Kristu Jayanti College

AUTONOMOUS Bengaluru

Reaccredited A++ Grade by NAAC | Affiliated to Bengaluru North University

LOFO

(LOST AND FOUND SYSTEM)

Project Report submitted in partial fulfillment of the requirements

for the reward of the degree of

BACHELOR OF COMPUTER APPLICATIONS (BCA)

Submitted by

U S ABHIRAM

(21BCAE10)

Under the guidance of

Prof. PRATHAP G

DEPARTMENT OF COMPUTER SCIENCE (UG)

BCA PROGRAMME

KRISTU JAYANTI COLLEGE (AUTONOMOUS)

K. NARAYANAPURA, KOTHANUR P.O., BENGALURU – 560077

Kristu Jayanti College
AUTONOMOUS Bengaluru
Reaccredited A++ Grade by NAAC | Affiliated to Bengaluru North University

DEPARTMENT OF COMPUTER SCIENCE (UG)

CERTIFICATE OF COMPLETION

This is to certify that the project entitled **LOFO** has been satisfactorily completed by **U S Abhiram (21BCAE10)** in partial fulfillment of the award of the Bachelor of Computer Applications degree requirements prescribed by Kristu Jayanti College (Autonomous), Bengaluru (Affiliated to Bengaluru North University) during the academic year 2022 -23.

Internal Guide ***Head of the Department***

Valued by Examiners

1: _____ ***Centre:*** **Kristu Jayanti College**

2: _____ ***Date:***

Kristu Jayanti College
AUTONOMOUS Bengaluru
Reaccredited A++ Grade by NAAC | Affiliated to Bengaluru North University

DECLARATION

I, U S ABHIRAM (21BCAE10) hereby declare that the project work entitled **LOFO** is an original project work carried out by me, under the guidance of Prof.PRATHAP G.

This project work has not been submitted earlier either to any University / Institution or any other body for the fulfillment of the requirement of a course of study.

Signature

U S ABHIRAM
21BCAE10

Bengaluru

Date:

ACKNOWLEDGEMENT

The success of the project depends upon the efforts invested. It's my duty to acknowledge and thank the individuals who has contributed in the successful completion of the project.

I take this opportunity to express my profound and whole hearted thanks to

Rev. Fr. Dr. AUGUSTINE GEORGE, PRINCIPAL and Rev. Fr. LIJO P THOMAS, VICE PRINCIPAL and CHIEF FINANCIAL OFFICER, KRISTU JAYANTI COLLEGE (AUTONOMOUS), BENGALURU for providing ample facilities made to undergo my project successfully.

I express my deep sense of gratitude and sincere thanks to **Prof. SEVUGA PANDIAN A**, **HEAD, DEPARTMENT OF COMPUTER SCIENCE** and **Dr. CALISTUS JUDE AL, DEAN, FACULTY OF SCIENCES** for their valuable advice.

I feel immense pleasure to thank my respected guide **Prof. PRATHAP G** for sustaining interest and providing dynamic guidance in aiding me to complete this project immaculately and impeccably and for being the source of my strength and confidence.

It is my duty to express my thanks to all teaching and non-teaching staff members of the computer science department who offered me help directly or indirectly by their suggestions. The successful completion of my project would not have been possible without my parent's sacrifice, guidance and prayers. I take this opportunity to thank everyone for their continuous happiness and agony.

Last but not the least I thank almighty God for giving me strength and good health throughout my project and enabling me to complete it successfully.

ABSTRACT

LOFO is a Windows-based lost and found application designed specifically for your campus. The application serves as a platform for students, faculty, and staff to report lost items or search for their lost belongings. When a user finds an item that has been lost by someone else, they can upload the item on LOFO with all the details such as found location, image etc. People looking for their lost item can go through the lost item catalog and claim it by answering the claim questionnaire. If the claim is accepted, you will receive notification of contact information in your mail. If your claim is rejected, you can claim again with more accurate details to prove your ownership of the item. The application is built using vb.net framework as the frontend and SQLite database as the backend. The application also has a few security features like password hashing. When an admin registers new users to the application by adding respective username and password the password is not stored as plain text, but they are encrypted and stored as hashes, now even an admin who has privileges to see the values stored in the database he cannot see the password text but sees the hashed text. While the user logins and the credentials are authenticated the entered password in textbox is again encrypted and compared with the stored hashed string, If both of them match then the user is allowed into the app.

All in all, the app solves the problem of misplacing items in the campus and never seeing them again. The pain of going through lot of manual work to find your way back to the item or the item finding its way back to you is made simpler with this application. The SMTP used will help you to get the contact information of the person who has found your item quickly. This is the case only if someone has already found your lost item but incase no one has found your item, you can add items to the log details in our app. When an item matching your log description matches you will be notified by the admin through the mail id given during uploading the log. This way any items lost in the campus is more probable reaching back to its owner.

TABLE OF CONTENTS

Sl. No	Topic	Page No
1	Introduction	1-3
1.1	Problem Definition	1
1.2	Scope of the Project	1
1.3	Modules in the Project	2-3
2	System Study	4-5
2.1	Existing System	4
2.2	Feasibility Study	4
2.3	Proposed System	5
3	System Design	6-37
3.1	ER Diagram	6-12
3.2	DFD[lev0, lev1]	12-16
3.3	Gantt Chart	16-20
3.4	Input / Output Design	21-37
4	System Configuration	38
4.1	Hardware Requirements	38
4.2	Software Requirements	38
5	Details of Software	39-43
5.1	Overview of Frontend	40
5.2	Overview of Backend	41-42
5.3	About the Platform	42-43
6	Testing	44-46
7	Conclusion and Future Enhancement	47
8	Bibliography	48-50
9	APPENDICES A-Table Structure	51-52
10	APPENDICES B-Screenshots	53-57
11	APPENDICES C-Sample Report of test cases	58

LIST OF FIGURES

Fig.no	Description	Page No
1	Er diagram for LOFO	11
2	DFD level-0 for LOFO	15
3	DFD level-1for LOFO	16
4	Gantt chart for LOFO	20
5	login	53
6	User homepage	53
7	Add log page	54
8	Add items	54
9	Find items	55
10	Admin homepage	55
11	Claim item	56
12	Resolve claims	56
13	Resolve logs	57

LIST OF TABLES

Table No.	Description	Page No
01	Hardware requirement	38
02	Software requirement	38
03	User table	51
04	Logs table	51
05	Found items table	52
06	Claims table	52
07	Sample report of test cases	58

1.INTRODUCTION

1.1 PROBLEM DEFINITION:

LOFO is a Windows-based lost and found application designed specifically for your campus. The application serves as a platform for students, faculty, and staff to report lost items or search for their lost belongings. When a user finds an item that has been lost by someone else, they can upload the item on LOFO with all the details such as found location, image etc. People looking for their lost item can go through the lost item catalog and claim it by answering the claim questionnaire. If the claim is accepted, you will receive notification of contact information in your mail.

1.2 SCOPE OF THE PROJECT:

- Campus-wide implementation: You could potentially roll out LOFO to the entire campus, which would require integration with existing campus systems and infrastructure, as well as extensive testing and training.

- Additional features: Depending on the needs of your users, you could consider adding features such as a chat function to connect lost item owners and finders, a map view to track lost item locations, or integration with social media platforms to increase visibility.

- Mobile app development: You could consider expanding LOFO to include a mobile app version, which would require additional development and testing.

- Integration with other institutions: If successful, LOFO could potentially be scaled to other institutions or organizations, such as other universities, airports, or large corporations.

1.3 MODULES IN THE PROJECT:

- User Authentication
- Add Items
- Find Items
- Claim Item
- Log Items
- Resolve Claims
- Resolve Logs
- Add new users/admins and #password.

- User Authentication:

  Here the username and password entered by the user is verified with the values stored in the database. If the credentials match, then the user-type field is verified and redirected based on that value. If the user-type is ‘user’ then user-home page is shown, else ‘admin’- homepage is shown.
- Add Items:

  People can upload items which they have found on campus that belong to people who have lost it. They can go to this ‘Add items’ section and fill out all the details such as their contact number, found location, item image etc.
- Find Items:

  If you have lost an item and wish to find it , you can visit ‘Find Items’ section where there are items posted by various people. You can filter according to item category or lost date. If you find an item that you believe is yours, you can click on that and go to the claim module.
- Claim Item:

  Enter the details to prove the ownership of the item you are trying to claim. Fill in all the fields like the description and your contact details. Once the admin accepts/ rejects your claim you will be notified through the SMTP to your given mail address.

- Log Items:

  If you have lost an item and were not able to find that item in the 'Find items' sections, then you can log that item in the database with the description of the lost item. You can provide an email address, and upon finding an item matching your description you will be notified through SMTP.

- Resolve Claims:

  This is an admin side module which allows the admins to see all the claims that are been posted by the user. Based on the uploader description and the claimant description, admin can either accept or reject the claim. The claimant will be mailed a response accordingly.

- Resolve Logs:

  This is also an admin side module, All the logs uploaded by the user are visible to the admin whenever a matching item is found, He can say 'item found' upon which the user who had logged that item gets notified through mail.

2.SYSTEM STUDY

2.1 EXISTING SYSTEM STUDY:

There is no existing digital system for lost and found on the campus. All the procedures involved are taken manually without any digital interference.

Some of the main problems in the Existing system is:

- It is time consuming.
- No digital data available for further reference.
- Manual work involved is more.
- No centralized system.

2.2 FEASIBILITY STUDY:

A feasibility study is an analysis of how successfully a project can be completed, accounting for factors that affect it such as economic, technological, legal and scheduling factors. Project managers use feasibility studies to determine potential positive and negative outcomes of a project before investing a considerable amount of time and money into it.

A feasibility study tests the visibility of an idea, a project or even a new business. The goal of a feasibility study is to place emphasis on potential problems that could occur if a project is pursued if, after all significant factors are considered, the project should be pursued.

The project **"LOFO"** can be designed and developed using .Net framework. The content and language are feasible to use, and the portal can be developed based on the requirements.

COMPONENTS:

- Technical feasibility
- Operational feasibility
- Schedule feasibility.

Every project is feasible for given unlimited resources and infinitive time. Feasibility study is an evaluation of the proposed system regarding its workability, impact on the organization, ability to meet user needs and effective use of resources. Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development. Feasibility and risk analysis are related in many ways. The feasibility analysis in this project has been discussed below based on the above-mentioned components of feasibility.

2.2.1 SCHEDULE:

Time duration of this project requires 66 days covering 36 days for initiation phase, 6 days for Design phase, 12 days for Development phase, 06 days for implementation phase, 2 days for Testing phase and 2 days for documentation. The resource person is U S Abhiram.

2.2.2 OPERATIONAL:

- Drastic reduction of paperwork.
- User information management.
- Easy to handle.
- Lastly it is secure and reliable.

2.3 PROPOSED SYSTEM:

The proposed system has got the following advantages over the existing system.

- Faster claim resolve.
- Easy to use.
- Item details are stored in the database for further study.
- Secure and reliable.
- Centralized platform for all.

3.SYSTEM DESIGN

In the design phase the architecture is established. This phase starts with the requirement document delivered by the requirement phase and maps the requirements into an architecture. The architecture defines the components, their interfaces and behaviors. The deliverable design document is architecture. The design document describes a plan to implement the requirements. This phase represents the ``how'' phase. Details on computer programming languages and environments, machines, packages, application architecture, distributed architecture layering, memory size, platform, algorithms, data structures, global type definitions, interfaces, and many other engineering details are established.

The design may include the usage of existing components. Analyzing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. The architecture team also converts the typical scenarios into a test plan. In our approach, the team, given a complete requirement document, must also indicate critical priorities for the implementation team. A critical implementation priority leads to a task that has to be done right. If it fails, the product fails. If it succeeds, the product might succeed. At the very least, the confidence level of the team producing a successful product will increase. This will keep the implementation team focused. Exactly how this information is conveyed is a skill based on experience more than a science based on fundamental foundations. System design is the process of defining the architecture components, modules, interfaces, and data for a system to satisfy specified requirements .Systems design could be seen as the application of systems theory to product development.
There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user. Until the 1990s, systems design had a crucial and respected role in the data processing industry.

In the 1990s, standardization of hardware and. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering. Object-oriented analysis and design methods are becoming the most widely used methods for computer systems design. UML has become the standard language in object-oriented analysis and design. It is widely used for modelling software systems and is increasingly used for high designing non software systems and organizations.

3.1 ARCHITECTURAL DESIGN:

The architectural design of a system emphasizes the design of the system architecture that describes the structure, behavior and more views that system and analysis.

3.1.1 LOGICAL DESIGN:

The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system. This is often conducted via modelling, using an over-abstract (and sometimes graphical) model of the actual system. In the context of systems, designs are included. Logical design includes entity-relationship diagrams (ER diagrams).

3.1.2 PHYSICAL DESIGN:

The physical design relates to the actual input and output processes of the system. This is explained in terms of how data is input into a system, how it is verified/authenticated, how it is processed, and how it is displayed. In physical design, the following requirements about the system are decided.

- Input requirement
- Output requirement
- Storage Requirement
- Processing Requirement
- User interface Design.
- Data design.
- Process design.

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system. Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system. At the end of the system design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase.

To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc. It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc. It

involves a detailed design of a user and a product database structure processor and a control processor. The H/S personal specification is developed for the proposed system.

3.2 ER DIAGRAM:

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases. At first glance an entity relationship diagram looks very much like a flowchart. It is the specialized symbols, and the meanings of those symbols, that make it unique. Because this ER tutorial focuses on beginners.

Below are some tips that will help you build effective ER diagrams:

- An entity should appear only once in a particular diagram.
- Provide a precise and appropriate name for each entity, attribute, and relationship in the diagram.
- Terms that are simple and familiar always beat vague, technical-sounding words.
- In naming entities, remember to use singular nouns. However, adjectives may be used to distinguish entities belonging to the same class (for example part-time employee and full -time employee).
- Meanwhile attribute names must be meaningful, unique, system independent, and easily understandable.
- Remove vague, redundant or unnecessary relationships between entities.
- Never connect a relationship to another relationship.
- Make effective use of colors.
- You can use colors to classify similar entities or to highlight key areas in your diagrams.
- You can draw entity relationship diagrams manually, especially when you are just informally showing simple systems to your peers.

However, for more complex systems and for external audiences, you need diagramming software such as Creately’s to craft visually engaging and precise ER diagrams.

- The ER Diagram Software offered by Creately as an online service is pretty easy to use and is a lot more affordable than purchasing licensed software.
- It is also perfectly suited for development teams because of its strong support for collaboration.

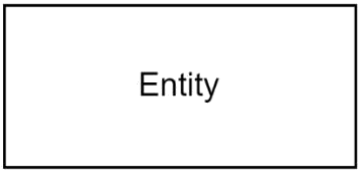
3.2.1 THE HISTORY OF ENTITY RELATIONSHIP DIAGRAMS:

Peter Chen developed ERDs in 1976. Since then, Charles Bachman and James Martin have added some slight refinements to the basic ERD principles.

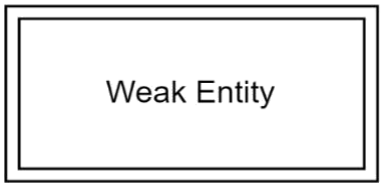
3.2.2 STRUCTURE OF ENTITY RELATIONSHIP DIAGRAM WITH COMMON ERD:

Notations: An entity relationship diagram is a means of visualizing how the information a system produces is related. There are five main components of an ERD:

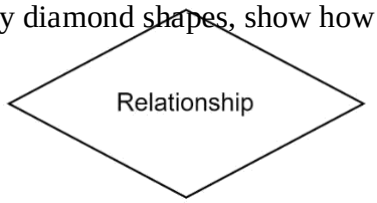
Entities: which are represented by rectangles. An entity is an object or concept about which you want to store information.



A weak entity is an entity that must be defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.

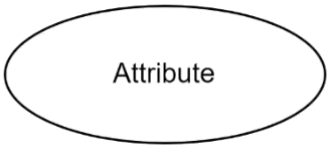


Actions: which are represented by diamond shapes, show how two entities share information in the database.

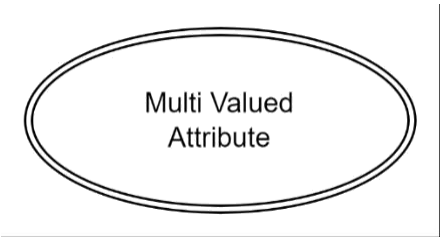


In some cases, entities can be self-linked. For example, employees can supervise other employees.

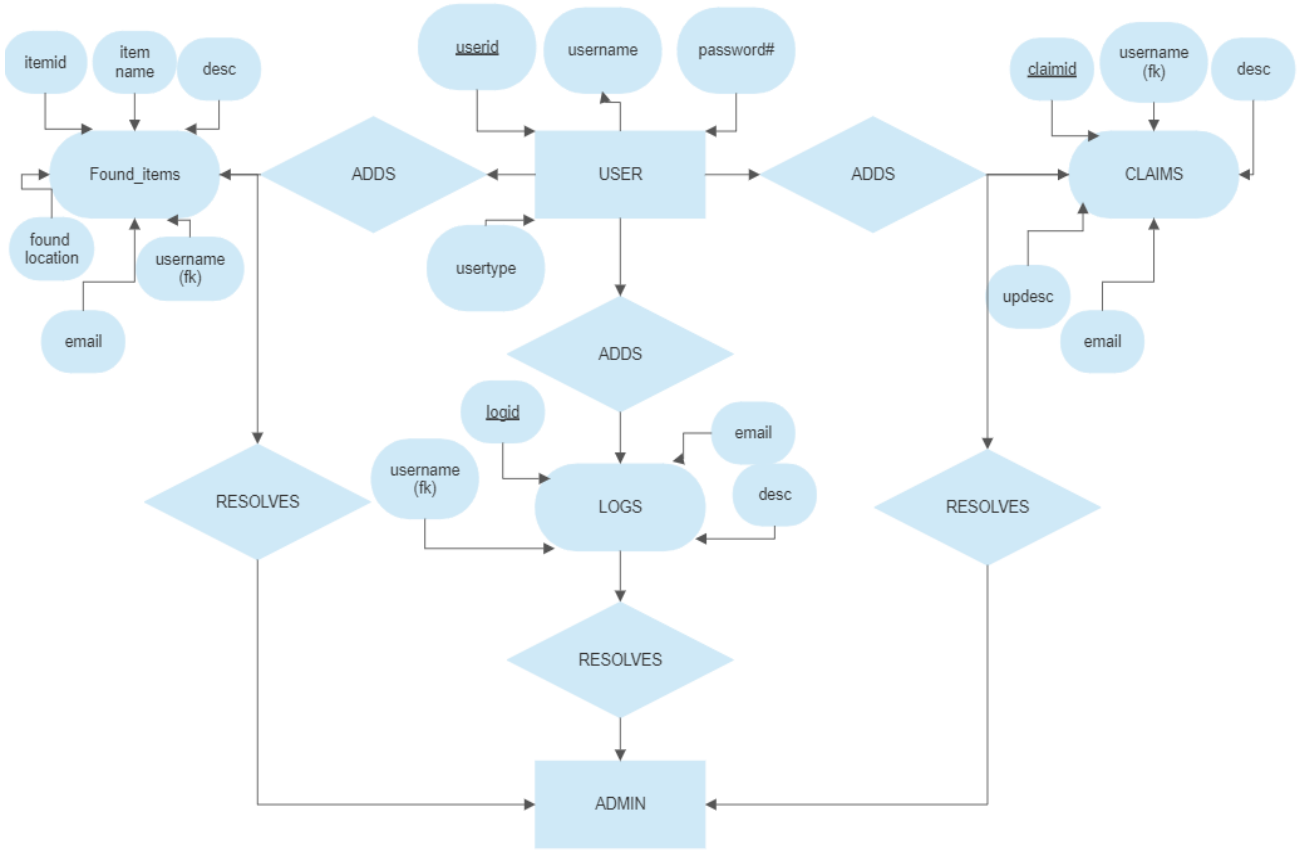
Attributes: which are represented by ovals. A key attribute is the unique, distinguishing characteristics of the entity. For example, an employee's social security number might be the employee's key attribute.



A multivalued attribute: can have more than one value. For example, an employee entity can have multiple skill values.



ER DIAGRAM FOR LOFO: e-LOST AND FOUND



3.3 DATA FLOW DIAGRAM (LEVEL 0 AND LEVEL 1):

The Data Flow Diagrams (DFDs) are used for structure analysis and design. DFDs show the flow of data from external entities into the system. DFDs also show how the data moves and is transformed from one process to another, as well as its logical storage. The following symbols are used within DFDs. For clarity, a key has been provided at the bottom of this page.

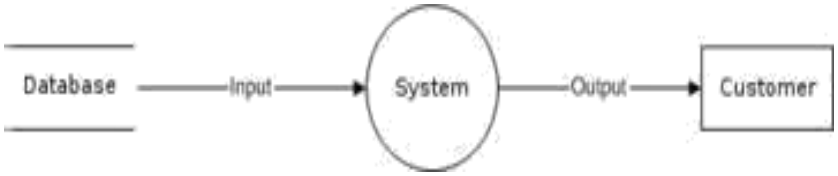
A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).

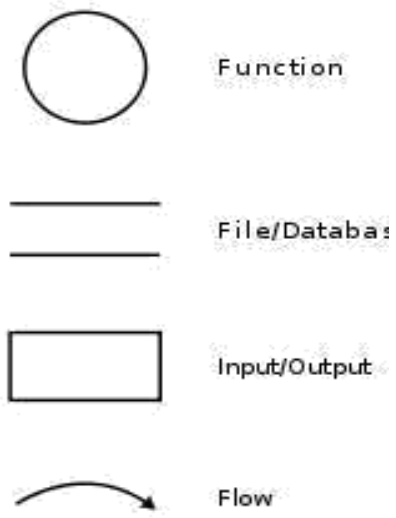
3.3.1 HISTORY

Larry Constantine, the original developer of structured design, based on Martin and Estrin's"Data Flow Graph" model of computation. Starting in the 1970s, data flow diagrams (DFD) became a popular way to visualize the major steps and data involved in software system processes. DFDs were usually used to show data flow in a computer system, although they could in theory be applied to business process modelling. DFD were useful to document the major data flows or to explore a new high-level design in terms of data flow.

3.3.2 THEORY:



3.3.3 DATA FLOW DIAGRAM EXAMPLE:



The Data Flow Diagrams (DFDs) are used for structure analysis and design. DFDs show the flow of data from external entities into the system. DFDs also show how the data moves and is transformed from one process to another, as well as its logical storage. The following symbols are used within DFDs. For clarity, a key has been provided at the bottom of this page.

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design)

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).

Data flow diagrams are one of the three essential perspectives of the structured-systems analysis and design method. The sponsor of a project and the end users will need to be briefed and consulted throughout all stages of a system's evolution. With a data flow diagram, users can visualize how the system will operate, what the system will accomplish, and how the system will be implemented.

The old system's data flow diagrams can be drawn up and compared with the new system's data flow diagrams to draw comparisons to implement a more efficient system.

Data flow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately influences the structure of the whole system from order to dispatch to report. How any system is developed can be determined through a data flow diagram model.

While developing a set of levelled data flow diagrams the analyst/designer is forced to address how the system may be decomposed into component sub-systems, and to identify the transaction data in the data model. Data flow diagrams can be used in both Analysis and Design phase of the SDLC.
There are different notations to draw data flow diagrams (Yourdon & Coad and Game & Sarson), defining different visual representations for processes, data stores, data flow, and external entities.

3.3.4 PHYSICAL VS LOGICAL DFD:

A logical DFD captures the data flows that are necessary for a system to operate. It describes the processes that are undertaken, the data required and produced by each process, and the stores needed to hold the data. On the other hand, a physical DFD shows how the system is implemented, either now (Current Physical DFD), or how the designer intends it to be in the future (Required Physical DFD).

Thus, a Physical DFD may be used to describe the set of data items that appear on each piece of paper that moves around an office, and the fact that a particular set of pieces of paper are stored together in a cabinet. It is quite possible that a Physical DFD will include references to data that are duplicated, or redundant, and that the data stores, if implemented as a set of database tables, would constitute an un-normalized (or demoralized) relational database. In contrast, a Logical DFD attempts to capture the data flow aspects of a system in a form that has neither redundancy nor duplication.

3.3.5 DATA FLOW SYMBOLS AND THEIR MEANINGS: - An entity- A

source of data or a destination for data.

Source/Sink: Represented by rectangles in the diagram. Sources and Sinks are external ,Entities which are sources or destinations of data, respectively.

Process: Represented by circles in the diagram. Processes are responsible for manipulating They take data as input and output an altered version of the data.

Data Store: Represented by a segmented rectangle with an open end on the right. Data Stores Are both electronic and physical locations of data. Examples include databases, directories, files, and even filing cabinets and stacks of paper.

When drawing Level-0 DFD's, we must first identify the process, all the external entities and all the data flows. We must also state any assumptions we make about the system. It is advised that we draw the process in the middle of the page. We then draw our external entities in the corners and finally connect our entities to our process with the data flows.

LEVEL ZERO DIAGRAM FOR LOFO:

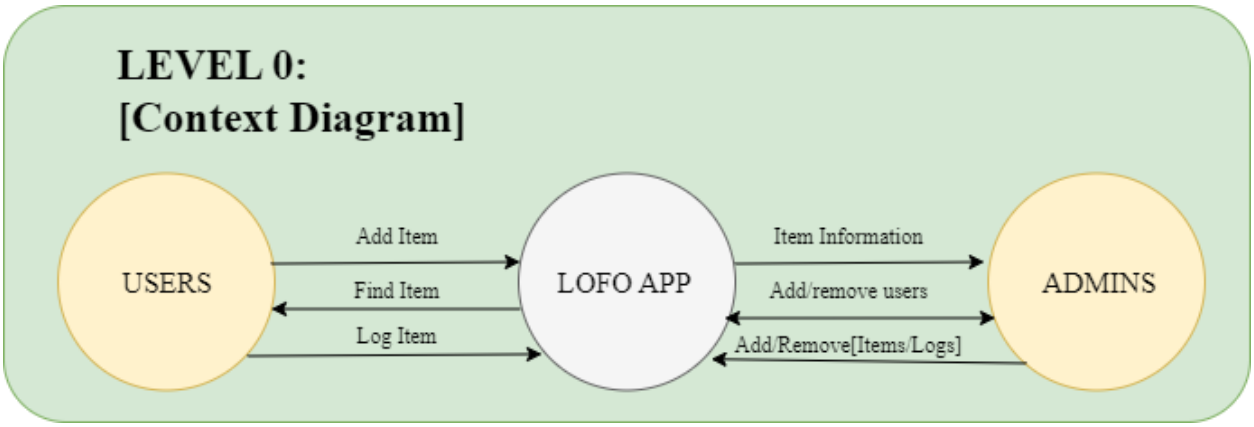


Fig 2 LEVEL 0 DFD

3.3.6 LEVEL 1 DFD's:

Level 1 DFD's aim is to give an overview of the full system. They look at the system in more detail. Major processes are broken down into subprocesses. Level 1 DFD's also identifies data stores that are used by the major processes. When constructing a Level 1 DFD, we must start by examining the Context Level DFD. We must break up the single process into its subprocesses.

We must then pick out the data stores from the text we are given and include them in our DFD. Like the Context Level DFD's, all entities, data stores and processes must be labelled. We must also state any assumptions made from the text.

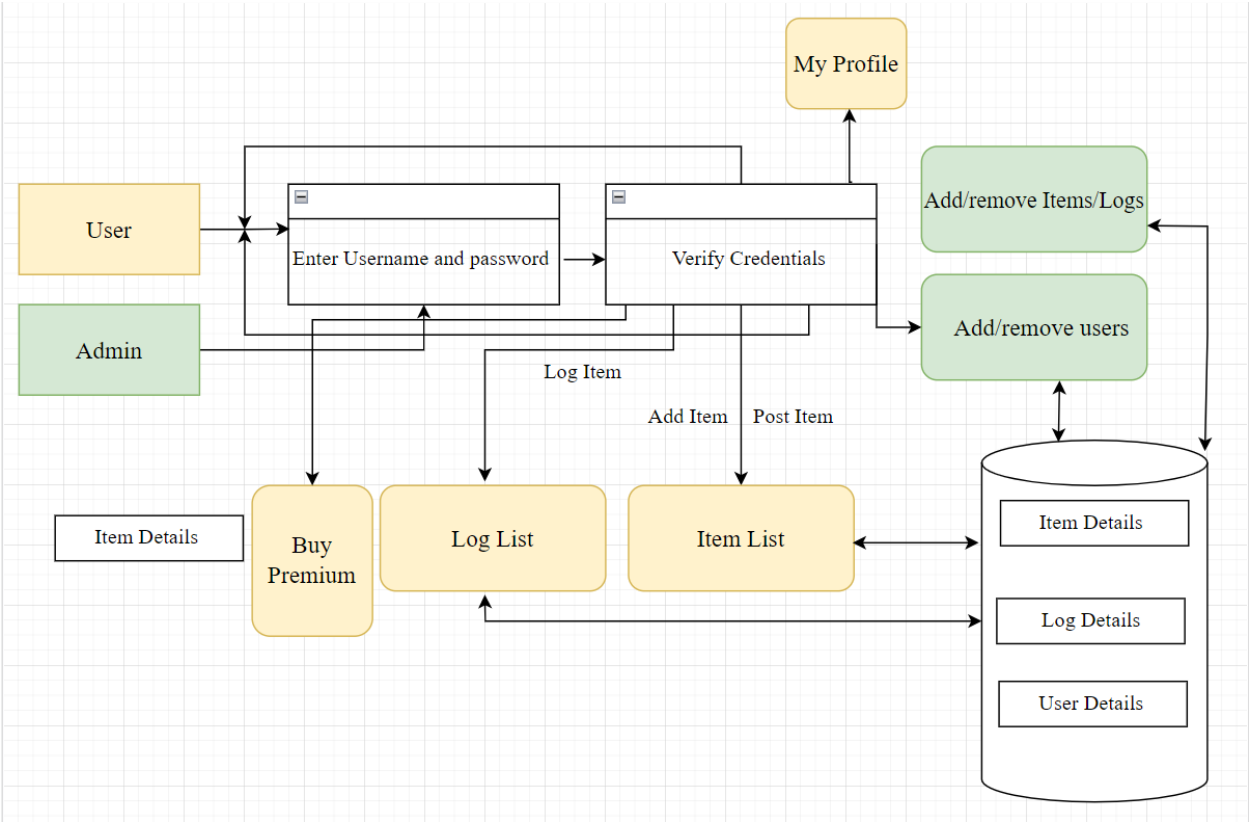


Fig 3 Level 1 DFD

3.4 GANTT CHART:

A Gantt chart is a type of bar chart, devised by Henry Gantt in the 1910s, that illustrates a project schedule. Gantt charts illustrate the start and finish dates of the terminal elements and summary elements of a project.

Terminal elements and summary elements comprise the work breakdown structure of the project.
Modern Gantt charts also show the dependency (i.e., precedence network) relationships between activities. Gantt charts can be used to show current schedule status using percent-complete shadings and a vertical "TODAY" line as shown here.

Although now regarded as a common charting technique, Gantt charts were considered revolutionary when first introduced. This chart is also used in information technology to represent data that has been collected.

3.4.1 HISTORICAL DEVELOPMENT:

The first known tool of this type was developed in 1896 by Karol Adamic, who called it a Har monogram, Adamic did not publish his chart until 1931, however, and only in Polish, which limited both its adoption and recognition of his authorship.

The chart is named after Henry Gantt (1861– 1919), who designed his chart around the years 1910– 1915.One of the first major applications of Gantt charts was by the United States during World War I, at the instigation of General William Crozier in the 1980s, personal computers allowed widespread creation of complex and elaborate Gantt charts. The first desktop applications were intended mainly for project managers and project schedulers. With the advent of the Internet and increased collaboration over networks at the end of the 1990s, Gantt charts became a common feature of web-based applications, including collaborative groupware.

3.4.2 GANTT CHART BENEFITS:

CLARITY: One of the biggest benefits of a Gantt chart is the tool's ability to boil down multiple tasks and timelines into a single document. Stakeholders throughout an organization can easily understand where teams are in a process while grasping the ways in which independent elements come together toward project completion.

COMMUNICATION: Teams can use Gantt charts to replace meetings and enhance other status updates. Simply clarifying easy, visual method to help team members understand task progress.

MOTIVATION: Some teams or team members become more effective when faced with a form of external motivation. Gantt charts offer teams the ability to focus work at the front of a task timeline, or at the tail end of a

chart segment. Both types of team members can find Gantt charts meaningful as they plug their own work habits into the overall project schedule.

COORDINATION: For project managers and resource schedulers, the benefits of a Gantt chart include the ability to sequence events and reduce the potential for overburdening team members. Some project managers even use combinations of charts to break down projects into more manageable sets of tasks.

CREATIVITY: Sometimes, a lack of time or resources forces project managers and teams to find creative solutions. Seeing how individual tasks intertwine on Gantt charts often encourages new partnerships and collaborations that might not have evolved under traditional task assignment systems.

TIME MANAGEMENT: Most managers regard scheduling as one of the major benefits of Gantt charts in a creative environment. Helping teams understand the overall impact of project delays can foster stronger collaboration while encouraging better task organization.

FLEXIBILITY: Whether you use Excel to generate Gantt charts or you load tasks into a more precise chart generator, the ability to issue new charts as your project evolves lets you react to unexpected changes in project scope or timeline. While revising your project schedule too frequently can eliminate some of the other benefits of Gantt charts, offering a realistic view of a project can help team members recover from setbacks or adjust to other changes.

MANAGEABILITY: For project managers handling complex assignments, like software publishing or event planning, the benefits of Gantt charts include externalizing assignments. By visualizing all the pieces of a project puzzle, managers can make more focused, effective decisions about resources and timetables.

EFFICIENCY: Another one of the benefits of Gantt charts is the ability for teams' members to leverage each other's deadlines for maximum efficiency. For instance, while one team member waits on the outcome of three other tasks before starting a crucial piece of the assignment, he or she can perform other project tasks. Visualizing resource usage during projects allows managers to make better use of people, places, and things.

ACCOUNTABILITY: When project teams face major organizational change, documenting effort and outcomes becomes crucial to career success. Using Gantt charts during critical projects allows both project managers and participants to track team progress, highlighting both big wins and major failures. During professional review periods, team members who frequently exceed expectations can leverage this documentation into larger raises or bonuses.

3.4.3 GANTT CHART IMPORTANCE:

The project's summary and terminal elements, which combine to form the project's internal structure, are shown on the Gantt chart. Many charts will also depict the precedence rankings and dependencies of various tasks within the project. The charts can illustrate the start and finish project terminal elements in project management.

It can also show summary elements and terminal dependencies. The smallest task tracked as part of the project effort is known as a terminal element. Gantt chart represents the tasks in most modern project scheduling packages. However other management applications use simpler communication tools such as message boards, to-do lists and simple scheduling etc., therefore, they do not use Gantt charts as heavily. The way to create this chart begins by determining and listing the necessary activities.

Next, sketch out how you expect the chart to look. This technique's primary advantage is its good graphical overview that is easy to understand for nearly all project participants and stakeholders. Its primary disadvantage is its limited applicability for many projects, since projects are often more complex than can be effectively communicated with this chart.

GANNT CHART OF LOFO:

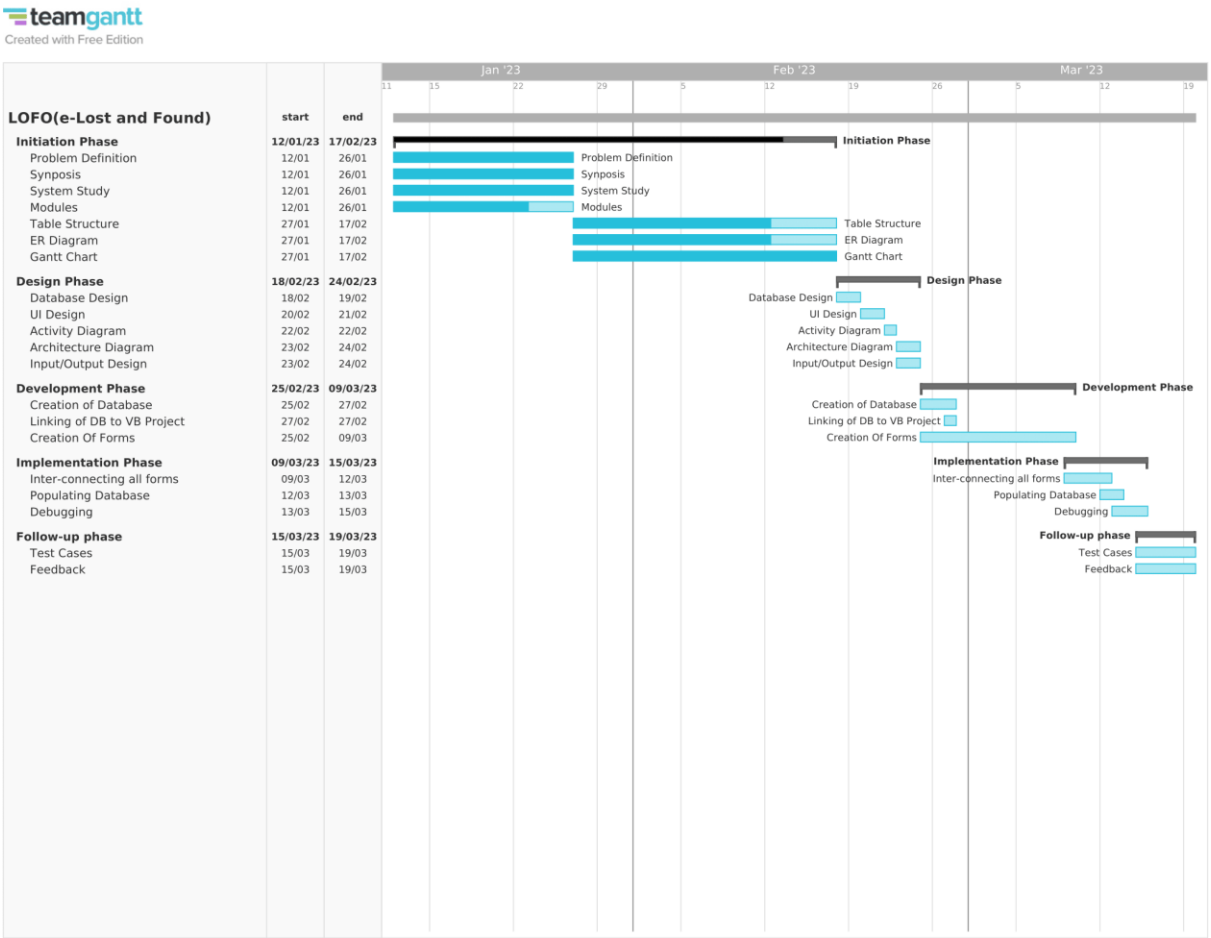


Fig 5

3.5 INPUT/OUTPUT DESIGN :

3.5.1 USER AUTHETICATION FORM:

```
Imports System.Windows
Imports System.Data.SQLite
Imports System.Collections.ObjectModel
Imports System.Security.Cryptography
Imports System.Text
Public Class Login

   Dim connectionString As String = "Data Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;"
   Public connection As New SQLiteConnection(connectionString)
   Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
      connection.Open()
      Me.FormBorderStyle = Windows.Forms.FormBorderStyle.FixedSingle
   End Sub

   Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click

      Dim username As String = TextBox1.Text
      Dim password As String = TextBox2.Text
#Region "Condition"
      If (String.IsNullOrWhiteSpace(TextBox1.Text) AndAlso String.IsNullOrWhiteSpace(TextBox2.Text))
Then
         MessageBox.Show("Please fill all the fields")
         Return
      End If

#End Region

      Dim sha256 As SHA256 = SHA256.Create()
```

```
    Dim hash As Byte() = sha256.ComputeHash(Encoding.UTF8.GetBytes(password))
Dim hashedPassword As String = Convert.ToBase64String(hash)
Dim query As String = "SELECT usertype FROM users WHERE username = @username AND password =
@password"
    Dim command As New SQLiteCommand(query, connection)
    command.Parameters.AddWithValue("@username", username)
    command.Parameters.AddWithValue("@password", hashedPassword)

    Dim reader As SQLiteDataReader = command.ExecuteReader()
    If reader.Read() Then
        Dim userType As String = reader.GetString(0)
        If userType = "admin" Then
            MessageBox.Show("Welcome, admin!")
            Me.Hide()
            Ahome.Show()
            Label3.Text = TextBox1.Text
            TextBox1.Clear()
            TextBox2.Clear()
        Else
            MessageBox.Show("Welcome, " + TextBox1.Text)
            Me.Hide()
            UHome.Show()
            TextBox1.Clear()
            TextBox2.Clear()

        End If

    Else
        MessageBox.Show("Incorrect username or password.")
        TextBox1.Clear()
        TextBox2.Clear()
    End If
End Sub
Private Sub TextBox1_KeyPress(sender As Object, e As KeyPressEventArgs) Handles TextBox1.KeyPress
    If e.KeyChar = ChrW(Keys.Enter) Then
        ' The "Enter" key was pressed
        e.Handled = True ' Prevents the "ding" sound
        Button1.PerformClick() ' Programmatically click the button

    End If

End Sub
```

```
    Private Sub TextBox2_KeyPress(sender As Object, e As KeyPressEventArgs) Handles TextBox2.KeyPress
        If e.KeyChar = ChrW(Keys.Enter) Then
            ' The "Enter" key was pressed
            e.Handled = True ' Prevents the "ding" sound
            Button1.PerformClick() ' Programmatically click the button
        End If
    End Sub
End Class
```

3.5.2 USER HOME FORM:

```
Public Class UHome
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
Me.Hide()
PostItem.ShowDialog()
End Sub

Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
Me.Hide()
FindItem.Show()
End Sub
Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
Me.Hide()
Login.Show()
End Sub
Private Sub Button4_Click(sender As Object, e As EventArgs) Handles Button4.Click
Me.Hide()
LogItem.Show()
End Sub
Private Sub UHome_Load(sender As Object, e As EventArgs) Handles MyBase.Load
Me.FormBorderStyle = Windows.Forms.FormBorderStyle.FixedSingle
End Sub
End Class
```

3.5.3 ADMIN HOME FORM:

```
Public Class Ahome
  Private Sub Ahome_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    Me.FormBorderStyle = Windows.Forms.FormBorderStyle.FixedSingle
  End Sub

  Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
    Me.Hide()
    ResolveLogs.Show()
  End Sub

  Private Sub Button4_Click(sender As Object, e As EventArgs) Handles Button4.Click
    Me.Hide()
    Login.Show()
  End Sub

  Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    Me.Hide()
    ResolveClaims.Show()
  End Sub

  Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
    Me.Hide()
    AddUsers.Show()
  End Sub
End Class
```

3.5.4 ADD USERS FORM:

```
Imports System.Data.SqlClient
Imports System.Data.SQLite
Imports System.Security.Cryptography
Imports System.Text
Public Class AddUsers
  Dim connectionString As String = "Data Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;"
  Public connection As New SQLiteConnection(connectionString)
  Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    Dim username As String = TextBox1.Text
    Dim password As String = TextBox2.Text
    Dim usertype As String = ComboBox1.Text
    Dim sha256 As SHA256 = SHA256.Create()
    Dim hash As Byte() = sha256.ComputeHash(Encoding.UTF8.GetBytes(password))
    Dim hashedPassword As String = Convert.ToBase64String(hash)
    connection.Open()
    Dim command As New SQLiteCommand("INSERT INTO users (username, password, usertype)
VALUES (@Username, @Password, @UserType);", connection)
    command.Parameters.AddWithValue("@Username", username)
    command.Parameters.AddWithValue("@Password", hashedPassword)
    command.Parameters.AddWithValue("@UserType", usertype)
    Dim rc As Integer = command.ExecuteNonQuery()
    If (rc > 0) Then
      MessageBox.Show("User is added, Thank you for your efforts!")
      Me.Close()
      Ahome.Show()
      connection.Close()
    Else
      MessageBox.Show("Oops, Item is not Uploaded.")
      TextBox1.Text = ""
      TextBox2.Text = ""
    End If
```

```
    End Sub
End Class
```

3.5.5 CLAIM ITEM FORM:

```
Imports System.Data.SQLite
Imports System.Net
Imports System.Net.Mail
Imports System.Reflection.Metadata

Public Class ClaimDetails
    Dim claimId As New Integer
    Dim itemId As New Integer
    Dim connectionString As String = "Data Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;"
    Public connection As New SQLiteConnection(connectionString)
    Public Sub New(ByVal cid As Integer, ByVal iid As Integer)

        ' This call is required by the designer.
        InitializeComponent()

        ' Add any initialization after the InitializeComponent() call.
        claimId = cid
        itemId = iid
    End Sub

    Protected Overrides Sub OnLoad(ByVal e As EventArgs)
        MyBase.OnLoad(e)
        Dim command As New SQLiteCommand("SELECT photo_path from found_items WHERE id=@id",
connection)
        command.Parameters.AddWithValue("@id", itemId)
        connection.Open()
        Dim result As Object = command.ExecuteScalar()
        Dim imagepath As String = result.ToString()
        PictureBox1.SizeMode = PictureBoxSizeMode.StretchImage
        PictureBox1.ImageLocation = imagepath

        Dim qry As New SQLiteCommand("SELECT item_title,date_found, contact_phone FROM found_items
WHERE id=@id", connection)
        qry.Parameters.AddWithValue("@id", itemId)
        Dim reader As SQLiteDataReader = qry.ExecuteReader()
        While reader.Read()
            Dim item_title As String = reader.GetString(0)
            Dim datetime As Date = reader.GetDateTime(1)
            Dim number As String = reader.GetString(2)
            Label3.Text = item_title
            TextBox3.Text = datetime
            Label6.Text = number
```

```
        End While

        Dim qry2 As New SQLiteCommand("SELECT claimant_desc, uploader_desc, claimant_mail FROM
claims WHERE id=@id", connection)
        qry2.Parameters.AddWithValue("@id", itemId)
        Dim read As SQLiteDataReader = qry2.ExecuteReader()
        While read.Read()
            Dim cdesc As String = read.GetString(0)
            Dim udesc As String = read.GetString(1)
            Dim cmail As String = read.GetString(2)
            TextBox1.Text = udesc
            TextBox2.Text = cdesc
            Label5.Text = cmail

        End While

        connection.Close()

    End Sub

    Private Sub Label3_Click(sender As Object, e As EventArgs) Handles Label3.Click

    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        MessageBox.Show("Claim Accepted! and the user is notified.")
        connection.Open()
        Dim command As New SQLiteCommand("UPDATE claims SET status = @new_value WHERE id =
@iid", connection)
        command.Parameters.AddWithValue("@iid", itemId)
        command.Parameters.AddWithValue("@new_value", "accepted")
        command.ExecuteNonQuery()
        Dim command2 As New SQLiteCommand("DELETE FROM found_items WHERE id = @iid",
connection)
        command2.Parameters.AddWithValue("@iid", itemId)
        command2.ExecuteNonQuery()
        connection.Close()
        SendEmail()
        Me.Hide()
        Ahome.Show()
    End Sub

    Public Sub SendEmail()
        Try
            connection.Open()

            Dim email As String = Label5.Text
```

```
        Dim command As New SQLiteCommand("SELECT contact_phone FROM found_items WHERE id =
@iid", connection)
        command.Parameters.AddWithValue("@iid", itemId)
        command.ExecuteNonQuery()
        Dim contact As String = command.ToString()


        Dim smtpClient As New SmtpClient("smtp.gmail.com", 587)
        smtpClient.EnableSsl = True
        smtpClient.Credentials = New Net.NetworkCredential("study.time0604@gmail.com",
"qiiobaukqjpwclli")

        MessageBox.Show(email)

        Dim message As New MailMessage()
        message.From = New MailAddress("study.time0604@gmail.com")
        message.To.Add(New MailAddress(email))
        message.Subject = "Mail Regarding your item Claim"
        message.Body = "Your Claim request is accepted! Please Contact " & Label6.Text & "And enquire
about your item. - LOFO TEAM"
        message.IsBodyHtml = False


        smtpClient.Send(message)
        connection.Close()
    Catch e As Exception
        MessageBox.Show(e.Message)
    End Try
  End Sub



  Public Sub SendEmail2()
    Try
        connection.Open()

        Dim email As String = Label5.Text
        Dim contact As String = ""


        Dim smtpClient As New SmtpClient("smtp.gmail.com", 587)
        smtpClient.EnableSsl = True
        smtpClient.Credentials = New Net.NetworkCredential("study.time0604@gmail.com",
"qiiobaukqjpwclli")

        MessageBox.Show(email)

        Dim message As New MailMessage()
```

```
            message.From = New MailAddress("study.time0604@gmail.com")
            message.To.Add(New MailAddress(email))
            message.Subject = "Mail Regarding your item Claim"
            message.Body = "Your Claim request is Rejected! Sorry to inform that you claim was not sactisfied.
Kindly raise another detailed claim for review . - LOFO TEAM"
            message.IsBodyHtml = False

            smtpClient.Send(message)
            connection.Close()
        Catch e As Exception
            MessageBox.Show(e.Message)
        End Try
    End Sub


    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
        SendEmail2()
        MessageBox.Show("Claim Rejected! and the user is notified.")
        connection.Open()
        Dim command2 As New SQLiteCommand("DELETE FROM claims WHERE claim_id = @iid",
connection)
        command2.Parameters.AddWithValue("@iid", claimId)
        command2.ExecuteNonQuery()
        connection.Close()
        Me.Hide()
    End Sub

    Private Sub ClaimDetails_Load(sender As Object, e As EventArgs) Handles MyBase.Load

    End Sub
End Class
```

3.5.6 CLAIM ITEM FORM:

```
Imports System.Windows.Forms.VisualStyles.VisualStyleElement

Imports System.Text

Imports System.IO

Imports System.Windows

Imports System.Data.SQLite

Imports System.Collections.ObjectModel

Imports System.Runtime.CompilerServices


Public Class FindItem

    Dim connectionString As String = "Data Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;"
```

```
Public connection As New SQLiteConnection(connectionString)
Private Sub FindItem_Load(sender As Object, e As EventArgs) Handles MyBase.Load
    connection.Open()
    DataGridView1.ReadOnly = True
    Dim sql As String = "SELECT id, item_title, photo_path FROM found_items"
    Dim cmd As New SQLiteCommand(sql, connection)
    Dim adapter As New SQLiteDataAdapter(cmd)
    Dim table As New DataTable()
    adapter.Fill(table)


    DataGridView1.AutoGenerateColumns = False
    DataGridView1.RowTemplate.Height = 200
    DataGridView1.AllowUserToAddRows = False


    Dim colId As New DataGridViewTextBoxColumn()
    colId.DataPropertyName = "id"
    colId.HeaderText = "Item ID"
    colId.Name = "colId"
    DataGridView1.Columns.Add(colId)

    Dim colTitle As New DataGridViewTextBoxColumn()
    colTitle.DataPropertyName = "item_title"
    colTitle.HeaderText = "Item Title"
    colTitle.Name = "colTitle"
    DataGridView1.Columns.Add(colTitle)

    Dim colImage As New DataGridViewImageColumn()
    colImage.DataPropertyName = "photo_path"
    colImage.HeaderText = "Image"
    colImage.Name = "colImage"
```

```
        colImage.ImageLayout = DataGridViewImageCellLayout.Stretch
        DataGridView1.Columns.Add(colImage)

        DataGridView1.Columns(0).Name = ""

        For Each row As DataRow In table.Rows
            Dim id As Integer = Convert.ToInt32(row("id"))
            Dim title As String = row("item_title").ToString()
            Dim imagePath As String = row("photo_path").ToString()
            Dim image As Image = Image.FromFile(imagePath)
            DataGridView1.Rows.Add(id, title, image)
        Next

        Me.Controls.Add(DataGridView1)

        DataGridView1.Columns(0).Width = 300
        DataGridView1.Columns(1).Width = 300
        DataGridView1.Columns(2).Width = 300
    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Me.Hide()
        UHome.Show()
    End Sub

    Private Sub DataGridView1_CellContentClick(sender As Object, e As DataGridViewCellEventArgs)
Handles DataGridView1.CellContentClick
        If e.ColumnIndex = 2 AndAlso e.RowIndex >= 0 Then
            ' Retrieve the information for the selected item from the database
            Dim itemID As Integer = CInt(DataGridView1.Rows(e.RowIndex).Cells(0).Value)
            Dim itemTitle As String = CStr(DataGridView1.Rows(e.RowIndex).Cells(1).Value)
            'Dim itemImage As Image = Image.FromFile(CStr(DataGridView1.Rows(e.RowIndex).Cells(2).Value))
```

```
        ' Create a new instance of the ItemDetailsForm form with the item information as arguments
        Dim itemDetailsForm As New ItemDetails(itemID, itemTitle)

        ' Show the new form as a dialog box
        itemDetailsForm.ShowDialog()
    End If
  End Sub
End Class
```

3.5.7 LOG ITEMS FORM:

```
Imports System.Data.SQLite
Public Class LogItem
  Dim connectionString As String = "Data
Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;Journal Mode=WAL"
  Public connection As New SQLiteConnection(connectionString)
Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
    Dim command As New SQLiteCommand("INSERT INTO logs(log_desc, log_mail) VALUES(@desc,
@mail)", connection)
    command.Parameters.AddWithValue("@desc", TextBox1.Text)
    command.Parameters.AddWithValue("@mail", TextBox2.Text)
    connection.Open()
    Dim rc As Integer = command.ExecuteNonQuery()
    If (rc > 0) Then
      MessageBox.Show("Log is Uploaded, We Will Get Back to you!")
      Me.Close()
      UHome.Show()
      connection.Close()
    Else
      MessageBox.Show("Oops, Log is not Uploaded.")
      Me.Refresh()
    End If
  End Sub
```

```
    Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
        Me.Hide()
        UHome.Show()
    End Sub

    Private Sub LogItem_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        TextBox1.PlaceholderText = "Enter details : " & Environment.NewLine & " Item Description" &
Environment.NewLine & " How you lost it" & Environment.NewLine & " etc."
        Me.FormBorderStyle = Windows.Forms.FormBorderStyle.FixedSingle
        Label3.Focus()
    End Sub
End Class
```

3.5.8 POST ITEMS FORM:

```
Imports System.Windows.Forms.VisualStyles.VisualStyleElement
Imports System.Text
Imports System.IO
Imports System.Windows
Imports System.Data.SQLite
Imports System.Collections.ObjectModel
Imports System.Diagnostics.Tracing

Public Class PostItem
    Dim connectionString As String = "Data
Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;Journal Mode=WAL"
    Public connection As New SQLiteConnection(connectionString)

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Dim img1 As OpenFileDialog = New OpenFileDialog()
        img1.Filter = "choose image(*.jpg;*.png;*.gif)|*.jpg;*.png;*.gif"
        If img1.ShowDialog() = DialogResult.OK Then
```

```
        Label3.Show()
        Label3.Text = img1.FileName
        PictureBox1.ImageLocation = Label3.Text
    End If
End Sub

Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
    Dim command As New SQLiteCommand("INSERT INTO found_items(item_type, item_description,
location_found, date_found, contact_phone, photo_path, username, item_title) VALUES (@item_type,
@item_description, @location_found, @date_found, @contact_phone, @photo_path, @username,
@item_title)", connection)

    command.Parameters.AddWithValue("@item_title", TextBox1.Text)
    command.Parameters.AddWithValue("@item_type", ComboBox1.Text)
    command.Parameters.AddWithValue("@item_description", TextBox2.Text)
    command.Parameters.AddWithValue("@location_found", TextBox3.Text)
    command.Parameters.AddWithValue("@date_found", DateTimePicker1.Value.ToString("yyyy-MM-dd"))
    command.Parameters.AddWithValue("@contact_phone", TextBox4.Text)
    command.Parameters.AddWithValue("@photo_path", Label3.Text)
    command.Parameters.AddWithValue("@username", Login.Label3.Text)
    connection.Open()
    Dim rc As Integer = command.ExecuteNonQuery()
    If (rc > 0) Then
        MessageBox.Show("Item is Uploaded, Thank you for your efforts!")
        Me.Close()
        UHome.Show()
        connection.Close()
    Else
        MessageBox.Show("Oops, Item is not Uploaded.")
        Me.Refresh()
    End If
    'End Using
```

```
    End Sub
    Private Sub Button3_Click(sender As Object, e As EventArgs) Handles Button3.Click
        Me.Hide()
        UHome.Show()
    End Sub

    Private Sub PostItem_Load(sender As Object, e As EventArgs) Handles MyBase.Load

    End Sub
End Class
```

3.5.9 RESOLVE CLAIMS FORM:

```
Imports System.Data.SQLite

Public Class ResolveClaims
    Dim connectionString As String = "Data Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;"
    Public connection As New SQLiteConnection(connectionString)
    Private Sub ResolveClaims_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Dim adapter As New SQLiteDataAdapter("SELECT claims.claim_id, found_items.id, claims.status FROM
claims INNER JOIN found_items ON claims.id = found_items.id", connection)
        Dim table As New DataTable()
        adapter.Fill(table)
        DataGridView1.DataSource = table
        Me.FormBorderStyle = Windows.Forms.FormBorderStyle.FixedSingle

        DataGridView1.Columns(0).Width = 135
        DataGridView1.Columns(1).Width = 135
        DataGridView1.Columns(2).Width = 139

        Dim font As New Font("Unispace", 14, FontStyle.Bold)
        DataGridView1.DefaultCellStyle.Font = font
        DataGridView1.Rows(0).Selected = False
```

```
    End Sub
Private Sub DataGridView1_CellContentClick(sender As Object, e As DataGridViewCellEventArgs) Handles
DataGridView1.CellContentClick
        If e.ColumnIndex = 2 AndAlso e.RowIndex >= 0 Then
            ' Retrieve the information for the selected item from the database
            Dim claimID As Integer = CInt(DataGridView1.Rows(e.RowIndex).Cells(0).Value)
            Dim itemId As Integer = CStr(DataGridView1.Rows(e.RowIndex).Cells(1).Value)
            'Dim itemImage As Image = Image.FromFile(CStr(DataGridView1.Rows(e.RowIndex).Cells(2).Value))

            ' Create a new instance of the ItemDetailsForm form with the item information as arguments
            Dim claimDetailsform As New ClaimDetails(claimID, itemId)

            ' Show the new form as a dialog box
            claimDetailsform.ShowDialog()
            Me.Hide()
            Ahome.Show()
        End If
    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Me.Close()
        Ahome.Show()
    End Sub
End Class
```

3.5.10 RESOLVE LOGS FORM:

```
Imports System.Data.SQLite

Public Class ResolveLogs
   Dim connectionString As String = "Data Source=C:\Users\mojog\Desktop\LoFo\mydatabase.db;Version=3;"
   Public connection As New SQLiteConnection(connectionString)
   Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
      Me.Hide()
      Ahome.Show()
   End Sub

   Private Sub ResolveLogs_Load(sender As Object, e As EventArgs) Handles MyBase.Load
      Dim adapter As New SQLiteDataAdapter("SELECT * from logs", connection)
      Dim table As New DataTable()
      adapter.Fill(table)
      DataGridView1.DataSource = table
      Me.FormBorderStyle = Windows.Forms.FormBorderStyle.FixedSingle

      DataGridView1.Columns(0).Width = 135
      DataGridView1.Columns(1).Width = 135
      DataGridView1.Columns(2).Width = 139

      Dim font As New Font("Unispace", 9, FontStyle.Bold)
      DataGridView1.DefaultCellStyle.Font = font
      DataGridView1.Rows(0).Selected = False
   End Sub

   Private Sub DataGridView1_CellContentClick(sender As Object, e As DataGridViewCellEventArgs)
Handles DataGridView1.CellContentClick
      If e.RowIndex >= 0 Then

         Dim logID As Integer = CInt(DataGridView1.Rows(e.RowIndex).Cells(0).Value)
         Dim logdesc As String = CStr(DataGridView1.Rows(e.RowIndex).Cells(1).Value)
         Dim logmail As String = CStr(DataGridView1.Rows(e.RowIndex).Cells(2).Value)

         Dim logDetailsform As New LogDetails(logID, logdesc, logmail)

         logDetailsform.ShowDialog()
         Me.Hide()
         Ahome.Show()
      End If
   End Sub
End Class
```

4. SYSTEM CONFIGURATION

4.1 SOFTWARE REQUIREMENT SPECIFICATIONS:

The production of the requirements stage of the software development process is Software Requirements Specifications (SRS) (also called a requirements document). This report lays a foundation for software engineering activities and is constructing when entire requirements are elicited and analyzed. SRS is a formal report, which acts as a representation of software that enables the customers to review whether it (SRS) is according to their requirements. Also, it comprises user requirements for a system as well as detailed specifications of the system requirements.

4.1.1 HARDWARE REQUIREMENTS:

TABLE1:HARDWARE REQUIREMENTS

PROCESSOR	INTEL CORE I5 9TH GEN
INSTALLED RAM	8 GB
SYSTEM TYPE	64 BIT OPERATING SYSTEM

4.1.2 SOFTWARE REQUIREMENTS:

TABLE2: SOFTWARE REQUIREMENTS

FRONT END	VISUAL BASIC 2022
BACK END	SQLITE SERVER
OPERATING SUSTEM	WINDOWS 11
DOCUMENTATION	MS WORD
GANTT CHAR	TEAMGANNT

5. DETAILS OF SOFTWARE

5.1 OVERVIEW OF FRONT-END:

Microsoft Visual Studio 2019 is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio supports different programming languages and allows the code editor and debugger to support nearly any programming language, provided a language-specific service exists. Built-in languages include C, C++, Visual C++ and VB.NET. Support for other languages such as Python, Ruby, Node.js, and M among others is available via language services installed separately.

It also supports XML/XSLT, HTML/XHTML, JavaScript and CSS. Java (and J#) were supported in the past**.** Microsoft provides a free version of Visual Studio called the Community edition that supports plugins and is available at no cost for all users. Support for programming languages is added by using a specific Package called a Language Service.

A language service defines various interfaces which the Package implementation can implement to add support for various functionalities. Functionalities that can be added this way include syntax coloring, statement completion, brace matching, parameter information tooltips, member lists and error markers for background compilation.

If the interface is implemented, the functionality will be available for the language. Language services are implemented on a per-language basis. The implementations can reuse code from the parser or the compiler for the language. Language services can be implemented either in native code or managed code. For native code, either the native COM interfaces or the Babel Framework can be used. For managed code, the MPF includes wrappers for writing managed language services.

5.1.1 FEATURES:

- Boolean Conditions
- Automatic Garbage Collection Standard Library
- Assembly Versioning
- Properties and Events
- Delegates and Events Management Easy-to-use Generics
    - Indexers
- Conditional Compilation Simple Multithreading.

5.1.2 ADVANTAGES:

- VB is not only a language but primarily an integrated, interactive development environment
- The graphical user interface of the VB-IDE provides intuitively appealing views for the management of the program structure in the large and the various types of entities (classes, modules, procedures, forms, …)
- When editing program text, the "IntelliSense" technology informs you in a little popup window about the types of constructs that may be entered at the current cursor location.
- VB is a component integration language which is attuned to Microsoft's Component Object Model "COM").
- COM components can be written in different languages and then integrated using VB.
- Interfaces of COM components can be easily called remotely via Distributed COM ("DCOM"), which makes it easy to construct distributed applications.

5.2 OVERVIEW OF BACK-END:

SQLite is a popular, lightweight, and self-contained relational database management system (RDBMS) that can serve as a backend for various applications. Unlike traditional client-server RDBMSs, SQLite does not have a separate server process that runs continuously and accepts incoming connections. Instead, SQLite stores the entire database as a single file on disk, and applications interact with the file through the SQLite library.
To use SQLite as a backend, an application needs to include the SQLite library and interact with it through API calls. The application can create, read, update, and delete data in the database by sending SQL commands to the SQLite engine. SQLite also provides various command-line tools and graphical interfaces that can be used to manage the database.

One of the main advantages of using SQLite as a backend is its ease of deployment. Since the database is stored as a single file, it can be easily copied, moved, and backed up. This makes it convenient for distributing applications that require a local database.

Overall, SQLite is a reliable and efficient choice for applications that require a lightweight and self-contained database backend.

5.2.1 FEATURES OF SQLITE SERVER:

- Self-contained: SQLite is a fully self-contained database engine that requires no separate server process or system to run. It operates as a library that can be embedded into an application, making it easy to deploy and maintain.

- Cross-platform: SQLite works on a variety of platforms, including Windows, macOS, Linux, iOS, and Android. It is compatible with multiple programming languages, such as C/C++, Java, Python, and PHP.

- Lightweight: SQLite is small in size and has a small memory footprint. It does not require a lot of system resources to operate, making it an excellent choice for embedded systems and mobile devices.
- Transactional: SQLite supports ACID (Atomicity, Consistency, Isolation, Durability) transactions, allowing multiple operations to be grouped into a single transaction for consistency and data integrity.

- No configuration: Since there is no separate server process, there is no need for complex configuration settings or installation procedures. Applications can simply use the SQLite library to access the database.

5.2.2 SECURITY:

A privilege and password system that is very flexible and secure, and that enables host-based verification. Password security by encryption of all password traffic when you connect to a server.

5.2.6 CLIENTS AND TOOLS:

- Microsoft provides both data management and business intelligence (BI) tools and services together with SQL Server.

- For data management, SQL Server includes SQL Server Integration Services (SSIS), SQL Server Data Quality Services, and SQL Server Master Data Services. To develop databases, SQL Server provides SQL Server Data tools; and to manage, deploy, and monitor databases SQL Server has SQL Server Management Studio (SSMS).

- For data analysis, SQL Server offers SQL Server Analysis Services (SSAS). SQL Server Reporting Services (SSRS) provides reports and visualization of data. The Machine Learning Services technology appeared first in SQL Server 2016 which was renamed from the R Services.

5.3 ABOUT THE PLATFORM:

Windows is a group of several proprietary graphical operating system families developed and marketed by Microsoft. Each family caters to a certain sector of the computing industry. For example, Windows NT for consumers, Windows Server for servers, and Windows IoT for embedded systems. Defunct Windows families include Windows 9x, Windows Mobile, and Windows Phone.

The first version of Windows was released on November 20, 1985, as a graphical operating system shell for MS-DOS in response to the growing interest in graphical user interfaces (GUIs).

Windows is the most popular desktop operating system in the world, with 75% market share as of April 2022, according to StatCounter. However, Windows is not the most used operating system when including both mobile and desktop OSes, due to Android's massive growth.

As of September 2022, the most recent version of Windows is Windows 11 for consumer PCs and tablets, Windows 11 Enterprise for corporations, and Windows Server 2022 for servers.

5.3.1 .NET FRAMEWORK:

.NET Framework (pronounced as "dot net") is a software framework developed by Microsoft that runs primarily on Microsoft Windows. It includes a large class library named Framework Class Library (FCL) and provides language interoperability (each language can use code written in other languages) across several programming languages.

Programs written for .NET Framework execute in a software environment (in contrast to a hardware environment) named Common Language Runtime (CLR), an application virtual machine that provides services such as security, memory management, and exception handling.

As such, computer code written using .NET Framework is called "managed code". FCL and CLR together constitute the .NET Framework. FCL provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. Programmers produce software by combining their source code with .NET Framework and other libraries.

The framework is intended to be used by the newest applications created for the Windows platform. Microsoft also produces an integrated development environment largely for .NET software called Visual Studio. .NET Framework began as proprietary software, although the firm worked to standardize the software stack almost immediately, even before its first release.

Despite the standardization efforts, developers, mainly those in the free and open-source software communities, expressed their unease with the selected terms and the prospects of any free and open- source implementation, especially regarding software patents. Since then, Microsoft has changed.

.NET development to more closely follow a contemporary model of a community developed software project, including issuing an update to its patent promising to address the concerns.

6. TESTING

Testing is a vital part of software development, and it is important to start it as early as possible, and to make testing a part of the process of deciding requirements. To get the most useful perspective on your development project, it is worthwhile devoting some thought to the entire lifecycle including how feedback from users will influence the future of the application.

The tools and techniques we've discussed in this book should help your team to be more responsive to changes without extra cost, despite the necessarily wide variety of different development processes. Nevertheless, new tools and process improvements should be adopted gradually, assessing the results after each step.

Testing is part of a lifecycle. The software development lifecycle is one in which you hear of a need, you write some code to fulfil it, and then you check to see whether you have pleased the stakeholders— the users, owners, and other people who have an interest in what the software does.

They like it, but would also like some additions or changes, so you update or augment your code; and so, the cycle continues. This cycle might happen every few days, as it does in Fabrikam's ice cream vending project, or every few years, as it does in Contoso's carefully specified and tested healthcare support system. Software development lifecycle Testing is a proxy for the customer.

You could conceivably do your testing by releasing it into the wild and waiting for the complaints and compliments to come back. Some companies have been accused of having such a strategy as their business model even before it became fashionable. But overall, the books are better balanced by trying to make sure that the software will satisfy the customer before we hand it over.

We therefore design tests based on the stakeholders' needs and run the tests before the product reaches the users. Preferably well before then, so as not to waste our time working on something that isn't going to do the job. In this light, two important principles become clear.

6.1 TESTS REPRESENT REQUIREMENTS:

Whether you write user stories on sticky notes on the wall, or use cases in a big thick document, your tests should be derived from and linked to those requirements. And as we've said, devising tests is a good vehicle for discussing the requirements.

WE ARE NOT DONE TILL THE TEST IS PASSED:

The only useful measure of completion is when tests have been performed successfully. Those principles apply no matter how you develop your software.

6.2 SOFTWARE TESTING TYPES:

BLACK BOX TESTING – Internal system design is not considered in this type of testing. Tests are based on requirements and functionality.

WHITE BOX TESTING – This testing is based on knowledge of the internal logic of an application's code. Also known as Glass box Testing. Internal software and code working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions.

UNIT TESTING – Testing of individual software components or modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. may require developing test driver modules or test harnesses.

INCREMENTAL INTEGRATION TESTING – Bottom-up approach for testing i.e., continuous testing of an application as new functionality is added; Application functionality and modules should be independent enough to test separately. done by programmers or by testers.

INTEGRATION TESTING – Testing of integrated modules to verify combined functionality after integration. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

FUNCTIONAL TESTING – This type of testing ignores the internal parts and focuses on the output is as per requirement or not. Black-box type testing geared to functional requirements of an application.

SYSTEM TESTING – Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system.

END-TO-END TESTING – Similar to system testing, involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

SANITY TESTING – Testing to determine if a new software version is performing well enough to accept it for a major testing effort. If the application is crashing for initial use, then system is not stable enough for further testing and build or application is assigned to fix.

REGRESSION TESTING – Testing the application for the modification in any module or functionality. Difficult to cover all the systems in regression testing so typically automation tools are used for these testing types.

ACCEPTANCE TESTING -Normally this type of testing is done to verify if the system meets the customer's specified requirements. User or customer does this testing to determine whether to accept application.

LOAD TESTING– It's a performance testing to check system behavior under load. Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.

STRESS TESTING– System is stressed beyond its specifications to check how and when it fails. Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load.

PERFORMANCE TESTING – Term often used interchangeably with 'stress' and 'load' testing. To check whether the system meets performance requirements. Used different performance and load tools to do this.

USABILITY TESTING – User-friendliness check. Application flow is tested, can new user understand the application easily, Proper help documented whenever user stuck at any point. Basically, system navigation is checked in this testing.

INSTALL/UNINSTALL TESTING– Tested for full, partial, or upgrade install/uninstall processes on different operating systems under different hardware, software environment.

RECOVERY TESTING – Testing how well a system recovers from crashes, hardware failures, or other catastrophic problems.

SECURITY TESTING – Can system be penetrated by any hacking way. Testing how well the system protects against unauthorized internal or external access.

COMPARISON TESTING– Comparison of product strengths and weaknesses with previous versions or other similar products.

ALPHA TESTING– In house virtual user environment can be created for this type of testing. Testing is done at the end of development. Still minor design changes may be made as a result of such testing.

BETA TESTING – Testing typically done by end-users or others. Final testing before releasing application for commercial purpose.

7. CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION:

LOFO is the ultimate solution of problems regarding LOST and FOUND in campus. The proposed project forms a bridge between finder and a person who has lost belongings in a big campus.

The proposed project performs all the item related management works such as post item, find item, log item, resolve claims, resolve log, add new users etc. It provides a strong security for the protection of data in the system database.

7.2 FUTURE ENHANCEMENT:

LOFO has a lot of scope in future where various modules can be upgraded.

- Online- LOFO project module can be implemented in the all the organization which helps them to have efficient lost and found.
- Biometric- The login can be taken using biometric and can be updated.
- Multi-platform- multi-platforms like mobile and web applications can be developed so that everyone can have access to this project and can view item details.

The implementation of the proposed project 'LOFO' reduces the number of man interference in the system. This project can easily work with new features and modules added to the system as per the user requirements.

8.BIBLIOGRAPHY

8.1 NEWSPAPER REFERENCES

[1] Software Engineering at Google, author: Hyrum Wright and Titus Delafayette Winters and Tom Manshreck, 2020.

[2] A Systematic Mapping Study on Software Reuse, authors: Bhargava Mithra Konda and Kranthi Kiran Mandava, 2010

[3]Ahmed Jilani, A. A., Nadeem, A., Kim, T. H. and Cho, E. S. (2008). Formal Representations of the Data Flow Diagram: A Survey. Proc. of the 2008 Advanced Software Engineering and Its Applications. Washington, USA: IEEE Computer Society. pp. 153-158.

[4]Lucas, F.J., Molina, F. and Toval, A. (2009). A Systematic Review of UML Model Consistency Management. Information and Software Technology, 51(12), pp. 1 – 15.

[5]S, Shylesh, A Study of Software Development Life Cycle Process Models (June 10, 2017). Available at SSRN: https://ssrn.com/abstract=2988291

[6]Kumar, Sujit & Dubey, Pushkar. (2013). SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC) ANALYTICAL COMPARISON AND SURVEY ON TRADITIONAL AND AGILE METHODOLOGY. ABHINAV NATIONAL MONTHLY REFEREED JOURNAL OF RESEARCH IN SCIENCE & TECHNOLOGY. 2. 22-30.

[7]Song, Il-Yeol & Evans, Mary & Park, Eui Kyun. (1995). A Comparative Analysis of Entity-Relationship Diagrams. Journal of Computer and Software Engineering. 3.

[8]Evdokimov, Ivan & Tsarev, Roman & Yamskikh, Tatiana & Pupkov, Alexandr. (2018). Using PERT and Gantt charts for planning software projects on the basis of distributed digital ecosystems. Journal of Physics: Conference Series. 1074. 012127. 10.1088/1742-6596/1074/1/012127.

[9]Databases in Software Engineering: A Roadmap, authors: Klaus R. Dittrich Dimitrios Tombros Andreas Geppert, Department of Information Technology, University of Zurich

[10]Bs, Soumya. (2015). .NET Paper. 2. 3434-3449.

[11]Zhu Li-juan, "Research and application of SQL Server in the trade management system," *2011 3rd International Conference on Computer Research and Development*, Shanghai, China, 2011, pp. 209-213, doi: 10.1109/ICCRD.2011.5764282.

[12]Software Engineers Develop New App to Simplify Task Management" - The New York Times

[13]"Local Tech Company Releases Revolutionary Video Editing Software" - The Los Angeles Times

[14]"Startup Raises $10 Million to Build Artificial Intelligence-Powered Chatbot" - The Wall Street Journal

[15]"Open-Source Software Project Allows for Easy Collaboration Among Developers" - The Guardian

[16]"Virtual Reality Startup Introduces Revolutionary Training Tool for Medical Professionals" - The Washington Post

[17]"Software Company Launches Innovative Cybersecurity Platform to Combat Online Threats" - The Boston Globe

[18]"Tech Giant Announces Major Updates to Cloud Computing Infrastructure" - Forbes

[19]"Software Developer Creates New Algorithm to Improve Data Analysis" - Wired

[20]"New Social Media Platform Takes the Internet by Storm, Attracting Millions of Users Worldwide" – CNN

[21]"Startup Launches Revolutionary E-commerce Platform with Built-in Machine Learning Capabilities" – TechCrunch

[22]"Artificial Intelligence Startup Creates Virtual Assistant that Can Schedule Your Entire Day" - CNBC

[23]"Software Company Uses Blockchain Technology to Improve Supply Chain Management" - Bloomberg

[24]"Startups Race to Develop Self-Driving Car Software" - The Verge

[25]"New Mobile App Lets You Easily Manage Your Finances and Invest in the Stock Market" - Financial Times

8.2 BOOK REFRENCES

[1]VB.NET Language in a Nutshell, Second Edition by Steven Roman PhD Released April 2002, Publisher(s): O'Reilly Media, Inc. ISBN: 9780596003081

[2]Dennis, A., Wixom, B.H. and Roth, R.M. (2006). Systems Analysis and Design. 3rd ed. Hoboken: John Wiley & Sons, Inc.

[3]Code Complete: A Practical Handbook of Software Construction by Steve McConnell, Publisher: Microsoft Press; Second Edition

[4]Software Engineering by Ian Sommerville, Publisher: Addison-Wesley Edition: 7

[5]Agile Management for Software Engineering: Applying the Theory of Constraints for Business Results, by David Anderson, Publisher : Prentice Hall (September 17, 2003), ISBN-10 : 9780131424609

[6] Pressman, Roger (2010) Software Engineering: A Practitioner's Approach, McGraw Hill, New York, NY.

[7]Sommerville, Ian (2011) Software Engineering, Addison-Wesley, Boston, MA.

[8] Stephens, Rod (2015) Beginning Software Engineering, Wrox.

[9]Tsui, Frank, Orlando Karam and Barbara Bernal (2013) Essentials of Software Engineering, Jones & Bartlett Learning, Sudbury, MA.

[10]Pfleeger, Shari (2001) Sofwtare Engineering: Theory and Practice, Prentice Hall, Upper Saddle River, NJ.

9.APPENDICES A-TABLE STRUCTURE:

9.1 LOGIN AND REGISTRATION:

TABLE 3: USERS TABLE.

mydatabase Table name: users WITHOUT ROWID STRICT

	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	id	INTEGER								NULL
2	username	TEXT								NULL
3	password	TEXT								NULL
4	usertype	TEXT								NULL

TABLE 4: LOGS TABLE.

mydatabase Table name: logs WITHOUT ROWID STRICT

	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	logid	INTEGER								NULL
2	log_desc	TEXT								NULL
3	log_mail	TEXT								NULL

TABLE 5: FOUND ITEMS TABLE.

mydatabase Table name: found_items WITHOUT ROWID STRICT

	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	id	INTEGER								NULL
2	item_type	TEXT								NULL
3	item_description	TEXT								NULL
4	location_found	TEXT								NULL
5	date_found	TEXT								NULL
6	contact_phone	TEXT								NULL
7	photo_path	TEXT								NULL
8	username	TEXT								NULL
9	item_title	TEXT								NULL

TABLE 6: CLAIMS TABLE.

mydatabase Table name: claims WITHOUT ROWID STRICT

	Name	Data type	Primary Key	Foreign Key	Unique	Check	Not NULL	Collate	Generated	
1	claim_id	INTEGER								NULL
2	id									NULL
3	claimant_desc	TEXT								NULL
4	claimant_mail	TEXT								NULL
5	uploader_desc	TEXT								NULL
6	status	TEXT								rejected

10.APPENDICES-B SCREENSHOTS:

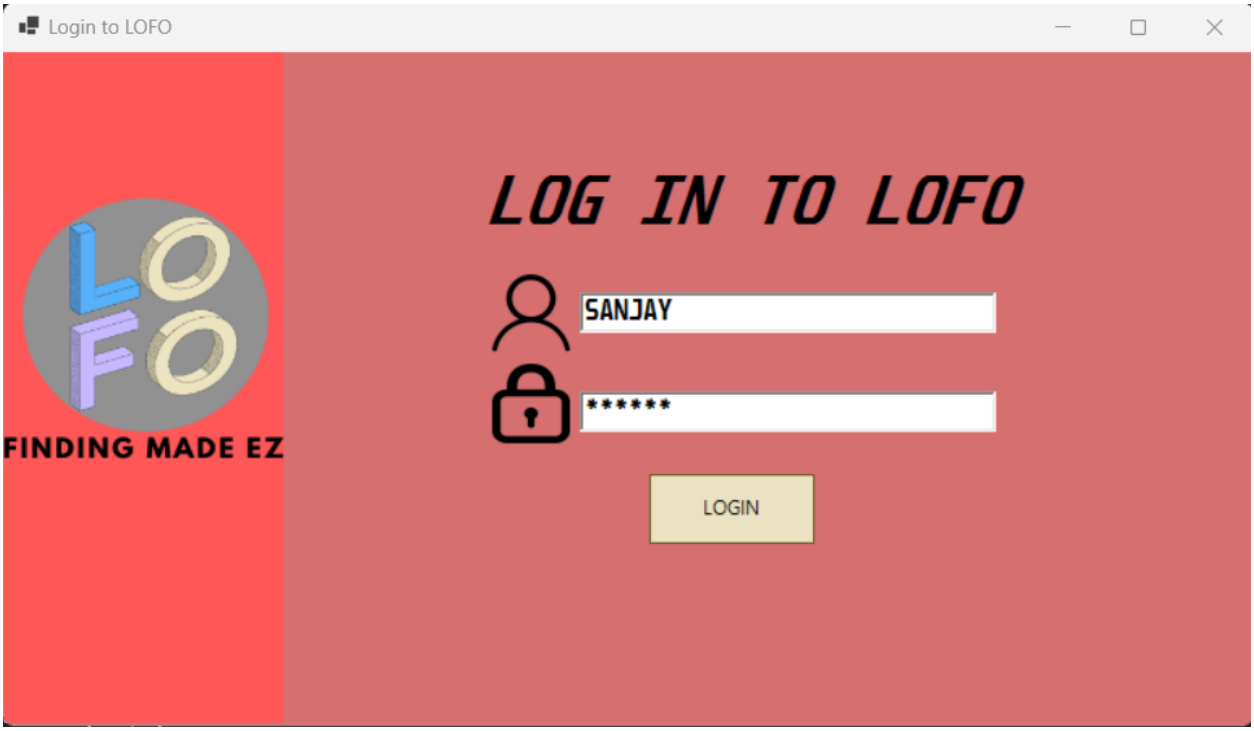


FIG5: LOGIN PAGE

FIG6: USER HOME PAGE

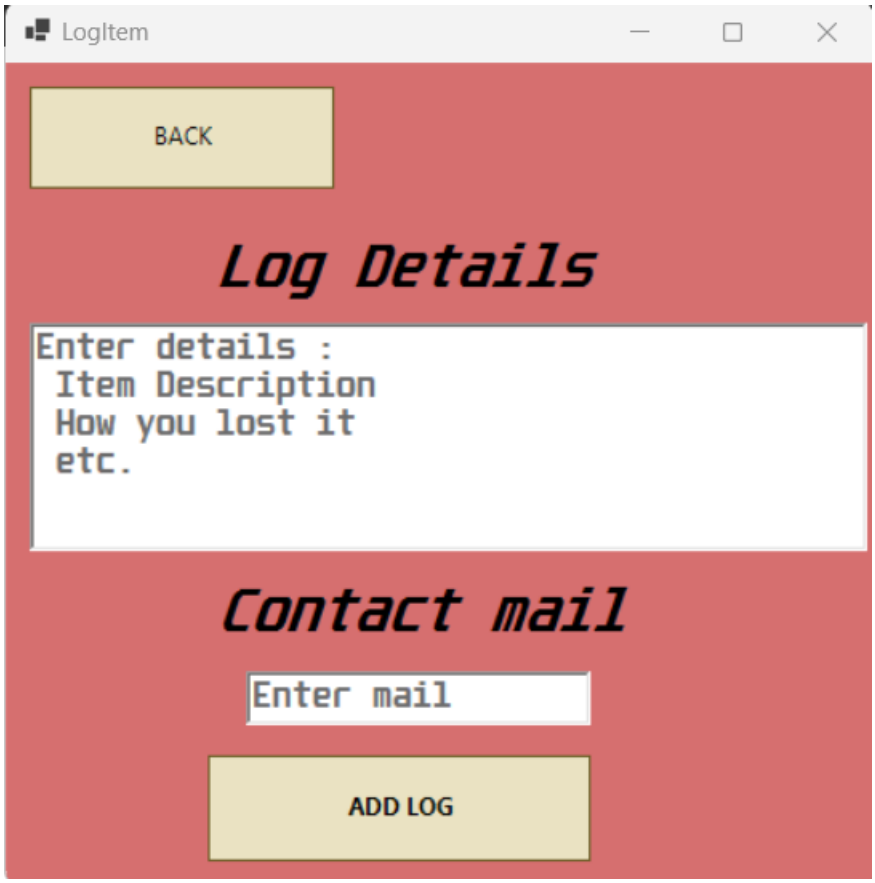


FIG7: ADD LOG PAGE

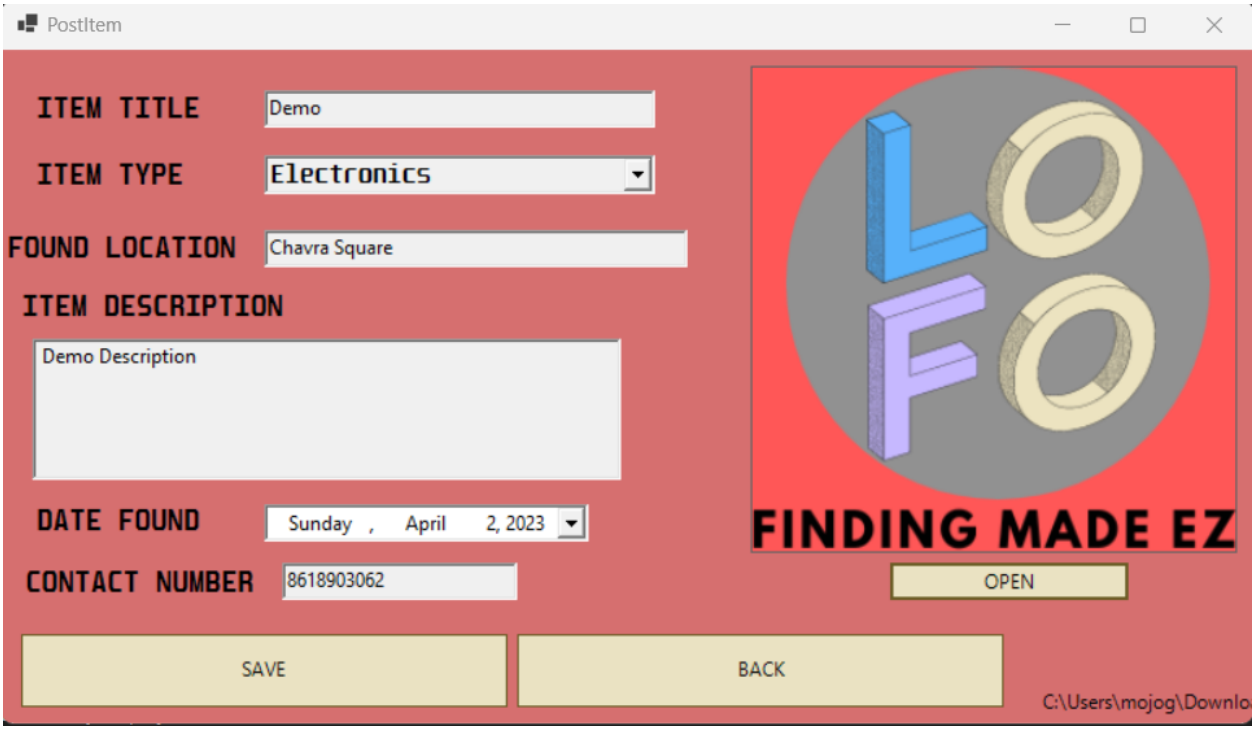


FIG8: ADD ITEMS PAGE

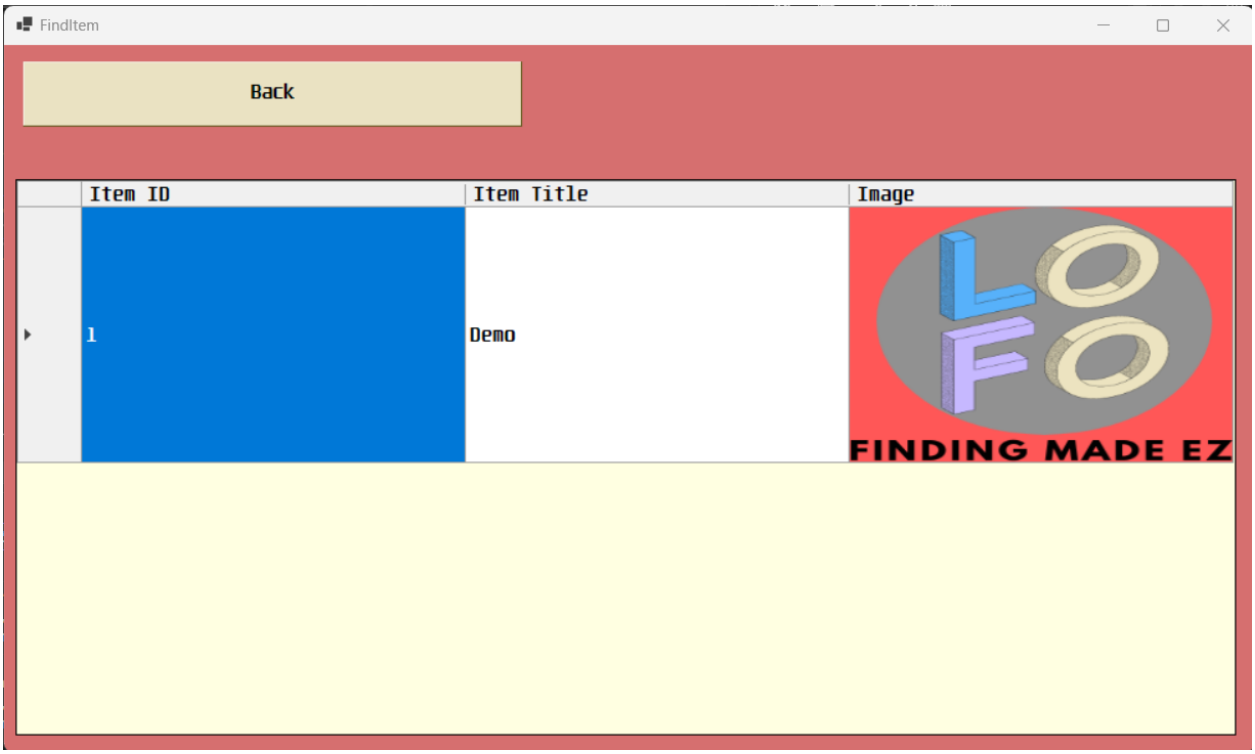


FIG9: FIND ITEMS PAGE

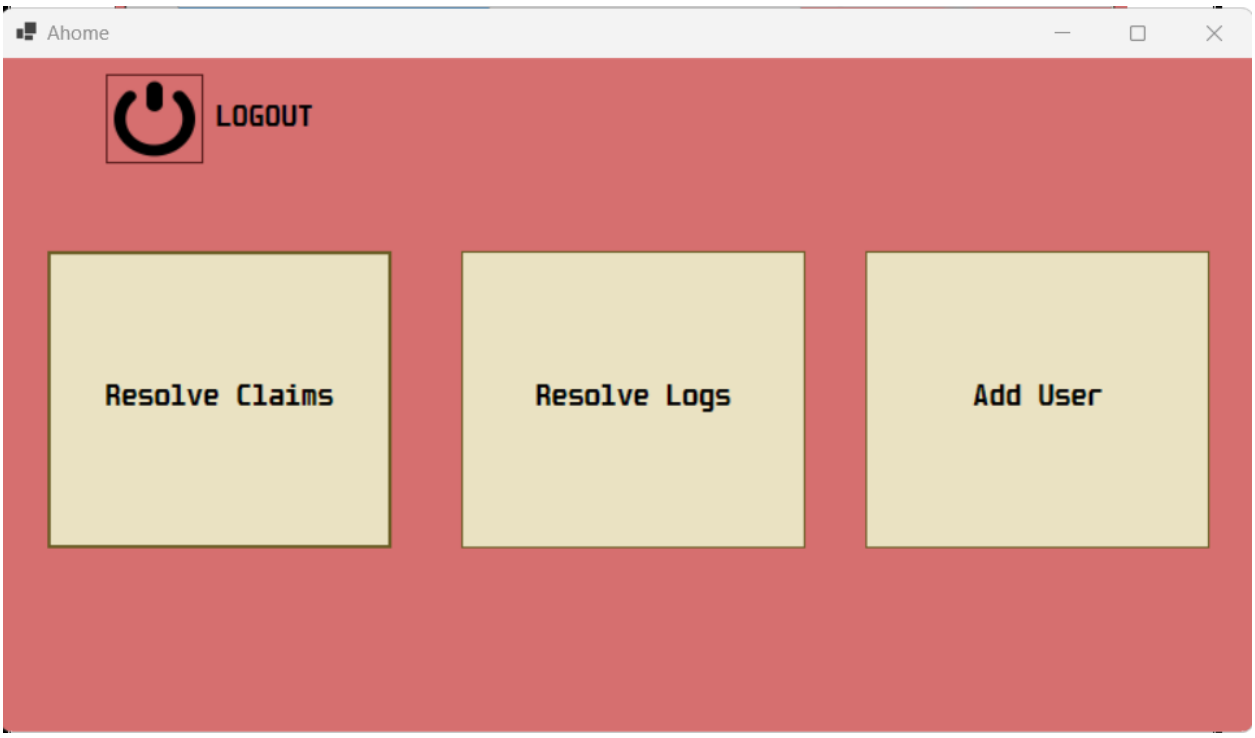


FIG10: ADMIN HOME PAGE

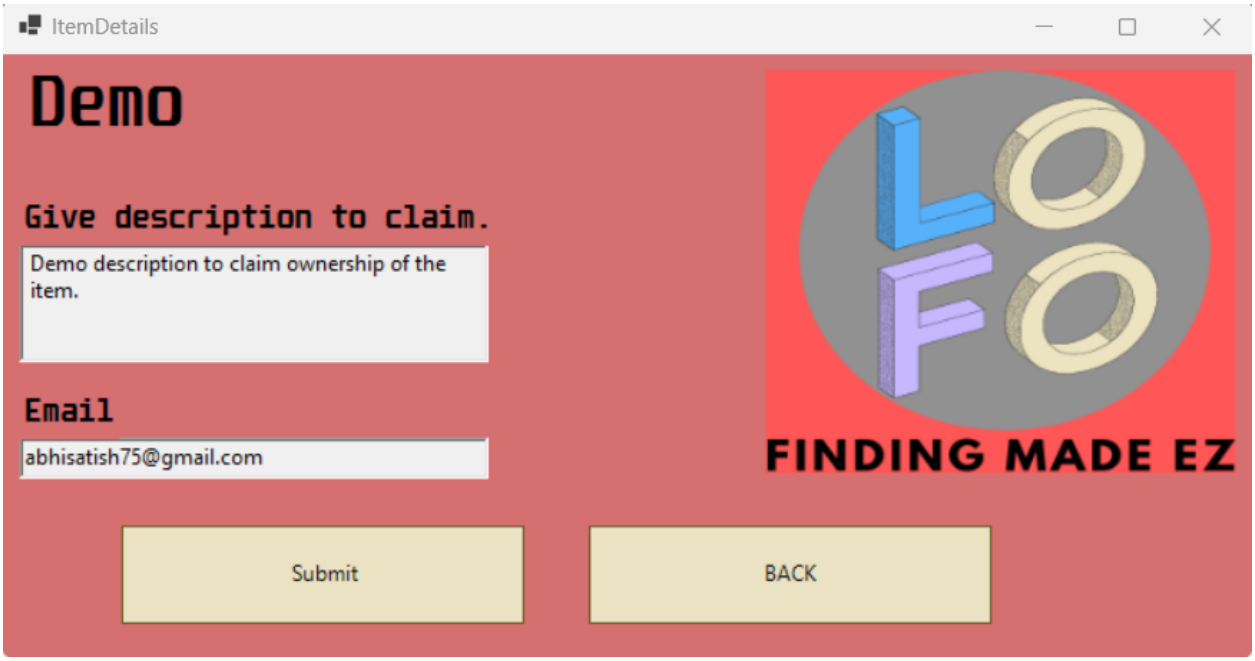


FIG11: CLAIM ITEM

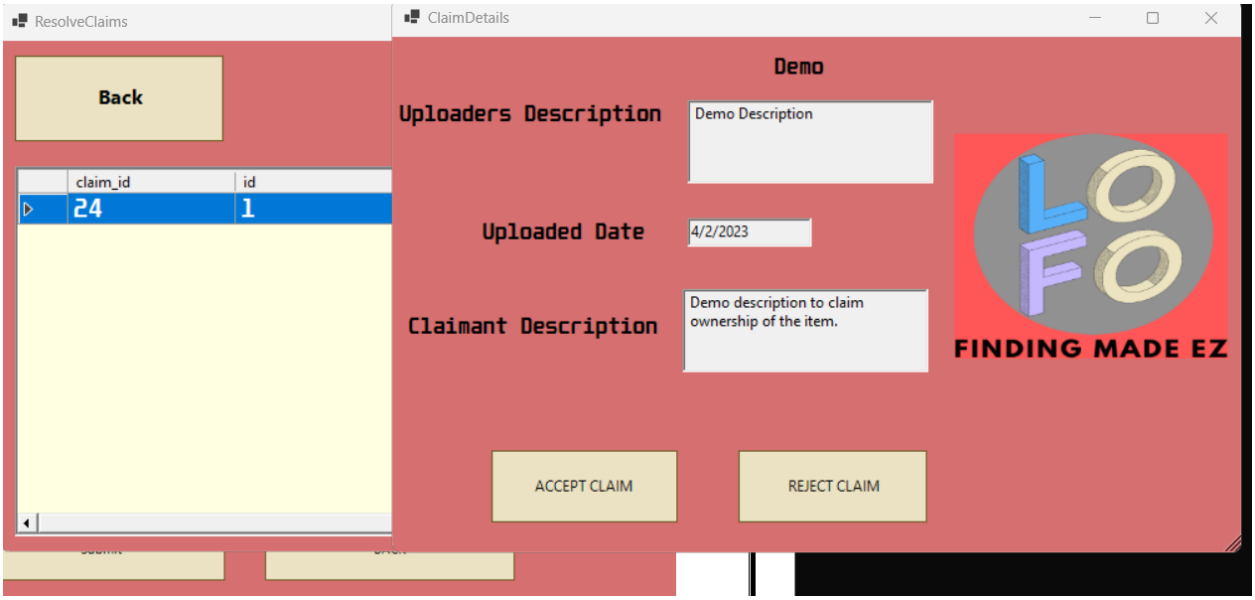


FIG12: RESOLVE CLAIM ITEM

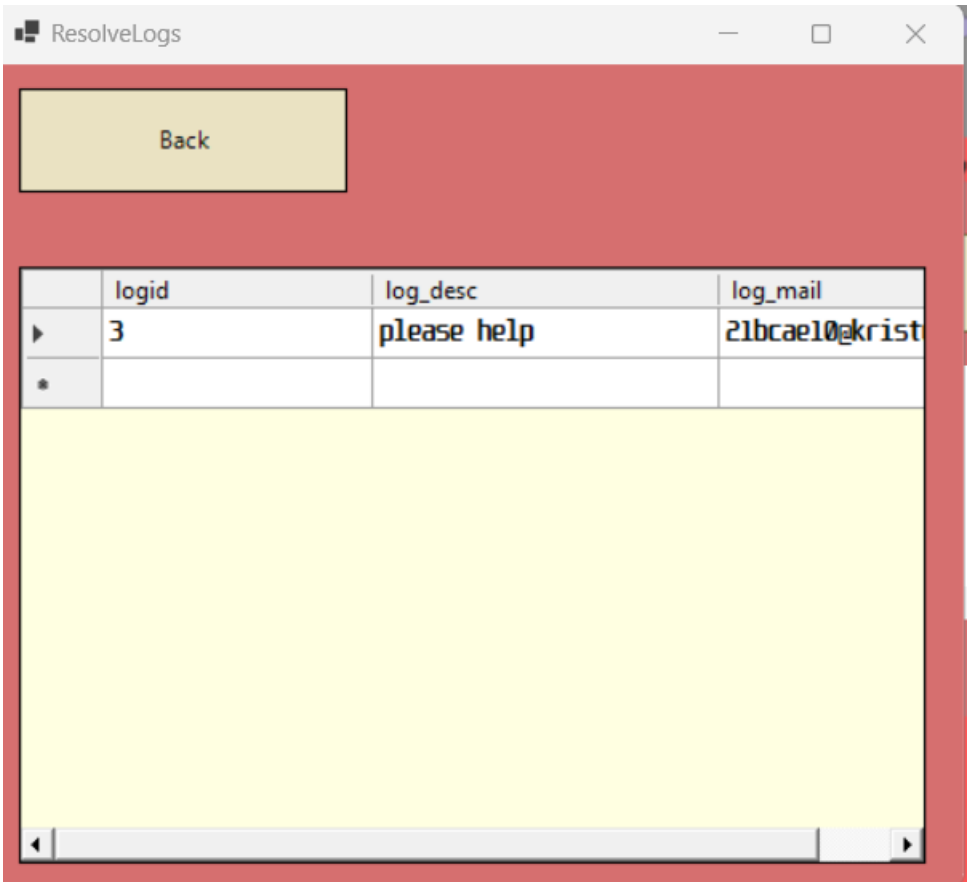


FIG13: RESOLVE LOGS ITEM

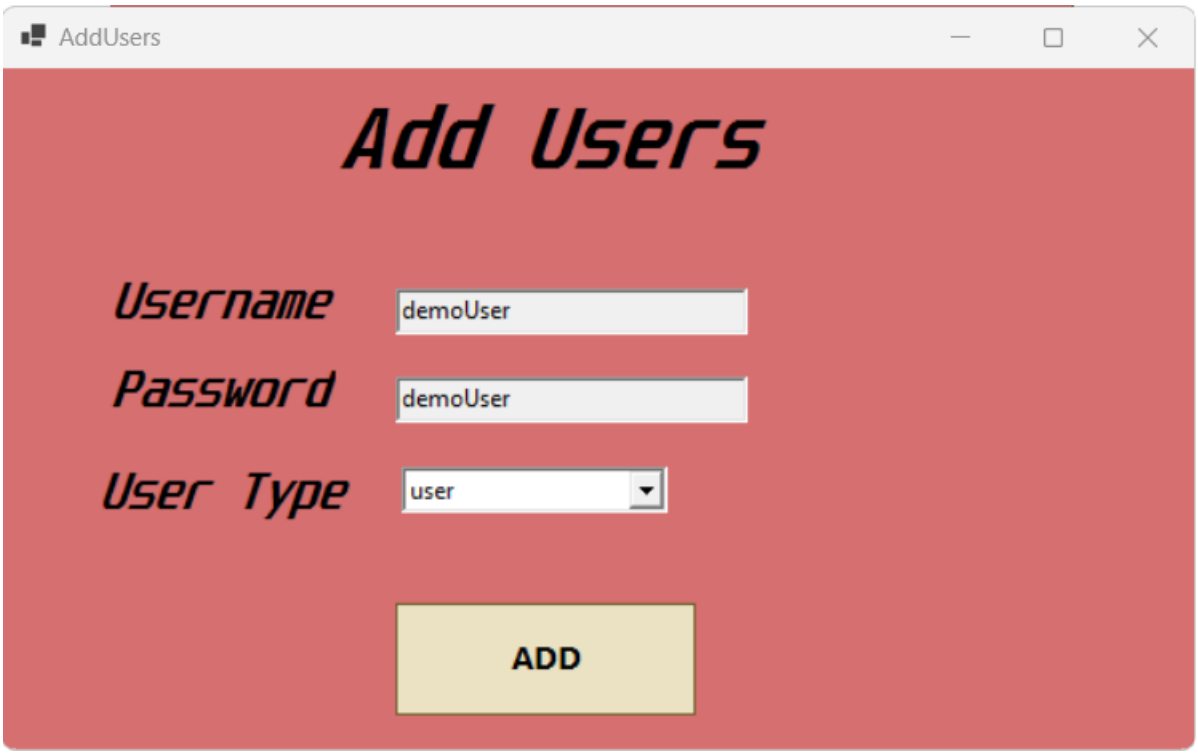


FIG14: ADD NEW USERS/ADMIN PAGE

11. APPENDICES C-SAMPLE REPORT OF TEST CASES:

TABLE 7: SAMPLE REPORT OF TEST CASES

SLNO.	TEST CASE ID	TESTCASE DESCRIPTION	STEPS TO EXECUTE	TEST DATA	EXPECTED RESULT	ACTUAL RESULT	STATUS
1	TID1	CORRECT USERNAME AND PASSWORD.	START AND ENTER CREDENTIALS.	CORRECT USERNAME AND PASSWORD	LOGIN SUCCESSFULL	LOGIN SUCCESS	PASS
2	TID2	WRONG USERNAME AND PASSWORD	START AND ENTER CREDENTIALS	WRONG USERNAME AND PASSWORD	LOGIN FAILED	LOGIN FAILED	PASS
3	TID3	ITEM DETAILS FORM TEST	LOGIN AND OPEN ADD ITEMS PAGE	ENTER ALL FIELDS	ITEM POSTED	ITEM POSTED	PASS
4	TID4	ITEM DETAILS FORM TEST	LOGIN AND OPEN ADD ITEMS PAGE	DO NOT ENTER ALL FIELDS	ITEM NOT POSTED	ITEM NOT POSTED	PASS
5	TID5	ADD LOG PAGE TEST	LOGIN AND OPEN ADD LOG PAGE	ENTER ALL FIELDS	LOG POSTED	LOG POSTED	PASS
6	TID6	ADD LOG PAGE TEST	LOGIN AND OPEN ADD LOG PAGE	DO NOT ENTER ALL FIELDS	LOG NOT POSTED	LOG NOT POSTED	PASS
7	TID7	RESOLVE CLAIM MODULE TEST	OPEN RESOLVE CLAIM PAGE AND PRESS ACCEPT CLAIM	PRESS ACCEPT CLAIM BUTTON	CLAIM ACCEPTED AND CLAIMANT RECIEVES MAIL	CLAIM ACCEPTED AND CLAIMANT RECIEVES MAIL	PASS
8	TID8	RESOLVE CLAIM MODULE TEST	OPEN RESOLVE CLAIM PAGE AND PRESS ACCEPT CLAIM	PRESS REJECT CLAIM BUTTON	CLAIM REJECTED AND CLAIMANT RECIEVES MAIL	CLAIM REJECTED AND CLAIMANT RECIEVES MAIL	PASS