

EFFECTIVE TRACKING OF FRAUDULENT ACTION FROM MULTI FUND TRANSACTION

Submitted in partial fulfilment of the required for the award of

Bachelor of Computer Science

By

Nellaippen Tharun Ram.K

And

Naveen.K

(REGISTER NO 41290037 AND 41290038)



DEPARTMENT OF COMPUTER SCIENCE

SCHOOL OF SCIENCE AND HUMANITIES

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY (DEEMED TO BE UNIVERSITY)

Accredited with Grade “A” by NAAC

JEPPIAR NAGAR, RAJIV GANDHI SALAI, CHENNAI 600 119 MAY 2024

SATHYABAMA



www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC I Approved by AICTE

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of Naveen (41290037) Nellaiappan Tharun Ram.K (41290038) who carried out the project entitled “EFFECTIVE TRACKING OF FRAUDULENT ACTION FROM MULTI FUND TRANSACTION ” under our supervision from JANUARY 2024 to APRIL 2024.

S. Rajeshwary

INTERNAL GUIDE

Ms.Rajeshwary .S, M.Sc, Assistant professor

DEAN AND HEAD OF THE DEPARTMENT

Dr. REKHA CHAKRAVARTHI M.E., Ph.D.

Submitted for Viva voce examination held on 26.04.24

Internal Examiner

External Examiner

DECLARATION

I am, **Nellaiappan Tharun Ram.K(41290038)** here by declare that the Project Report entitled “EFFECTIVE TRACKING OF FRAUDULENT ACTION FROM MULTI FUND TRANSACTION ” done by us under the guidance of **MS. Rajeshwary .S, M.Sc**, Assistant Professor, Department of Computer Science at **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY**, Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai-600 119 is submitted in partial fulfillment of the requirements for the award of Master of Science degree in Computer Science.

Naveen.K

Nellaiappan Tharun Ram.K

DATE:26.04.24

PLACE:CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them. I convey my thanks to **Dr. REKHA CHAKRAVARTHI M.E., Ph.D. Dean of Arts and Science** for providing me necessary support and details at the right time during the progressive reviews. I would like to express my sincere and deep sence of gratitude to my project guide **MS.Rajeshwary.S M.Sc.**, Assistant Professor, Department of Computer Science for their valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work. I wish to express my thanks to all Teaching and Non-teaching staff member of the Department of Computer Science who were helpful in many ways for the completion of the project.

Naveen.K

Nellaiappan Tharun Ram .K

ABSTRACT

Our approach to navigating the complexities of cloud computing revolves around a tenant-driven model, giving users control over connectivity options. By integrating Blockchain, we ensure transparency and security in data transactions for both Public and Private cloud storage. In the banking sector, this empowers stakeholders with comprehensive insights into user behavior across transactions, enabling the detection of anomalies and the preservation of financial integrity.

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	IV
1	INTRODUCTION	1
2	LITERATURE SURVEY	2
3	METHODOLOGY	5
3.1	HARDWARE SPECIFICATION	5
3.2	SOFTWARE SPECIFICATION	5
3.3	EXISTING SYSTEM	5
3.4	DISADVANTAGES	5
3.5	PROPOSED SYSTEM	6
3.6	ADVANTAGES	6
4	SYSTEM DESIGN	7
4.1	ARCHITECTURE DIAGRAM	7
4.2	DATA FLOW DIAGRAM	8
4.3	DATA FLOW DIAGRAM	9
4.4	USE CASE DIAGRAM	11
4.5	CLASS DIAGRAM	12
4.6	SEQUENCE DIAGRAM	14
4.7	COLLABORATION DIAGRAM	15
5	SYSTEM SPECIFICATION	16
5.1	JAVA PROGRAMMING LANGUAGE	17
5.2	HARDWARE ENVIRONMENT	17
5.3	SOFTWARE ENVIRONMENT	18

5.4	JAVA	18
5.5	BUSINESS BENEFITS	18
5.6	FEATURES OF JAVA	19
5.7	JAVA PROGRAMMING LANGUAGE	19
5.8	JAVA PLATFORM	20
5.9	JAVA VIRTUAL MACHINE	20
5.10	JAVA API	21
6	MODULES	24
6.1	USER REGISTRATION	25
6.2	BANK SERVER	25
6.3	LAND REGISTRATION AND GOLD PURCHASE	25
7	SYSTEM SECURITY	26
7.1	TAXONOMY OF TESTING	26
7.2	TESTING IN PARTICULAR	27
7.3	INTEGRATION TESTING	28
8	APPENDICES	29
8.1	SAMPLE CODE	29
8.2	SCREEN SHOTS	44
9	CONCLUSION AND FUTUREWORK	48
10	REFERENCE	49

S.NO	LIST OF FIGURES	PAGE NO
1.1	BLOCK CHAIN	1
4.1	ARCHITECTURE DIAGRAM	7
4.2	DATA FLOW DIAGRAM	8
4.3	DATA FLOW DIAGRAM	9
4.4	USE CASE DIAGRAM	11
4.5	CLASS DIAGRAM	12
4.6	SEQUENCE DIAGRAM	14
4.7	COLLABRATION DIAGRAM	15
5.1	MySQL ARCHITECTURE	21
5.2	INTERNAL COMPONENTS	22
6.1	USER REGISTRATION	24
6.2	BANK SERVER	24
6.3	LAND REGISTRATION AND GOLD PURCHASE	24
7.1	TAXONOMY OF TESTING	25
8.1	LOGIN DIAGRAM	41
8.2	TRANSACTION AND USER PAGE	41
8.3	USERREGISTRATION	42
8.4	ADD BANK DETAILS	42
8.5	USER REGISTRATION	43
8.6	LOGIN AND LOGOUT PAGE	43
8.7	LAND REGISTRATION PAGE	44
8.8	BANKING LAND AND GOLD	44

CHAPTER 1

INTRODUCTION

DOMAIN INTRODUCTION

BLOCK CHAIN

The block chain is an undeniably ingenious invention – the brainchild of a person or group of people known by the pseudonym, Satoshi Nakamoto. But since then, it has evolved into something greater, and the main question every single person is asking is: What is Block chain?

By allowing digital information to be distributed but not copied, block chain technology created the backbone of a new type of internet. Originally devised for the digital currency, Bitcoin, (Buy Bitcoin) the tech community is now finding other potential uses for the technology.

Bitcoin has been called “digital gold,” and for a good reason. To date, the total value of the currency is close to \$112 billion US. And block chains can make other types of digital value. Like the internet (or your car), you don’t need to know how the block chain works to use it. However, having a basic knowledge of this new technology shows why it’s considered revolutionary. So, we hope you enjoy this, What Is Block chain Guide. And if you already know what block chain is and want to become a block chain developer (2018 – currently in high demand!) please check out our in-depth block chain tutorial and create your very first block chain.

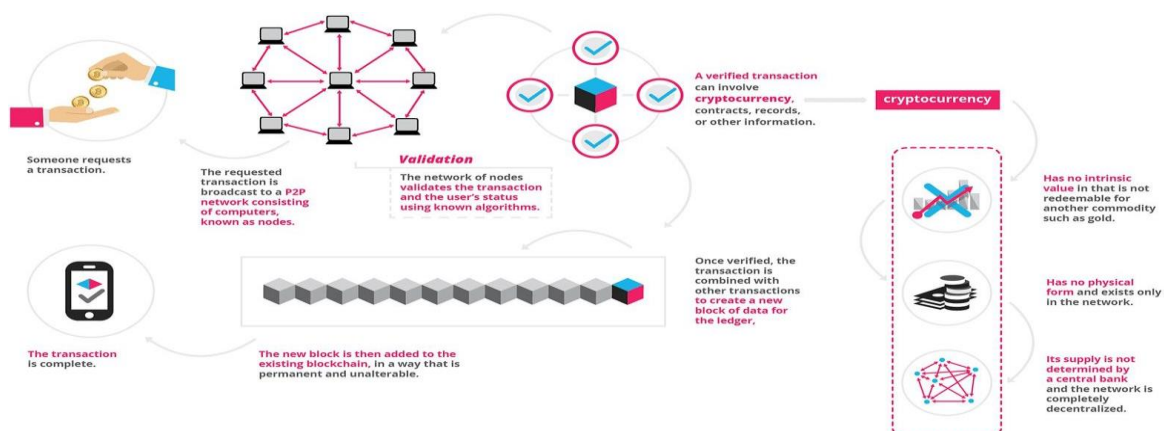


FIG 1.1 BLOCK CHAIN

CHAPTER 2

LITERATURE SURVEY

Title 1: Tax Evasion in the Bitcoin Network: Perspective of Mixing Services Author:

JunwooSeo,Mookyu Park,Haengrok Oh,Kyungho Lee

Abstract:

Various crimes using Bitcoin are highlighted. Among various crimes using Bitcoin, this paper suggests a method to detect Tax Evasion focusing on mixing service that provides Tax Evasion. This is part of the anti-money-laundering (AML) strategy, which can determine whether the mixer service is used in certain transactions by using transaction sample data using mixer. Tax Evasion using Bitcoin is often used to avoid fund tracking in the underground world and analyzing it is essential in situational awareness of fund tracking.

Title 2: Identifying Tax Evasion Accounts Author: Chih-Hua Tai,Tai-Jung Kan Abstract:

Tax Evasion is often associated with criminal activities. Anti-Tax Evasion is thus regarded as an important task in many countries. However, as it is common that money launderers divide the dirty money into multiple parts and make sequences of banking transfers or commercial transactions, manually detecting activities of Tax Evasion is challenging. To ease the task, this work establishes a two-phase intelligent method based on machine learning and data analysis techniques for identifying suspicious Tax Evasion accounts from the transaction data. The first phase emphasizes on identifying every suspicious Tax Evasion account while the second phase further retrieves highly suspicious ones so that both the recall and precision for the identification of Tax Evasion accounts can be somewhat taken care of. Evaluated on the data given by Bank SinoPac, the established intelligent method achieves a recall rate of 26.3%, which is three times the recall rate (8.6%) of the Tax Evasion Control Act in Taiwan, in the first phase, and later the precision rate can be increased up to 87.04% in the second phase.

Title 3: Cryptocurrency: The New Face of Cyber Tax Evasion Author: SagwadiMabunda

Abstract:

Virtual currencies are on the rise and so is Tax Evasion. While there are efforts to combat Tax Evasion through various intergovernmental bodies, many have expressed concern over the rise of virtual currencies. Some cryptocurrencies such as Bitcoin have played a major role in the proliferation of online Tax Evasion as it possesses characteristics that criminals are found Bitcoin and other cryptocurrencies are decentralised, anonymous/pseudonymous and irreversible. They provide the means to skirt the Anti-Tax Evasion safeguards that have been put in place. This paper discusses the intersection between Anti-Tax Evasion efforts and the challenges that are introduced by cryptocurrencies such as Bitcoin. It also looks at the case of Liberty Reserve to highlight these challenges.

Title 4: A Novel Multiobjective Approach for Detecting Tax Evasion with a Neuro- Fuzzy Technique

Author: Mohammad (Behdad) Jamshidi, Mohammadreza Gorjiankhanzad, Ali Lalbakhsh

Abstract:

Using the computational inelegant methods in processing financial data is a practicable action to reduce a wide variety of crime in this domain. In this paper, a new intelligent multi objective to recognize Tax Evasion in banks and currency exchanges is presented. The introduced approach is based on Adaptive Neuro-Fuzzy Inference System (ANFIS) which is set up by MATLAB software. The proposed method can replace conventional methods to detect the risk of Tax Evasion in suspicious banking transaction. In addition, this approach can be used in banking systems as an online technique to analyze the data of customers' accounts. Also, the probability of Tax Evasion's risk for each exchange is processed and monitored. One of the main advantages of the system is categorizing customers for different customers. The results illustrate the accuracy of this system in filtration of accounts infected by Tax Evasion is acceptable.

Title 5: A New Algorithm for Tax Evasion Detection Based on Structural Similarity

Author: Reza Soltani, UyenTrang Nguyen, Yang Uyen Trang, Mohammad Faghani, AlaaYagoub, Aijun An

Tax Evasion (ML) is the process of cleaning ‘dirty’ money, thereby making the source of funds no longer identifiable. Detecting Tax Evasion activities is a challenging task due to huge volumes of financial transactions being made in a global market on a daily basis. This paper proposes a novel approach for detecting Tax Evasion transactions among large volumes of financial data in an efficient and accurate manner. We propose a framework that applies case reduction methods to progressively reduce the input data set to a significantly smaller size. The framework then scans the reduced data to find pairs of transactions with common attributes and behaviours that are potentially involved in ML activities. It then applies a clustering method to detect potential ML groups. We present preliminary experimental results that demonstrate the effectiveness of the proposed framework.

CHAPTER 3

METHODOLOGY

3.1 HARDWARE SPECIFICATION

HDD	:	500GB
RAM	:	4 GB
Processor	:	Core i3, i5

3.2 SOFTWARE SPECIFICATION

Operating system	:	Windows 7/ 8.1
Front End	:	JDK 7/8,
Back End	:	MySQL
Tools	:	HDFS

3.3 EXISITING SYSTEM

There is no automatic process in identifying the misbehaving activities in the transactions process. All the transactions are manually verified so lots of records may not pulled out easily. So money laundering process happens very easily in the existing system.

3.4 DISADVANTAGES

- There is no monitoring process to detect Money Laundering activity.
- There is no methodology to detect / stop Black money rotation.
- PAN number is mandatory only when the limit is crossed.
- Multiple small transactions cannot be tracked.

3.5 PROPOSED SYSTEM

Alike land Registrations and Gold purchase. We consider in 3 major ways user could do money laundering process, 1. Bank Transactions, 2. Land purchase and 3. Gold purchase. We combine all these 3 activities of user transactions and analyze the money laundering concept. part from the proposed work from the base paper; we also improvise the concept by analyzing other purchase made by the user

3.6 ADVANTAGES

- All the transactions are monitored.
- Different bank transactions or even the small transactions are concatenated and analyzed
- Black money is tracked and detected.
- Big data is used for Overall transactions like Bank, Land registration and Gold Purchase.
- Application notifies the Government about misbehavioural activity of the user.

CHAPTER 4

SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE DIAGRAM



Fig 4.1 ARCHITECTURE DIAGRAM

The Overall architecture explain about the entire process of project. User will register their basic information with their aadhar number and they made transaction on their bank account , purchase gold and land using their account. Those transaction information is stored on blockchain and cloud, but in cloud it will stored in three folder like gold, land and bank. Big data will analyze the those information using MapReduce and send those information to RBI what are the transaction are having more than 20K .

4.2 DATA FLOW DIAGRAM

A data flow diagram shows the way information flows through a process or system. It includes data inputs and outputs, data stores, and the various subprocesses the data moves through. DFDs are built using standardized symbols and notation to describe various entities and their relationships.

Data flow diagrams visually represent systems and processes that would be hard to describe in a chunk of text. You can use these diagrams to map out an existing system and make it better or to plan out a new system for implementation. Visualizing each element makes it easy to identify inefficiencies and produce the best possible system.

4.3 LEVEL 0:

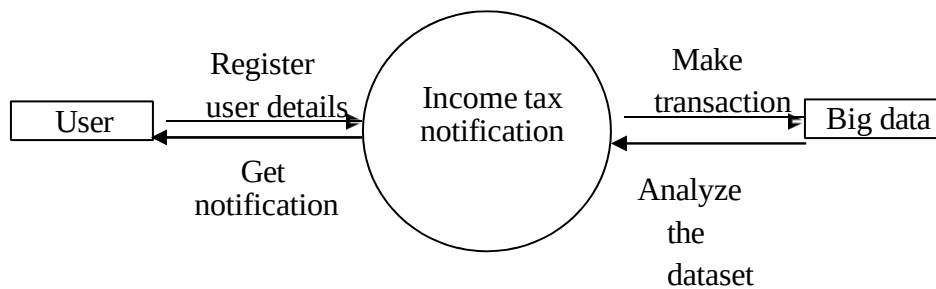


Fig4.2 Data Flow Diagram

4.3 LEVEL 1:

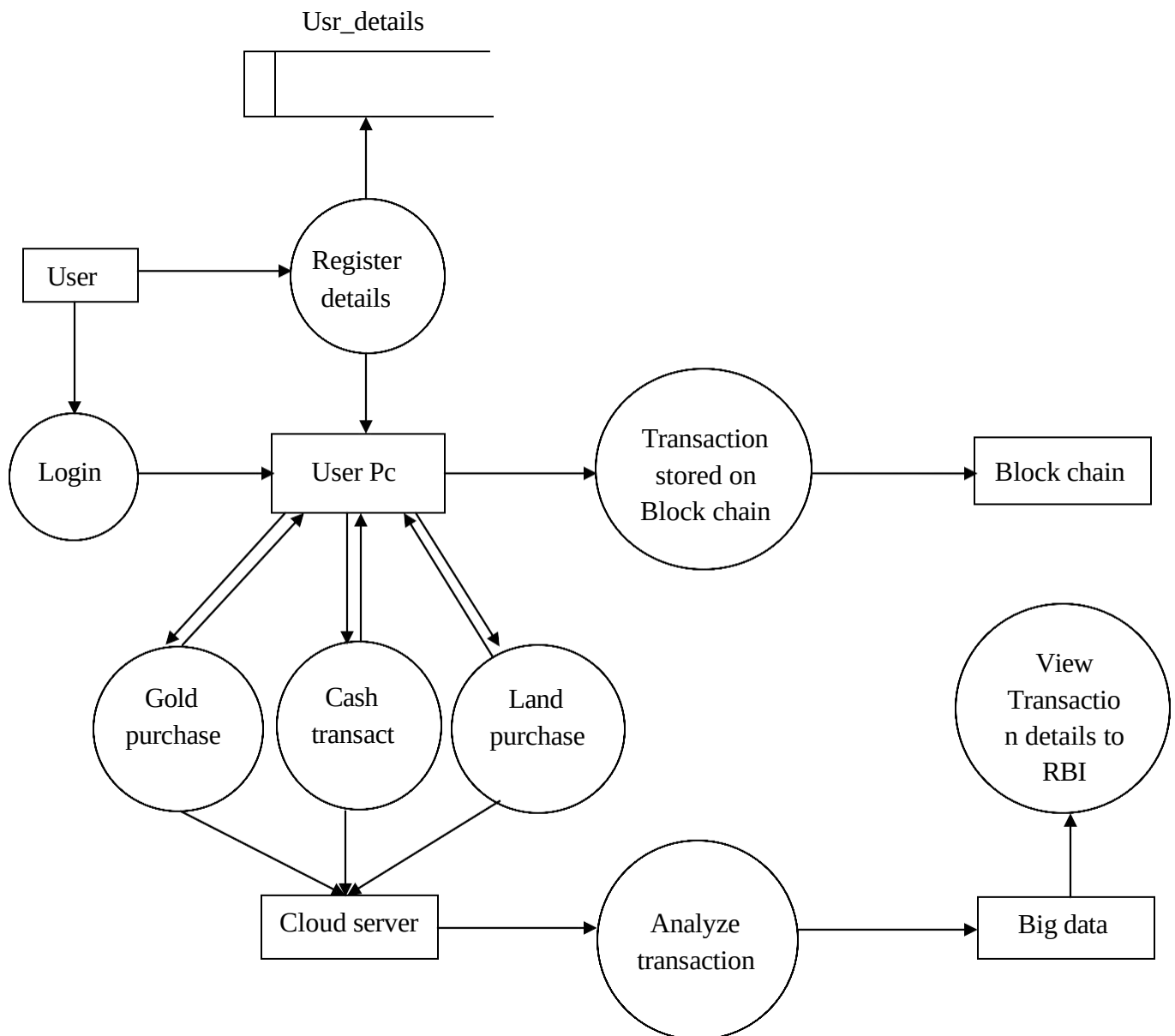


Fig. No 4.3 Data Flow Diagram

4.3 UML DIAGRAMS

The Unified Modeling Language (UML) was created to forge a common, semantically and syntactically rich visual modeling language for the architecture, design, and implementation of complex software systems both structurally and behaviorally. UML has applications beyond software development, such as process flow in manufacturing.

t is analogous to the blueprints used in other fields, and consists of different types of diagrams. In the aggregate, UML diagrams describe the boundary, structure, and the behavior of the system and the objects within it.

UML is not a programming language but there are tools that can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design.

4.4 USE CASE DIAGRAM:

A use case is a list of steps that define interaction between an actor (a human who interacts with the system or an external system) and the system itself. Use case diagrams depict the specifications of a use case and model the functional units of a system. These diagrams help development teams understand the requirements of their system, including the role of human interaction therein and the differences between various use cases. A use case diagram might display all use cases of the system, or just one group of use cases with similar functionality.

1. To begin a use case diagram, add an oval shape to the center of the drawing.
2. Type the name of the use case inside the oval.
3. Represent actors with a stick figure near the diagram, then use lines to model relationships between actors and use cases.

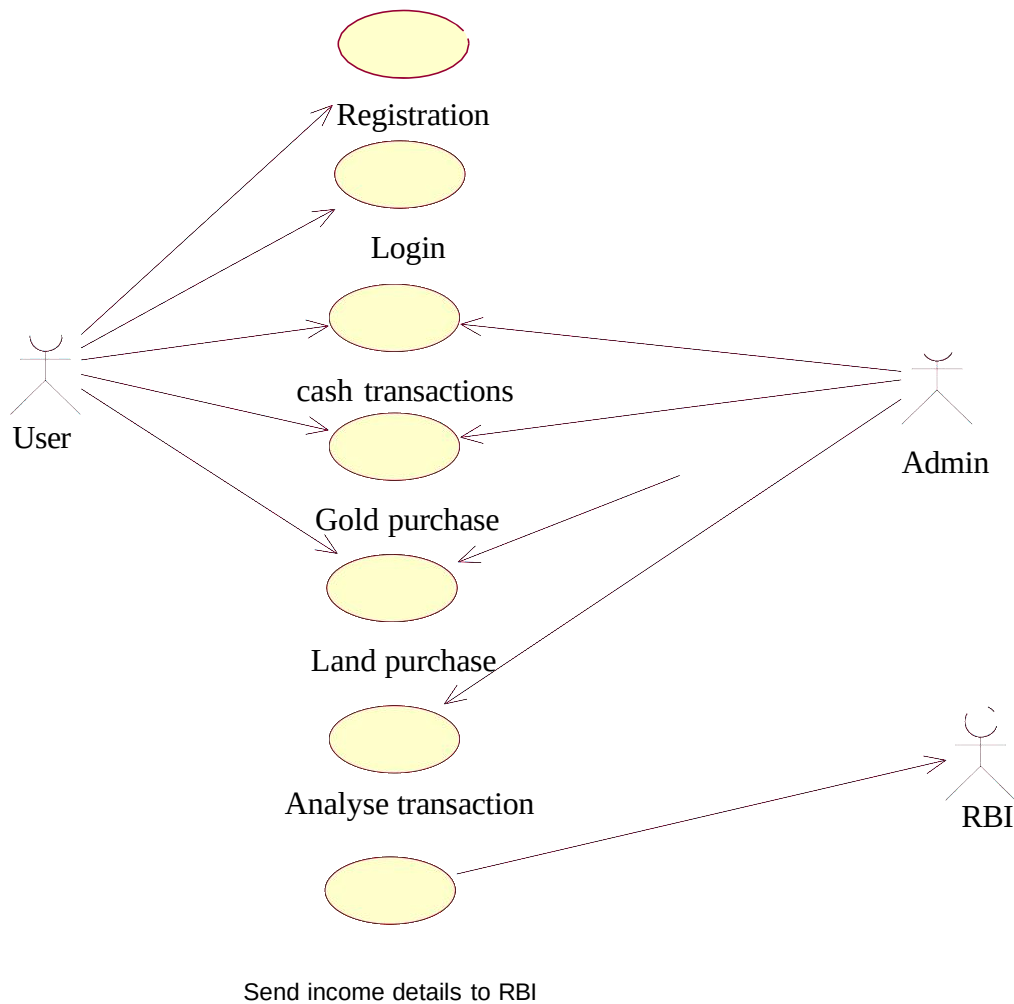


Fig 4.4 Use Case Diagram

4.5 CLASS DIAGRAM

Class diagrams represent the static structures of a system, including its classes, attributes, operations, and objects. A class diagram can display computational data or organizational data in the form of implementation classes and logical classes, respectively. There may be overlap between these two groups.

1. Classes are represented with a rectangular shape that is split into thirds. The top section displays the class name, while the middle section contains the class' attributes. The bottom section features the class operations (also known as methods).
2. Add class shapes to your class diagram to model the relationship between those objects. You may need to add subclasses, as well.
3. Use lines to represent association, inheritance, multiplicity, and other relationships between classes and subclasses. Your preferred notation style will inform the notation of these lines.

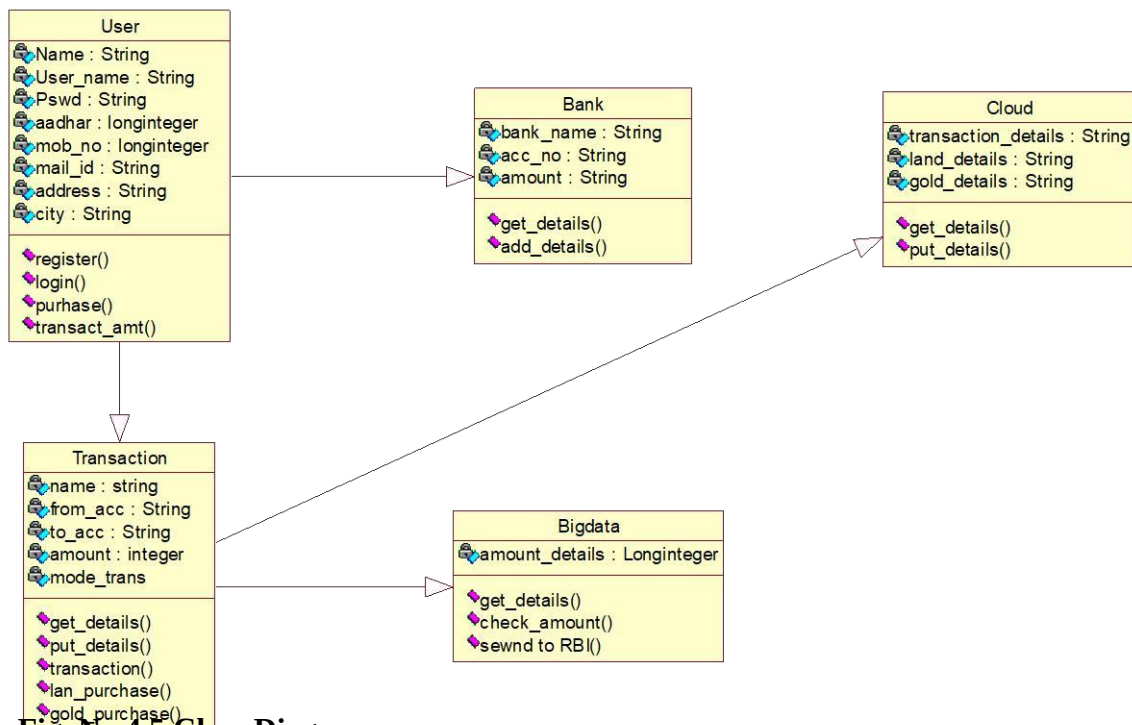


Fig. No 4.5 Class Diagram

4.6 SEQUENCE DIAGRAM

Sequence diagrams, also known as event diagrams or event scenarios, illustrate how processes interact with each other by showing calls between different objects in a sequence. These diagrams have two dimensions: vertical and horizontal. The vertical lines show the sequence of messages and calls in chronological order, and the horizontal elements show object instances where the messages are relayed.

1. To create a sequence diagram, write the class instance name and class name in a rectangular box.

2. Draw lines between class instances to represent the sender and receiver of messages.
3. Use solid arrowheads to symbolize synchronous messages, open arrowheads for asynchronous messages, and dashed lines for reply messages.

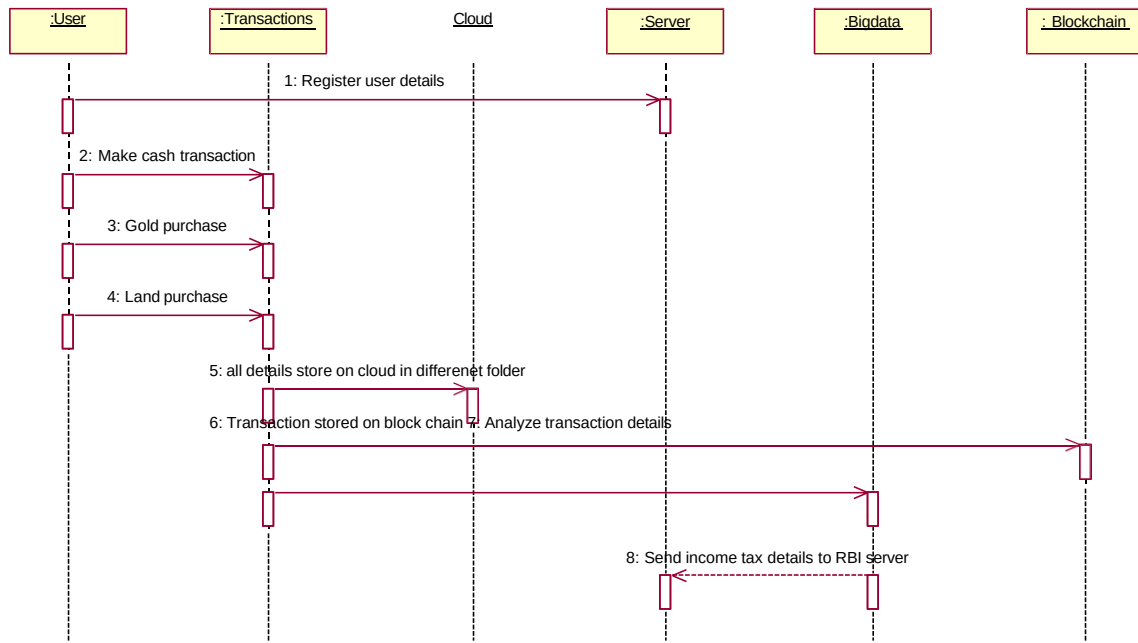


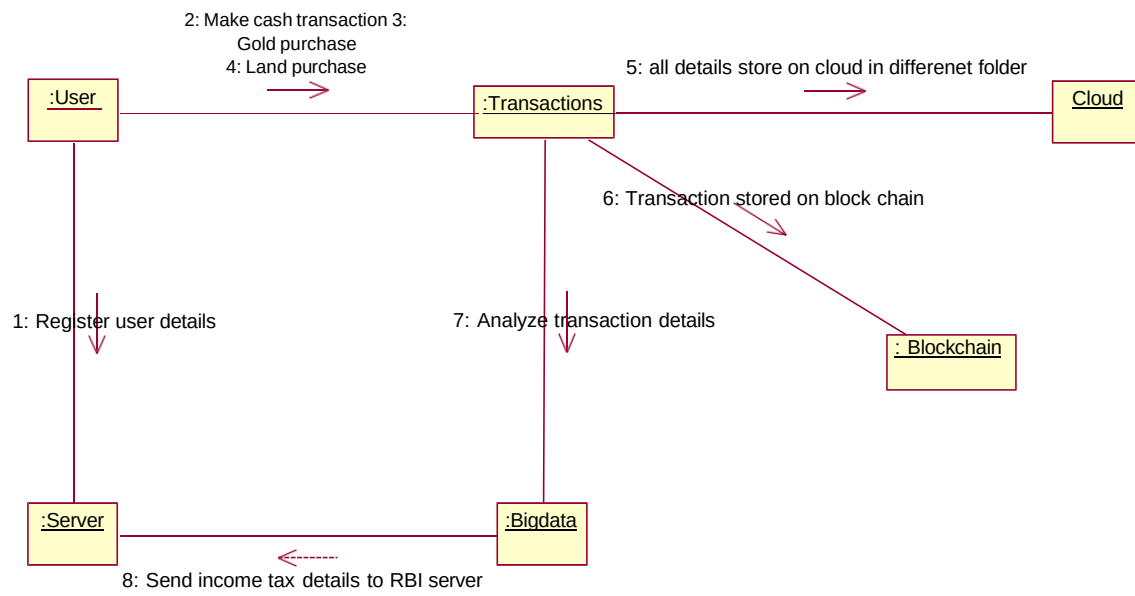
Fig. No: 4.6 Sequence diagram

4.7 COLLABORATION DIAGRAM:

Communication diagrams offer benefits similar to sequence diagrams, but they will offer a better understanding of how components communicate and interact with each other rather than solely emphasizing the sequence of events. They can be a useful reference for businesses, organizations, and engineers who need to visualize and understand the physical communications within a program. Try drawing a sequence diagram to:

- Model the logic of a sophisticated procedure, function, or operation.
- Identify how commands are sent and received between objects or components of a process.
- Visualize the consequences of specific interactions between various components in a process.

- Plan and understand the detailed functionality of an existing or future scenario.



CHAPTER 5

SYSTEM SPECIFICATION

5.1 JAVA PROGRAMMING LANGUAGE

REQUIREMENT ANALYSIS

Requirement analysis determines the requirements of a new system. This project analyses on product and resource requirement, which is required for this successful system. The product requirement includes input and output requirements it gives the wants in term of input to produce the required output. The resource requirements give in brief about the software and hardware that are needed to achieve the required functionality.

5.2 Hardware Environment

The hardware requirements may serve as the basis for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole system. They are used by software engineers as the starting point for the system design. It shows what the systems do and not how it should be implemented.

- Hard disk : 500 GB
- RAM : 4 GB
- PROCESSOR : CORE i5/ i7

5.3 Software Environment

The software requirements are the specification of the system. It should include both a definition and a specification of requirements. It is a set of what the system should do rather than how it should do it. The software requirements provide a basis for creating the software requirements specification. It is useful in estimating cost, planning team activities, performing tasks and tracking the team's and tracking the team's progress throughout the development activity.

- Operating system : Windows 7/8.1
- Languages : Java , Solidity, go
- Data Base : Mysql , JSON
- IDE : Net Beans 8.2

5.4 JAVA

- The Java platform is the ideal platform for network computing. Running across all platforms -- from servers to cell phones to smart cards -- Java technology unifies business infrastructure to create a seamless, secure, networked platform for your business. The Java platform benefits from a massive community of developers and supporters that actively work on delivering Java technology-based products and services as well as evolving the platform through an open, community-based, standards organization known as the Java Community Process program. You can find Java technology in cell phones, on laptop computers, on the Web, and even trackside at Formula One Grand Prix races. The fact is today, you can find Java technology just about everywhere!

5.5 Business Benefits

A richer user experience - Whether you're using a Java technology-enabled mobile phone to play a game or to access your company's network, the Java platform provides the foundation for true mobility. The unique blend of mobility and security in Java technology makes it the ideal development and deployment vehicle for mobile and

wireless solutions.

- The ideal execution environment for Web services - The Java and XML languages are the two most extensible and widely accepted computing languages on the planet, providing maximum reach to everyone, everywhere, every time, to every device and platform.
- Enabling business from end to end - Java offers a single, unifying programming model that can connect all elements of a business infrastructure.

5.6 FEATURES OF JAVA

Java is a programming language originally developed by James Gosling at Sun Microsystems (which is now a subsidiary of Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code(class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is general- purpose, concurrent, class-based, and object- oriented, and is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere". Java is considered by many as one of the most influential programming languages of the 20th century, and widely used from application software to web application.

The original and reference implementation Java compilers, virtual machines, and class libraries were developed by Sun from 1995. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of their Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java and GNU Class path.

5.7 JAVA PROGRAMMING LANGUAGE

The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portal
- Distributed
- High performance

Each of the preceding buzzwords is explained in *The Java Language Environment*, a white paper written by James Gosling and Henry Chilton. In the Java programming language, all source code is first written in plain text files ending with the .java extension. Those source files are then compiled into .class files by the javac compiler. A .class file does not contain code that is native to your processor; it instead contains *byte codes* — the machine language of the Java Virtual Machine (Java VM). The java launcher tool then runs your application with an instance of the Java Virtual Machine.

Because the Java VM is available on many different operating systems, the same .class files are capable of running on Microsoft Windows, the Solaris Operating System (Solaris OS), Linux, or Mac OS. Some virtual machines, such as the Java Hotspot virtual machine, perform additional steps at runtime to give your application a performance boost. This includes various tasks such as finding performance bottlenecks and recompiling (to native code) frequently used sections of code.

5.8 JAVA PLATFORM

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Microsoft Windows, Linux, Solaris OS, and Mac OS. Most platforms can be described as a combination of the operating system and underlying hardware. 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.

The Java platform has two components:

- The Java Virtual Machine
- The Java Application Programming Interface (API)

5.9 JAVA VIRTUAL MACHINE

A **Java virtual machine (JVM)** is a virtual machine that can execute Java byte code. It is the code execution component of the Java software platform. A Java virtual machine is a program which executes certain other programs, namely those containing Java byte code instructions. JVMs are most often implemented to run on an existing operating system, but can

also be implemented to run directly on hardware. A JVM provides an environment in which Java byte code can be executed, enabling such features as automated exception handling, which provides root-cause debugging information for every software error (exception), independent of the source code.

A JVM is distributed along with a set of standard class libraries that implement the Java application programming interface (API). These libraries, bundled together with the JVM, form the Java Runtime Environment (JRE). JVMs are available for many hardware and software platforms. The use of the same byte code for all JVMs on all platforms allows Java to be described as a writeonce, runanywhere programming language, versus writeonce, compileanywhere, which describes cross-platform compiled languages. Thus, the JVM is a crucial component of the Java platform.

5.10 JAVA API

The API is a large collection of ready-made software components that provide many useful capabilities. It is grouped into libraries of related classes and interfaces; these libraries are known as *packages*.

As a platform-independent environment, the Java platform can be a bit slower than native code. However, advances in compiler and virtual machine technologies are bringing performance close to that of native code without threatening portability.

BACK END

MYSQL IS A DATABASE MANAGEMENT SYSTEM.

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

MySQL software is Open Source.

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. If you want to redistribute altered versions MySQL, other licences are available

MySQL Server works in client/server or embedded systems.

The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

A large amount of contributed MySQL software is available.

MySQL is a widely supported database, and very likely that most of your applications supports it.

MySQL Architecture

Three layer model:

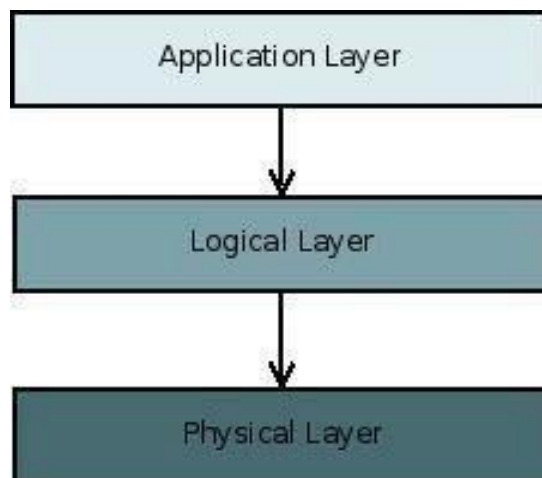


FIG 5.1 MySQL ARCHITECTURE

The application layer contains common network services for connection handling, authentication and security. This layer is where different clients interact with MySQL these clients can be written in different APIs: .NET, Java, C, C++, PHP, Python, Ruby, Tcl, Eiffel, etc...

The Logical Layer is where the MySQL intelligence resides, it includes functionality for query parsing, analysis, caching and all built-in functions (math, date...). This layer also provides functionality common across the storage engines.

The Physical Layer is responsible for storing and retrieving all data stored in “MySQL”. Associated with this layer are the storage engines, which MySQL interacts with very basic standard APIs. Each storage engine has its strengths and weaknesses, some of these engines are MyISAM, InnoDB, CSV, NDB Cluster, Falcon, etc...

SOME INTERNAL COMPONENTS:

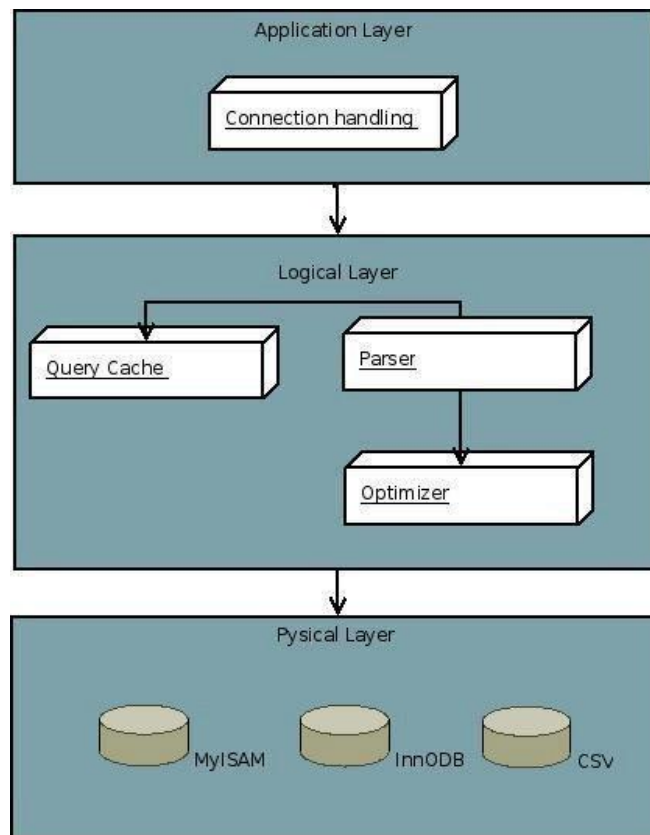


FIG 5.2 INTERNAL SOMPONENTS

CHAPTER 6

MODULES

A modular design reduces complexity, facilitates change (a critical aspect of software maintainability), and results in easier implementation by encouraging parallel development of different parts of the system. Software with effective modularity is easier to develop because functions may be compartmentalized and interfaces are simplified. Software architecture embodies modularity that is software is divided into separately named and addressable components called modules that are integrated to satisfy problem requirements.

Modularity is the single attribute of software that allows a program to be intellectually manageable. The five important criteria that enable us to evaluate a design method with respect to its ability to define an effective modular design are: Modular decomposability, Modular Composability, Modular Understandability, Modular continuity, Modular Protection.

The following are the modules of the project, which is planned in aid to complete the project with respect to the proposed system, while overcoming existing system and also providing the support for the future enhancement.

MODULE LIST:

- User registration
- Bank server
- Land registration and gold purchase
- Cloud deployment
- Blockchain deployment
- Bigdata Analysis & black money notification

USER REGISTRATION:

In this module we are going to create a User application by which the User is allowed to access the data from the Server. Here first the User wants to create an account and then only they are allowed to access the Network. Once the User creates an account, they are to login into their account and request the Job from the Server. Based on the User's request, the service Provider will process the User requested Job and respond to them. All the User details will be stored in the Database.

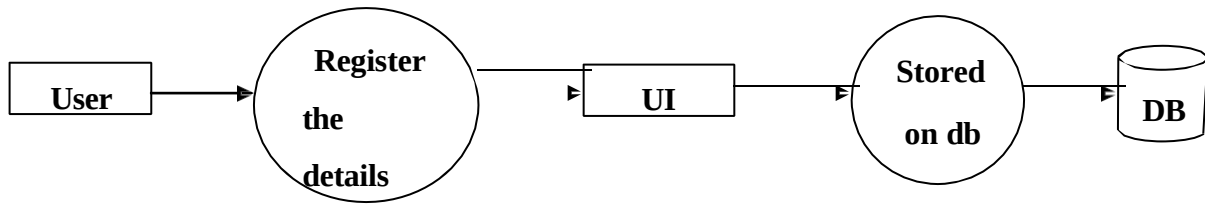


FIG 6.1 USER REGISTRATION

BANK SERVER:

Bank Service Provider will contain information about the user in their Data Storage. Also the Bank Service provider will maintain the all the User information to authenticate when they want to login into their account. The User information will be stored in the Database of the Bank Service Provider. To communicate with the Client and with the other modules of the Company server, the Bank Server will establish connection between them. For this Purpose we are going to create a User Interface Frame.

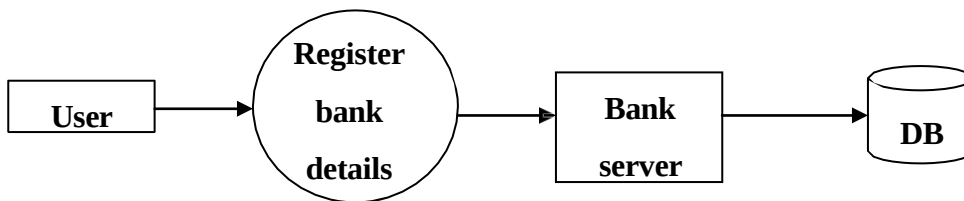


FIG 6.2 BANK SERVER

LAND REGISTRATION AND GOLD PURCHASE

In this module we implement land registration and purchased details to be monitor . Here, user name, land documents, price and selling price land. And also we monitor the gold purchase of every user and all other property details will be monitored based on user' Id.

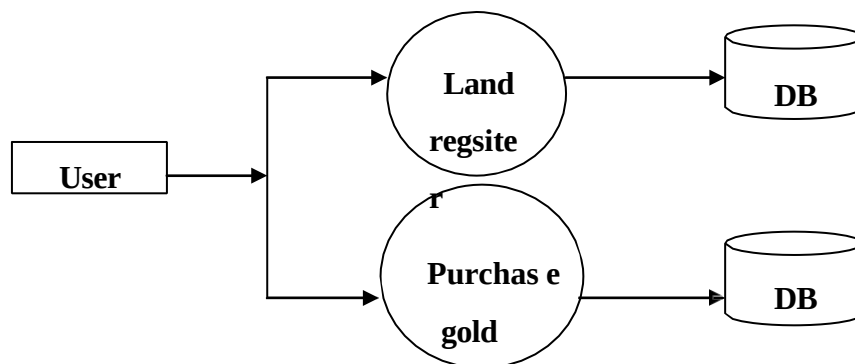


FIG 6.3 LAND REGISTRATION AND GOLD PURCHASE

CHAPTER 7

SYSTEM TESTING

Generally testing process is carried out to ensure that the system has been developed according to the required specifications and the expected output is properly obtained. There are two main categories of testing namely, the White Box Testing and the Black Box Testing. Each of this testing in turn consists of many types of testing.

7.1 TAXONOMY OF TESTING

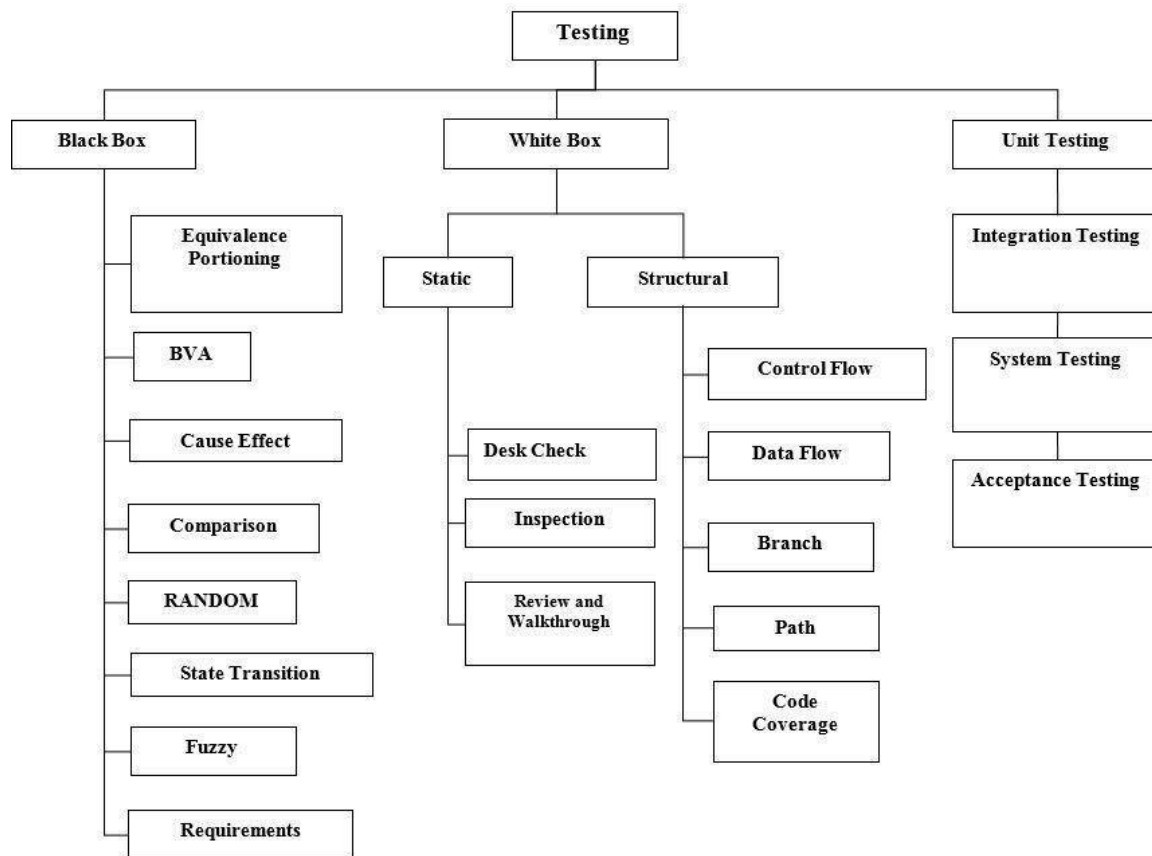


Fig 7.1 Taxonomy of Testing

➤ **SYSTEM TESTING:**

System testing of software or hardware is testing conducted on complete, integrated system to evaluate the system's compliance with its specified requirements. Once all of the modules have been completely developed and both unit testing and integration testing is done on the various parts of the modules later the system testing is done so as to ensure that the requirements are fulfilled properly. All it basically does is it performs tests to find the discrepancies between the system and its original objective, current specifications and system documentation. If any discrepancy is to be found the respective errors will be rectified and again system testing will be performed to make sure the rectification does not introduce a new error into the system.

➤ **STRUCTURE TESTING:**

It is concerned with exercising the internal logic of a program and traversing certain execution paths. Structure testing takes into account of all process that works internally to make the entire system to work properly. The basic structure and the background codes and fragments that help in upholding the system are used.

This goes layer by layer as in till reaching the core process. The various layers or levels depends on the type of the project and the domain it comes under. This is considered to be a part of the White Box Testing Process. This is done so as to make sure all the internal processes work properly. If They don't then the probability that the entire process may collapse is a possibility and this causes a grave danger to the project leading to failure.

7.2 TESTING IN PARTICULAR

UNIT TESTING

Unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use.

In this project, all statements are executed properly. All units of program programs are tested in different computer. And the result of the project is same in all system.

7.4 INTEGRATION TESTING

Integration testing (sometimes called **integration and testing**, abbreviated **I&T**) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

CHAPTER 8

APPENDIX

8.1 SAMPLE CODE:

Home Page:

```
public class HomePage extends javax.swing.JFrame {
```

```
/**
```

```
 * Creates new form HomePage
```

```
 */
```

```
public HomePage()
```

```
{ initComponents();
```

```
}
```

```
/**
```

```
 * This method is called from within the constructor to initialize the form.
```

```
 * WARNING: Do NOT modify this code. The content of this method is always
```

```
 * regenerated by the Form Editor.
```

```
@SuppressWarnings("unchecked")
```

```
desc="Generated
```

```
Code">//GEN-
```

```
/**/
```

```
<editor-fold
```

```
defaultstate="collapsed"
```

```
BEGIN:initComponents
```

```
private void initComponents() {
```

```
    buttonGroup1 = new javax.swing.ButtonGroup();
```

```
    jPanel1 = new javax.swing.JPanel();
```

Menu:

```
private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    menu_landregister = new javax.swing.JButton();
    menu_cashtransactions = new javax.swing.JButton();
    menu_banking = new javax.swing.JButton();
    menu_back = new javax.swing.JButton();
    menu_cashtransactions1 = new javax.swing.JButton();


    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);


    jPanel1.setBackground(new java.awt.Color(0, 153, 153));


    menu_landregister.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
    menu_landregister.setText("LAND REGISTRATION");
    menu_landregister.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt)
        { menu_landregisterActionPerformed(evt);

        }

    });

    menu_cashtransactions.setFont(new java.awt.Font("Times New Roman", 1, 24)); //
NOI18N

    menu_cashtransactions.setText("CASH TRANSACTIONS");
```

```

menu_cashtransactions.addActionListener(new java.awt.event.ActionListener()
    { public void actionPerformed(java.awt.event.ActionEvent evt) {

menu_cashtransactionsActionPerformed(evt);

    }

});
menu_banking.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
menu_banking.setText("BANKING");
menu_banking.addActionListener(new java.awt.event.ActionListener() { public
void actionPerformed(java.awt.event.ActionEvent evt) {

menu_bankingActionPerformed(evt);

    }

});
menu_back.setFont(new java.awt.Font("Times New Roman", 1, 24)); // NOI18N
menu_back.setText("BACK");

menu_back.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt)
{ menu_backActionPerformed(evt);

    }

});
menu_cashtransactions1.setFont(new java.awt.Font("Times New Roman", 1, 24)); //
NOI18N
menu_cashtransactions1.setText("GOLD PURCHASE");

```

```

menu_cashtransactions1.addActionListener(new java.awt.event.ActionListener()
{
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        menu_cashtransactions1ActionPerformed(evt);
    }
});
javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
jPanel1.setLayout(jPanel1Layout);
jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(add Component, jPanel1Layout.createSequentialGroup()
            .addContainerGap(80, Short.MAX_VALUE)

            .addGroup(jPanel1Layout.createParallelGroup(add Component
                , false)

                    .addComponent(menu_cashtransactions,
                        javax.swing.GroupLayout.DEFAULT_SIZE,
                        javax.swing.GroupLayout.DEFAULT_SIZE,
                        Short.MAX_VALUE)

                    .addComponent(menu_landregister,
                        javax.swing.GroupLayout.DEFAULT_SIZE,
                        javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

                    .addComponent(menu_banking,
                        javax.swing.GroupLayout.PREFERRED_SIZE,
                        303, javax.swing.GroupLayout.PREFERRED_SIZE)

                    .addComponent(menu_back, javax.swing.GroupLayout.PREFERRED_SIZE, 303,
                        javax.swing.GroupLayout.PREFERRED_SIZE)

                    .addComponent(menu_cashtransactions1,
                        javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
                        javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))

            .addGap(74, 74, 74))

```

```
this.dispose();
```

```
new CashTrans().setVisible(true);
```

```
}//GEN-LAST:event_menu_cashtransactionsActionPerformed
```

```
public static void main(String args[])
```

```
{ try {
```

```
    for (javax.swing.UIManager.LookAndFeelInfo info :  
    javax.swing.UIManager.getInstalledLookAndFeels()) { if ("Nimbus".equals(info.getName())) {  
    javax.swing.UIManager.setLookAndFeel(info.getClassName()); break;
```

```
}
```

```
}
```

```
} catch (ClassNotFoundException ex) {
```

```
    java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE,  
        null, ex);
```

```
} catch (InstantiationException ex) {
```

```
    java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE,  
        null, ex);
```

```
} catch (IllegalAccessException ex) {
```

```
    java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE,  
        null, ex);
```

```
} catch (javax.swing.UnsupportedLookAndFeelException ex) {
```

```
java.util.logging.Logger.getLogger(Menu.class.getName()).log(java.util.logging.Level.SEVERE,  
null, ex);
```

```
}
```

```
//</editor-fold>
```

```
/* Create and display the form */
```

```
java.awt.EventQueue.invokeLater(new Runnable() {
```

```
public void run() {
```

```
new Menu().setVisible(true);
```

```
}
```

```
});
```

```
}
```

Upload Page:

```
public class UploadPageAdmin extends javax.swing.JFrame
```

```
{ private String userName;
```

```
private List<File> selFiles = new ArrayList<File>(); private
```

```
String[] filesList = null;
```

```
private Set<String> fileNamesLst = new HashSet<String>();
```

```
private Set<String> roleSel = new HashSet<String>(); private
```

```
File roles El;
```

```
/**
```

```
* Creates new form UploadPageAdmin
```

```
*/
```

```
public UploadPageAdmin() {
```

```

        initComponents();

    }

/**

 * This method is called from within the constructor to initialize the form.

 * WARNING: Do NOT modify this code. The content of this method is always

@SuppressWarnings("unchecked")
 * regenerated by the Form Editor.
//      <editor-fold          desc="Generated Code">//GEN-

    */      defaultstate="collapsed"
BEGIN: initComponents
private void initComponents() {

    jPanel1 = new javax.swing.JPanel(); add_bt
    = new javax.swing.JButton();

    fileLst_cb = new javax.swing.JComboBox();
    jLabel2 = new javax.swing.JLabel();
    upload_bt = new javax.swing.JButton();
    back_bt = new javax.swing.JButton();

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

    jPanel1.setBackground(new java.awt.Color(0, 153, 153));

    add_bt.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N

```

```

add_bt.setText("Add");
add_bt.addActionListener(new java.awt.event.ActionListener()
{ public void actionPerformed(java.awt.event.ActionEvent evt) {

add_btActionPerformed(evt);

}

});
fileLst_cb.setModel(new
    javax.swing.DefaultComboBoxModel( new String[] { "" }));

jLabel2.setFont(new java.awt.Font("Trebuchet MS", 1, 18)); // NOI18N
jLabel2.setText("Select Files:-");

upload_bt.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
upload_bt.setText("Upload");
upload_bt.addActionListener(new java.awt.event.ActionListener() { public
void actionPerformed(java.awt.event.ActionEvent evt) {

upload_btActionPerformed(evt);

}

});

back_bt.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
back_bt.setText("Exit");

back_bt.addActionListener(new java.awt.event.ActionListener() {

public void actionPerformed(java.awt.event.ActionEvent evt) {

```

```

getContentPane().setLayout(layout); layout.setHorizontalGroup(

layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addComponent(jPanel1,    javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

);

layout.setVerticalGroup
( layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)

.addGroup(layout.createSequentialGroup()

.addComponent(jPanel1,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)

.addGap(0, 0, Short.MAX_VALUE))

)p;ack();

} // </editor-fold> //GEN-END:initComponents

private void add_btActionPerformed(java.awt.event.ActionEvent evt) { //GEN-
FIRST:event_add_btActionPerformed

// TODO add your handling code here:

file = com.nura.filehandling.FileBrowser.selectFile();
if (this.fileNamesLst.add(file.getName())) { //If file added then update combo box and list
selFiles.add(file);

DefaultComboBoxModel dcb = new DefaultComboBoxModel(); for (String fileName :
fileNamesLst) {

dcb.addElement(fileName);

```

```
java.util.logging.Logger.getLogger(UploadPageAdmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
} catch (IllegalAccessException ex) {
```

```
java.util.logging.Logger.getLogger(UploadPageAdmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
} catch (javax.swing.UnsupportedLookAndFeelException ex) {
```

```
java.util.logging.Logger.getLogger(UploadPageAdmin.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
}
```

```
//</editor-fold>
```

```
/* Create and display the form */
```

```
java.awt.EventQueue.invokeLater(new Runnable() {
```

```
public void run() {
```

```
new UploadPageAdmin().setVisible(true);
```

```
}
```

```
});
```

```
}
```

Upload Page:

```
        upload_btActionPerformed(java.awt.event.ActionEvent evt)    {//GEN-
private    void FIRST:event_upload_btActionPerformed
// TODO add your handling code here: try {

//Update the user details in the db first BufferedReader br = null;

String line = "";

String cvsSplitBy = "\\|";

UserDetailsDAOImpl _uDAOImpl = new UserDetailsDAOImpl();

    try {
        br = new BufferedReader(new FileReader(fileSelected.getAbsolutePath()));
        while ((line = br.readLine()) != null) {
            }

        } catch (FileNotFoundException e) {

            e.printStackTrace();

        } catch (IOException e) { e.printStackTrace();
        } catch (Exception ex) { ex.printStackTrace();

        } finally {
            if (br != null) { try {

                br.close();
```

```

    } catch (IOException e) { e.printStackTrace();

    }

}

}

Admin_FileProcess _af=new
    Admin_FileProcess();
    Drive service = GetGoogleDriveService.getDriveService(); GoogleDriveAccess
_gda = new GoogleDriveAccess();

    if(bank_cb.getSelectedItem().equals(constants.Constants.UPLOAD_MENU
[0])){
}

else if(bank_cb.getSelectedItem().equals(constants.Constants.UPLOAD_MENU[1])){

}

else if(bank_cb.getSelectedItem().equals(constants.Constants.UPLOAD_MENU[2])){

String part1_Loc
this.fileSelected.get      =      constants.Constants.FILE_UPLOAD_LOCATION      +
Name();

String part1Name = this.fileSelected.getName();

```

```

if(fileSelected_TF.getText().toString().startsWith(bank_cb.getSelectedItem().toString())){

    if ( !_af.GetFiledetails(bank_cb.getSelectedItem().toString()).equals(""))
        { System.out.println("Updating the existing file");

        _gda.updateFile(service, new File(part1_Loc), part1Name, FileDetails.getFileUrl());

        JOptionPane.showMessageDialog(this, "File updated successfully");

    } else {
        FileDetails _ff=new FileDetails(); System.out.println("Uploading file because not exist");
        String fileId1 = _gda.uploadFile(service, new File(part1_Loc), part1Name,
            bank_cb.getSelectedItem().toString());

        FileDetails fd = new FileDetails();

        _ff.setFileUrl(fileId1);

        if (new Admin_FileProcess().SetFiledetails(bank_cb.getSelectedItem().toString()))
            { JOptionPane.showMessageDialog(this, "File Uploaded ");

            fileSelected_TF.setText("");

            } else {
                JOptionPane.showMessageDialog(this, "Unable to upload. Retry after
sometime");

            }

        }

        JOptionPane.showMessageDialog(rootPane, "Category and File are Not
Matched..!");
    }else{

```

```

    }else{
    if(new
GoldValidateDao().Gold_purchase())
    { JOptionPane.showMessageDialog(rootPane, "Purchased success");

    }else{

JOptionPane.showMessageDialog(rootPane, "Declined");

    }
} //GEN-LAST:event_jButton3ActionPerformed

    }

private void jComboBox_jewelleryItemStateChanged(java.awt.event.ItemEvent evt) { //GEN-
FIRST:event_jComboBox_jewelleryItemStateChanged

// TODO add your handling code here:

} //GEN-LAST:event_jComboBox_jewelleryItemStateChanged

private void jComboBox_cashmodeItemStateChanged(java.awt.event.ItemEvent evt) { //GEN-
FIRST:event_jComboBox_cashmodeItemStateChanged

// TODO add your handling code here:

if(jComboBox_cashmode.getSelectedItem().toString().equals(constants.Constants.CASH_MO
D E[1])){
acc_no.setEnabled(true); pin.setEnabled(true);

}

```

```
java.util.logging.Logger.getLogger(Withdrawwindow.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
} catch (javax.swing.UnsupportedLookAndFeelException ex) {
```

```
java.util.logging.Logger.getLogger(Withdrawwindow.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
```

```
}    /* Create and display the form */  
    java.awt.EventQueue.invokeLater(new Runnable() {  
//</editor-fold>  
        public void run() {  
  
            new Withdrawwindow().setVisible(true);  
  
        }  
  
    });  
  
}
```

SCREEN SHOT

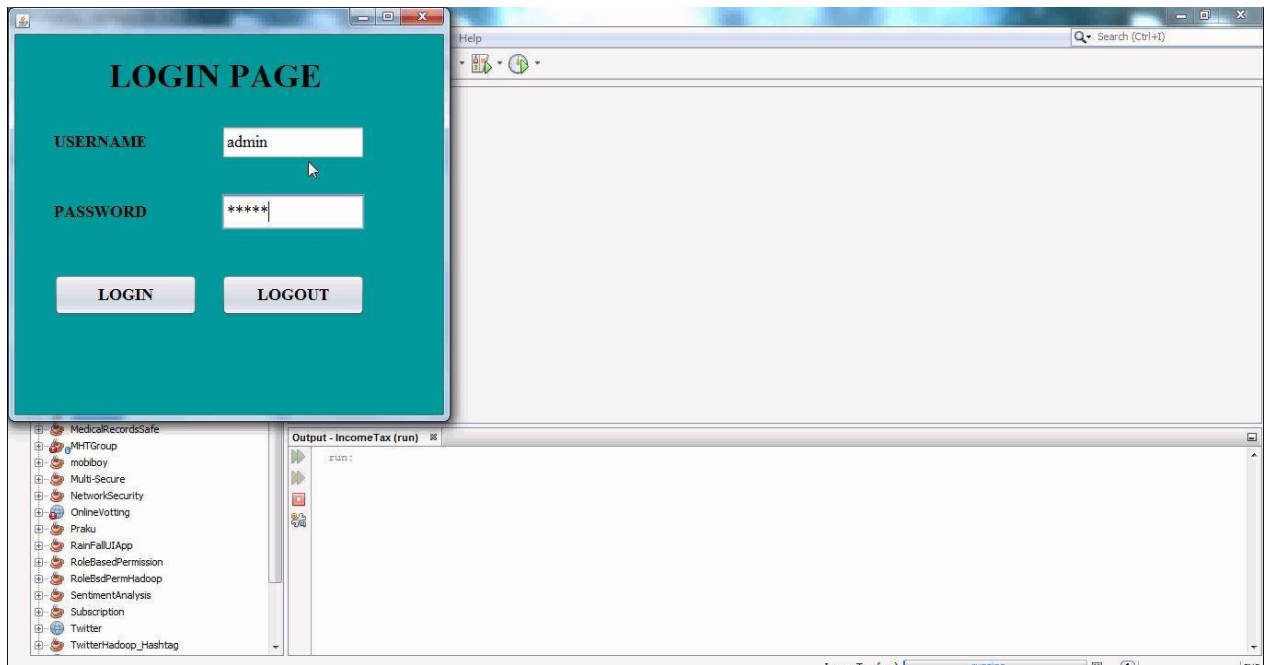


FIG 8.1 LOGIN PAGE

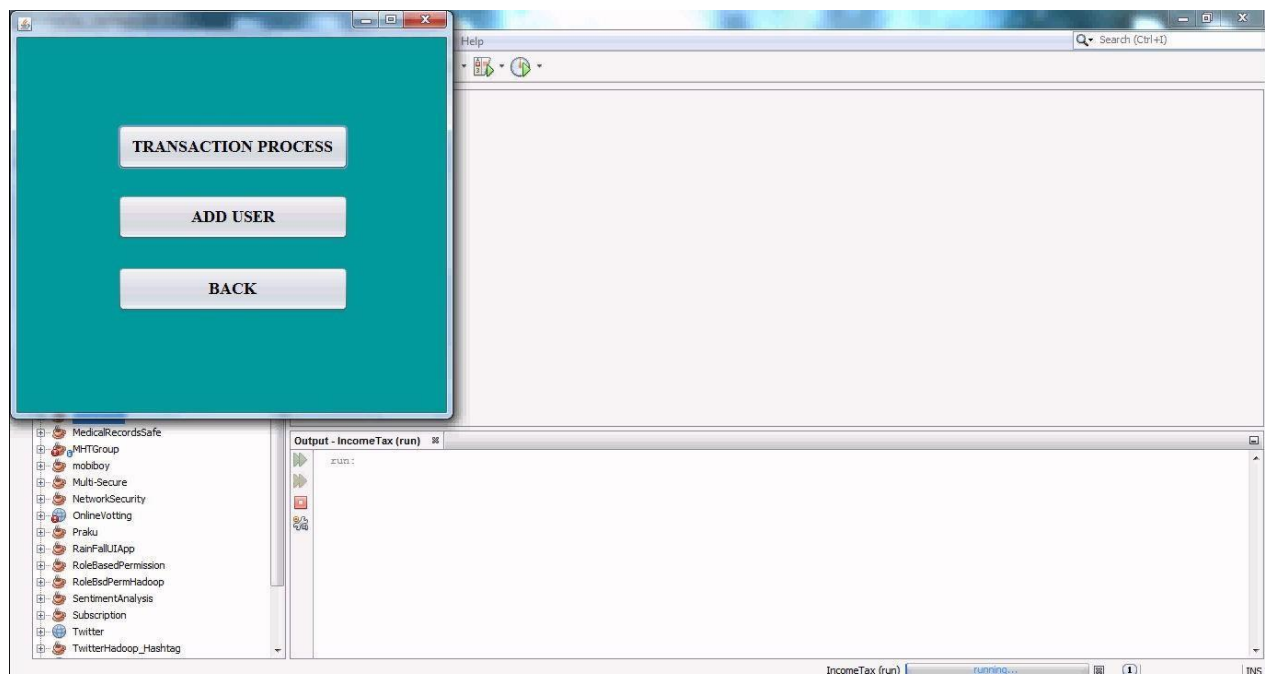


FIG 8.2 TRANSACTION AND ADD USER PAGE

USER REGISTRATION

NAME	siva
USER NAME	siva
PASSWORD	siva
AADHAAR NUMBER	123456789123
CONTACT NUMBER	9790930
EMAIL	
ADDRESS	
CITY	

REGISTER **BACK**

FIG 8.3 USER REGISTRATION

ADD BANK DETAILS

BANK NAME	INDIAN BANK
PIN	12345
AMOUNT	100000

ADD **BACK**

FIG 8.4 ADD BANK DETAILS

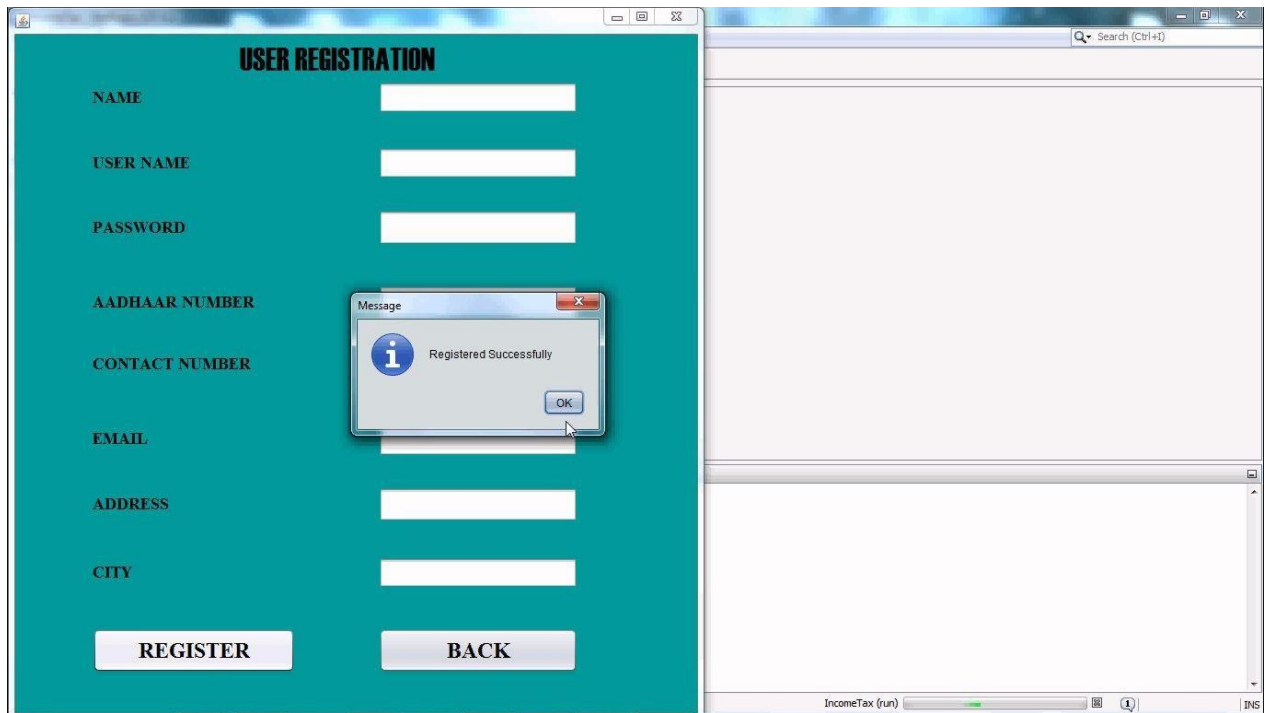


FIG 8.5 USER REGISTRATION

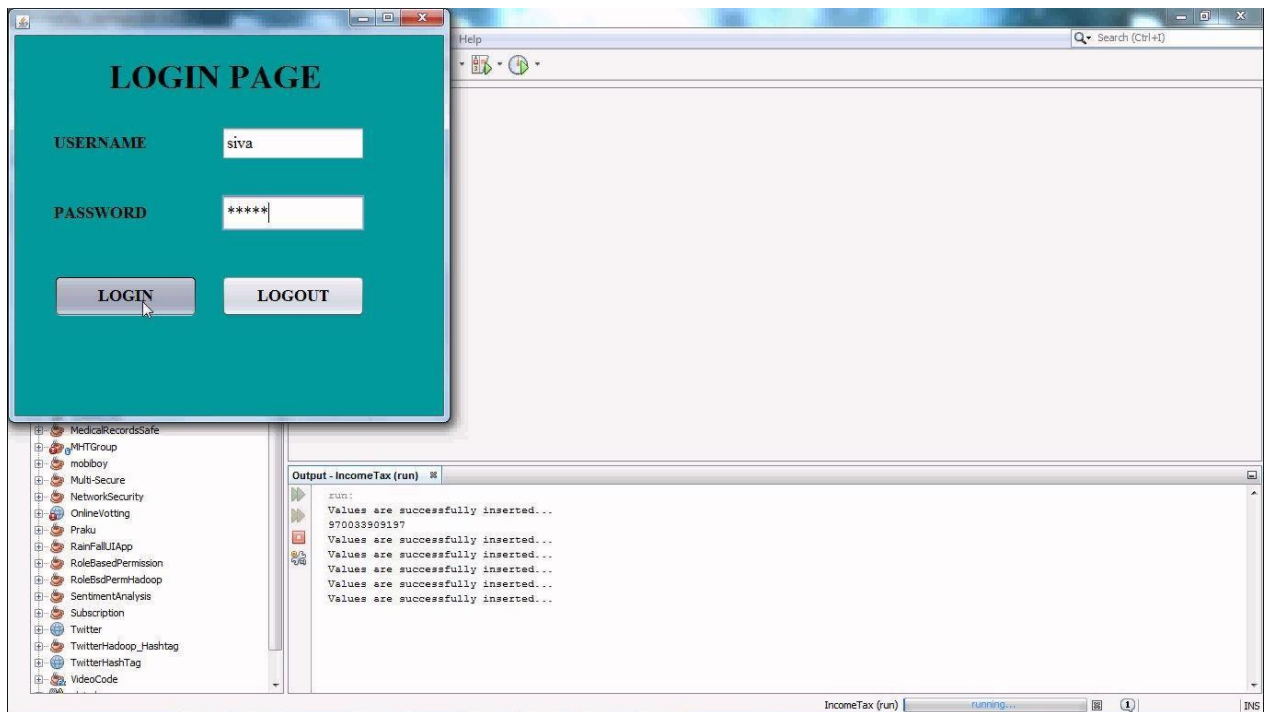


FIG 8.6 LOGIN AND LOGOUT PAGE

Land Registration

SELLER-AADHAAR NO :- 123456789987

BUYER-AADHAAR NO :- 123456789123

LAND-LOCATION :- T.NAGAR

LAND-SQ FEET :-

LAND DOC -NO :-

PROPERTY-VALUE :-

LAND-PRICE :-

TRANSACTION MODE :- CASH

ACC-NO :-

PIN :-

REGISTER CLEAR BACK

IncomeTax (run) running... INS

FIG 8.7 LAND REGISTRATION PAGE

BANKING LAND AND GOLD

BANKING

LAND REGISTRATION

CASH TRANSACTIONS

GOLD PURCHASE

BACK

IncomeTax (run) running... INS

Praku
RainFallUIApp
RoleBasedPermission
RoleBasedPermHadoop
SentimentAnalysis
Subscription
Twitter
TwitterHadoop_HashTag
TwitterHashTag
VideoCode

970033505197
Values are successfully inserted...
Values are successfully inserted...
Values are successfully inserted...
Values are successfully inserted...
Values are successfully inserted...
Values are successfully inserted...

FIG 8.6 BANKING LAND AND GOLD

CHAPETER 9

CONCLUSION

9.1 CONCLUSION

To address the rising sophistication of forgery, especially in high-value transactions like gold purchases over \$20,000, a robust tracking system is proposed. Leveraging blockchain technology offers a promising solution. By recording transactions securely and transparently, blockchain ensures authenticity at each stage. This decentralized approach enhances security and instills confidence in stakeholders, effectively combating fraud in modern financial transactions.

CHAPTER 10

REFERENCE

- [1] "History of Anti-Money Laundering Laws". United States Department of the Treasury. 30 June 2015. Retrieved 30 June 2015. <https://www.fincen.gov/history-anti-money-laundering-laws>
- [2] "Money Laundering Act 2012 amended". Resource Portal of OGR Legal. OGR Legal. Retrieved 2 November 2015. <https://resource.ogrlegal.com/money-laundering-act-2012-amended/>
- [3] "OPSI: Proceeds of Crime Act". Retrieved 14 February 2009. http://www.opsi.gov.uk/acts/acts2002/ukpga_20020029_en_1
- [4] "Suspicious Money Laundering Transaction: Q&A". Anti-Money Laundering Division, Investigation Bureau, Ministry of Justice, Taiwan. Retrieved 20 November 2017. <https://goo.gl/jbu6Ln>
- [5] "Money Laundering". Wikipedia. Retrieved 18 January 2018. https://en.wikipedia.org/wiki/Money_laundering
- [6] "Suspicious Money Laundering or Terrorist Trading" Anti-Money Laundering Division, Investigation Bureau, Ministry of Justice, Taiwan. Retrieved 30 January 2018. <https://goo.gl/SfDrvC>
- [7] "Don't Trust Your Instincts-Benford's law" PanSci. Retrieved 25 December 2017. <https://pansci.asia/archives/54264>
- [8] "Money Laundering Control Act". Wikipedia. Retrieved 25 May 2019. https://en.wikipedia.org/wiki/Money_Laundering_Control_Act
- [9] Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System. Retrieved from <https://bitcoin.org/bitcoin.pdf>
- [10] Tapscott, D., & Tapscott, A. (2016). Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world. Penguin.
- [11] Swan, M. (2015). Blockchain: Blueprint for a new economy. " O'Reilly Media, Inc."
- [12] Raval, S. (2016). Decentralized Applications: Harnessing Bitcoin's Blockchain Technology. " O'Reilly Media, Inc."
- [13] Antonopoulos, A. M. (2014). Mastering Bitcoin: Unlocking Digital Cryptocurrencies. " O'Reilly Media, Inc."