MOTILAL DULICHAND PVT. LTD.

A Project Report

Submitted As Part of the Internship Program

Project Titled:-Return Challan

Submitted By

Vanshika Thavnani

Under The Guidance Of

Mr.Ram Chander

20, Industrial Estate, Kalpi Road,

Shastri Nagar, Kanpur Nagar-208012,

Uttar Pradesh, India

ACKNOWLEGEMENT

I am immensely grateful to Motilal Dulichand for providing me with the opportunity to undertake my internship and complete the project titled "Return Challan System" during the period from 3 June 2024 to 17 July 2024. I extend my heartfelt thanks to Mr. Ram Chander, Head of the EDP Department, whose unwavering guidance, expertise, and encouragement were invaluable throughout the project. His insightful feedback and constructive criticism greatly contributed to refining the project and enhancing my skills in software development and return challan management.

I would also like to acknowledge the support and cooperation received from the entire team at Motilal Dulichand. Their willingness to share knowledge, provide assistance, and offer practical solutions were instrumental in overcoming challenges and achieving project milestones effectively.

Lastly, I express my appreciation to all those who supported and contributed to my internship experience at Motilal Dulichand.

NAME: Vanshika Thavnani
COLLEGE NAME: Kristu Jayanti College, Autonomous Bengaluru
DATE: 17 July 2024

ABSTRACT

This documentation provides a comprehensive overview of a return challan system developed during an internship at Motilal Dulichand, utilizing Visual Basic (VB) for frontend development and SQL for backend database management. The project aimed to improve the efficiency of handling return challans by automating the creation, management, and tracking of these documents. The system's architecture was designed to ensure scalability and reliability, with VB providing a user-friendly interface for users to seamlessly manage return entries, while SQL facilitated secure storage and efficient retrieval of data. Key components covered include a detailed exploration of the database schema, outlining tables, relationships, and data structures employed to manage return challan information and related transactions. This project not only enriched technical skills in software development and database integration but also highlighted the practical application of technology in optimizing business processes and decision-making. Overall, the return challan system project exemplifies the successful application of theoretical knowledge in a real-world setting, underscoring its significance in professional growth and proficiency in information technology.

TABLE OF CONTENTS

CONTENT OF TABLES

INTRODUCTION

During my internship at Motilal Dulichand, I developed a robust return challan software using Visual Basic (VB) and SQL. This software was designed to streamline the process of generating return challans, which are essential documents for recording the return of goods. By automating the creation of these documents, the software ensures that the necessary details are accurately captured and documented, reducing the likelihood of errors and enhancing overall efficiency.

The software boasts several features that significantly enhance operational efficiency. Users can easily post challans, print pending reports in Excel format, and manage entries with a high degree of flexibility. This includes adding new entries, updating existing ones, and deleting challans when necessary. The ability to print pending reports in Excel facilitates better tracking and management of outstanding challans, aiding in timely follow-ups and ensuring smooth business operations. Additionally, the software's user-friendly interface makes it accessible to users with varying levels of technical expertise.

One of the key strengths of this software is its ability to handle large volumes of data with ease, thanks to its underlying SQL database. The database ensures that all entries are stored securely and can be retrieved quickly, providing a reliable and efficient way to manage return challans. The software's integration with Excel further enhances its utility, as it allows for the seamless generation of reports that can be used for analysis and decision-making. This integration ensures that users can maintain comprehensive records of all transactions and easily share them with relevant stakeholders.

This project, developed solely for my learning purposes, provided me with invaluable hands-on experience in software development and database management. It not only deepened my understanding of programming in VB and SQL but also offered insights into the practical aspects of managing return challans in a business environment. Collaborating with professionals at Motilal Dulichand allowed me to create a tool that, while not implemented for operational use, served as a vital learning platform for developing efficient and user-friendly business solutions. This experience has significantly enhanced my skills and prepared me for future challenges in the field of software development.

DATABASE SCHEMA

A database schema is like a blueprint that shows how data is organized in a database. It includes tables with information, how these tables are connected, and rules to keep the data accurate. The database schema for the return challan system I developed during my internship at Motilal Dulichand is very important for making sure everything works well and efficiently. This schema is created using SQL and is designed to store and manage key information related to return challans, working seamlessly with the Visual Basic frontend.

Here are the main parts of the schema:

Challan table:-This table stores the details of each challan, including the ID, liable person, receiver, and dates (both the challan date and return date). It helps in easily accessing and managing return challan information.

Product table:-This table contains details about the products involved in the return challans. It includes information like product ID, item name, quantity, type, and purpose. This table ensures that all product details are accurately recorded and can be retrieved efficiently.

Party table:-This table stores information about the parties involved in the transactions. It includes fields like receiver's name, address, phone number, city, and pincode. This helps in managing contact details and ensuring the correct party is associated with each return challan.

Liable table:-This table stores information about the liable persons involved in the transactions. It includes fields like liable person name, address, phone number, city, and pincode. This helps in managing contact details and ensuring the correct liable person is associated with each return challan.

Status table:-This table tracks the status of each return challan. It includes fields such as ID, post status, and additional details like phone number and address. This table helps in monitoring the progress of each return challan and ensuring that all steps are completed.

These tables are linked together using keys. For example, the Challan table is connected to the Liable Person table, ensuring that each challan is linked to a valid person responsible for the return. Similarly, the Product table is linked to the Challan table to connect product

details with the right challan. These links help keep the data accurate and make it easier to retrieve complex information.

The schema also has rules to ensure data accuracy. For instance, each challan must have a unique identifier, and product quantities should be valid. These rules help prevent errors and keep the data consistent.

This well-designed database schema makes sure that data is stored and retrieved securely and efficiently, following industry standards. It highlights the technical details and careful planning needed to meet the organization's needs. The schema ensures the return challan system can handle different scenarios, adapt to changes, and support future improvements, providing a reliable solution for managing return challans.

Here is the table structure of the tables used:

1-CHALLAN

FIELD	TYPE	NULL	KEY
Id	Numeric(18, 0)	Not Null	Primary Key
Liable Person	Varchar(50)	Null	
Date	Date	Null	
Return Date	Date	Null	
Reciever	Varchar(50)	Null	

TABLE 1:CHALLAN

2-PRODUCT

FIELD	TYPE	NULL	KEY
Id	Numeric(18, 0)	Not Null	Foreign Key
Item	Varchar(50)	Null	
Quantity	Numeric(18, 0)	Null	
Type	Varchar(50)	Null	
Purpose	Varchar(50)	Null	
Si	Int	Null	

TABLE 2:PRODUCT

3-PARTY

FIELD	TYPE	NULL	KEY
Receiver	Varchar(50)	Not	
Address	Varchar(50)	Null	
City	Varchar(50)	Null	
Pincode	Numeric(18, 0)	Null	
Phone	Numeric(18, 0)	Null	

TABLE 3:PARTY

4-LIABLE PERSON

FIELD	TYPE	NULL	KEY
Liable	Varchar(50)	Not	
Address	Varchar(50)	Null	
City	Varchar(50)	Null	
Pincode	Numeric(18, 0)	Null	
Phone	Numeric(18, 0)	Null	

TABLE 4:LIABLE

5-STATUS

FIELD	TYPE	NULL	KEY
Id	Numeric(18, 0)	Null	
Post	Varchar(50)	Null	
Si	Int	Null	

TABLE 5:STATUS

ER DIAGRAM

Entity-Relationship (ER) diagram is a visual representation of the data and the relationships between data within a database. It is a crucial tool in database design and helps in understanding how different entities (objects) interact with each other within the system. Below is an explanation of the components and symbols used in an ER diagram, tailored to the return challan system developed during my internship at Motilal Dulichand.

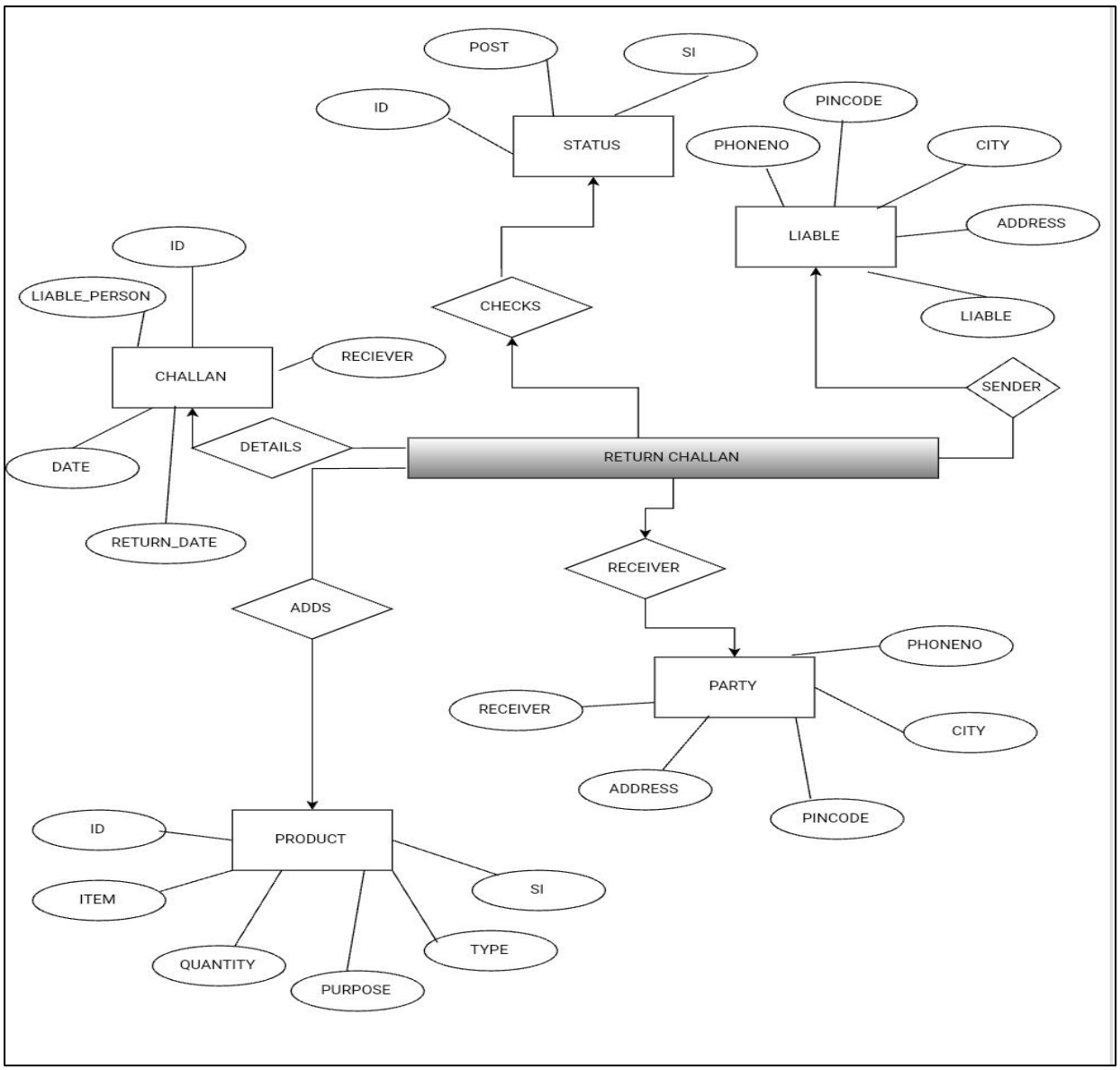
COMPONENTS AND SYMBOLS OF AN ER DIAGRAM

1. **Entity**
    - **Symbol:** Rectangles
    - **Description:** Entities represent real-world objects or concepts that have data stored about them in the database. Examples in the return challan system include "Challan," "Product," and "Party."
2. **Attributes**
    - **Symbol:** Ovals
    - **Description:** Attributes are the properties or details of an entity. For instance, a "Challan" entity may have attributes such as Date, Return Date, and Liable Person. A "Product" entity may have attributes like Item, Quantity, and Type.
3. **Relationships**
    - **Symbol:** Diamonds
    - **Description:** Relationships show how entities are related to each other. The relationship is usually labeled with verbs like "contains," "is sent to," or "is associated with."
4. **Primary Key**
    - **Symbol:** Underlined text inside the entity rectangle
    - **Description:** A primary key is a unique identifier for an entity. It ensures that each record within the entity is unique. For example, ChallanID can be the primary key for the Challan entity.
5. **Cardinality**
    - **Symbol:** Notations like 1:1, 1, N
    - **Description:** Cardinality defines the numerical relationship between entities:
        - **1:1 (One-to-One):** One entity instance is associated with one instance of another entity.

- **1(One-to-Many):** One entity instance is associated with multiple instances of another entity.
- **N(Many-to-Many):** Multiple instances of one entity are associated with multiple instances of another entity.

Understanding ER diagrams and their symbols is essential for designing efficient and well-structured databases, ensuring data integrity, and supporting the system's operational needs effectively. The ER diagram for the return challan system provides a clear blueprint for managing the data and relationships, ensuring smooth operation and future scalability.

ER DIAGRAM FOR RETURN CHALLAN

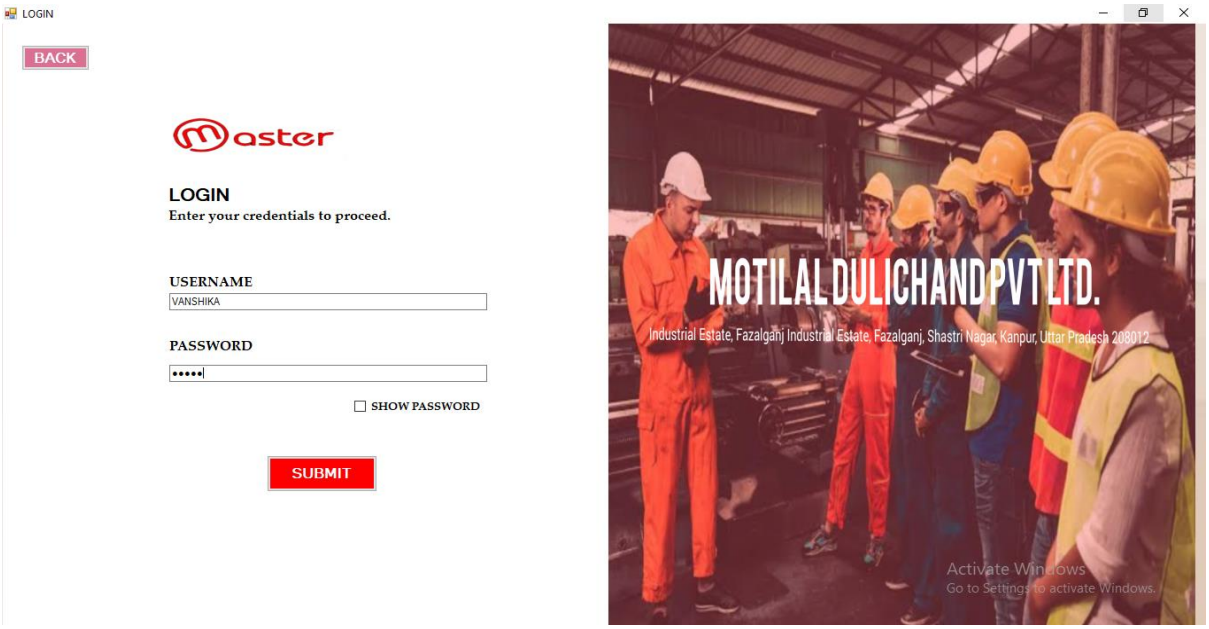


USER INTERFACE

User Interface (UI) in the context of software refers to the visual elements, controls, and interactions through which a user interacts with a system. It encompasses the layout, design, and functionality of the graphical interface that users interact with to perform tasks and access information within an application or system. Below are the components and a description of the UI for the return challan system developed during my internship at Motilal Dulichand.

FORM1

LOGIN

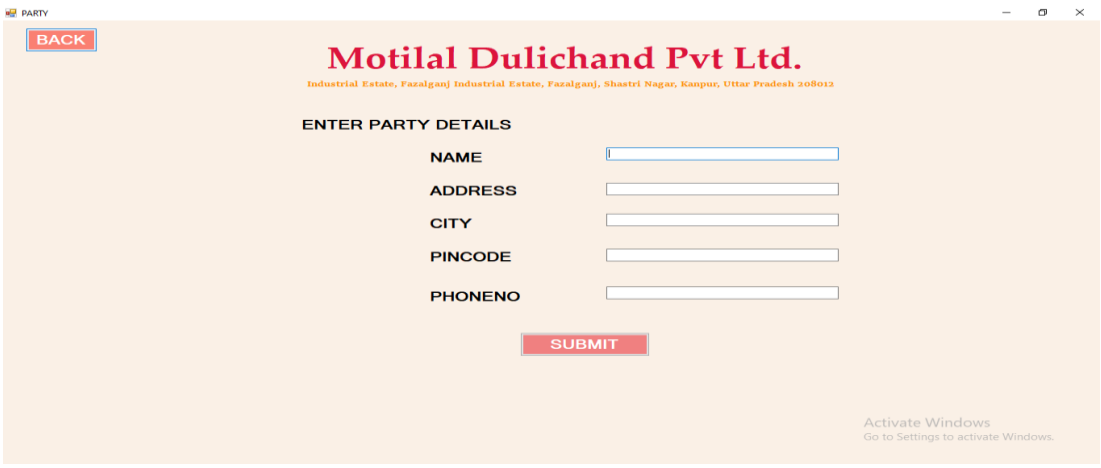


HOME PAGE

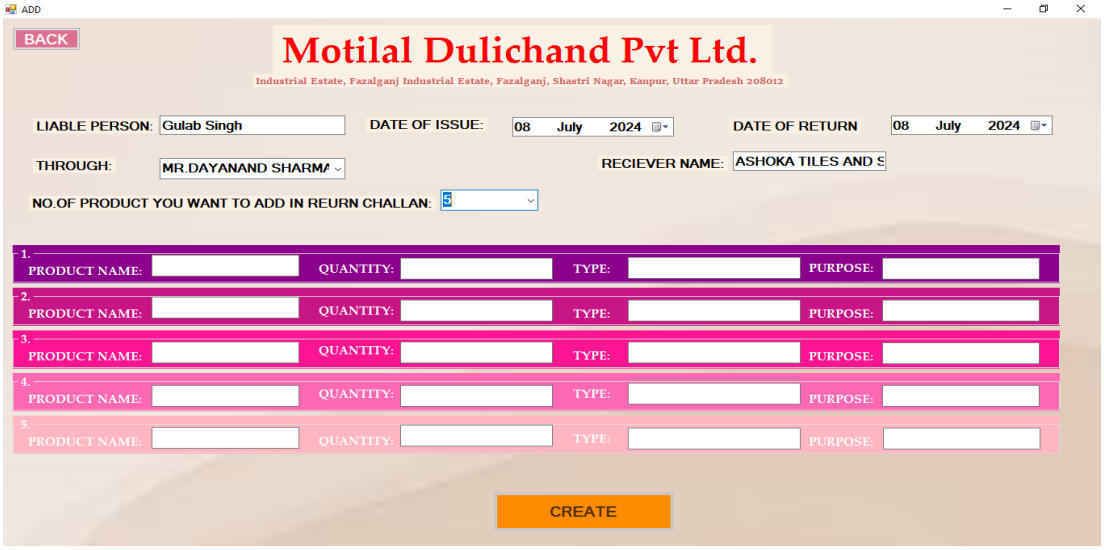
ADD

1.1 LIABLE PERSON

1.2 PARTY



1.3 CHALLAN ENTRY



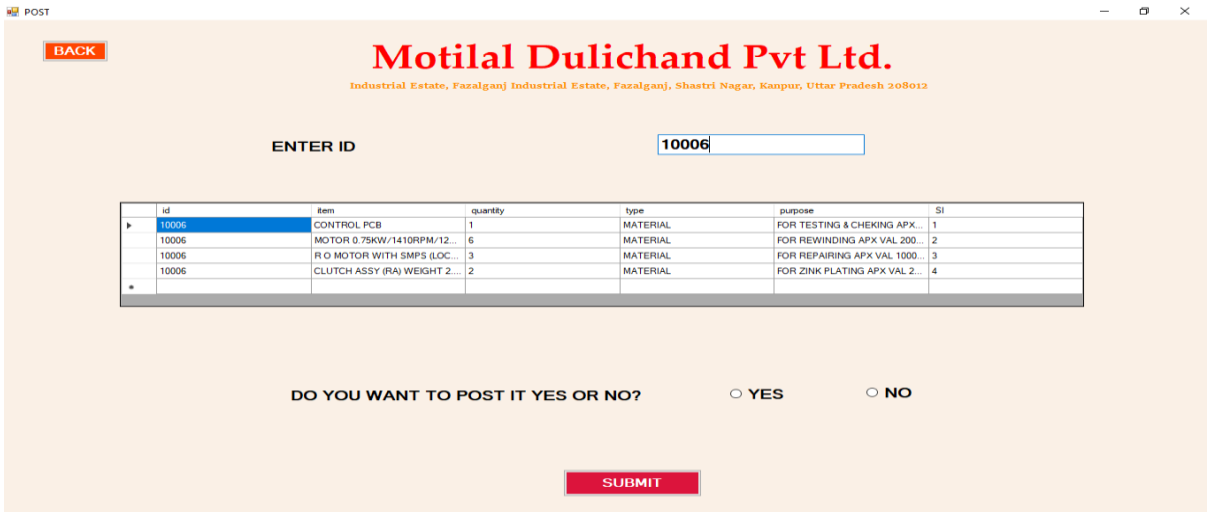
CHECK STATUS

DELETE

GENERATE

id	liable person	Date	return_date	reciever	item	quantity	type	purpose
10005	SARVE DEO PANDEY	28/06/2024	30/06/2024	KUMAR BERTAN B...	PULLY FOR SAMPLE	5	SAMPLE	FOR SAMPLE APX V...
10006	Naresh Narain Mishra	28/06/2024	28/06/2024	TEXPLUS FIBRES P...	CONTROL PCB	1	MATERIAL	FOR TESTING & CH...
10006	Naresh Narain Mishra	28/06/2024	28/06/2024	TEXPLUS FIBRES P...	MOTOR 0.75KW/14...	6	MATERIAL	FOR REWINDING A...
10006	Naresh Narain Mishra	28/06/2024	28/06/2024	TEXPLUS FIBRES P...	R O MOTOR WITH ...	3	MATERIAL	FOR REPAIRING AP...
10006	Naresh Narain Mishra	28/06/2024	28/06/2024	TEXPLUS FIBRES P...	CLUTCH ASSY (RA) ...	2	MATERIAL	FOR ZINK PLATING ...
10007	SHIV KUMAR	28/06/2024	28/06/2024	LOHIA STARLINGE...	MONITOR	1	IP	FOR ZINK PLATING ...
10007	SHIV KUMAR	28/06/2024	28/06/2024	LOHIA STARLINGE...	COMPUTER	2	IP	FOR ZINK PLATING ...

POST



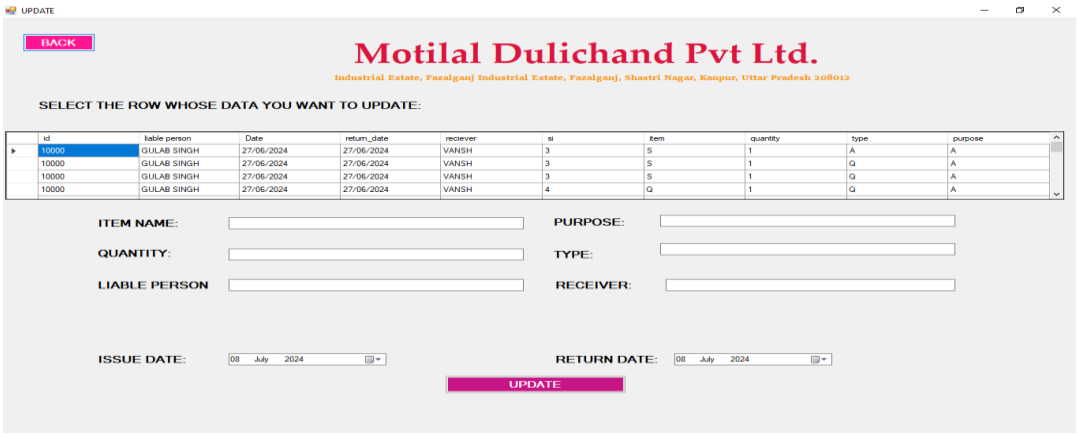
id	item	quantity	type	purpose	Sl
10006	CONTROL PCB	1	MATERIAL	FOR TESTING & CHEKING APX...	1
10006	MOTOR 0.75KW/1410RPM/12...	6	MATERIAL	FOR REWINDING APX VAL 200...	2
10006	R O MOTOR WITH SMPS (LOC...	3	MATERIAL	FOR REPAIRING APX VAL 1000...	3
10006	CLUTCH ASSY (RA) WEIGHT 2....	2	MATERIAL	FOR ZINK PLATING APX VAL 2...	4

PENDING

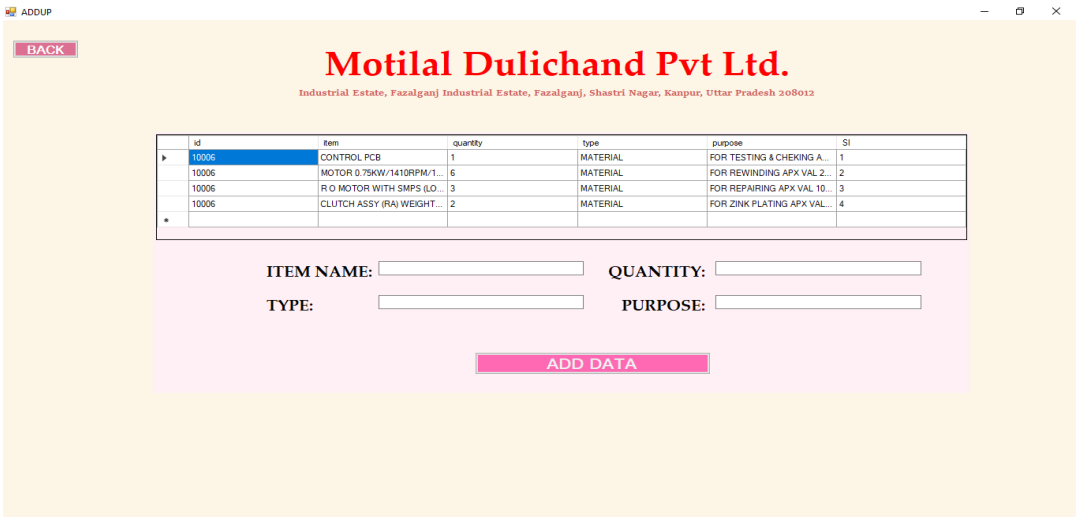
ChallanID	liable person	reciever	date	RETURN_DATE	item	quantity	TYPE	PURPOSE
10003	Gulab Singh	LOHIA STARLINGER LIMITED	28/06/2024	28/06/2024	BOBBIN 1680/1/100 (Z) WITH 3 KG MATERIAL	2	MATERIAL	USE
10004	ANIL KUMAR SAVITA	KHANNA PHARMACUTICALS AND CHEMICALS	28/06/2024	28/06/2024	SPINDLE ST	1	MATERIAL	FOR REPAIRING APX VAL 500.00
10004	ANIL KUMAR SAVITA	KHANNA PHARMACUTICALS AND CHEMICALS	28/06/2024	28/06/2024	BUSH 8X4	1	MATERIAL	FOR REPAIRING APX VAL 500.00
10005	SARVE DEO PANDEY	KUMAR BERTAN BHANDAR	28/06/2024	30/06/2024	PULLY FOR SAMPLE	5	SAMPLE	FOR SAMPLE APX VAL 500.00
10006	Naresh Narain Mishra	TEXPLUS FIBRES Private LTD	28/06/2024	28/06/2024	CONTROL PCB	1	MATERIAL	FOR TESTING & CHEKING APX VAL
10006	Naresh Narain Mishra	TEXPLUS FIBRES Private LTD	28/06/2024	28/06/2024	MOTOR 0.75KW/1410RPM/1211000	6	MATERIAL	FOR REWINDING APX VAL 2000.00
10006	Naresh Narain Mishra	TEXPLUS FIBRES Private LTD	28/06/2024	28/06/2024	R O MOTOR WITH SMPS (LOCAL)	3	MATERIAL	FOR REPAIRING APX VAL 1000.00
10006	Naresh Narain Mishra	TEXPLUS FIBRES Private LTD	28/06/2024	28/06/2024	CLUTCH ASSY (RA) WEIGHT 2.320 KGS	2	MATERIAL	FOR ZINK PLATING APX VAL 2000.
10007	SHIV KUMAR	LOHIA STARLINGER LIMITED	28/06/2024	28/06/2024	MONITOR	1	IP	FOR ZINK PLATING APX VAL 2000.
10007	SHIV KUMAR	LOHIA STARLINGER LIMITED	28/06/2024	28/06/2024	COMPUTER	2	IP	FOR ZINK PLATING APX VAL 2000.
10008	Gulab Singh	LOHIA STARLINGER LIMITED	06/07/2024	06/07/2024	FFFFFFFFFFFFF	1	FFFF	LLL
10009	Gulab Singh	KHANNA PHARMACUTICALS AND CHEMICALS	11/07/2024	25/07/2024	COMPUTR(SALES)	1	D	LLL
10009	Gulab Singh	KHANNA PHARMACUTICALS AND CHEMICALS	11/07/2024	25/07/2024	MOUSE	2	D	W
10010	SARVE DEO PANDEY	LOHIA STARLINGER LIMITED	06/07/2024	06/07/2024	DDD	2	S	S
10011	ANIL KUMAR SAVITA	KUMAR BERTAN BHANDAR	06/07/2024	06/07/2024	SS	1	D	X
10014	Gulab Singh	LOHIA STARLINGER LIMITED	06/07/2024	06/07/2024	DD	1	T	W
10014	Gulab Singh	LOHIA STARLINGER LIMITED	06/07/2024	06/07/2024	D	2	G	Q
10014	Gulab Singh	LOHIA STARLINGER LIMITED	06/07/2024	06/07/2024	F	3	V	W
10001	LALIT	DD	27/06/2024	27/06/2024	W	1	W	E

UPDATE

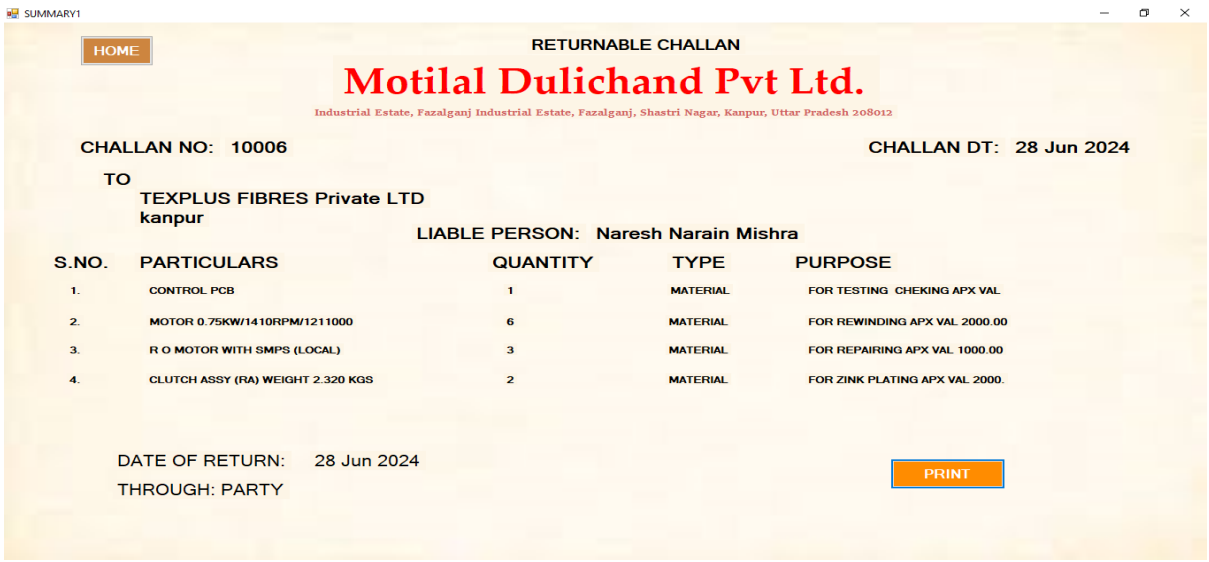
1.1 EDIT



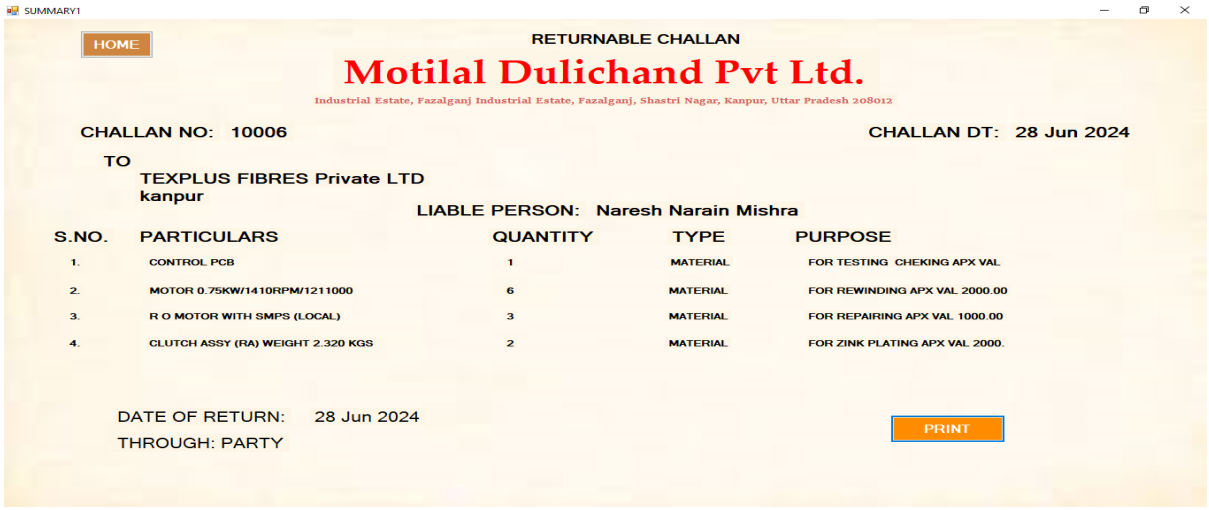
1.2 ADD



SUMMARY



SUMMARY1



REPORT PRINT

GSTIN NO: 09AAAACD8231ZZ

RETURNABLE CHALLAN

Motilal Dulichand (P) Ltd.

20, Industrial Estate, Kanpur-208012

Challan No: 10006 Challan Dt: 28 Jun 2024

TO,
TEXPLUS FIBRES Private LTD
kanpur

Liable Person: Naresh Narain Mishra

S. No.	PARTICULARS	Quantity	TYPE	Purpose
1	CONTROL PCB	1	MATERIAL	FOR TESTING & CHEKING APX VAL
2	MOTOR 0.75KW/1410RPM/1211000	6	MATERIAL	FOR REWINDING APX VAL 2000.00
3	R O MOTOR WITH SMPS (LOCAL)	3	MATERIAL	FOR REPAIRING APX VAL 1000.00
4	CLUTCH ASSY (RA) WEIGHT 2.320 KGS	2	MATERIAL	FOR ZINK PLATING APX VAL 2000.

Date of Return: 28 Jun 2024 Through: PARTY

Sign. Mech Store Incharge | Forwarded By (Section Head) | Approved By

material returned in store sign mech store incharge | quality checked by mech Section Head) | posted computer dt. by.

PENDING CHALLANS EXCEL SHEET

Motilal Dulichand

Pending Challan

ChallanID	liable person	reciever	date	RETURN_DATE	item	quantity	TYPE	PURPOSE
10003	Gulab Singh	LOHIA STARLINGER LIMITED	28-06-2024	28-06-2024	BOBBIN 1680/1/100 (Z) WITH 3 KG MATERIA	2	MATERIAL	USE
10004	ANIL KUMAR SAVITA	KHANNA PHARMACUTICALS AND CHEMICALS	28-06-2024	28-06-2024	SPINDLE ST	1	MATERIAL	FOR REPAIRING APX VAL 500.00
10004	ANIL KUMAR SAVITA	KHANNA PHARMACUTICALS AND CHEMICALS	28-06-2024	28-06-2024	BUSH 8X4	1	MATERIAL	FOR REPAIRING APX VAL 500.00
10005	SARVE DEO PANDEY	KUMAR BERTAN BHANDAR	28-06-2024	30-06-2024	PULLY FOR SAMPLE	5	SAMPLE	FOR SAMPLE APX VAL 500.00
10006	Naresh Narain Mishi	TEXPLUS FIBRES Private LTD	28-06-2024	28-06-2024	CONTROL PCB	1	MATERIAL	FOR TESTING & CHEKING APX VAL
10006	Naresh Narain Mishi	TEXPLUS FIBRES Private LTD	28-06-2024	28-06-2024	MOTOR 0.75KW/1410RPM/1211000	6	MATERIAL	FOR REWINDING APX VAL 2000.00
10006	Naresh Narain Mishi	TEXPLUS FIBRES Private LTD	28-06-2024	28-06-2024	R O MOTOR WITH SMPS (LOCAL)	3	MATERIAL	FOR REPAIRING APX VAL 1000.00
10006	Naresh Narain Mishi	TEXPLUS FIBRES Private LTD	28-06-2024	28-06-2024	CLUTCH ASSY (RA) WEIGHT 2.320 KGS	2	MATERIAL	FOR ZINK PLATING APX VAL 2000.
10007	SHIV KUMAR	LOHIA STARLINGER LIMITED	28-06-2024	28-06-2024	MONITOR	1	IP	FOR ZINK PLATING APX VAL 2000.
10007	SHIV KUMAR	LOHIA STARLINGER LIMITED	28-06-2024	28-06-2024	COMPUTER	2	IP	FOR ZINK PLATING APX VAL 2000.

CODE IMPLEMENTATION

Code implementation refers to the process of writing the actual code that brings a software design to life. It involves translating design specifications and requirements into a functional program using programming languages. This includes developing the frontend and backend components, integrating them, performing initial testing, and optimizing for efficiency.

Here is the code for all the forms in the return challan:

FORM1

```
Public Class Form1
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Me.Hide()
        LOGIN.Show()
    End Sub

    Private Sub PictureBox3_Click(sender As Object, e As EventArgs) Handles
PictureBox3.Click
        Dim result As DialogResult = MessageBox.Show("Are you sure you want to
exit the application?",
"Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Question)
        ' If the user confirms, exit the application
        If result = DialogResult.Yes Then
            Application.Exit()
        End If
    End Sub
End Class
```

LOGIN

```
Public Class LOGIN
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Dim username As String = TextBox1.Text.Trim()
        Dim password As String = TextBox2.Text
        If String.IsNullOrWhiteSpace(username) OrElse
String.IsNullOrWhiteSpace(password) Then
            MessageBox.Show("Username and password are required.")
            Return
        End If
        If username = "VANSHIKA" AndAlso password = "12345" Then
            Me.Hide()
            ADMINHOME.Show()
        Else
            MessageBox.Show("Invalid username or password.")
        End If
    End Sub
    Private Sub CheckBox1_CheckedChanged(sender As Object, e As EventArgs)
Handles CheckBox1.CheckedChanged
        If CheckBox1.Checked = True Then
            TextBox2.UseSystemPasswordChar = False
```

```
        ElseIf CheckBox1.Checked = False Then
            TextBox2.UseSystemPasswordChar = True
        End If
    End Sub
    Private Sub LOGIN_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        If CheckBox1.Checked = False Then
            TextBox2.UseSystemPasswordChar = True
        End If
    End Sub
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
        Form1.Show()
    End Sub
End Class
```

HOME PAGE

```
Public Class ADMINHOME
    Private Sub LOGOUTToolStripMenuItem_Click(sender As Object, e As
EventArgs) Handles LOGOUTToolStripMenuItem.Click
        Dim result As DialogResult
        result = MessageBox.Show("ARE YOU SURE YOU WANT TO LOGOUT",
"Confirmation",
        MessageBoxButtons.OKCancel, MessageBoxIcon.Question)
        If result = DialogResult.OK Then
            Me.Close()
            Form1.Show()
        End If
    End Sub
    Private Sub GENERATEToolStripMenuItem_Click(sender As Object, e As
EventArgs) Handles GENERATEToolStripMenuItem.Click
        Me.Close()
        GENERATE.Show()
    End Sub

    Private Sub CHECKSTATUSToolStripMenuItem_Click(sender As Object, e As
EventArgs) Handles CHECKSTATUSToolStripMenuItem.Click
        Me.Close()
        STATUS.Show()
    End Sub

    Private Sub DELETEToolStripMenuItem_Click(sender As Object, e As
EventArgs) Handles DELETEToolStripMenuItem.Click
        Me.Close()
        DEL.Show()
    End Sub

    Private Sub POSTToolStripMenuItem_Click(sender As Object, e As EventArgs)
Handles POSTToolStripMenuItem.Click
        Me.Close()
        POST.Show()
    End Sub

    Private Sub PRODUCTToolStripMenuItem_Click(sender As Object, e As
EventArgs) Handles PRODUCTToolStripMenuItem.Click
        Me.Close()
        ADD.Show()
    End Sub
```

```
    Private Sub LIABLEPERSONToolStripMenuItem_Click(sender As Object, e As
EventArgs) Handles LIABLEPERSONToolStripMenuItem.Click
        Me.Close()
        LIABLE.Show()
    End Sub

    Private Sub PARTYToolStripMenuItem_Click(sender As Object, e As EventArgs)
Handles PARTYToolStripMenuItem.Click
        Me.Close()
        PARTY.Show()
    End Sub
    Private Sub PENDINNGToolStripMenuItem_Click(sender As Object, e As
EventArgs) Handles PENDINNGToolStripMenuItem.Click
        Me.Close()
        PENDING.Show()
    End Sub
    Private Sub EDITToolStripMenuItem_Click(sender As Object, e As EventArgs)
Handles EDITToolStripMenuItem.Click
        Me.Close()
        UPDATE1.Show()
    End Sub
    Private Sub ADDToolStripMenuItem_Click(sender As Object, e As EventArgs)
Handles ADDToolStripMenuItem.Click
        Me.Close()
        ADDUP.Show()
    End Sub
End Class
```

ADD

LIABLE PERSON

```
Imports System.Data.SqlClient

Public Class LIABLE

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

        Dim liable As String = TextBox1.Text.Trim()

        Dim address As String = TextBox2.Text.Trim()

        Dim city As String = TextBox3.Text.Trim()

        Dim pincode As String = TextBox4.Text.Trim()

        Dim phone As String = TextBox5.Text.Trim()

        ' Check if any field is empty

        If String.IsNullOrEmpty(liable) OrElse

           String.IsNullOrEmpty(address) OrElse

           String.IsNullOrEmpty(city) OrElse

           String.IsNullOrEmpty(pincode) OrElse
```

```
        String.IsNullOrEmpty(phone) Then


            MessageBox.Show("Please fill in all fields.", "Missing Information",
MessageBoxButtons.OK, MessageBoxIcon.Warning)

            Return

        End If

        ' Validate phone number format

        If Not System.Text.RegularExpressions.Regex.IsMatch(phone, "^\d{10}$")
Then

            MessageBox.Show("Phone number should be exactly ten digits.", "Invalid
Phone Number", MessageBoxButtons.OK, MessageBoxIcon.Warning)

            Return

        End If

        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

        ' SQL query to insert data into the LIABLE table

        Dim query As String = "INSERT INTO [dbo].[LIABLE] (LIABLE, ADDRESS,
CITY, PINCODE, PHONE) " &

                              "VALUES (@LIABLE, @Address, @City, @Pincode, @Phone)"

        Try

            Using connection As New SqlConnection(connectionString)

                Using command As New SqlCommand(query, connection)

                    ' Add parameters to prevent SQL injection

                    command.Parameters.AddWithValue("@LIABLE", liable)

                    command.Parameters.AddWithValue("@Address", address)

                    command.Parameters.AddWithValue("@City", city)

                    command.Parameters.AddWithValue("@Pincode", pincode)

                    command.Parameters.AddWithValue("@Phone", phone)

                    connection.Open()

                    command.ExecuteNonQuery()

                    MessageBox.Show("Data inserted successfully.")

                End Using

            End Using
```

```
        Catch ex As Exception
            MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
        ADMINHOME.Show()
    End Sub
End Class
```

PARTY

```
  Imports System.Data.SqlClient
Public Class PARTY
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Dim receiver As String = TextBox1.Text.Trim()
        Dim address As String = TextBox2.Text.Trim()
        Dim city As String = TextBox3.Text.Trim()
        Dim pincode As String = TextBox4.Text.Trim()
        Dim phone As String = TextBox5.Text.Trim()
        ' Check if any field is empty
        If String.IsNullOrEmpty(receiver) OrElse
           String.IsNullOrEmpty(address) OrElse
           String.IsNullOrEmpty(city) OrElse
           String.IsNullOrEmpty(pincode) OrElse
           String.IsNullOrEmpty(phone) Then
            MessageBox.Show("Please fill in all fields.", "Missing Information",
MessageBoxButtons.OK, MessageBoxIcon.Warning)
            Return
        End If
        ' Validate phone number format
        If Not System.Text.RegularExpressions.Regex.IsMatch(phone, "^\d{10}$")
Then
            MessageBox.Show("Phone number should be exactly ten digits.", "Invalid
Phone Number", MessageBoxButtons.OK, MessageBoxIcon.Warning)
            Return
        End If
        ' Validate pincode format (if necessary)
        ' Example: Ensure pincode is exactly six digits
        If Not System.Text.RegularExpressions.Regex.IsMatch(pincode, "^\d{6}$")
Then
            MessageBox.Show("Pincode should be exactly six digits.", "Invalid Pincode",
MessageBoxButtons.OK, MessageBoxIcon.Warning)
            Return
        End If
```

```
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

        ' SQL query to insert data into the PARTY table
        Dim query As String = "INSERT INTO [dbo].[PARTY] (RECIEVER,
ADDRESS, CITY, PINCODE, PHONE) " &
                              "VALUES (@Receiver, @Address, @City, @Pincode, @Phone)"

        Try
            Using connection As New SqlConnection(connectionString)
                Using command As New SqlCommand(query, connection)
                    ' Add parameters to prevent SQL injection
                    command.Parameters.AddWithValue("@Receiver", receiver)
                    command.Parameters.AddWithValue("@Address", address)
                    command.Parameters.AddWithValue("@City", city)
                    command.Parameters.AddWithValue("@Pincode", pincode)
                    command.Parameters.AddWithValue("@Phone", phone)

                    connection.Open()
                    command.ExecuteNonQuery()

                    MessageBox.Show("PARTY ADDED SUCESSFULLY.")
                End Using
            End Using
        Catch ex As Exception
            MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
        ADMINHOME.Show()
    End Sub
End Class
```

CHALLAN ENTRY

```
Imports System.Data.SqlClient


Public Class ADD

    Dim CONNECTION_STRING As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

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

        ListBox1.Visible = False

        ListBox2.Visible = False

        GroupBox1.Visible = False

        GroupBox2.Visible = False

        GroupBox3.Visible = False
```

```
        GroupBox4.Visible = False
        GroupBox5.Visible = False
        Button1.Enabled = False
    End Sub
    Private Sub ComboBox1_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles ComboBox1.SelectedIndexChanged
        If ComboBox1.SelectedItem = "1" Then
            GroupBox1.Visible = True
            GroupBox2.Visible = False
            GroupBox3.Visible = False
            GroupBox4.Visible = False
            GroupBox5.Visible = False
        ElseIf ComboBox1.SelectedItem = "2" Then
            GroupBox1.Visible = True
            GroupBox2.Visible = True
            GroupBox3.Visible = False
            GroupBox4.Visible = False
            GroupBox5.Visible = False
        ElseIf ComboBox1.SelectedItem = "3" Then
            GroupBox1.Visible = True
            GroupBox2.Visible = True
            GroupBox3.Visible = True
            GroupBox4.Visible = False
            GroupBox5.Visible = False
        ElseIf ComboBox1.SelectedItem = "4" Then
            GroupBox1.Visible = True
            GroupBox2.Visible = True
            GroupBox3.Visible = True
            GroupBox4.Visible = True
            GroupBox5.Visible = False
        ElseIf ComboBox1.SelectedItem = "5" Then
            GroupBox1.Visible = True
```

```
            GroupBox2.Visible = True

            GroupBox3.Visible = True

            GroupBox5.Visible = True

            GroupBox4.Visible = True

        Else

            MessageBox.Show("PLEASE SELECT ")

        End If

        ValidateForm()

    End Sub

    Private Sub ValidateForm()

        ' Check if all required textboxes are filled

        Dim allFilled As Boolean = Not (String.IsNullOrWhiteSpace(TextBox22.Text)
Or

                                        String.IsNullOrWhiteSpace(TextBox21.Text) Or

                                        (GroupBox1.Visible AndAlso
(String.IsNullOrWhiteSpace(TextBox1.Text) Or
String.IsNullOrWhiteSpace(TextBox2.Text) Or
String.IsNullOrWhiteSpace(TextBox3.Text) Or
String.IsNullOrWhiteSpace(TextBox4.Text))) Or

                                        (GroupBox2.Visible AndAlso
(String.IsNullOrWhiteSpace(TextBox8.Text) Or
String.IsNullOrWhiteSpace(TextBox7.Text) Or
String.IsNullOrWhiteSpace(TextBox6.Text) Or
String.IsNullOrWhiteSpace(TextBox5.Text))) Or

                                        (GroupBox3.Visible AndAlso
(String.IsNullOrWhiteSpace(TextBox12.Text) Or
String.IsNullOrWhiteSpace(TextBox11.Text) Or
String.IsNullOrWhiteSpace(TextBox10.Text) Or
String.IsNullOrWhiteSpace(TextBox9.Text))) Or

                                        (GroupBox4.Visible AndAlso
(String.IsNullOrWhiteSpace(TextBox16.Text) Or
String.IsNullOrWhiteSpace(TextBox15.Text) Or
String.IsNullOrWhiteSpace(TextBox14.Text) Or
String.IsNullOrWhiteSpace(TextBox13.Text))) Or

                                        (GroupBox5.Visible AndAlso
(String.IsNullOrWhiteSpace(TextBox20.Text) Or
String.IsNullOrWhiteSpace(TextBox19.Text) Or
String.IsNullOrWhiteSpace(TextBox18.Text) Or
String.IsNullOrWhiteSpace(TextBox17.Text))))
```

```
        ' Enable or disable the button based on whether all required fields are filled
        Button1.Enabled = allFilled
    End Sub
    Private Sub TextBox_TextChanged(sender As Object, e As EventArgs) Handles
TextBox22.TextChanged, TextBox21.TextChanged, TextBox1.TextChanged,
TextBox2.TextChanged, TextBox3.TextChanged, TextBox4.TextChanged,
TextBox8.TextChanged, TextBox7.TextChanged, TextBox6.TextChanged,
TextBox5.TextChanged, TextBox12.TextChanged, TextBox11.TextChanged,
TextBox10.TextChanged, TextBox9.TextChanged, TextBox16.TextChanged,
TextBox15.TextChanged, TextBox14.TextChanged, TextBox13.TextChanged,
TextBox20.TextChanged, TextBox19.TextChanged, TextBox18.TextChanged,
TextBox17.TextChanged
        ValidateForm()
    End Sub
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Using connection As New SqlConnection(CONNECTION_STRING)
            Try
                connection.Open()
                ' Generate unique ID
                Dim newID As Integer = GenerateUniqueID(connection)
                ' Insert into challan table
                Dim liablePerson As String = TextBox22.Text
                Dim issueDate As DateTime = DateTimePicker1.Value
                Dim returnDate As DateTime = DateTimePicker2.Value
                Dim receiver As String = TextBox21.Text
                InsertChallan(connection, newID, liablePerson, issueDate, returnDate,
receiver)
                ' Insert into product table
                ' Insert into product table
                Dim products As List(Of (String, Integer, String, String, Integer)) =
GetProducts()
                For Each product In products
                    InsertProduct(connection, newID, product.Item1, product.Item2,
product.Item3, product.Item4, product.Item5)
                Next
```

```
            Me.Close()
            Dim COMBO As String = ComboBox2.SelectedItem.ToString()
            ' Show the summary form
            Dim summaryForm As New Summary()
            summaryForm.UpdateSummary(newID, liablePerson, issueDate,
returnDate, receiver, products, COMBO)
            summaryForm.ShowDialog()
            MessageBox.Show("Records inserted successfully.", "Success")
        Catch ex As Exception
            MessageBox.Show("An error occurred: " & ex.Message, "Error")
        Finally
            If connection.State = ConnectionState.Open Then
                connection.Close()
            End If
        End Try
    End Using
End Sub
Private Function GenerateUniqueID(connection As SqlConnection) As Integer
    Dim newID As Integer
    ' Define the query to get the highest ID from the table
    Dim query As String = "SELECT ISNULL(MAX([id]), 9999) FROM [challan]"
    Using command As New SqlCommand(query, connection)
        ' Execute the query and get the highest ID
        Dim hiGhestID As Integer = Convert.ToInt32(command.ExecuteScalar())
        ' Increment the highest ID by one for the new ID
        newID = highestID + 1
    End Using
    Return newID
End Function
Private Sub InsertChallan(connection As SqlConnection, id As Integer,
liablePerson As String, issueDate As DateTime, returnDate As DateTime, receiver
As String)
```

```
        Dim query As String = "INSERT INTO [challan] ([id], [liable person], [date], [return_date], [reciever]) VALUES (@id, @liablePerson, @date, @returnDate, @receiver)"

        Using command As New SqlCommand(query, connection)

            command.Parameters.AddWithValue("@id", id)

            command.Parameters.AddWithValue("@liablePerson", liablePerson)

            command.Parameters.AddWithValue("@date", issueDate)

            command.Parameters.AddWithValue("@returnDate", returnDate)

            command.Parameters.AddWithValue("@receiver", receiver)

            command.ExecuteNonQuery()

        End Using

    End Sub

    Private Sub InsertProducts(connection As SqlConnection, id As Integer)

        Dim products As New List(Of (String, Integer, String, String, Integer))

        ' Read data from GroupBox1

        If Not String.IsNullOrWhiteSpace(TextBox1.Text) Then

            products.Add((TextBox1.Text, Convert.ToInt32(TextBox2.Text), TextBox3.Text, TextBox4.Text, 1))

        End If

        ' Read data from GroupBox2

        If Not String.IsNullOrWhiteSpace(TextBox8.Text) Then

            products.Add((TextBox8.Text, Convert.ToInt32(TextBox7.Text), TextBox6.Text, TextBox5.Text, 2))

        End If

        ' Read data from GroupBox3

        If Not String.IsNullOrWhiteSpace(TextBox12.Text) Then

            products.Add((TextBox12.Text, Convert.ToInt32(TextBox11.Text), TextBox10.Text, TextBox9.Text, 3))

        End If

        ' Read data from GroupBox4

        If Not String.IsNullOrWhiteSpace(TextBox16.Text) Then

            products.Add((TextBox16.Text, Convert.ToInt32(TextBox15.Text), TextBox14.Text, TextBox13.Text, 4))

        End If
```

```
        ' Read data from GroupBox5
        If Not String.IsNullOrWhiteSpace(TextBox20.Text) Then
            products.Add((TextBox20.Text, Convert.ToInt32(TextBox19.Text),
TextBox18.Text, TextBox17.Text, 5))
        End If
        ' Insert each product into the product table
        For Each product In products
            InsertProduct(connection, id, product.Item1, product.Item2, product.Item3,
product.Item4, product.Item5)
        Next
    End Sub
    Private Sub InsertProduct(connection As SqlConnection, id As Integer, item As
String, quantity As Integer, type As String, purpose As String, si As Integer)
        Dim query As String = "INSERT INTO [product] ([id], [item], [quantity],
[type], [purpose], [si]) VALUES (@id, @item, @quantity, @type, @purpose, @si)"
        Using command As New SqlCommand(query, connection)
            command.Parameters.AddWithValue("@id", id)
            command.Parameters.AddWithValue("@item", item)
            command.Parameters.AddWithValue("@quantity", quantity)
            command.Parameters.AddWithValue("@type", type)
            command.Parameters.AddWithValue("@purpose", purpose)
            command.Parameters.AddWithValue("@si", si)
            command.ExecuteNonQuery()
        End Using
    End Sub
    Private Function GetProducts() As List(Of (String, Integer, String, String, Integer))
        Dim products As New List(Of (String, Integer, String, String, Integer))

        ' Read data from GroupBox1
        If Not String.IsNullOrWhiteSpace(TextBox1.Text) Then
            products.Add((TextBox1.Text, Convert.ToInt32(TextBox2.Text),
TextBox3.Text, TextBox4.Text, 1))
        End If
```

```
        ' Read data from GroupBox2
        If Not String.IsNullOrWhiteSpace(TextBox8.Text) Then
            products.Add((TextBox8.Text, Convert.ToInt32(TextBox7.Text),
TextBox6.Text, TextBox5.Text, 2))
        End If

        ' Read data from GroupBox3
        If Not String.IsNullOrWhiteSpace(TextBox12.Text) Then
            products.Add((TextBox12.Text, Convert.ToInt32(TextBox11.Text),
TextBox10.Text, TextBox9.Text, 3))
        End If

        ' Read data from GroupBox4
        If Not String.IsNullOrWhiteSpace(TextBox16.Text) Then
            products.Add((TextBox16.Text, Convert.ToInt32(TextBox15.Text),
TextBox14.Text, TextBox13.Text, 4))
        End If

        ' Read data from GroupBox5
        If Not String.IsNullOrWhiteSpace(TextBox20.Text) Then
            products.Add((TextBox20.Text, Convert.ToInt32(TextBox19.Text),
TextBox18.Text, TextBox17.Text, 5))
        End If

        Return products
    End Function
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
        ADMINHOME.Show()
    End Sub
    Private Sub TextBox22_TextChanged(sender As Object, e As EventArgs) Handles
TextBox22.TextChanged
        ListBox1.Visible = False
```

```
        FetchLiablePersons(TextBox22.Text.Trim())
    End Sub
    Private Sub FetchLiablePersons(partialName As String)
        ListBox1.Visible = True
        ' Clear existing items in ListBox1
        ListBox1.Items.Clear()

        If String.IsNullOrEmpty(partialName) Then
            Return
        End If

        Dim query As String = "SELECT LIABLE FROM LIABLE WHERE LIABLE
LIKE @PartialName"
        Try
            Using connection As New SqlConnection(CONNECTION_STRING)
                Using command As New SqlCommand(query, connection)
                    ' Add parameter for partial name search
                    command.Parameters.AddWithValue("@PartialName", partialName +
"%")

                    connection.Open()

                    Using reader As SqlDataReader = command.ExecuteReader()
                        While reader.Read()
                            ListBox1.Items.Add(reader("LIABLE").ToString())
                        End While
                    End Using

                    ' Check if no names were found
                    If ListBox1.Items.Count = 0 Then
                        ListBox1.Items.Add("No liable person found.")
                    End If
                End Using
```

```
            End Using
        Catch ex As Exception
            MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub

    Private Sub ListBox1_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles ListBox1.SelectedIndexChanged
        ' Set selected item in TextBox1
        If ListBox1.SelectedIndex <> -1 Then
            TextBox22.Text = ListBox1.SelectedItem.ToString()

        End If
        ListBox1.Visible = False
    End Sub
    Private Sub TextBox21_TextChanged(sender As Object, e As EventArgs) Handles
TextBox21.TextChanged
        ListBox2.Visible = False
        FetchLiablePersons1(TextBox21.Text.Trim())
    End Sub

    Private Sub FetchLiablePersons1(partialName As String)
        ListBox2.Visible = True
        ' Clear existing items in ListBox1
        ListBox2.Items.Clear()

        If String.IsNullOrEmpty(partialName) Then
            Return
        End If

        Dim query As String = "SELECT RECIEVER FROM PARTY WHERE
RECIEVER LIKE @PartialName"
        Try
```

```
            Using connection As New SqlConnection(CONNECTION_STRING)
                Using command As New SqlCommand(query, connection)
                    ' Add parameter for partial name search
                    command.Parameters.AddWithValue("@PartialName", partialName +
"%")

                    connection.Open()

                    Using reader As SqlDataReader = command.ExecuteReader()
                        While reader.Read()
                            ListBox2.Items.Add(reader("RECIEVER").ToString())
                        End While
                    End Using

                    ' Check if no names were found
                    If ListBox2.Items.Count = 0 Then
                        ListBox2.Items.Add("No PARTY found.")
                    End If
                End Using
            End Using
        Catch ex As Exception
            MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub

    Private Sub ListBox2_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles ListBox2.SelectedIndexChanged
        ' Set selected item in TextBox1
        If ListBox2.SelectedIndex <> -1 Then
            TextBox21.Text = ListBox2.SelectedItem.ToString()
        End If
        ListBox2.Visible = False
    End Sub
```

```
End Class
```

CHECK STATUS

```
Imports System.Data.SqlClient

Public Class STATUS
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
        ADMINHOME.Show()
    End Sub

    Private Sub STATUS_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        DataGridView1.Visible = False
    End Sub
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        ' Get the entered Challan ID
        Dim challanID As String = TextBox1.Text.Trim()

        ' Check if the Challan ID is not empty
        If String.IsNullOrEmpty(challanID) Then
            MessageBox.Show("Please enter a Challan ID")
            Return
        End If

        ' Connection string to your database
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

        ' SQL Query to check Challan ID and get details
        Dim query As String = "SELECT  c.id AS challan_id, c.[liable person], c.[date],
c.[return_date], c.[reciever], p.[item], p.[quantity], p.[type], p.[purpose]
                               FROM challan c
                               LEFT JOIN product p ON c.id = p.id
                               WHERE c.id = @challan_id"

        ' Using block to ensure the connection is closed and disposed properly
        Using connection As New SqlConnection(connectionString)
            Try
                ' Open the connection
                connection.Open()

                ' Create the command
                Using command As New SqlCommand(query, connection)
                    ' Add parameter to the query
                    command.Parameters.AddWithValue("@challan_id", challanID)

                    ' Create a data adapter
                    Using adapter As New SqlDataAdapter(command)
                        ' Create a data table to hold the results
                        Dim dataTable As New DataTable()

                        ' Fill the data table
                        adapter.Fill(dataTable)
```

```
                ' Check if any rows were returned
                If dataTable.Rows.Count > 0 Then

                    ' Show the status in a message box
                    MessageBox.Show("Status: Active")
                    DataGridView1.Visible = True
                    ' Bind the data table to the DataGridView
                    DataGridView1.DataSource = dataTable
                Else
                    DataGridView1.Visible = False
                    ' Show a message if the Challan ID does not exist
                    MessageBox.Show("Status: Inactive. Challan ID does not exist.")
                End If
            End Using
        End Using
    Catch ex As Exception
        ' Show an error message if something goes wrong
        MessageBox.Show("An error occurred: " & ex.Message)
    End Try
  End Using
 End Sub
End Class
```

DELETE

```
Imports System.Data.SqlClient

Public Class DEL
  Private Sub DEL_Load(sender As Object, e As EventArgs) Handles MyBase.Load

    Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

    Dim sqlQuery As String = "Select c.id, c.[liable person], c.Date, c.return_date,
c.reciever, p.item, p.quantity, p.type, p.purpose ,P.SI
From challan c
INNER Join product p ON c.id = p.ID
ORDER BY c.id;"




    Dim connection As New SqlConnection(connectionString)
    Dim adapter As New SqlDataAdapter(sqlQuery, connection)
    ' Add username parameter

    Dim dataSet As New DataSet()


    connection.Open()
    adapter.Fill(dataSet)

    If dataSet.Tables.Count > 0 Then
      DataGridView2.DataSource = dataSet.Tables(0)
```

```
        End If


        If connection.State = ConnectionState.Open Then
            connection.Close()
        End If

    End Sub


    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        If DataGridView2.Rows.Count > 0 Then
            ' Get the index of the last row
            Dim lastRowIndex As Integer = DataGridView2.Rows.Count - 2 ' -2 because
of the new row at the e
            ' Get the index of the currently selected row
            Dim selectedRowIndex As Integer = DataGridView2.CurrentCell.RowIndex

            ' Check if the selected row is the last row
            If selectedRowIndex = lastRowIndex Then
                ' Get the ID of the last row
                Dim lastRowId As Integer =
Convert.ToInt32(DataGridView2.Rows(lastRowIndex).Cells("id").Value)
                Dim lastRowId1 As Integer =
Convert.ToInt32(DataGridView2.Rows(lastRowIndex).Cells("SI").Value)

                ' Delete the last row from the database
                Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"
                Using connection As New SqlConnection(connectionString)
                    connection.Open()
                    Dim deleteQuery As String = "DELETE p
FROM PRODUCT p
INNER JOIN CHALLAN C ON p.id = c.id
WHERE p.id = @id AND p.SI = @sI;
"
                    Using command As New SqlCommand(deleteQuery, connection)
                        command.Parameters.AddWithValue("@id", lastRowId)
                        command.Parameters.AddWithValue("@SI", lastRowId1)
                        command.ExecuteNonQuery()
                    End Using
                End Using

                ' Delete the last row from the DataGridView
                DataGridView2.Rows.RemoveAt(lastRowIndex)
            Else
                ' Show a message box if the selected row is not the last row
                MessageBox.Show("You can only delete the last row.", "Delete Row",
MessageBoxButtons.OK, MessageBoxIcon.Warning)
            End If
        Else
            ' Show a message box if there are no rows to delete
            MessageBox.Show("There are no rows to delete.", "Delete Row",
MessageBoxButtons.OK, MessageBoxIcon.Information)
        End If
    End Sub
```

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

GENERATE

```
Imports System.Data.SqlClient

Public Class GENERATE

    Private Sub GENERATE_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        Label2.Text = LOGIN.TextBox1.Text
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

        Dim sqlQuery As String = "SELECT c.id , c.[liable person], c.Date,
c.return_date, c.reciever, " &
                    "p.item, p.quantity, p.type, p.purpose " &
                    "FROM challan c " &
                    "INNER JOIN product p ON c.id = p.ID " &  ' Assuming corrected
JOIN condition
                    "ORDER BY c.id"
        Dim connection As New SqlConnection(connectionString)
        Dim adapter As New SqlDataAdapter(sqlQuery, connection)
        ' Add username parameter

        Dim dataSet As New DataSet()


        connection.Open()
            adapter.Fill(dataSet)

            If dataSet.Tables.Count > 0 Then
                DataGridView1.DataSource = dataSet.Tables(0)

        End If


            If connection.State = ConnectionState.Open Then
                connection.Close()
            End If

    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"
        Dim id As String = TextBox1.Text

        Using connection As New SqlConnection(connectionString)
            Dim query As String = "SELECT p.item, p.quantity, p.type, p.purpose,
c.[liable person], c.date, c.return_date, c.reciever " &
```

```
                    "FROM product p " &
                    "INNER JOIN challan c ON p.id = c.id " &
                    "WHERE p.id = @id"
        Dim command As New SqlCommand(query, connection)
        command.Parameters.AddWithValue("@id", id)

        connection.Open()
        Dim reader As SqlDataReader = command.ExecuteReader()

        ' Check if any data is returned
        If reader.HasRows Then
            ' Read the common data for the challan
            reader.Read()
            Dim challanID As Integer = id
            Dim liablePerson As String = reader("liable person").ToString()
            Dim issueDate As DateTime = DateTime.Parse(reader("date").ToString())
            Dim returnDate As DateTime =
DateTime.Parse(reader("return_date").ToString())
            Dim receiver As String = reader("reciever").ToString()

            ' Collect the product data
            Dim products As New List(Of (String, Integer, String, String))
            Do
                products.Add((reader("item").ToString(),
                        Convert.ToInt32(reader("quantity")),
                        reader("type").ToString(),
                        reader("purpose").ToString()))
            Loop While reader.Read()
            Me.Close()
            ' Create a new instance of SUMMARY1 form
            Dim summaryForm As New SUMMARY1()
            summaryForm.UpdateSummary(challanID, liablePerson, issueDate,
returnDate, receiver, products)

            ' Open SUMMARY1 form
            summaryForm.Show()
        Else
            MessageBox.Show("No data found for the entered ID.")
        End If

        connection.Close()
    End Using
End Sub

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

POST

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

Public Class POST
```

```
    Dim CONNECTION_STRING As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

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

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        PostSelectedRowToStatus()
    End Sub

    Private Sub PostSelectedRowToStatus()
        ' Check if a row is selected
        If DataGridView1.SelectedRows.Count > 0 Then
            Dim selectedRow As DataGridViewRow = DataGridView1.SelectedRows(0)

            ' Get values from the selected row
            Dim id As String = selectedRow.Cells("id").Value.ToString()
            Dim si As String = selectedRow.Cells("SI").Value.ToString()

            ' Determine the value for the status based on the selected radio button
            Dim status As String
            If RadioButton1.Checked Then
                status = "YES"
            ElseIf RadioButton2.Checked Then
                status = "NO"
            Else
                MessageBox.Show("Please select a status (Yes or No).")
                Return
            End If

            ' Example SQL query to insert into STATUS table
            Dim insertQuery As String = "INSERT INTO STATUS (ID, POST, SI)
VALUES (@ID, @POST, @SI)"

            Try
                Using connection As New SqlConnection(CONNECTION_STRING)
                    connection.Open()

                    ' Insert data into the STATUS table
                    Using insertCommand As New SqlCommand(insertQuery, connection)
                        ' Parameters to prevent SQL injection
                        insertCommand.Parameters.AddWithValue("@ID", id)
                        insertCommand.Parameters.AddWithValue("@POST", status)
                        insertCommand.Parameters.AddWithValue("@SI", si)

                        insertCommand.ExecuteNonQuery()
                        MessageBox.Show("Data inserted into STATUS table successfully.")

                        ' Remove the posted row from DataGridView1
                        DataGridView1.Rows.Remove(selectedRow)
                    End Using
                End Using
            Catch ex As Exception
                MessageBox.Show($"An error occurred: {ex.Message}")
            End Try
```

```
        Else
            MessageBox.Show("Please select a row to post to STATUS.")
        End If
    End Sub


    Private Sub TextBox1_TextChanged(sender As Object, e As EventArgs) Handles
TextBox1.TextChanged
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

        ' Get the value from TextBox1 (assuming it's the ID you want to search for)
        Dim valueD As String = TextBox1.Text.Trim()

        ' SQL query to select data from PRODUCT table based on ID
        Dim selectProductQuery As String = "SELECT id, item, quantity, type, purpose,
SI FROM PRODUCT WHERE id = @ID"

        Try
            Using connection As New SqlConnection(connectionString)
                connection.Open()

                ' Retrieve data from PRODUCT table for the selected ID
                Using selectProductCommand As New SqlCommand(selectProductQuery,
connection)
                    selectProductCommand.Parameters.AddWithValue("@ID", valueD)

                    ' Create a DataTable to hold the results
                    Dim dataTable As New DataTable()
                    Dim adapter As New SqlDataAdapter(selectProductCommand)
                    adapter.Fill(dataTable)

                    ' Bind the DataTable to DataGridView1
                    DataGridView1.DataSource = dataTable
                End Using
            End Using
        Catch ex As Exception
            MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub
End Class
```

PENDING

```
Imports System.Data.SqlClient
Imports Microsoft.Office.Interop.Excel
Imports System.Runtime.InteropServices
Imports System.Data
Imports System.Drawing
Imports System.IO

Public Class PENDING
    Private Sub PENDING_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
```

```
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

        ' SQL query to fetch data from PRODUCT and CHALLAN tables based on
STATUS table's conditions
        Dim query As String = "
          SELECT
           C.id AS ChallanID, C.[liable person], C.[reciever],
C.date,C.RETURN_DATE, P.item, P.quantity,P.TYPE,P.PURPOSE
        FROM PRODUCT P
        JOIN CHALLAN C ON P.ID = C.id -- Ensure this is the correct join condition
        WHERE NOT EXISTS (
          SELECT 1 FROM STATUS S WHERE S.id = P.id AND S.SI = P.SI
        )"

        Try
          Using connection As New SqlConnection(connectionString)
            Using command As New SqlCommand(query, connection)
              connection.Open()
              Dim adapter As New SqlDataAdapter(command)
              Dim dataTable As New System.Data.DataTable()
              adapter.Fill(dataTable)

              ' Set DataGridView properties
              DataGridView1.DataSource = dataTable

              ' Auto-resize columns
              DataGridView1.AutoResizeColumns()
            End Using
          End Using
        Catch ex As Exception
          MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub

    Public Sub ExportDataGridViewToExcel()
        Dim excelApp As New Application()
        Dim excelWorkbook As Workbook = Nothing
        Dim excelWorksheet As Worksheet = Nothing

        Try
          excelWorkbook = excelApp.Workbooks.Add()
          excelWorksheet = excelWorkbook.Sheets(1)
          excelWorksheet.Name = "Pending Challan"

          ' Merge and center-align cells for header
          Dim headerRange As Range = excelWorksheet.Range("A1:H1")
          headerRange.Merge()
          headerRange.HorizontalAlignment = XlHAlign.xlHAlignCenter
          headerRange.Value = "Motilal Dulichand"
          headerRange.Font.Size = 36
          headerRange.Font.Bold = True

          ' Merge and center-align cells for title
          Dim titleRange As Range = excelWorksheet.Range("A2:H2")
          titleRange.Merge()
          titleRange.HorizontalAlignment = XlHAlign.xlHAlignCenter
          titleRange.Value = "Pending Challan"
          titleRange.Font.Size = 24
```

```
            titleRange.Font.Bold = True

            ' Adding DataGridView headers horizontally
            For A As Integer = 1 To DataGridView1.Columns.Count
                excelWorksheet.Cells(4, A).Value = DataGridView1.Columns(A -
1).HeaderText
                excelWorksheet.Cells(4, A).Font.Bold = True
                excelWorksheet.Cells(4, A).Interior.Color = RGB(195, 240, 247)
                excelWorksheet.Cells(4, A).Font.Color = RGB(0, 0, 0)
            Next A

            For I As Integer = 0 To DataGridView1.Rows.Count - 1
                For j As Integer = 0 To DataGridView1.Columns.Count - 1
                    Dim columnName As String = DataGridView1.Columns(j).Name
                    If columnName = "date" Or columnName = "RETURN_DATE" Then
                        If DataGridView1.Rows(I).Cells(j).Value IsNot Nothing AndAlso Not
String.IsNullOrEmpty(DataGridView1.Rows(I).Cells(j).Value.ToString()) Then
                            Try
                                Dim dateValue As Date =
Date.Parse(DataGridView1.Rows(I).Cells(j).Value.ToString())
                                excelWorksheet.Cells(5 + I, 1 + j).Value =
dateValue.ToString("dd-MM-yyyy")
                            Catch ex As Exception
                                ' Handle parsing errors, if necessary
                                excelWorksheet.Cells(5 + I, 1 + j).Value =
DataGridView1.Rows(I).Cells(j).Value.ToString()
                            End Try
                        End If
                    Else
                        If DataGridView1.Rows(I).Cells(j).Value IsNot Nothing Then
                            excelWorksheet.Cells(5 + I, 1 + j).Value =
DataGridView1.Rows(I).Cells(j).Value.ToString()
                        End If
                    End If
                Next
            Next

            ' Auto-fit the columns
            excelWorksheet.Columns.AutoFit()

            ' Open SaveFileDialog to ask for the save location
            Dim saveFileDialog As New SaveFileDialog()
            saveFileDialog.Filter = "Excel Files|*.xlsx"
            saveFileDialog.Title = "Save an Excel File"
            saveFileDialog.FileName =
$"PendingChallan_{DateTime.Now.ToString("yyyyMMdd_HHmmss")}.xlsx"

            If saveFileDialog.ShowDialog() = DialogResult.OK Then
                ' Save the Excel file to the selected path
                Dim filePath As String = saveFileDialog.FileName
                excelWorkbook.SaveAs(filePath)

                ' Notify the user
                MessageBox.Show($"File saved to: {filePath}")
            End If


        Catch ex As Exception
            MessageBox.Show($"An error occurred while exporting to Excel and
printing: {ex.Message}")
```

```
    Finally
        ' Release resources
        If excelWorksheet IsNot Nothing Then releaseObject(excelWorksheet)
        If excelWorkbook IsNot Nothing Then
            excelWorkbook.Close(False)
            releaseObject(excelWorkbook)
        End If
        If excelApp IsNot Nothing Then
            excelApp.Quit()
            releaseObject(excelApp)
        End If
    End Try
End Sub

Public Sub PrintExcelFile(filePath As String)
    Dim excelApp As New Application()
    Dim excelWorkbook As Workbook = Nothing

    Try
        excelWorkbook = excelApp.Workbooks.Open(filePath)

        ' Set print settings
        With excelWorkbook.ActiveSheet.PageSetup
            .PrintTitleRows = ""
            .PrintTitleColumns = ""
            .CenterHorizontally = False
            .CenterVertically = False
            .Orientation = XlPageOrientation.xlLandscape
            .FitToPagesWide = 1
            .FitToPagesTall = False
            .TopMargin = excelApp.InchesToPoints(0.75)

            .LeftMargin = excelApp.InchesToPoints(0.5)
            .RightMargin = excelApp.InchesToPoints(0.5)
            .BottomMargin = excelApp.InchesToPoints(0.5)
        End With

        ' Print the Excel file
        excelWorkbook.PrintOut()

        ' Close the workbook without saving changes
        excelWorkbook.Close(False)
    Catch ex As Exception
        MessageBox.Show($"An error occurred while printing: {ex.Message}")
    Finally
        ' Release resources
        If excelWorkbook IsNot Nothing Then releaseObject(excelWorkbook)
        If excelApp IsNot Nothing Then
            excelApp.Quit()
            releaseObject(excelApp)
        End If
    End Try
End Sub

Private Sub releaseObject(ByVal obj As Object)
    Try
        Marshal.ReleaseComObject(obj)
        obj = Nothing
    Catch ex As Exception
        obj = Nothing
```

```
        Finally
            GC.Collect()
        End Try
    End Sub

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

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

UPDATE

EDIT

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

Public Class UPDATE1
    Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"
    Private Sub UPDATE_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        LoadDataGridView() ' Load data into DataGridView on form load
        ListBox1.Visible = False
        ListBox2.Visible = False
    End Sub

    ' Method to load data into DataGridView
    Private Sub LoadDataGridView()
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"
        Dim sqlQuery As String = "SELECT c.id, c.[liable person], c.Date,
c.return_date, c.reciever, " &
                                 "p.si,p.item, p.quantity, p.type, p.purpose " &
                                 "FROM challan c " &
                                 "INNER JOIN product p ON c.id = p.ID " &
                                 "ORDER BY c.id"

        Dim connection As New SqlConnection(connectionString)
        Dim adapter As New SqlDataAdapter(sqlQuery, connection)
        Dim dataSet As New DataSet()
        connection.Open()
        adapter.Fill(dataSet)
        If dataSet.Tables.Count > 0 Then
            DataGridView1.DataSource = dataSet.Tables(0)
        End If
```

```
        connection.Close()
    End Sub

    ' Event handler for row selection in DataGridView
    Private Sub DataGridView1_SelectionChanged(sender As Object, e As
EventArgs) Handles DataGridView1.SelectionChanged
        If DataGridView1.SelectedRows.Count > 0 Then
            Dim selectedRow As DataGridViewRow = DataGridView1.SelectedRows(0)
            TextBox1.Text = selectedRow.Cells("id").Value.ToString()
            TextBox8.Text = selectedRow.Cells("si").Value.ToString() ' Assuming you
have a hidden TextBox for the ID
            TextBox2.Text = selectedRow.Cells("item").Value.ToString()
            TextBox3.Text = selectedRow.Cells("quantity").Value.ToString()
            TextBox4.Text = selectedRow.Cells("type").Value.ToString()
            TextBox5.Text = selectedRow.Cells("purpose").Value.ToString()
            TextBox6.Text = selectedRow.Cells("liable person").Value.ToString()
            DateTimePicker1.Value =
DateTime.Parse(selectedRow.Cells("date").Value.ToString())
            DateTimePicker2.Value =
DateTime.Parse(selectedRow.Cells("return_date").Value.ToString())
            TextBox7.Text = selectedRow.Cells("reciever").Value.ToString()
        End If
    End Sub

    ' Button2_Click event handler to update the data
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        If DataGridView1.SelectedRows.Count = 0 Then
            MessageBox.Show("Please select a row to update.")
            Return
        End If
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"
        Dim id As String = TextBox1.Text ' Use the hidden TextBox to get the ID
        Dim si As String = TextBox8.Text
        Using connection As New SqlConnection(connectionString)
            Dim query As String = "UPDATE product SET item = @item, quantity =
@quantity, type = @type, purpose = @purpose WHERE id = @id AND si = @si;" &
                                  "UPDATE challan SET [liable person] = @liablePerson, date =
@date, return_date = @returnDate, reciever = @reciever WHERE id = @id;"
            Dim command As New SqlCommand(query, connection)
            command.Parameters.AddWithValue("@id", id)
            command.Parameters.AddWithValue("@si", si)
            command.Parameters.AddWithValue("@item", TextBox2.Text)
            command.Parameters.AddWithValue("@quantity",
Convert.ToInt32(TextBox3.Text))
            command.Parameters.AddWithValue("@type", TextBox4.Text)
            command.Parameters.AddWithValue("@purpose", TextBox5.Text)
            command.Parameters.AddWithValue("@liablePerson", TextBox6.Text)
            command.Parameters.AddWithValue("@date", DateTimePicker1.Value)
            command.Parameters.AddWithValue("@returnDate",
DateTimePicker2.Value)
            command.Parameters.AddWithValue("@reciever", TextBox7.Text)
            connection.Open()
            Dim rowsAffected As Integer = command.ExecuteNonQuery()
            If rowsAffected > 0 Then
                MessageBox.Show("Data updated successfully.")
                LoadDataGridView() ' Reload data in DataGridView
```

```
            Else
                MessageBox.Show("Update failed.")
            End If

            connection.Close()
        End Using
    End Sub
    Private Sub TextBox6_TextChanged(sender As Object, e As EventArgs)
        ListBox1.Visible = False
        FetchLiablePersons(TextBox6.Text.Trim())
    End Sub

    Private Sub FetchLiablePersons(partialName As String)
        ListBox1.Visible = True
        ' Clear existing items in ListBox1
        ListBox1.Items.Clear()

        If String.IsNullOrEmpty(partialName) Then
            Return
        End If

        Dim query As String = "SELECT LIABLE FROM LIABLE WHERE LIABLE
LIKE @PartialName"
        Try
            Using connection As New SqlConnection(connectionString)
                Using command As New SqlCommand(query, connection)
                    ' Add parameter for partial name search
                    command.Parameters.AddWithValue("@PartialName", partialName +
"%")

                    connection.Open()

                    Using reader As SqlDataReader = command.ExecuteReader()
                        While reader.Read()
                            ListBox1.Items.Add(reader("LIABLE").ToString())
                        End While
                    End Using

                    ' Check if no names were found
                    If ListBox1.Items.Count = 0 Then
                        ListBox1.Items.Add("No liable person found.")
                    End If
                End Using
            End Using
        Catch ex As Exception
            MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub

    Private Sub ListBox1_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles ListBox1.SelectedIndexChanged
        ' Set selected item in TextBox1
        If ListBox1.SelectedIndex <> -1 Then
            TextBox6.Text = ListBox1.SelectedItem.ToString()
        End If
        ListBox1.Visible = False
    End Sub
    Private Sub TextBox7_TextChanged(sender As Object, e As EventArgs)
        ListBox2.Visible = True
        FetchLiablePersons1(TextBox7.Text.Trim())
```

```
    End Sub

    Private Sub FetchLiablePersons1(partialName As String)
        ListBox2.Visible = True
        ' Clear existing items in ListBox1
        ListBox2.Items.Clear()

        If String.IsNullOrEmpty(partialName) Then
            Return
        End If

        Dim query As String = "SELECT RECIEVER FROM PARTY WHERE
RECIEVER LIKE @PartialName"
        Try
            Using connection As New SqlConnection(connectionString)
                Using command As New SqlCommand(query, connection)
                    ' Add parameter for partial name search
                    command.Parameters.AddWithValue("@PartialName", partialName +
"%")

                    connection.Open()

                    Using reader As SqlDataReader = command.ExecuteReader()
                        While reader.Read()
                            ListBox2.Items.Add(reader("RECIEVER").ToString())
                        End While
                    End Using

                    ' Check if no names were found
                    If ListBox2.Items.Count = 0 Then
                        ListBox2.Items.Add("No PARTY found.")
                    End If
                End Using
            End Using
        Catch ex As Exception
            MessageBox.Show($"An error occurred: {ex.Message}")
        End Try
    End Sub

    Private Sub ListBox2_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles ListBox2.SelectedIndexChanged
        ' Set selected item in TextBox1
        If ListBox2.SelectedIndex <> -1 Then
            TextBox7.Text = ListBox2.SelectedItem.ToString()

        End If
        ListBox2.Visible = False
    End Sub

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

ADD

```
Imports System.Data.SqlClient
```

```
Public Class ADDUP

    Private connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Dim id As Integer
        If Integer.TryParse(TextBox1.Text, id) Then
            CheckAndDisplayProducts(id)
            Panel2.Visible = True
            Panel1.Visible = False
        Else
            MessageBox.Show("Please enter a valid ID.")
            Panel2.Visible = False
        End If
    End Sub

    Private Sub CheckAndDisplayProducts(id As Integer)
        Using conn As New SqlConnection(connectionString)
            conn.Open()
            Dim query As String = "SELECT * FROM Product WHERE id = @id
ORDER BY SI"
            Using cmd As New SqlCommand(query, conn)
                cmd.Parameters.AddWithValue("@id", id)
                Dim adapter As New SqlDataAdapter(cmd)
                Dim table As New DataTable()
                adapter.Fill(table)
                If table.Rows.Count > 0 Then
                    DataGridView1.DataSource = table
                    If table.Rows.Count >= 5 Then
                        MessageBox.Show("Space full. Cannot add more items.")
                    Else
                        MessageBox.Show("Products found. You can add details.")
                    End If
                Else
                    MessageBox.Show("No products found for the given ID.")
                End If
            End Using
        End Using
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Dim id As Integer
        If Integer.TryParse(TextBox1.Text, id) Then
            AddProductDetail(id)
        Else
            MessageBox.Show("Please enter a valid ID.")
        End If
    End Sub

    Private Sub AddProductDetail(id As Integer)
        Dim item As String = TextBox2.Text
        Dim quantity As Integer
        Dim type As String = TextBox4.Text
        Dim purpose As String = TextBox5.Text
```

```
        If String.IsNullOrEmpty(item) OrElse Not Integer.TryParse(TextBox3.Text,
quantity) OrElse String.IsNullOrEmpty(type) OrElse String.IsNullOrEmpty(purpose)
Then
            MessageBox.Show("Please fill all the details correctly.")
            Return
        End If

        ' Check the current maximum SI value for the given ID
        Dim currentSI As Integer = GetCurrentMaxSI(id)
        If currentSI >= 5 Then
            MessageBox.Show("Space full. Cannot add more items.")
            Return
        End If

        ' Insert the new product detail
        Using conn As New SqlConnection(connectionString)
            conn.Open()

            Dim insertQuery As String = "INSERT INTO Product (id, item, quantity,
type, purpose, SI) VALUES (@id, @item, @quantity, @type, @purpose, @SI)"
            Using insertCmd As New SqlCommand(insertQuery, conn)
                insertCmd.Parameters.AddWithValue("@id", id)
                insertCmd.Parameters.AddWithValue("@item", item)
                insertCmd.Parameters.AddWithValue("@quantity", quantity)
                insertCmd.Parameters.AddWithValue("@type", type)
                insertCmd.Parameters.AddWithValue("@purpose", purpose)
                insertCmd.Parameters.AddWithValue("@SI", currentSI + 1)
                insertCmd.ExecuteNonQuery()
            End Using

            MessageBox.Show("Product detail inserted successfully.")
            CheckAndDisplayProducts(id) ' Refresh the data grid view
        End Using
    End Sub

    Private Function GetCurrentMaxSI(id As Integer) As Integer
        Using conn As New SqlConnection(connectionString)
            conn.Open()
            Dim query As String = "SELECT MAX(SI) FROM Product WHERE id =
@id"
            Using cmd As New SqlCommand(query, conn)
                cmd.Parameters.AddWithValue("@id", id)
                Dim result = cmd.ExecuteScalar()
                If result IsNot DBNull.Value AndAlso result IsNot Nothing Then
                    Return Convert.ToInt32(result)
                Else
                    Return 0 ' If no SI values are found, start with 0
                End If
            End Using
        End Using
    End Function

    Private Sub ADDUP_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        Panel2.Visible = False
    End Sub

    Private Sub Button3_Click(sender As Object, e As EventArgs) Handles
Button3.Click
        Me.Close()
```

```
        ADMINHOME.Show()

    End Sub
End Class
```

SUMMARY

```
Imports System.Drawing.Printing
Imports System.Text
Imports System.Data.SqlClient

Public Class Summary
    Private _addFormInstance As ADD

    ' Constructor to receive ADD form instance
    Public Sub New(addFormInstance As ADD)
        InitializeComponent()
        _addFormInstance = addFormInstance
    End Sub
    Public Sub New()
        InitializeComponent()
    End Sub

    Public Sub UpdateSummary(CHALLANID As Integer, liablePerson As String,
issueDate As DateTime, returnDate As DateTime, receiver As String, products As
List(Of (String, Integer, String, String, Integer)), COMBO As String)
        Label3.Text = CHALLANID.ToString()
        Label9.Text = liablePerson
        Label11.Text = issueDate.ToString("dd MMM yyyy")
        Label43.Text = returnDate.ToString("dd MMM yyyy")
        Label7.Text = receiver
        Label45.Text = COMBO
        UpdateAddress()
        ' Update product labels and set visibility
        If products.Count > 0 Then
            Label17.Visible = True
            Label22.Text = products(0).Item1
            Label22.Visible = True
            Label23.Text = products(0).Item2.ToString()
            Label23.Visible = True
            Label24.Text = products(0).Item3
            Label24.Visible = True
            Label25.Text = products(0).Item4
            Label25.Visible = True
        Else
            Label22.Visible = False
            Label23.Visible = False
            Label24.Visible = False
            Label25.Visible = False
            Label17.Visible = False
        End If

        If products.Count > 1 Then
            Label18.Visible = True
            Label26.Text = products(1).Item1
            Label26.Visible = True
            Label27.Text = products(1).Item2.ToString()
            Label27.Visible = True
            Label28.Text = products(1).Item3
            Label28.Visible = True
            Label29.Text = products(1).Item4
```

```
    Label29.Visible = True
Else
    Label26.Visible = False
    Label27.Visible = False
    Label28.Visible = False
    Label29.Visible = False
    Label18.Visible = False
End If

If products.Count > 2 Then
    Label19.Visible = True
    Label30.Text = products(2).Item1
    Label30.Visible = True
    Label31.Text = products(2).Item2.ToString()
    Label31.Visible = True
    Label32.Text = products(2).Item3
    Label32.Visible = True
    Label33.Text = products(2).Item4
    Label33.Visible = True
Else
    Label30.Visible = False
    Label31.Visible = False
    Label32.Visible = False
    Label33.Visible = False
    Label19.Visible = False
End If

If products.Count > 3 Then
    Label20.Visible = True
    Label34.Text = products(3).Item1
    Label34.Visible = True
    Label35.Text = products(3).Item2.ToString()
    Label35.Visible = True
    Label36.Text = products(3).Item3
    Label36.Visible = True
    Label37.Text = products(3).Item4
    Label37.Visible = True
Else
    Label34.Visible = False
    Label35.Visible = False
    Label36.Visible = False
    Label37.Visible = False
    Label20.Visible = False
End If

If products.Count > 4 Then
    Label21.Visible = True
    Label38.Text = products(4).Item1
    Label38.Visible = True
    Label39.Text = products(4).Item2.ToString()
    Label39.Visible = True
    Label40.Text = products(4).Item3
    Label40.Visible = True
    Label41.Text = products(4).Item4
    Label41.Visible = True
Else
    Label38.Visible = False
    Label39.Visible = False
    Label40.Visible = False
    Label41.Visible = False
```

```
            Label21.Visible = False
        End If
    End Sub
    Private Sub PrintDocument1_PrintPage(sender As Object, e As
PrintPageEventArgs) Handles PrintDocument1.PrintPage
        Dim font As New Font("Arial", 10)
        Dim headerFont As New Font("Arial", 12, FontStyle.Bold)
        Dim linePen As New Pen(Color.Black, 1)
        Dim headerFont1 As New Font("Arial", 24, FontStyle.Bold, 1.5)
        Dim font1 As New Font("Arial", 10, FontStyle.Bold)

        ' Set margins
        Dim leftMargin As Integer = e.MarginBounds.Left
        Dim topMargin As Integer = e.MarginBounds.Top
        Dim rightMargin As Integer = e.MarginBounds.Right

        ' Company Information
        e.Graphics.DrawString("RETURNABLE CHALLAN", headerFont,
Brushes.Black, leftMargin + 440, topMargin)
        e.Graphics.DrawString("Motilal  Dulichand  (P)   Ltd.", headerFont1,
Brushes.Black, leftMargin + 390, topMargin + 20)
        e.Graphics.DrawString("20, Industrial Estate, Kanpur-208012", font1,
Brushes.Gray, leftMargin + 440, topMargin + 50)
        e.Graphics.DrawString("GSTIN NO: 09AAAACD8231ZZ", font,
Brushes.Black, leftMargin, topMargin)

        ' Challan Information
        e.Graphics.DrawString($"Challan No: {Label3.Text}", font, Brushes.Black,
leftMargin, topMargin + 80)
        e.Graphics.DrawString($"Challan Dt: {Label11.Text}", font, Brushes.Black,
leftMargin + 760, topMargin + 80)

        ' Recipient Information
        e.Graphics.DrawString("TO,", font, Brushes.Black, leftMargin, topMargin +
120)
        e.Graphics.DrawString(Label7.Text, font, Brushes.Black, leftMargin, topMargin
+ 140)
        e.Graphics.DrawString(Label46.Text, font, Brushes.Black, leftMargin,
topMargin + 160)
        e.Graphics.DrawString($"Liable Person: {Label9.Text}", font, Brushes.Black,
leftMargin + 400, topMargin + 160)

        ' Table Header
        Dim yPos As Integer = topMargin + 200
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        e.Graphics.DrawString("S. No.", font, Brushes.Black, leftMargin, yPos)
        e.Graphics.DrawString("PARTICULARS", font, Brushes.Black, leftMargin +
50, yPos)
        e.Graphics.DrawString("Quantity", font, Brushes.Black, leftMargin + 500,
yPos)
        e.Graphics.DrawString("TYPE", font, Brushes.Black, leftMargin + 610, yPos)
        e.Graphics.DrawString("Purpose", font, Brushes.Black, leftMargin + 700, yPos)

        ' Draw lines for the table header
        yPos += 20
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        ' Table Data
        If Label22.Visible = True Then
```

```
            yPos += 20
            e.Graphics.DrawString("1", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label22.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label23.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label24.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label25.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If

        If Label26.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("2", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label26.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label27.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label28.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label29.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
        If Label30.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("3", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label30.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label31.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label32.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label33.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
        If Label34.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("4", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label34.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label35.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label36.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label37.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
        If Label38.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("5", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label38.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label39.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label40.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label41.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
```

```
        yPos += 20
        ' Draw lines for the table data
        yPos += 40
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        ' Draw vertical lines to form the table
        Dim columnPositions As Integer() = {leftMargin, leftMargin + 40, leftMargin +
490, leftMargin + 590, leftMargin + 690, rightMargin}
        For Each xPos As Integer In columnPositions
            e.Graphics.DrawLine(linePen, xPos, topMargin + 200, xPos, yPos)
        Next

        ' Footer Information
        yPos += 20
        e.Graphics.DrawString($"Date of Return: {Label43.Text}", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString($"Through: {Label45.Text}", font, Brushes.Black,
leftMargin + 700, yPos)
        yPos += 20
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        yPos += 20
        e.Graphics.DrawString("Sign. Mech Store Incharge", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString("Forwarded By (Section Head)", font, Brushes.Black,
leftMargin + 350, yPos)
        e.Graphics.DrawString("Approved By", font, Brushes.Black, leftMargin + 700,
yPos)

        ' Draw lines for the footer
        yPos += 20
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)
        yPos += 20
        e.Graphics.DrawString("material returned in store", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString("quality checked", font, Brushes.Black, leftMargin +
350, yPos)
        e.Graphics.DrawString("posted computer", font, Brushes.Black, leftMargin +
700, yPos)
        yPos += 15
        e.Graphics.DrawString("sign mech store incharge", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString("by mech Section Head)", font, Brushes.Black,
leftMargin + 350, yPos)
        e.Graphics.DrawString("dt.       by.", font, Brushes.Black, leftMargin + 700,
yPos)
        yPos += 30
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
        ADMINHOME.Show()
    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        PrintDocument1.DefaultPageSettings.Landscape = True
        PrintDocument1.Print()
```

```
    End Sub
    Public Sub UpdateAddress()
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"
        Dim query As String = "SELECT Address FROM Party WHERE [RECIEVER]
= @Receiver"

        Using connection As New SqlConnection(connectionString)
            Using command As New SqlCommand(query, connection)
                command.Parameters.AddWithValue("@Receiver", Label7.Text)

                Try
                    connection.Open()
                    Dim address As Object = command.ExecuteScalar()

                    If address IsNot Nothing Then
                        Label46.Text = address.ToString()
                    Else
                        Label46.Text = "Address not found."
                    End If
                Catch ex As Exception
                    MessageBox.Show("An error occurred: " & ex.Message)
                End Try
            End Using
        End Using

    End Sub
    Private Sub SomeMethod()
        ' Set Label7.Text value here
        Label7.Text = "Receiver Name"

        ' Call the method to update Label46
        UpdateAddress()
    End Sub
End Class
```

SUMMARY1

```
Imports System.Data.SqlClient
Imports System.Drawing.Printing

Public Class SUMMARY1

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
        ADMINHOME.Show()
    End Sub
    Public Sub New()
        ' Initialize form components
        InitializeComponent()
    End Sub

    ' Method to update the form with provided data
    Public Sub UpdateSummary(challanID As Integer, liablePerson As String,
issueDate As DateTime, returnDate As DateTime, receiver As String, products As
List(Of (String, Integer, String, String)))
        Label3.Text = challanID.ToString()
        Label9.Text = liablePerson
        Label11.Text = issueDate.ToString("dd MMM yyyy")
        Label43.Text = returnDate.ToString("dd MMM yyyy")
```

```
Label7.Text = receiver
UpdateAddress()
' Update product labels and set visibility
If products.Count > 0 Then
    Label17.Visible = True
    Label22.Text = products(0).Item1
    Label22.Visible = True
    Label23.Text = products(0).Item2.ToString()
    Label23.Visible = True
    Label24.Text = products(0).Item3
    Label24.Visible = True
    Label25.Text = products(0).Item4
    Label25.Visible = True
Else
    Label22.Visible = False
    Label23.Visible = False
    Label24.Visible = False
    Label25.Visible = False
    Label17.Visible = False
End If

If products.Count > 1 Then
    Label18.Visible = True
    Label26.Text = products(1).Item1
    Label26.Visible = True
    Label27.Text = products(1).Item2.ToString()
    Label27.Visible = True
    Label28.Text = products(1).Item3
    Label28.Visible = True
    Label29.Text = products(1).Item4
    Label29.Visible = True
Else
    Label26.Visible = False
    Label27.Visible = False
    Label28.Visible = False
    Label29.Visible = False
    Label18.Visible = False
End If

If products.Count > 2 Then
    Label19.Visible = True
    Label30.Text = products(2).Item1
    Label30.Visible = True
    Label31.Text = products(2).Item2.ToString()
    Label31.Visible = True
    Label32.Text = products(2).Item3
    Label32.Visible = True
    Label33.Text = products(2).Item4
    Label33.Visible = True
Else
    Label30.Visible = False
    Label31.Visible = False
    Label32.Visible = False
    Label33.Visible = False
    Label19.Visible = False
End If

If products.Count > 3 Then
    Label20.Visible = True
    Label34.Text = products(3).Item1
```

```
            Label34.Visible = True
            Label35.Text = products(3).Item2.ToString()
            Label35.Visible = True
            Label36.Text = products(3).Item3
            Label36.Visible = True
            Label37.Text = products(3).Item4
            Label37.Visible = True
        Else
            Label34.Visible = False
            Label35.Visible = False
            Label36.Visible = False
            Label37.Visible = False
            Label20.Visible = False
        End If

        If products.Count > 4 Then
            Label21.Visible = True
            Label38.Text = products(4).Item1
            Label38.Visible = True
            Label39.Text = products(4).Item2.ToString()
            Label39.Visible = True
            Label40.Text = products(4).Item3
            Label40.Visible = True
            Label41.Text = products(4).Item4
            Label41.Visible = True
        Else
            Label38.Visible = False
            Label39.Visible = False
            Label40.Visible = False
            Label41.Visible = False
            Label21.Visible = False
        End If
    End Sub
    Private Sub PrintDocument_PrintPage(sender As Object, e As
PrintPageEventArgs) Handles PrintDocument1.PrintPage
        Dim font As New Font("Arial", 10)
        Dim headerFont As New Font("Arial", 12, FontStyle.Bold)
        Dim linePen As New Pen(Color.Black, 1)
        Dim headerFont1 As New Font("Arial", 24, FontStyle.Bold, 1.5)
        Dim font1 As New Font("Arial", 10, FontStyle.Bold)

        ' Set margins
        Dim leftMargin As Integer = e.MarginBounds.Left
        Dim topMargin As Integer = e.MarginBounds.Top
        Dim rightMargin As Integer = e.MarginBounds.Right

        ' Company Information
        e.Graphics.DrawString("RETURNABLE CHALLAN", headerFont,
Brushes.Black, leftMargin + 440, topMargin)
        e.Graphics.DrawString("Motilal  Dulichand  (P)   Ltd.", headerFont1,
Brushes.Black, leftMargin + 390, topMargin + 20)
        e.Graphics.DrawString("20, Industrial Estate, Kanpur-208012", font1,
Brushes.Gray, leftMargin + 440, topMargin + 50)
        e.Graphics.DrawString("GSTIN NO: 09AAAACD8231ZZ", font,
Brushes.Black, leftMargin, topMargin)

        ' Challan Information
        e.Graphics.DrawString($"Challan No: {Label3.Text}", font, Brushes.Black,
leftMargin, topMargin + 80)
```

```
        e.Graphics.DrawString($"Challan Dt: {Label11.Text}", font, Brushes.Black,
leftMargin + 760, topMargin + 80)

        ' Recipient Information
        e.Graphics.DrawString("TO,", font, Brushes.Black, leftMargin, topMargin +
120)
        e.Graphics.DrawString(Label7.Text, font, Brushes.Black, leftMargin, topMargin
+ 140)
        e.Graphics.DrawString(Label44.Text, font, Brushes.Black, leftMargin,
topMargin + 160)
        e.Graphics.DrawString($"Liable Person: {Label9.Text}", font, Brushes.Black,
leftMargin + 400, topMargin + 160)

        ' Table Header
        Dim yPos As Integer = topMargin + 200
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        e.Graphics.DrawString("S. No.", font, Brushes.Black, leftMargin, yPos)
        e.Graphics.DrawString("PARTICULARS", font, Brushes.Black, leftMargin +
50, yPos)
        e.Graphics.DrawString("Quantity", font, Brushes.Black, leftMargin + 500,
yPos)
        e.Graphics.DrawString("TYPE", font, Brushes.Black, leftMargin + 610, yPos)
        e.Graphics.DrawString("Purpose", font, Brushes.Black, leftMargin + 700, yPos)

        ' Draw lines for the table header
        yPos += 20
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        ' Table Data
        If Label22.Visible = True Then
            yPos += 20
            e.Graphics.DrawString("1", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label22.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label23.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label24.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label25.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If

        If Label26.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("2", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label26.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label27.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label28.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label29.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
        If Label30.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("3", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label30.Text, font, Brushes.Black, leftMargin + 50,
yPos)
```

```
            e.Graphics.DrawString(Label31.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label32.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label33.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
        If Label34.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("4", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label34.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label35.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label36.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label37.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
        If Label38.Visible = True Then
            yPos += 30
            e.Graphics.DrawString("5", font, Brushes.Black, leftMargin + 10, yPos)
            e.Graphics.DrawString(Label38.Text, font, Brushes.Black, leftMargin + 50,
yPos)
            e.Graphics.DrawString(Label39.Text, font, Brushes.Black, leftMargin + 500,
yPos)
            e.Graphics.DrawString(Label40.Text, font, Brushes.Black, leftMargin + 610,
yPos)
            e.Graphics.DrawString(Label41.Text, font, Brushes.Black, leftMargin + 700,
yPos)
        End If
        yPos += 20
        ' Draw lines for the table data
        yPos += 40
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        ' Draw vertical lines to form the table
        Dim columnPositions As Integer() = {leftMargin, leftMargin + 40, leftMargin +
490, leftMargin + 590, leftMargin + 690, rightMargin}
        For Each xPos As Integer In columnPositions
            e.Graphics.DrawLine(linePen, xPos, topMargin + 200, xPos, yPos)
        Next

        ' Footer Information
        yPos += 20
        e.Graphics.DrawString($"Date of Return: {Label43.Text}", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString($"Through: PARTY", font, Brushes.Black, leftMargin +
700, yPos)
        yPos += 20
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)

        yPos += 20
        e.Graphics.DrawString("Sign. Mech Store Incharge", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString("Forwarded By (Section Head)", font, Brushes.Black,
leftMargin + 350, yPos)
        e.Graphics.DrawString("Approved By", font, Brushes.Black, leftMargin + 700,
yPos)
```

```
        ' Draw lines for the footer
        yPos += 20
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)
        yPos += 20
        e.Graphics.DrawString("material returned in store", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString("quality checked", font, Brushes.Black, leftMargin +
350, yPos)
        e.Graphics.DrawString("posted computer", font, Brushes.Black, leftMargin +
700, yPos)
        yPos += 15
        e.Graphics.DrawString("sign mech store incharge", font, Brushes.Black,
leftMargin, yPos)
        e.Graphics.DrawString("by mech Section Head)", font, Brushes.Black,
leftMargin + 350, yPos)
        e.Graphics.DrawString("dt.        by.", font, Brushes.Black, leftMargin + 700,
yPos)
        yPos += 30
        e.Graphics.DrawLine(linePen, leftMargin, yPos, rightMargin, yPos)
    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        PrintDocument1.DefaultPageSettings.Landscape = True
        PrintDocument1.Print()
    End Sub
    Public Sub UpdateAddress()
        Dim connectionString As String = "Data Source=DESKTOP-
DL4DPHJ\SQLEXPRESS;Initial Catalog=return;Integrated
Security=True;Encrypt=False"
        Dim query As String = "SELECT Address FROM Party WHERE [RECIEVER]
= @Receiver"
        Using connection As New SqlConnection(connectionString)
            Using command As New SqlCommand(query, connection)
                command.Parameters.AddWithValue("@Receiver", Label7.Text)

                Try
                    connection.Open()
                    Dim address As Object = command.ExecuteScalar()

                    If address IsNot Nothing Then
                        Label44.Text = address.ToString()
                    Else
                        Label44.Text = "Address not found."
                    End If
                Catch ex As Exception
                    MessageBox.Show("An error occurred: " & ex.Message)
                End Try
            End Using
        End Using

    End Sub
    Private Sub SomeMethod()
        ' Set Label7.Text value here
        Label7.Text = "Receiver Name"

        ' Call the method to update Label46
        UpdateAddress()
    End Sub
End Class
```

CONCLUSION

Reflecting on my internship at Motilal Dulichand, developing the return challan system was a valuable learning experience in software development and database management. Although this project was primarily for training, it taught me important skills in creating easy-to-use interfaces, efficiently handling data, and working well in a corporate environment.

The return challan system was designed to automate the creation and management of return challans, ensuring accurate records and efficient processing of returns. Using Visual Basic for the frontend, I focused on making a user-friendly interface. SQL was used for the backend to ensure secure and efficient data storage, which was essential for keeping accurate records of returned products and parties involved.

During the project, I collaborated closely with various team members to make sure the system met industry standards and the company's needs. This teamwork not only improved the system's functionality but also highlighted the importance of user feedback in improving software. Working on real-world scenarios helped me understand the challenges in software development and reinforced my problem-solving skills.

In conclusion, while the return challan system was mainly a learning project, it provided a strong foundation for my future in software development. It enhanced my technical skills, gave me a better understanding of how technology can improve business processes, and increased my enthusiasm for contributing to future IT projects.