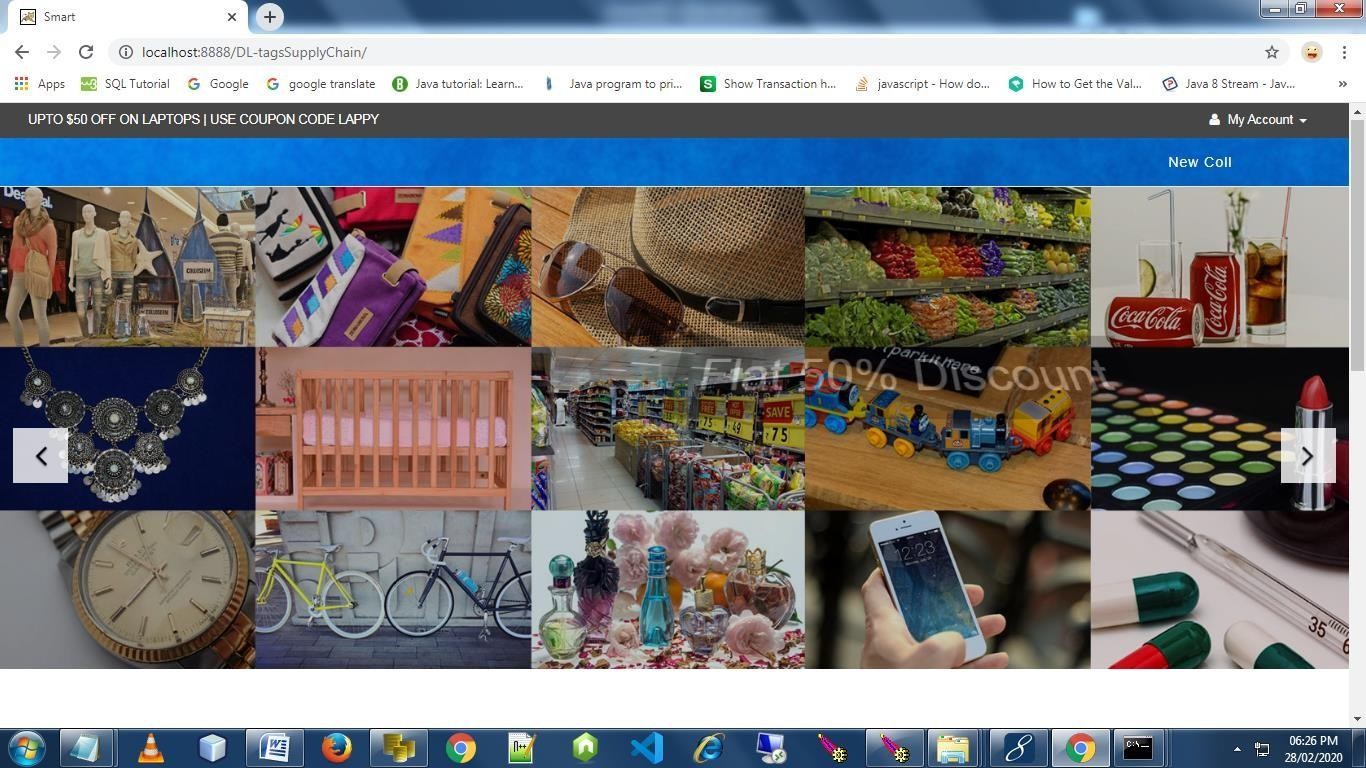


Fig.9.2. Welcome page for e-commerce

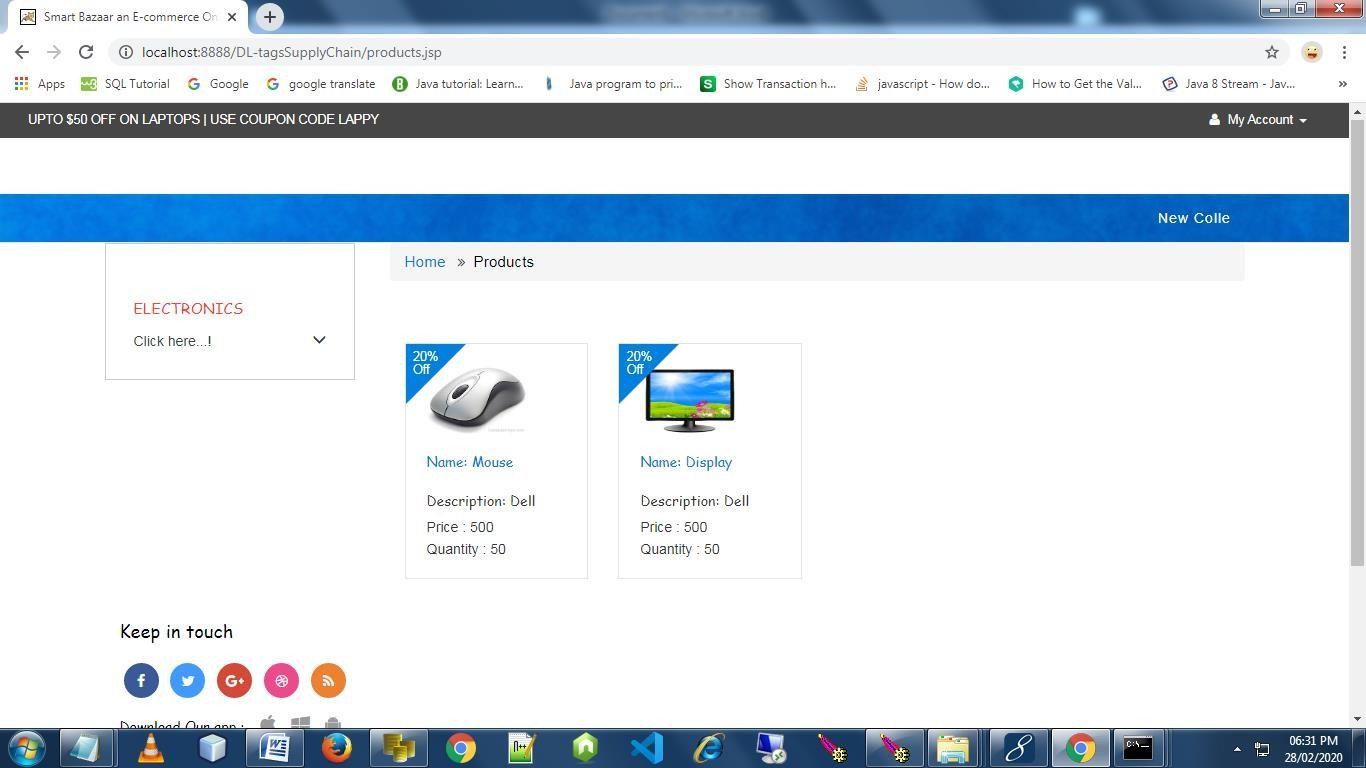


Fig.9.3.Supplier products

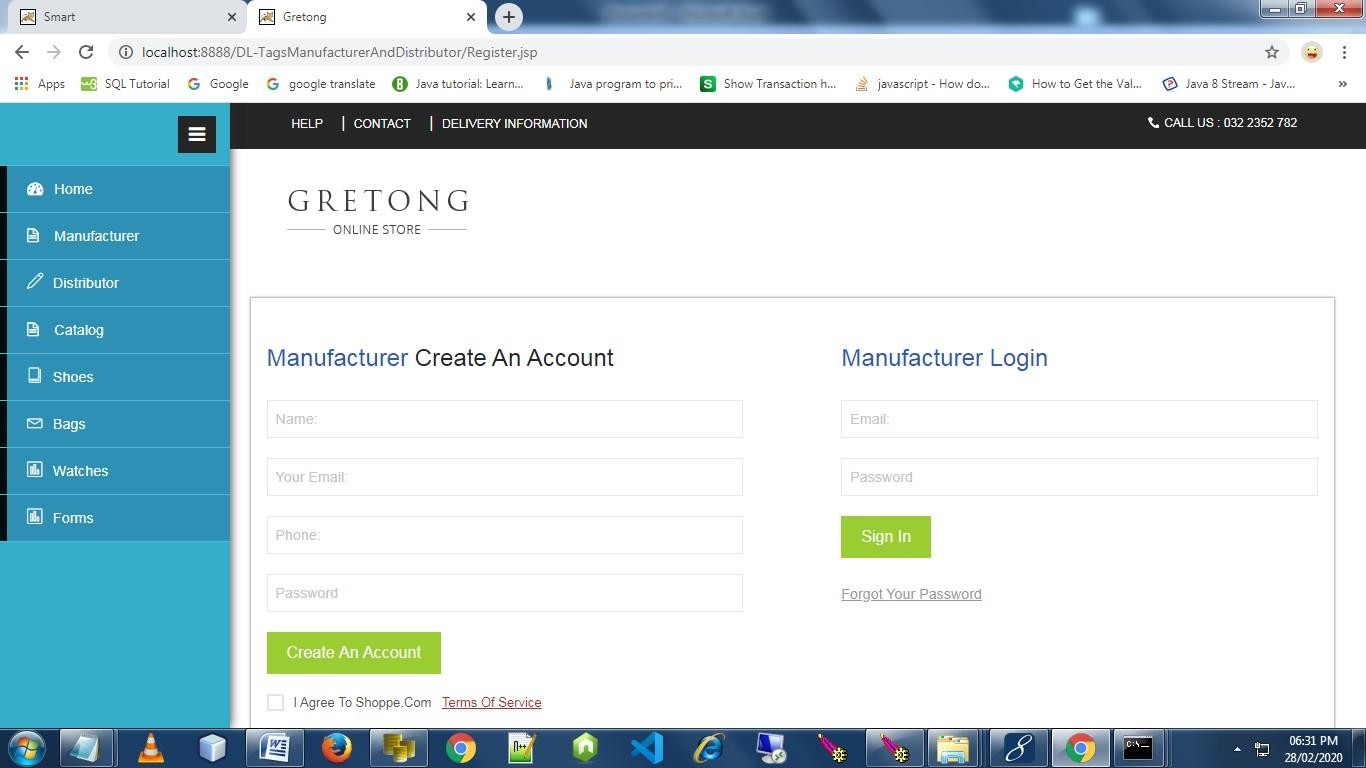


Fig. 9.4. Manufacturer account login

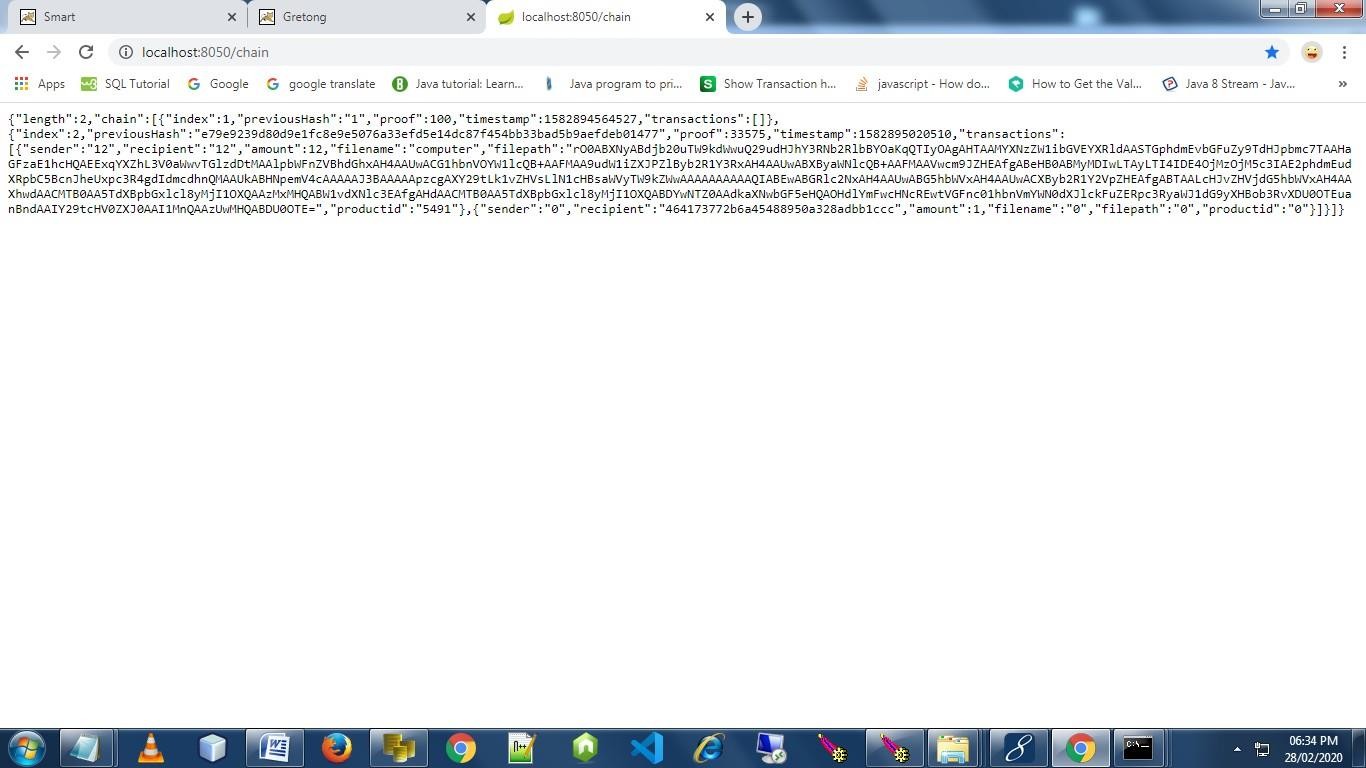


Fig.9.5. Blockchain Hashing

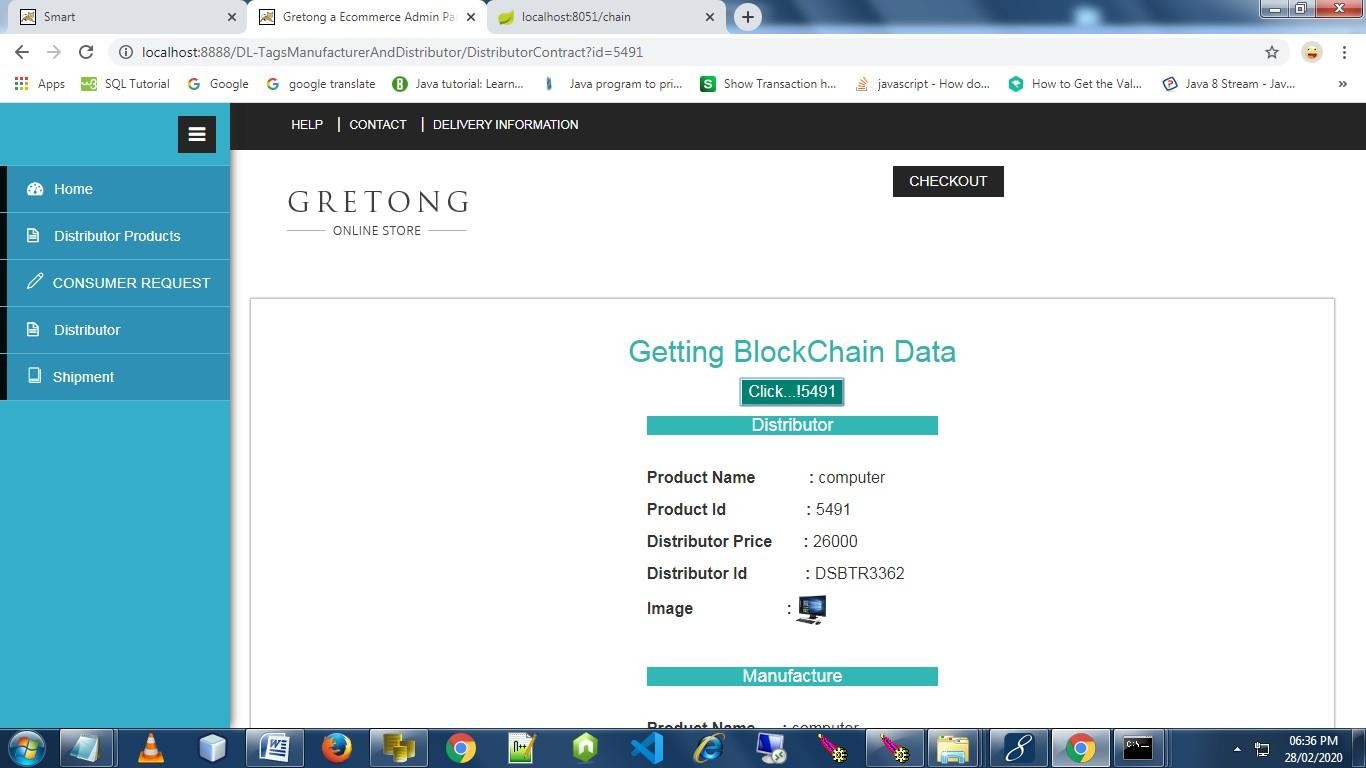


Fig. 9.6. Blockchain Data

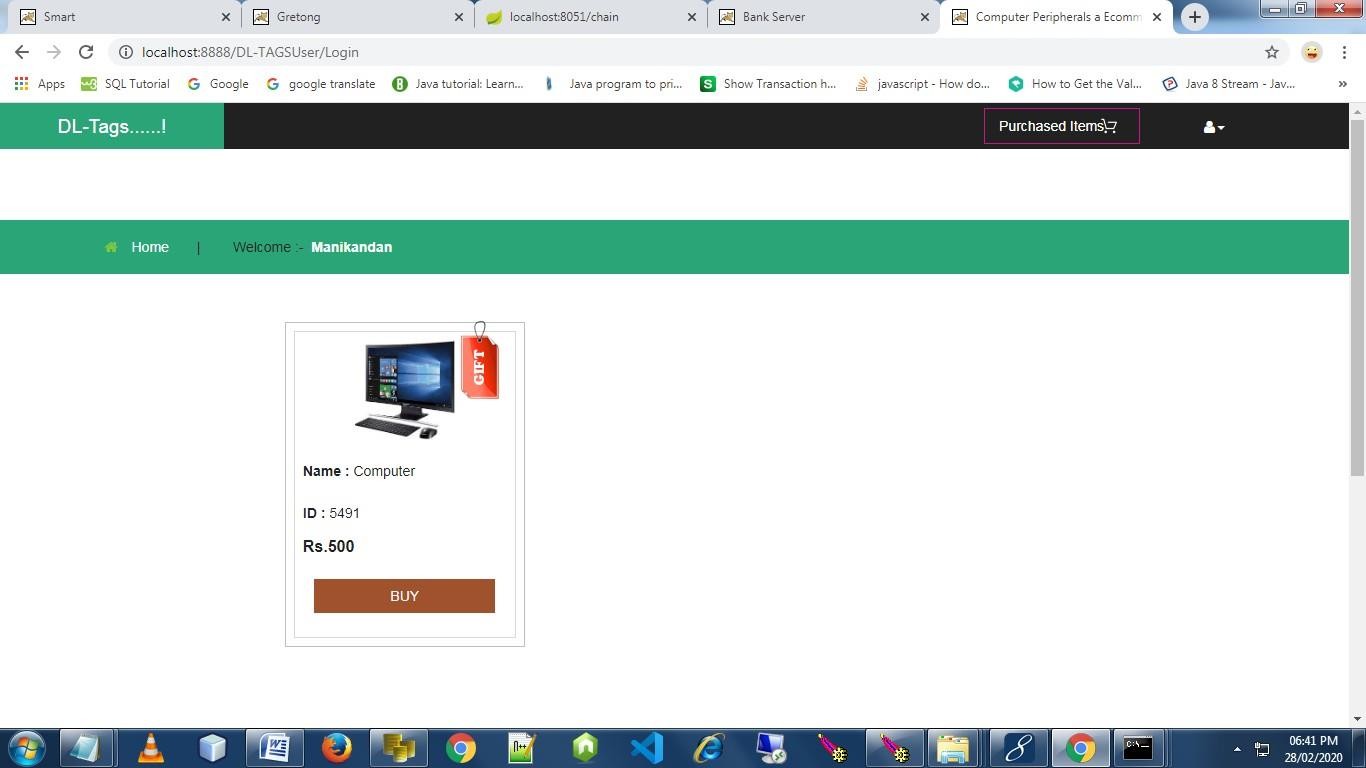


Fig.9.7. Distributor products

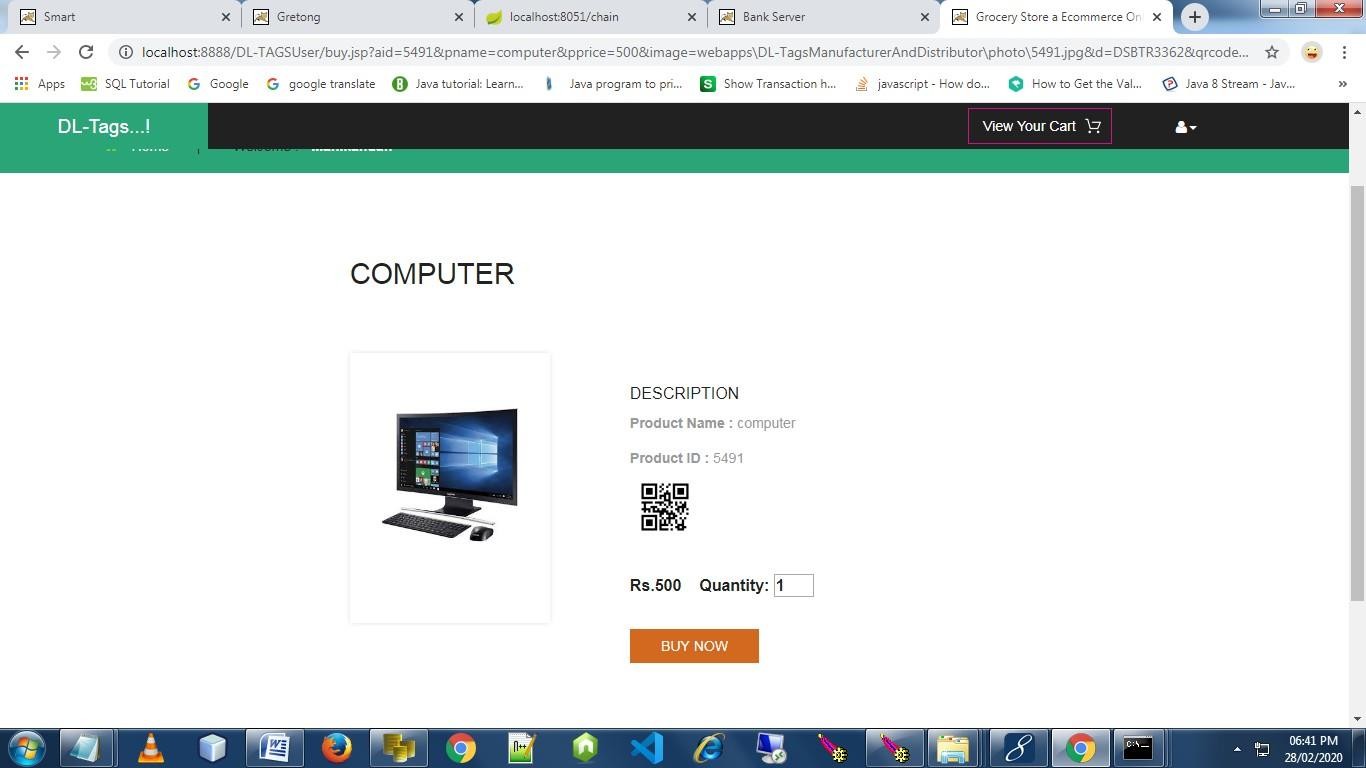


FIG.9.8.Product Description

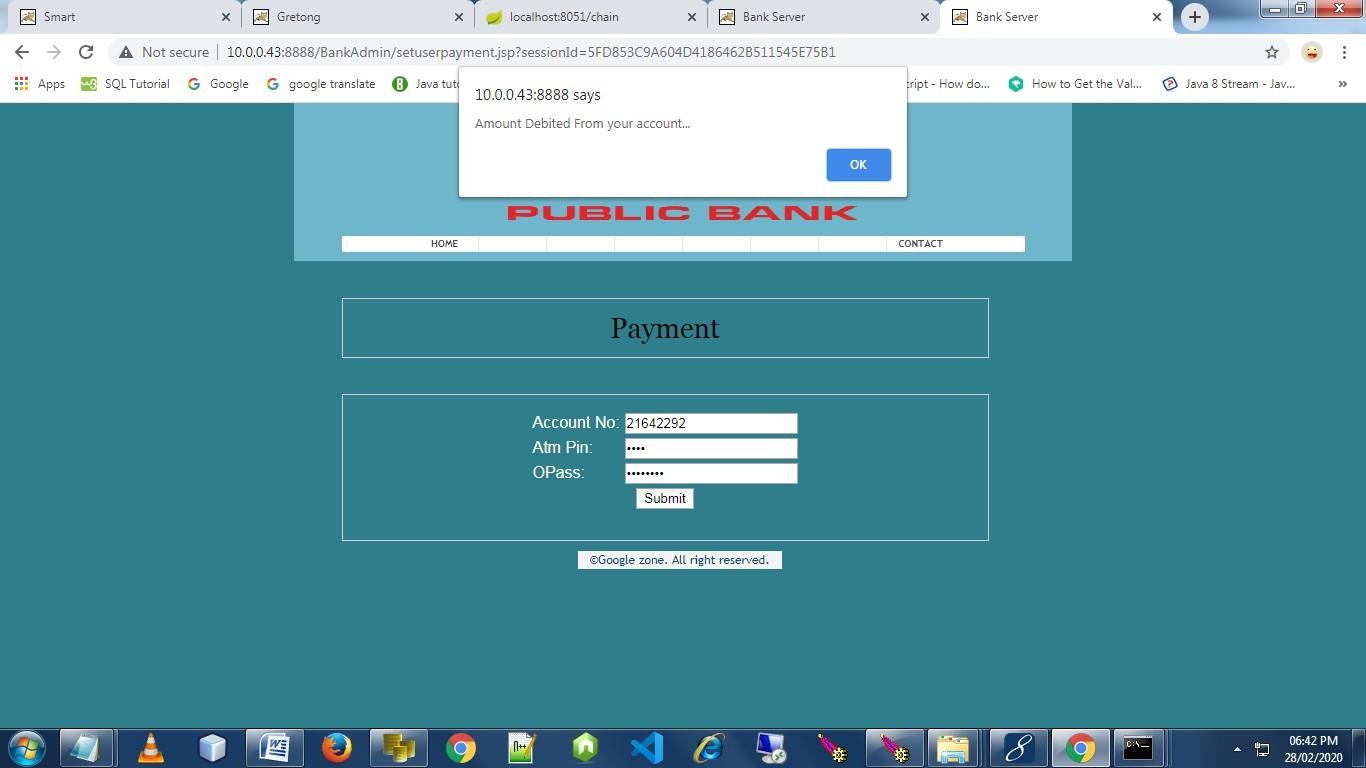


Fig.9.9. Public Bank

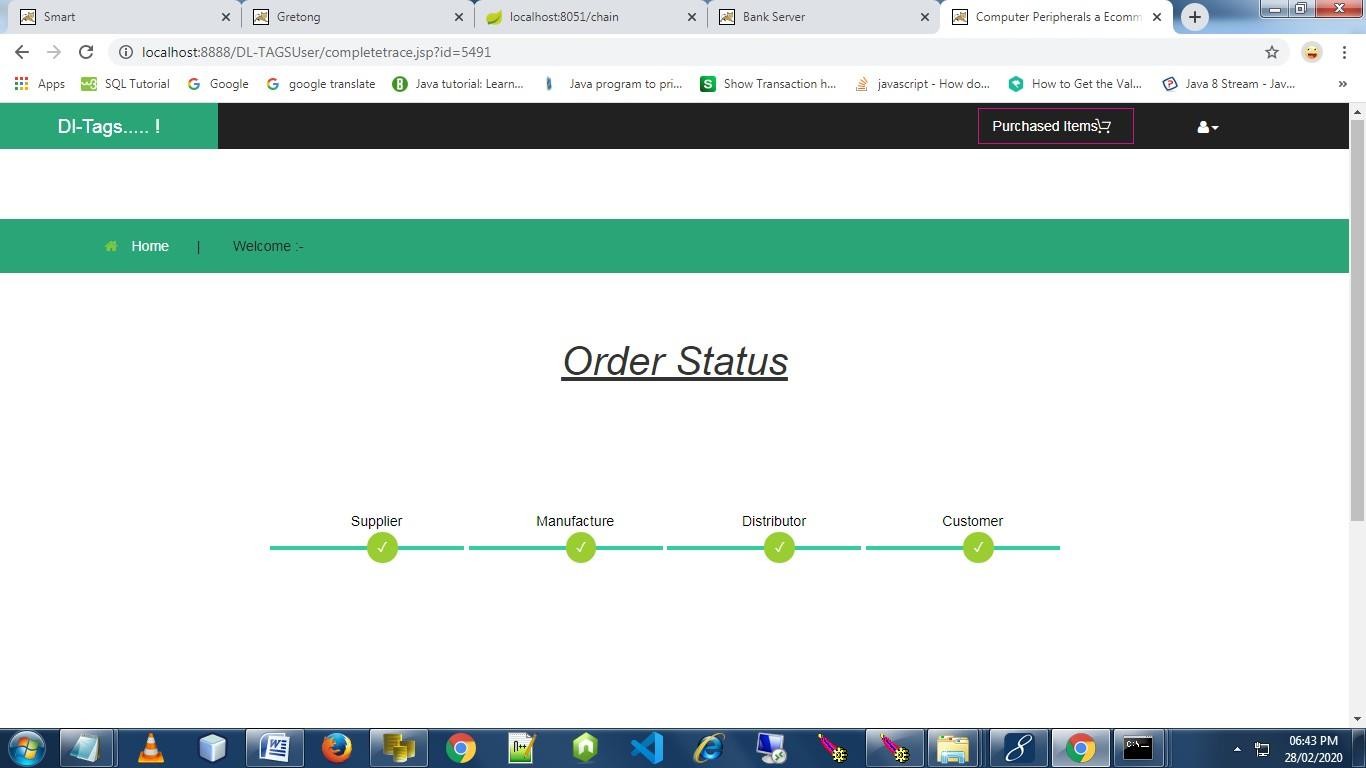


Fig.9.10.Order Status

## PUBLICATIONS

PRODUCT TRACKING AND TRACING WITH DECENTRALISED

BLOCKCHAIN - K Kiruthika, Aishwarya Suresh, P Deepika, S Kiruthika -

## [www.jetir.org](http://www.jetir.org/) (ISSN-2349-5162) © 2021 JETIR April 2021, Volume 8, Issue 4

**REFERENCES**

1. M. Christopher, ``Logistics and supply chain management: Strategies for reducing cost and improving service (second edition),'' Int. J. Logistics Res. Appl., vol. 2, no. 1, pp. 103\_104, 1999.
2. M. S. Ali, M. Vecchio, M. Pincheira, K. Dolui, F. Antonelli, and M. H. Rehmani,

``Applications of blockchains in the Internet of Things: A comprehensive survey,'' IEEE Commun. Surveys Tuts., to be published.

1. S. Georgoulas, S. Krco, and R. van Kranenburg, ``TagItSmart\_SmartTags for unlocking business potential,'' IEEE IoT Newslett., Nov. 2017. [Online]. Available: https://iot.ieee.org/newsletter/september-

2017/tagitsmart-smarttags-for-unlocking-business-potential.html

1. G. Zyskind, O. Nathan, and A. S. Pentland, ``Decentralizing privacy: Using blockchain to protect personal data,'' in Proc. IEEE Secur. Privacy Workshops (SPW), May 2015, pp. 180\_184.
2. O. Svein, Beyond Bitcoin Enabling Smart Government Using Blockchain Technology (Lecture Notes in Computer Science), vol. 9820. Cham, Switzerland: Springer, 2016, pp. 253\_264.
3. A. Narayanan and J. Clark, ``Bitcoin's academic pedigree,'' Commun. Acm, vol. 60, no. 12, pp. 36\_45, 2017.
4. P. Franco, The Blockchain-Understanding Bitcoin. New York, NY, USA: Wiley, 2014, pp. 95\_122.
5. F. M. Ben£i¢ and I. P. .arko, ``Distributed ledger technology: Blockchain compared to directed acyclic graph,'' in Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst., Jul. 2018, pp. 1569\_1570.
6. S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System. 2008, p. 9. [Online]. Available: <https://bitcoin.org/bitcoin.pdf>
7. V. Buterin, ``A next-generation smart contract and decentralized application platform,'' White Paper, 2014. [Online]. Available: https:// blockchainlab.com
8. B. Group and J. Garzik, Public Versus Private Blockchains. Part 1: Per- missionless Blockchains. Washington, DC, USA: Bitfury Group, 2015, pp. 1\_23.
9. G. Wood, ``Ethereum: A secure decentralised generalised transaction ledger,'' Ethereum & Ethcore, Ethereum Project Yellow Paper 151, 2014, pp. 1\_32.
10. S. Underwood, ``Blockchain beyond Bitcoin,'' Commun. ACM, vol. 59, no. 11, pp. 15\_17, 2016.
11. T. M. Fernández-Caramés and P. Fraga-Lamas, ``A review on the use of blockchain for the Internet of Things,'' IEEE Access, vol. 6, pp. 32979\_33001, 2018.
12. Q. He, Y. Xu, Z. Liu, J. He, Y. Sun, and R. Zhang, ``A privacy-preserving Internet of Things device management scheme based on blockchain,'' Int. J. Distrib. Sensor Netw., vol. 14, no. 11, pp. 1\_12, 2018.
13. K. Christidis and M. Devetsikiotis, ``Blockchains and smart contracts for the Internet of Things,'' IEEE Access, vol. 4, pp. 2292\_2303, 2016.
14. M. Petersen, N. Hackius, and B. von See, ``Mapping the sea of opportunities: Blockchain in supply chain and logistics,'' Inf. Technol., vol. 60, nos. 5\_6, pp. 263\_271, 2018.
15. F. Tian, ``A supply chain traceability system for food safety based on HACCP, blockchain&Internet of Things,'' in Proc. Int. Conf. Service Syst. Service Manage., Jun. 2017, pp. 1\_6.
16. How BigchainDB is Immutable\_BigchainDB Documentation. Accessed: Dec. 14, 2018. [Online]. Available: <http://docs.bigchaindb.com/>en/latest/immutable.html
17. T. Bocek, B. B. Rodrigues, T. Strasser, and B. Stiller, ``Blockchains everywhere-a use-case of blockchains in the pharma supply-chain,'' in Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM), May 2017, pp. 772\_777.
18. B. Rakic, T. Levak, Z. Drev, S. Savic, and A.Veljkovic, ``First purpose built protocol for supply chains based on blockchain,'' OriginTrail, Ljubljana, Slovenia,

Tech. Rep. 1, 2017. [Online]. Available: [www.origintrail.io](http://www.origintrail.io/)

1. TagItSmart consortium, D2.3-Final Enabler for Funcodes, Project Deliverable, Feb. 2018.
2. R. De Angelis, M. Howard, and J. Miemczyk, ``Supply chain management and the circular economy: Towards the circular supply chain,'' Prod. Planning Control, vol. 29, no. 6, pp. 425\_437, 2018.
3. I. Askoxylakis, ``A framework for pairing circular economy and the Internet of Things,'' in Proc. IEEE Int. Conf. Commun. (ICC), May 2018,

pp. 1\_6