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

SALES INVENTORY SOFTWARE

Project Report submitted in partial fulfillment of the requirements
for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS (BCA)

Submitted by

AKASH SEERVI N (21BCAD06)
DARREN IMMANUEL FERNANDES (21BCAD18)

Under the guidance of

DR. THENMOZHI K

DEPARTMENT OF COMPUTER SCIENCE(UG)
BCA PROGRAMME
KRISTU JAYANTI COLLEGE (AUTONOMOUS)
K. NARAYANAPURA, KOTHANUR P.O., BENGALURU – 560077

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

DEPARTMENT OF COMPUTER SCIENCE (UG)

CERTIFICATE OF COMPLETION

This is to certify that the project entitled SALES INVENTORY SOFTWARE has been satisfactorily completed by **DARREN IMMANUEL FERNANDES (21BCAD18)** in partial fulfillment of the award of the Bachelor of Computer Applications degree requirements prescribed by Kristu Jayanti College (Autonomous), Bengaluru (Affiliated to Bengaluru North University) during the academic year 2022 -23.

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

Valued by Examiners

1:________________________ ***Centre:*** **Kristu Jayanti College**

2:_______________________ ***Date:***

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

DECLARATION

I, DARREN IMMANUEL FERNANDES (21BCAD18) hereby declare that the project work entitled **SALES INVENTORY SOFTWARE** is an original project work carried out by me, under the guidance of DR THENMOZHI K.

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

Signature
DARREN IMMANUEL FERNANDES
21BCAD18

Bengaluru
Date:

ACKNOWLEDGEMENT

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

I take this opportunity to express my profound and whole hearted thanks to **Rev. Fr. Dr. AUGUSTINE GEORGE, PRINCIPAL and Rev. Fr. LIJO P THOMAS, VICE PRINCIPAL and CHIEF FINANCIAL OFFICER, KRISTU JAYANTI COLLEGE (AUTONOMOUS), BENGALURU** for providing ample facilities made to undergo my project successfully.

I express my deep sense of gratitude and sincere thanks to our **Head of the Department Prof. A. SEVUGA PANDIAN** and **Dean Dr. CALISTUS JUDE AL** for their valuable advice.

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

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

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

SYNOPSIS

The primary goal of this project, SALES INVENTORY SOFTWARE is to manage the stock (i.e., the products) in the business. This software will offer a user-friendly interface to manage inventory and sales transactions. Using this software, the Admin and the Employee in the business can maintain a record of the products purchased from the supplier and the products sold the company. Thus, through this software they will be able to manage the supplier and company details. Payment made to supplier and Payment from company can be managed. With this system, they can easily generate reports and analyze data to make informed decisions. It also enables the business to maintain accurate record of all transactions, which will help to prevent loss due to thrift or error.

TABLE OF CONTENTS

1. INTRODUCTION

SALES INVENTORY SOFTWARE is a software project developed in VB.Net which aims to provide a comprehensive solution for managing the Stock of the product which is available in the business. This project is designed to keep a record of product purchased and product sold. This project will help to see the products which were purchased and sold. The main goal is to manage and keep a record of the stock.

1.1 PROBLEM DEFINITION:

It helps the admin of the software to store the details of the products in the business. It also stores the details of purchase, supplier, sales, company, payment details (i.e., Payment to supplier and payment to company). Thus, this software helps to manage the stock in the business.

1.2 SCOPE OF THE PROJECT:

- The information of the product can be added to the stock and could be updated or deleted by the Admin / Employee.
- The registration of the employee is done by the Admin.
- Stock, Purchase and Sales report can be maintained.
- The administration does not require any documentation.
- The Admin / Employee can add, delete and update and view the stock, purchase, sales in the business.
- The Admin / Employee can purchase the product from Supplier and keep a record of the payment made to supplier as well as the item that he has purchased.
- The Admin / Employee can also see the product sold to the Company and keep a record of the payment made by the company as well as the items that he has sold.
- Admin has the access regarding the Log-in and Log-out activity.

1.3 MODULES IN THE PROJECT:

- Login
- Registration
- Product (Stock)
- Purchase / Supplier
- Payment
- Log Manager
- Report

1.3.1 MODULE DESCRIPTIONS:

LOGIN: The module enables the Admin or the Employee (User) to enter the User Name, Password, User Type and therefore gives access to the SALES INVENTORY SOFTWARE, if the login details are correct.

REGISTRATION: This module enables the admin to register the new Employee to access the software.

PRODUCT (STOCK): This Stock module allows to view the stock (i.e., Phones, Laptop) . It also allows to add, update, delete and view the stock.

PURCHASE / SUPPLIER: This module enables the Admin / Employee to keep a record of the stock (i.e., Phones, Laptop) purchased from the Supplier. It also stores the details of the Supplier.

SALES / COMPANY: This module enables the user to keep a record of the stock (i.e., Phones, Laptops) sold to the company. It also stores the details of the Company.

PAYMENT: This Payment module enables to keep a track of the payment which was made to the supplier which purchasing the stock (i.e., Phones, Laptops). It also helps to keep a track of the payment made by the company, while purchasing a stock.

LOG MANAGER: This module enables the admin to keep a track of when the Admin / User log in and out.

REPORT: In this module, Admin / Employee can view the report of purchase, stock sales in the business

2. SYSTEM STUDY

2.1 EXISTING SYSTEM:

The In the existing system of Inventory management, the details relating to stock, purchase or sales are manually maintained. Thus, keeping a record of the inventory of the level manually are time consuming. There is also existing system available. In the software the admin can enter the details of the product, and sold. Therefore, the admin can view the stock in the business. In the existing software, the major problem is:

- Poor reporting: Reports in the existing system may be limited, in only shows the stock details products purchased and sold.
- It does not show payment details, supplier details and company details.

2.2 FEASIBILITY STUDY:

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

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

The project SALES INVENTORY SOFTWARE can be designed and developed using .Net framework. The content, language is feasible to use and the portal can be developed based on the requirements.

2.2.1 COMPONENTS OF FEASIBLE STUDY:

- Technical feasibility
- Operational feasibility
- Schedule Feasibility

Every project is feasible for given unlimited resources and infinitive time. Feasibility study is an evaluation of the proposed system regarding its workability, impact on the organization, ability to meet the user needs and effective use of resources.

Thus, when a new application is proposed it normally goes through a feasibility study before it is approved for development. Feasibility and risk analysis are related in many ways. The feasibility analysis in this project has been discussed below based on the above-mentioned components of feasibility.

2.2.2 SCHEDULE:

Time duration of this project requires 58 days covering 13 days for initiation phase, 8 days for definition phase, 17 days for Design phase, 10 days for implementation phase, 6 days for Testing phase and 4 days for documentation.

2.2.3 OPERATIONAL:

- Reduction of paper work.
- Human effort or Manual Labour can be reduced drastically.
- Major operations that are done manually can be done within a matter of seconds.
- Data storing is easier.

2.3 PROPOSED SYSTEM:

In this project in, order to solve the issues / problems of the existing system, by which the admin / employee can maintain the purchase details, supplier, details, product details, sales details, company details, payment details to supplier and payment details from company. And report will be maintained. This the admin / employee can maintain all these details in the software.

3. SYSTEM DESIGN:

In the design phase the architecture is established. This phase starts with the requirement document delivered by the requirement phase and maps the requirements into an architecture. The architecture defines the components, their interfaces and behaviors. The deliverable design document is the architecture. The design document describes a plan to implement the requirements. This phase represents the ``how" phase.

Details on computer programming languages and environments, machines, packages, application architecture, distributed architecture layering, memory size, platform, algorithms, data structures, global type definitions, interfaces, and many other engineering details are established.

The design may include the usage of existing components. Analyzing the trade-offs of necessary complexity allows for many things to remain simple which, in turn, will eventually lead to a higher quality product. The architecture team also converts the typical scenarios into a test plan. In our approach, the team, given a complete requirement document, must also indicate critical priorities for the implementation team.

A critical implementation priority leads to a task that has to be done right. If it fails, the product fails.If it succeeds, the product might succeed. At the very least, the confidence level of the team producinga successful product will increase. This will keep the implementation team focused. Exactly how this information is conveyed is a skill based on experience more than a science based on fundamental foundations. System design is the process of defining the architecture components, modules, interfaces, and data for a system to satisfy specified requirements.

Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. If the broader topic of product development blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user. Until the 1990s, systems design had a crucial and respected role in the data processing industry. In the 1990s, standardization of hardware and software resulted in the ability to build modular systems. The increasing importance of software running on generic platforms has enhanced the discipline of software engineering.

Object-oriented analysis and design methods are becoming the most widely used methods for computer systems design. The UML has become the standard language in object-oriented analysis and design.

It is widely used for modelling software systems and is increasingly used for high designing non software systems and organizations.

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

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

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

- Input requirement
- Output requirements
- Storage requirements
- Processing requirements
- System control and backup or recovery

Put another way, the physical portion of system design can generally be broken down into three sub-tasks:

- User Interface Design
- Data Design
- Process Design

User Interface Design is concerned with how users add information to the system and with how the system presents information back to them. Data Design is concerned with how the data is represented and stored within the system.

Finally, Process Design is concerned with how data moves through the system, and with how and where it is validated, secured and/or transformed as it flows into, through and out of the system.

At the end of the system design phase, documentation describing the three sub-tasks is produced and made available for use in the next phase. Physical design, in this context, does not refer to the tangible physical design of an information system.

To use an analogy, a personal computer's physical design involves input via a keyboard, processing within the CPU, and output via a monitor, printer, etc.

It would not concern the actual layout of the tangible hardware, which for a PC would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots, etc.

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

3.1 ER-DIAGRAM:

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

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

- Identify all the relevant entities in a given system and determine the relationships among these entities.
- An entity should appear only once in a particular diagram.
- Provide a precise and appropriate name for each entity, attribute, and relationship in the diagram.
- Terms that are simple and familiar always beats vague, technical-sounding words.
- In naming entities, remember to use singular nouns. However, adjectives may be used to distinguish entities belonging to the same class (part-time employee and full -time employee, for example).
- Meanwhile attribute names must be meaningful, unique, system independent, and easily understandable.
- Remove vague, redundant or unnecessary relationships between entities. Never connect a relationship to another relationship.
- Make effective use of colours.
- You can use colours to classify similar entities or to highlight key areas in your diagrams. You can draw entity relationship diagrams manually, especially when you are just informally showing simple systems to your peers.
- However, for more complex systems and for external audiences, you need diagramming software such as Creately’s to craft visually engaging and precise ER diagrams.
- The ER Diagram Software offered by Creately as an online service is pretty easy to use and is a lot more affordable than purchasing licensed software.
- It is also perfectly suited for development teams because of its strong support for collaboration.

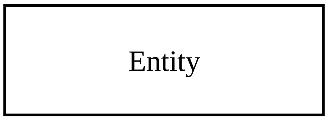
3.1.1 THE HISTORY OF ENTITY RELATIONSHIP DIAGRAMS:

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

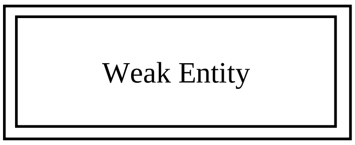
3.1.2 STRUCTURE OF ENTITY RELATIONSHIP DIAGRAM WITH COMMONERD:

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

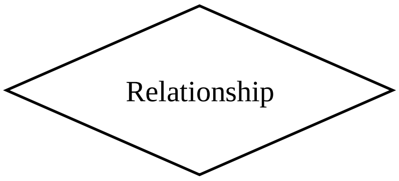
Entities, which are represented by rectangles. An entity is an object or concept about which you want to store information.



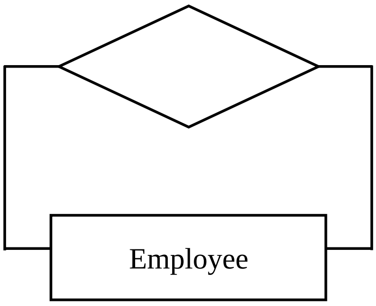
A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.



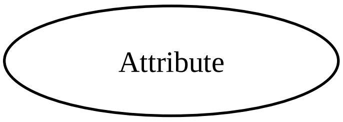
Actions, which are represented by diamond shapes, show how two entities share information in the database.



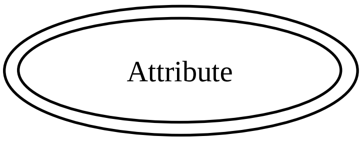
In some cases, entities can be self-linked. For example, employees can supervise other employees.



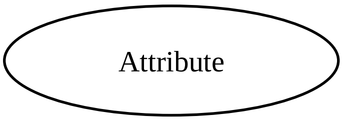
Attributes, which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the entity. For example, an employee's social security number might be the employee's key attribute.



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

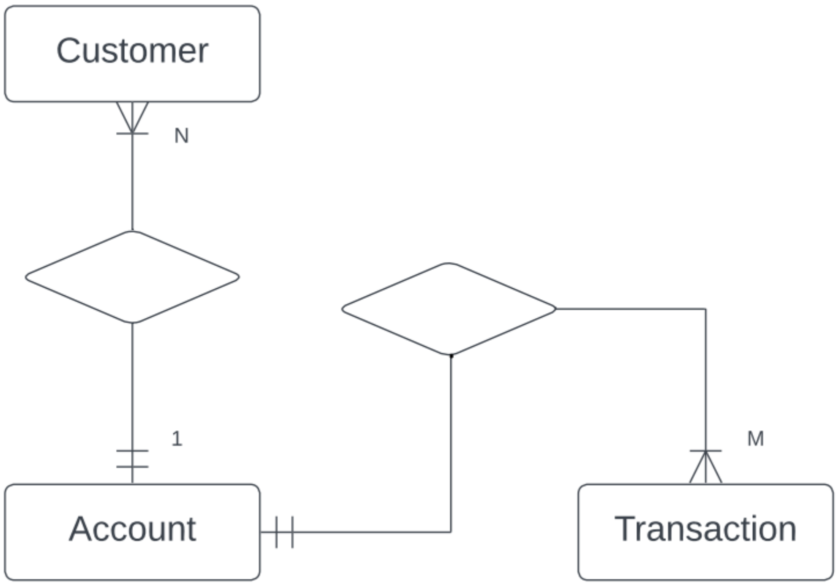


A derived attribute is based on another attribute. For example, an employee's monthly salary is based on the employee's annual salary.

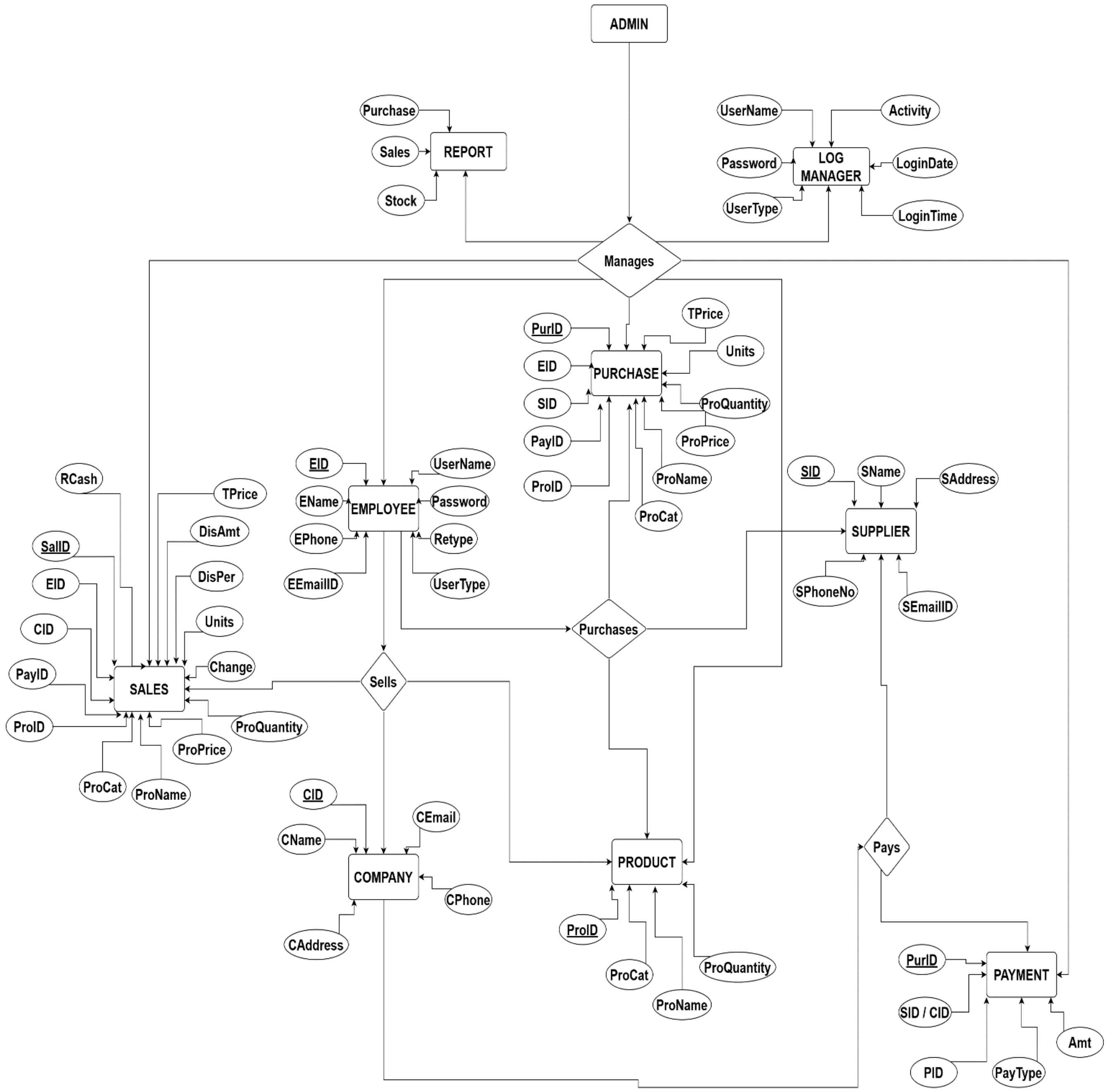


Connecting lines, solid lines that connect attributes to show the relationships of entities in the ER Diagram.

Cardinality specifies how many instances of an entity relate to one instance of another entity. Ordinality is also closely linked to cardinality. While cardinality specifies the occurrences of a relationship, ordinality describes the relationship as either mandatory or optional. In other words, cardinality specifies the maximum number of relationships and ordinality specifies the absolute minimum number of relationships. There are many notation styles that express cardinality.



ER DIAGRAM FOR SALES INVENTORY SOFTWARE:



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

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

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

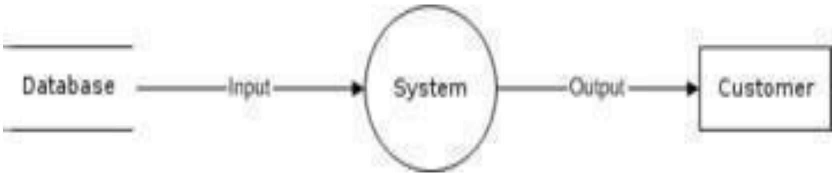
A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel (which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

3.2.1 HISTORY:

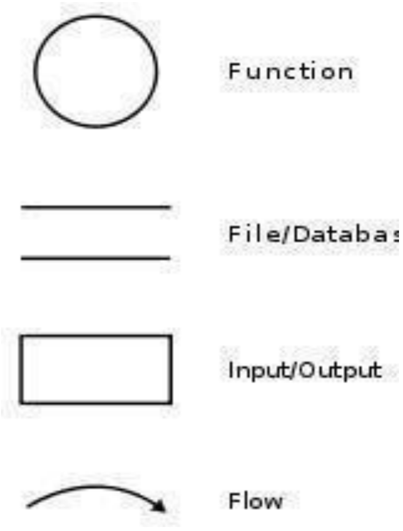
Larry Constantine, the original developer of structured design, based on Martin and Estrin's "Data Flow Graph" model of computation.

Starting in the 1970s, data flow diagrams (DFD) became a popular way to visualize the major steps and data involved in software system processes. DFDs were usually used to show data flow in a computer system, although they could in theory be applied to business process modelling. DFD were useful to document the major data flows or to explore a new high-level design in terms of data flow.

3.2.2 THEORY:



3.2.3 DATA FLOW DIAGRAM EXAMPLE:



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

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

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

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

The old system's dataflow diagrams can be drawn up and compared with the new system's data flow diagrams to draw comparisons to implement a more efficient system. Data flow diagrams can be used to provide the end user with a physical idea of where the data they input ultimately has an effect upon the structure of the whole system from order to dispatch to report. How any system is developed can be determined through a data flow diagram model.

In the course of developing a set of levelled data flow diagrams the analyst/designer is forced to address how the system may be decomposed into component sub-systems, and to identify the transaction data in the data model.

Data flow diagrams can be used in both Analysis and Design phase of the SDLC.
There are different notations to draw data flow diagrams (Yourdon & Coad and Gane & Sarson), defining different visual representations for processes, data stores, data flow, and external entities.

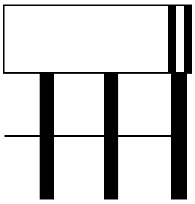
3.2.4 PHYSICAL VS LOGICAL DFD:

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

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

3.2.5 DATA FLOW SYMBOLS AND THEIR MEANINGS:

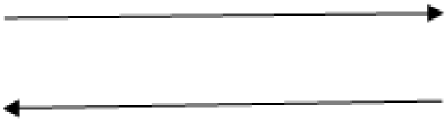
An entity*:* A source of data or a destination for data.



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

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

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



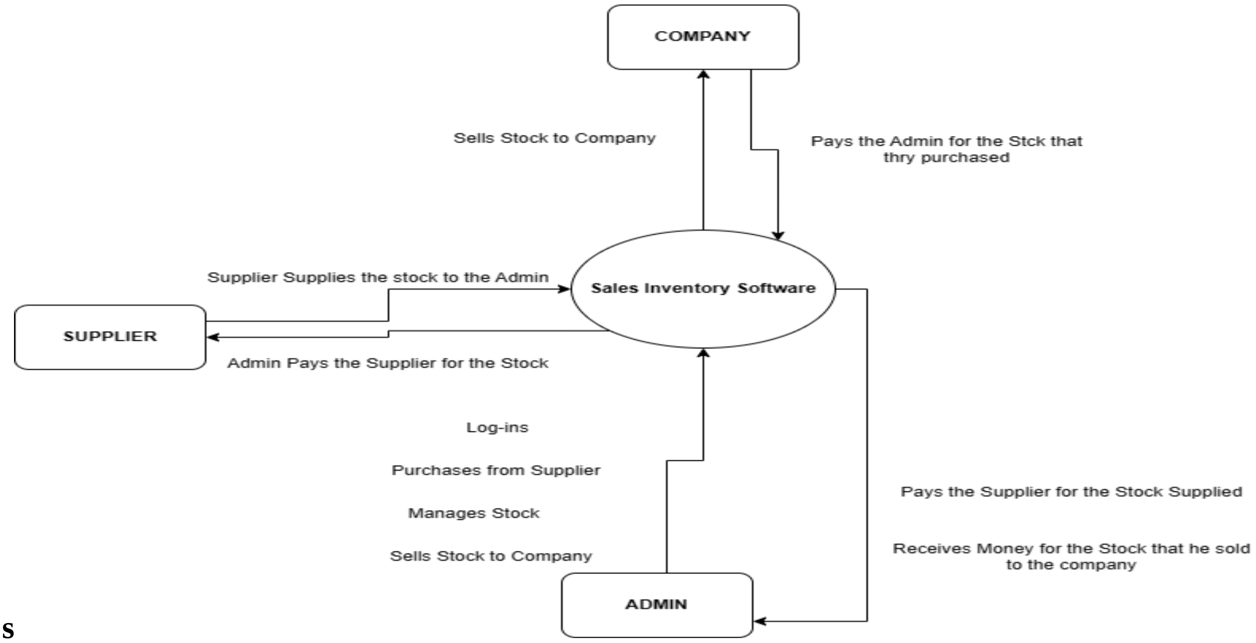
In our course, we need to understand and be able to draw 2 types of Data Flow Diagrams, they are Level-0 and Level 1 DFD's. In this blog, I will hopefully make it easier to understand the differences between the two types of DFD's and help understand how to draw a DFD for the exam.

Firstly, we will look at level-0 DFD's and give an example. Then we will look at Level 1 DFD's and give an example.

A level-0 DFD is the most basic form of DFD. It aims to show how the entire system works at a glance. There is only one process in the system and all the data flows either into or out of this process. Level- 0 DFD's demonstrates the interactions between the process and external entities. They do not contain Data Stores.

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

LEVEL ZERO DIAGRAM FOR SALES INVENTORY SOFTWARE:


COMPANY
Sells Stock to Company
Pays the Admin for the Stck that thry purchased
Supplier Supplies the stock to the Admin
Sales Inventory Software
SUPPLIER
Admin Pays the Supplier for the Stock
Log-ins
Purchases from Supplier
Manages Stock
Sells Stock to Company
Pays the Supplier for the Stock Supplied
Receives Money for the Stock that he sold to the company
ADMIN

s

LEVEL 1 DFD’s:

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

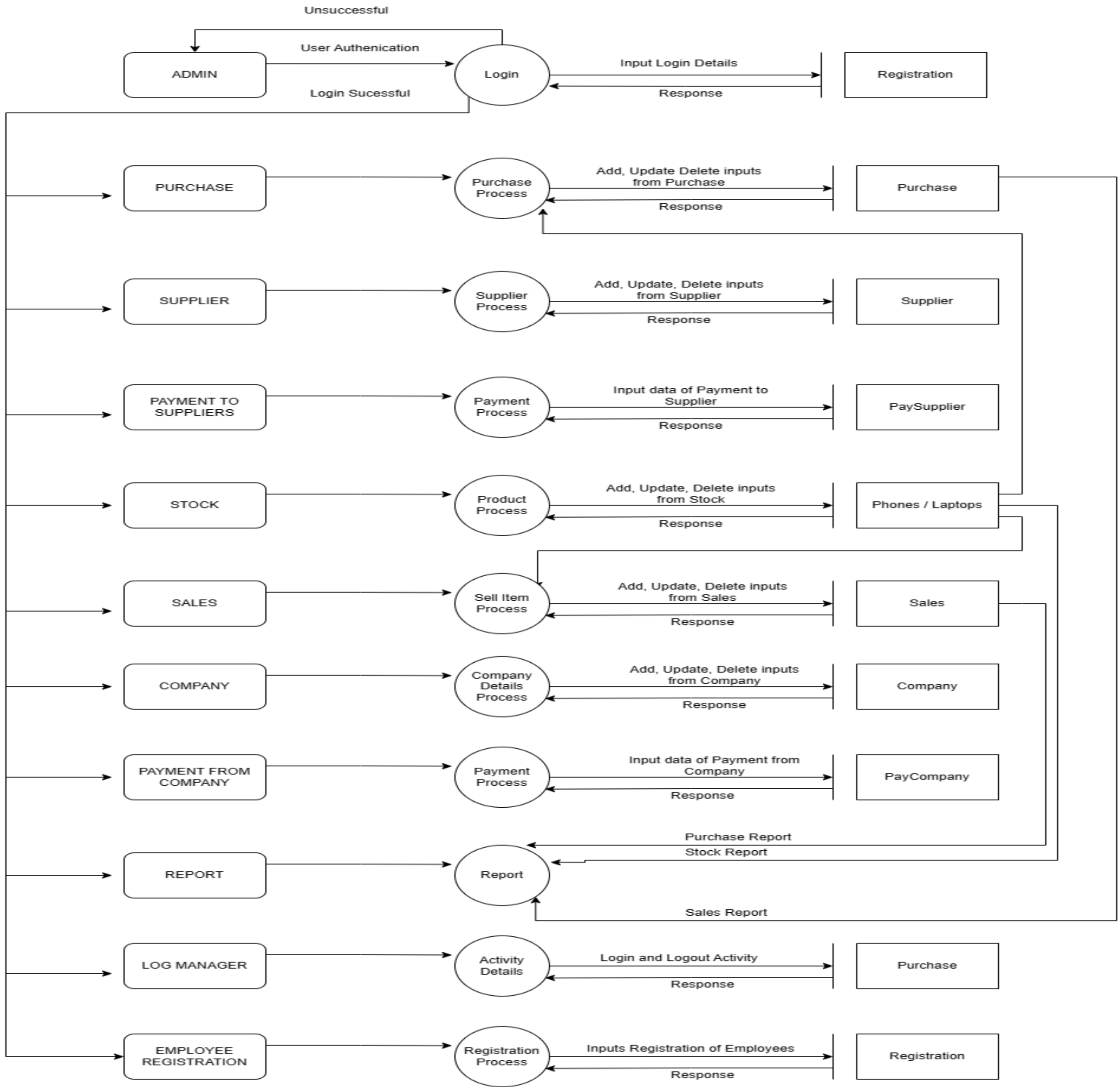
We must break up the single process into its subprocesses.

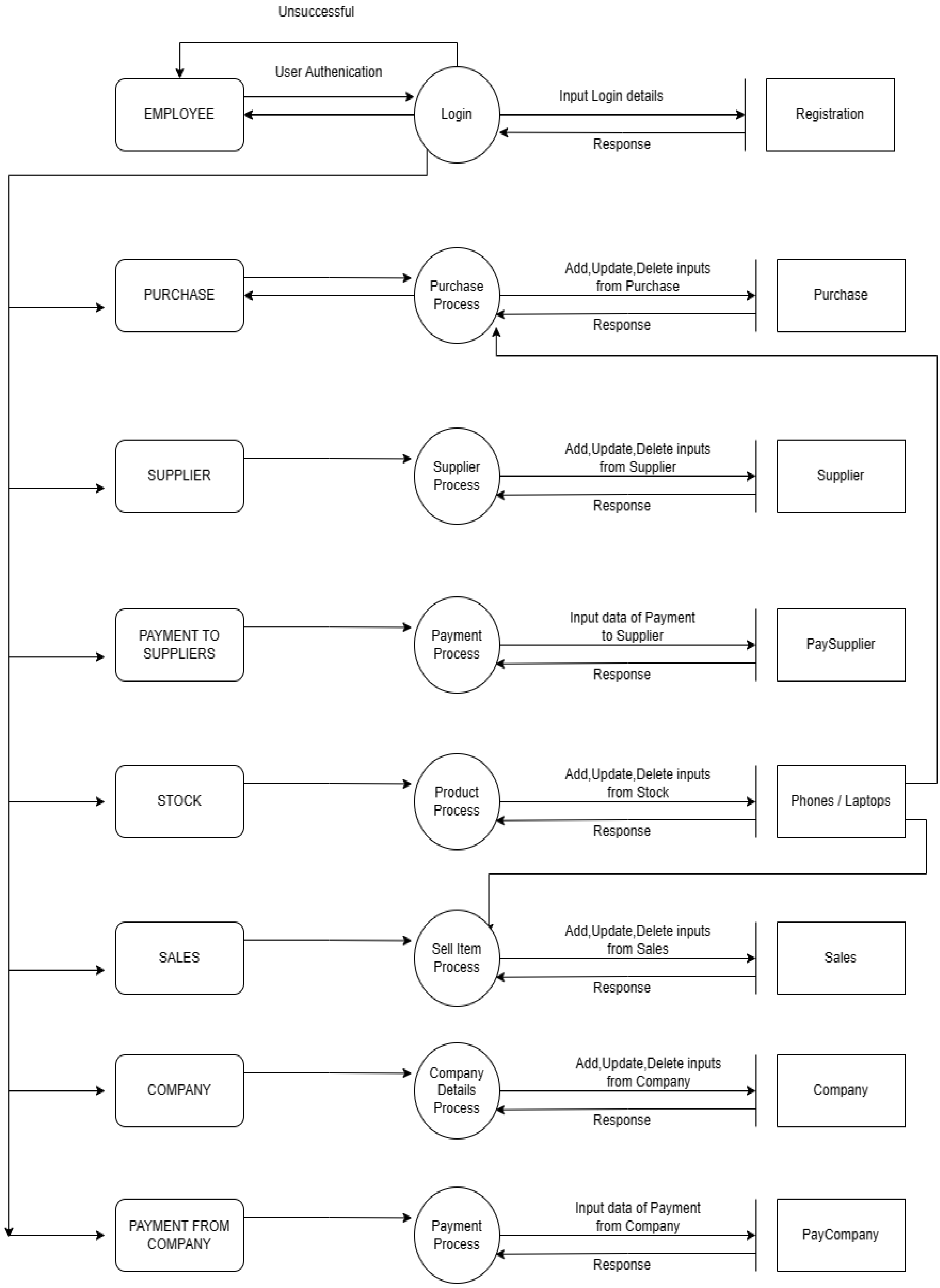
We must then pick out the data stores from the text we are given and include them in our DFD.

Like the Context Level DFD’s, all entities, data stores and processes must be labelled.

We must also state any assumptions made from the text.

LEVEL ONE DIAGRAM FOR SALES INVENTORY SOFTWARE:




Unsuccessful
User Authenication
EMPLOYEE
Login
Input Login details
Response
Registration
PURCHASE
Purchase Process
Add,Update,Delete inputs from Purchase
Response
Purchase
SUPPLIER
Supplier Process
Add,Update,Delete inputs from Supplier
Response
Supplier
PAYMENT TO SUPPLIERS
Payment Process
Input data of Payment to Supplier
Response
PaySupplier
STOCK
Product Process
Add,Update,Delete inputs from Stock
Response
Phones / Laptops
SALES
Sell Item Process
Add,Update,Delete inputs from Sales
Response
Sales
COMPANY
Company Details Process
Add,Update,Delete inputs from Company
Response
Company
PAYMENT FROM COMPANY
Payment Process
Input data of Payment from Company
Response
PayCompany

3.3 GANTT CHART:

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

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

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

3.3.1 HISTORICAL DEVELOPMENT:

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

The chart is named after Henry Gantt (1861– 1919), who designed his chart around the years 1910–1915.

One of the first major applications of Gantt charts was by the United States during World War I, at the instigation of General William Crozier

In the 1980s, personal computers allowed widespread creation of complex and elaborate Gantt charts. The first desktop applications were intended mainly for project managers and project schedulers.

With the advent of the Internet and increased collaboration over networks at the end of the 1990s, Gantt charts became a common feature of web-based applications, including collaborative groupware.

3.3.2 GANTT CHART BENEFITS:

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

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

MOTIVATION: Some teams or team members become more effective when faced with a form of external motivation. Gantt charts offer teams the ability to focus work at the front of a task timeline, or at the tail end of a chart segment. Both types of team members can find Gantt charts meaningful as they plug their own work habits into the overall project schedule.

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

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

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

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

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

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

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

3.3.3 GANTT CHART IMPORTANCE:

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

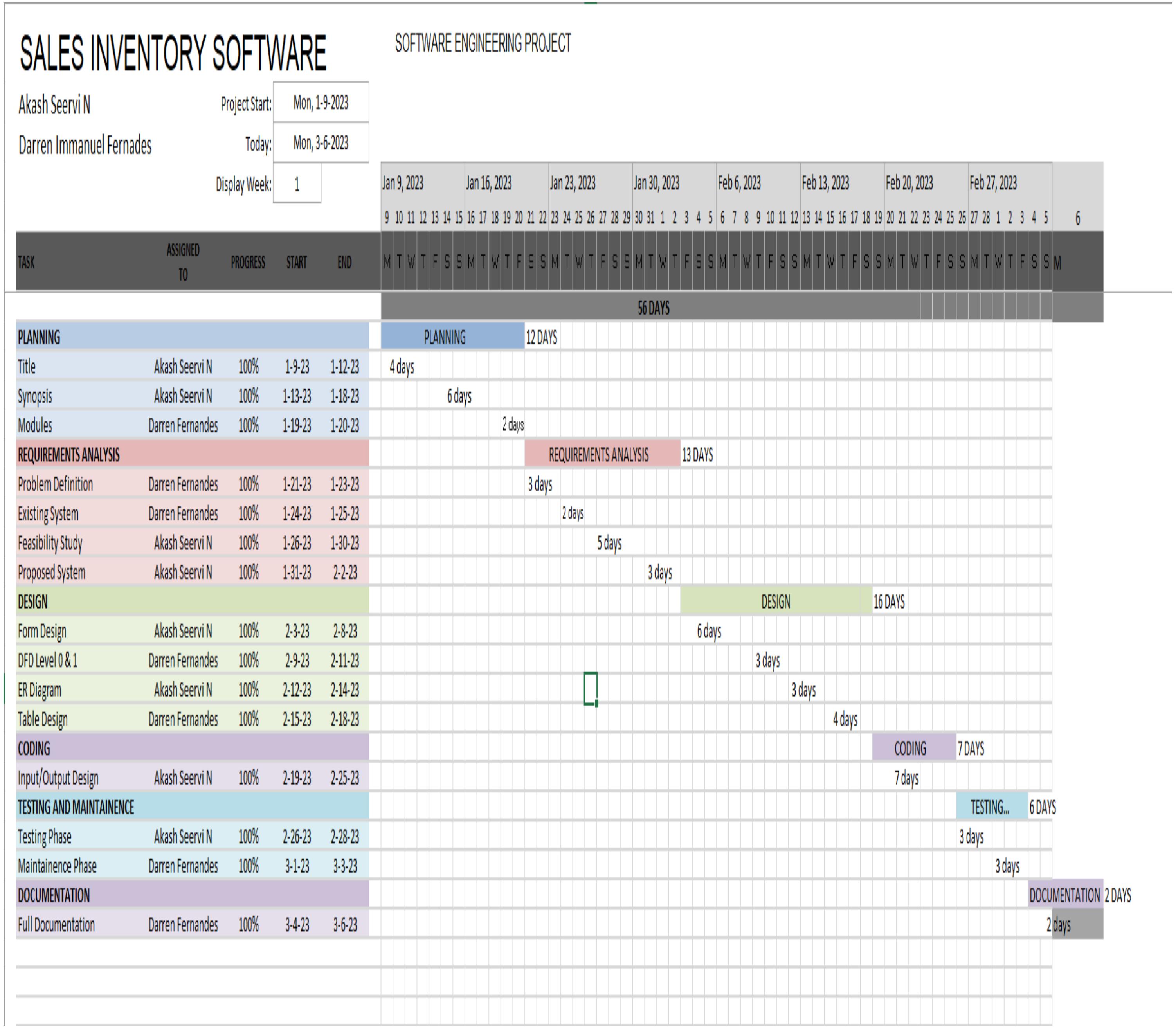
It can also show summary elements and terminal dependencies. The smallest task tracked as part of the project effort is known as a terminal element. Gantt chart represents the tasks in most modern project scheduling packages. However other management applications use simpler communication tools such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

The way to create this chart begins by determining and listing the necessary activities. Next, sketch out how you expect the chart to look. List which items depend on others and what activities take place when. For each activity, list how many man-hours it will require, and who is responsible.

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

Herewith is the Gantt chart for SALES INVENTORY SOFTWARE.

GANNT CHART OF SALES INVENTORY SOFTWARE:



SALES INVENTORY SOFTWARE

SOFTWARE ENGINEERING PROJECT

Akash Seervi N

Darren Immanuel Fernades

Project Start:	Mon, 1-9-2023
Today:	Mon, 3-6-2023
Display Week:	1

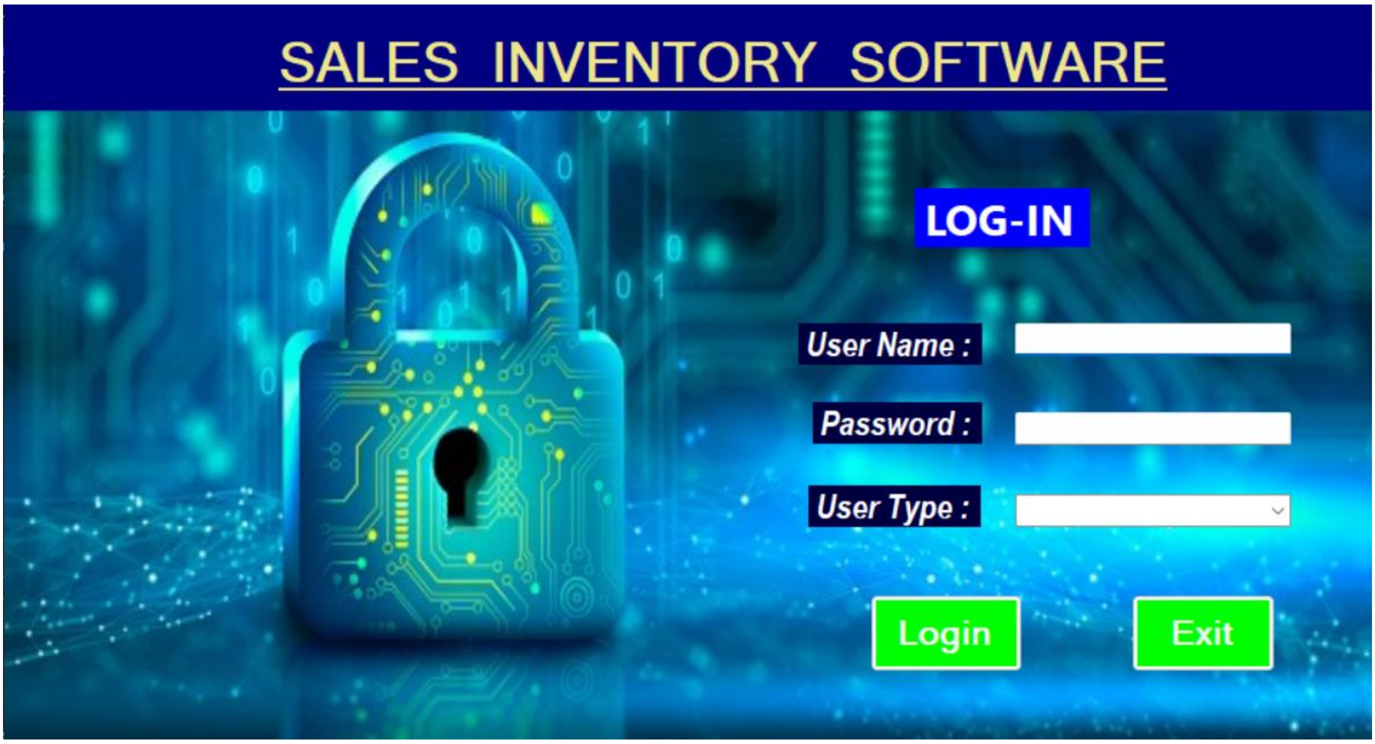
TASK	ASSIGNED TO	PROGRESS	START	END
PLANNING				
Title	Akash Seervi N	100%	1-9-23	1-12-23
Synopsis	Akash Seervi N	100%	1-13-23	1-18-23
Modules	Darren Fernandes	100%	1-19-23	1-20-23
REQUIREMENTS ANALYSIS				
Problem Definition	Darren Fernandes	100%	1-21-23	1-23-23
Existing System	Darren Fernandes	100%	1-24-23	1-25-23
Feasibility Study	Akash Seervi N	100%	1-26-23	1-30-23
Proposed System	Akash Seervi N	100%	1-31-23	2-2-23
DESIGN				
Form Design	Akash Seervi N	100%	2-3-23	2-8-23
DFD Level 0 & 1	Darren Fernandes	100%	2-9-23	2-11-23
ER Diagram	Akash Seervi N	100%	2-12-23	2-14-23
Table Design	Darren Fernandes	100%	2-15-23	2-18-23
CODING				
Input/Output Design	Akash Seervi N	100%	2-19-23	2-25-23
TESTING AND MAINTAINENCE				
Testing Phase	Akash Seervi N	100%	2-26-23	2-28-23
Maintainence Phase	Darren Fernandes	100%	3-1-23	3-3-23
DOCUMENTATION				
Full Documentation	Darren Fernandes	100%	3-4-23	3-6-23

3.4 INPUT/OUTPUT DESIGN:

LOGIN FORM:

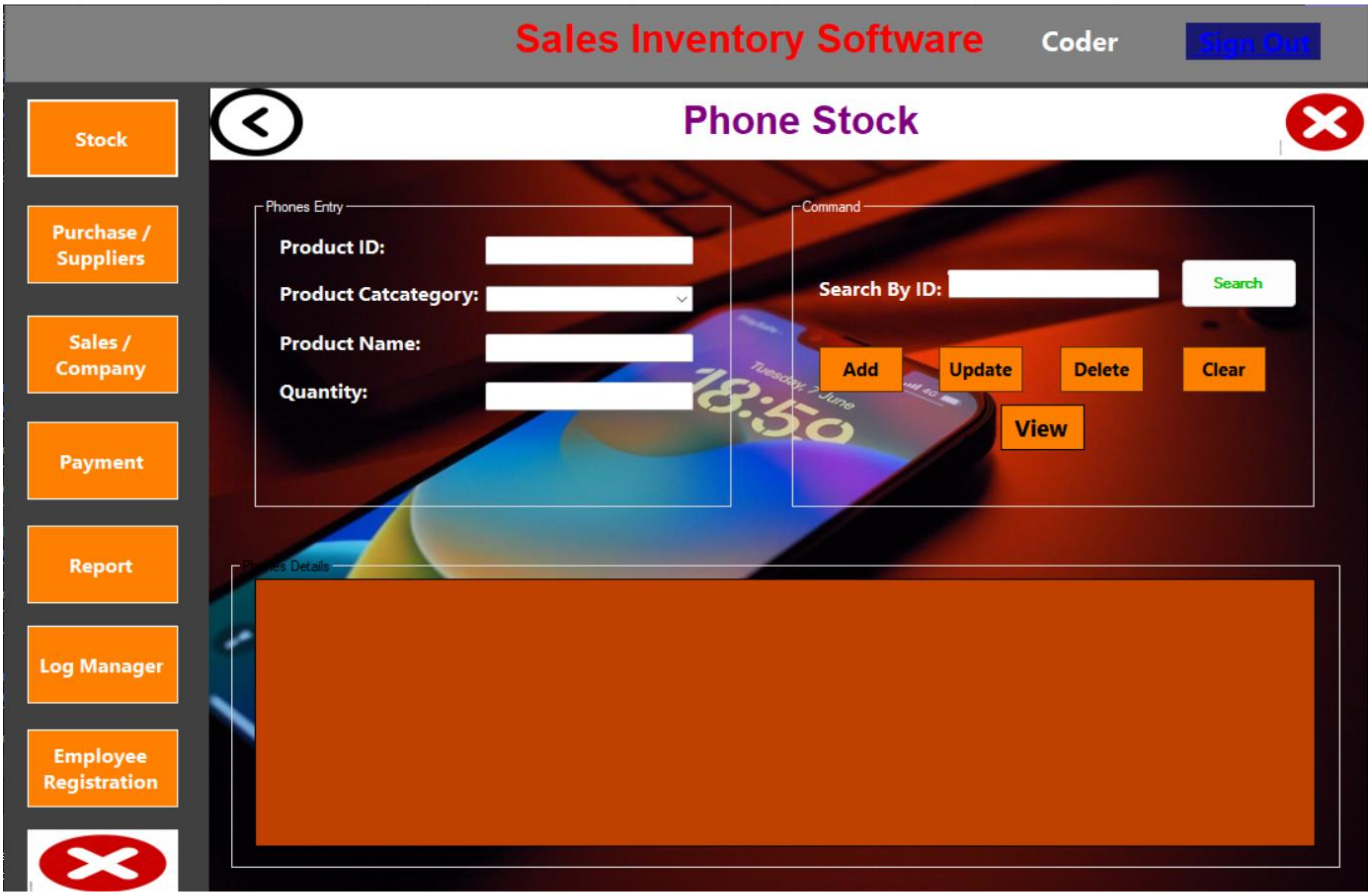
```
<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()>
Partial Class LoginForm
    Inherits System.Windows.Forms.Form
    'Form overrides dispose to clean up the component list.
    <System.Diagnostics.DebuggerNonUserCode()>
    Protected Overrides Sub Dispose(ByVal disposing As Boolean)
        Try
            If disposing AndAlso components IsNot Nothing Then
                components.Dispose()
            End If
        Finally
            MyBase.Dispose(disposing)
        End Try
    End Sub
    'Required by the Windows Form Designer
    Private components As System.ComponentModel.IContainer
    'NOTE: The following procedure is required by the Windows Form Designer
    'It can be modified using the Windows Form Designer.
    'Do not modify it using the code editor.
    <System.Diagnostics.DebuggerStepThrough()>
    Private Sub InitializeComponent()
        Me.TextBox1 = New System.Windows.Forms.TextBox()
        Me.TextBox2 = New System.Windows.Forms.TextBox()
        Me.ComboBox1 = New System.Windows.Forms.ComboBox()
        Me.Label1 = New System.Windows.Forms.Label()
        Me.Label2 = New System.Windows.Forms.Label()
        Me.Label3 = New System.Windows.Forms.Label()
        Me.Panel1 = New System.Windows.Forms.Panel()
        Me.Label4 = New System.Windows.Forms.Label()
        Me.btnLogin = New System.Windows.Forms.Button()
        Me.Button2 = New System.Windows.Forms.Button()
        Me.Label5 = New System.Windows.Forms.Label()
        Me.Panel1.SuspendLayout()
        Me.SuspendLayout()
        '
        'TextBox1
        '
        Me.TextBox1.Font = New System.Drawing.Font("Segoe UI", 9.0!, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, CType(0, Byte))
        Me.TextBox1.Location = New System.Drawing.Point(938, 273)
        Me.TextBox1.Name = "TextBox1"
        Me.TextBox1.Size = New System.Drawing.Size(254, 27)
        Me.TextBox1.TabIndex = 0
        '
        'TextBox2
        '
        Me.TextBox2.Font = New System.Drawing.Font("Segoe UI", 9.0!, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, CType(0, Byte))
```

```
        Me.TextBox2.Location = New System.Drawing.Point(938, 349)
        Me.TextBox2.Name = "TextBox2"
        Me.TextBox2.Size = New System.Drawing.Size(254, 27)
        Me.TextBox2.TabIndex = 1
        Me.TextBox2.UseSystemPasswordChar = True
        '
        'ComboBox1
        '
        Me.ComboBox1.Font = New System.Drawing.Font("Segoe UI", 9.0!, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, CType(0, Byte))
        Me.ComboBox1.FormattingEnabled = True
        Me.ComboBox1.Items.AddRange(New Object() {"Admin", "Employee"})
        Me.ComboBox1.Location = New System.Drawing.Point(938, 420)
        Me.ComboBox1.Name = "ComboBox1"
        Me.ComboBox1.Size = New System.Drawing.Size(254, 28)
        Me.ComboBox1.TabIndex = 2
        '
        'Label1
        '
        Me.Label1.AutoSize = True
        Me.Label1.BackColor = System.Drawing.Color.FromArgb(CType(CType(0, Byte), Integer),
CType(CType(0, Byte), Integer), CType(CType(64, Byte), Integer))
        Me.Label1.Font = New System.Drawing.Font("Arial Narrow", 18.0!,
CType((System.Drawing.FontStyle.Bold Or System.Drawing.FontStyle.Italic), System.Drawing.FontStyle),
System.Drawing.GraphicsUnit.Point, CType(0, Byte))
        Me.Label1.ForeColor = System.Drawing.Color.White
        Me.Label1.Location = New System.Drawing.Point(737, 273)
        Me.Label1.Name = "Label1"
        Me.Label1.Size = New System.Drawing.Size(161, 35)
        Me.Label1.TabIndex = 3
        Me.Label1.Text = "User Name :"
        '
        'Label2
        '
        Me.Label2.AutoSize = True
        Me.Label2.BackColor = System.Drawing.Color.FromArgb(CType(CType(0, Byte), Integer),
CType(CType(0, Byte), Integer), CType(CType(64, Byte), Integer))
        Me.Label2.Font = New System.Drawing.Font("Arial Narrow", 18.0!,
CType((System.Drawing.FontStyle.Bold Or System.Drawing.FontStyle.Italic), System.Drawing.FontStyle),
System.Drawing.GraphicsUnit.Point, CType(0, Byte))
End Class
```



STOCK:

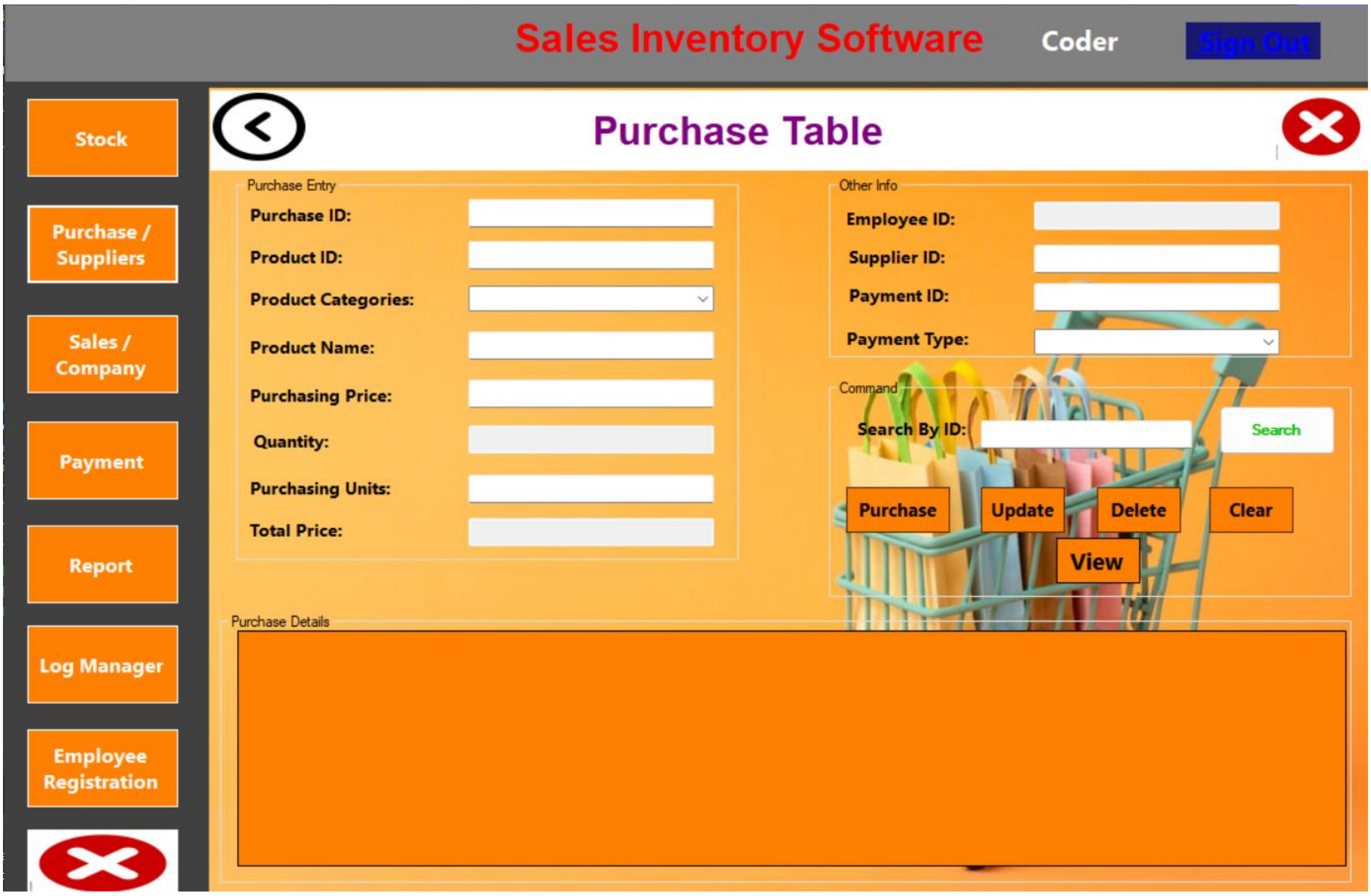
```
<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()> _
Partial Class Phones
    Inherits System.Windows.Forms.Form
    'Form overrides dispose to clean up the component list.
    <System.Diagnostics.DebuggerNonUserCode()> _
    Protected Overrides Sub Dispose(ByVal disposing As Boolean)
        Try
            If disposing AndAlso components IsNot Nothing Then
                components.Dispose()
            End If
        Finally
            MyBase.Dispose(disposing)
        End Try
    End Sub
    'Required by the Windows Form Designer
    Private components As System.ComponentModel.IContainer
    'NOTE: The following procedure is required by the Windows Form Designer
    'It can be modified using the Windows Form Designer.
    'Do not modify it using the code editor.
    <System.Diagnostics.DebuggerStepThrough()> _
    Private Sub InitializeComponent()
        Me.Panel1 = New System.Windows.Forms.Panel()
        Me.PictureBox2 = New System.Windows.Forms.PictureBox()
        Me.PictureBox1 = New System.Windows.Forms.PictureBox()
        Me.Label7 = New System.Windows.Forms.Label()
        Me.GroupBox2 = New System.Windows.Forms.GroupBox()
        Me.Button2 = New System.Windows.Forms.Button()
        Me.btnCancel = New System.Windows.Forms.Button()
        Me.btnDelete = New System.Windows.Forms.Button()
        Me.btnUpdate = New System.Windows.Forms.Button()
        Me.btnAdd = New System.Windows.Forms.Button()
        Me.btnSearch = New System.Windows.Forms.Button()
        Me.Label1 = New System.Windows.Forms.Label()
        Me.TextBox1 = New System.Windows.Forms.TextBox()
        Me.GroupBox1 = New System.Windows.Forms.GroupBox()
        Me.ComboBox1 = New System.Windows.Forms.ComboBox()
        Me.Label6 = New System.Windows.Forms.Label()
        Me.TextBox5 = New System.Windows.Forms.TextBox()
        Me.TextBox6 = New System.Windows.Forms.TextBox()
        Me.Label5 = New System.Windows.Forms.Label()
        Me.Label2 = New System.Windows.Forms.Label()
        Me.Label3 = New System.Windows.Forms.Label()
        Me.TextBox3 = New System.Windows.Forms.TextBox()
        Me.GroupBox3 = New System.Windows.Forms.GroupBox()
        Me.DataGridView1 = New System.Windows.Forms.DataGridView()
        Me.Panel1.SuspendLayout()
        CType(Me.PictureBox2, System.ComponentModel.ISupportInitialize).BeginInit()
        CType(Me.PictureBox1, System.ComponentModel.ISupportInitialize).BeginInit()
        Me.GroupBox2.SuspendLayout()
        Me.GroupBox1.SuspendLayout()
        Me.GroupBox3.SuspendLayout()
        CType(Me.DataGridView1, System.ComponentModel.ISupportInitialize).BeginInit()
        Me.SuspendLayout()
        '
        'Panel1
        '
        Me.Panel1.BackColor = System.Drawing.Color.White
        Me.Panel1.Controls.Add(Me.PictureBox2)
        Me.Panel1.Controls.Add(Me.PictureBox1)
        Me.Panel1.Controls.Add(Me.Label7)
        Me.Panel1.Location = New System.Drawing.Point(1, 0)
        Me.Panel1.Name = "Panel1"
        Me.Panel1.Size = New System.Drawing.Size(1256, 71)
        Me.Panel1.TabIndex = 8
        '
        'PictureBox2
        Me.PictureBox2.Image =
Global.SIS.My.Resources.Resources._1000_F_145200273_450ViYipr5uU3WIwqzwjsRDHYTMcUH9P
        Me.PictureBox2.Location = New System.Drawing.Point(1153, 0)
        Me.PictureBox2.Name = "PictureBox2"
End Class
```



PURCHASE:

```
<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()>
Partial Class buy
    Inherits System.Windows.Forms.Form
    'Form overrides dispose to clean up the component list.
    <System.Diagnostics.DebuggerNonUserCode()>
    Protected Overrides Sub Dispose(ByVal disposing As Boolean)
        Try
            If disposing AndAlso components IsNot Nothing Then
                components.Dispose()
            End If
        Finally
            MyBase.Dispose(disposing)
        End Try
    End Sub
```

```
'Required by the Windows Form Designer
Private components As System.ComponentModel.IContainer
'NOTE: The following procedure is required by the Windows Form Designer
'It can be modified using the Windows Form Designer.
'Do not modify it using the code editor.
<System.Diagnostics.DebuggerStepThrough()>
Private Sub InitializeComponent()
    Me.Panel1 = New System.Windows.Forms.Panel()
    Me.PictureBox2 = New System.Windows.Forms.PictureBox()
    Me.PictureBox1 = New System.Windows.Forms.PictureBox()
    Me.Label7 = New System.Windows.Forms.Label()
    Me.GroupBox1 = New System.Windows.Forms.GroupBox()
    Me.TextBox11 = New System.Windows.Forms.TextBox()
    Me.Label13 = New System.Windows.Forms.Label()
    Me.TextBox4 = New System.Windows.Forms.TextBox()
    Me.ComboBox1 = New System.Windows.Forms.ComboBox()
    Me.TextBox8 = New System.Windows.Forms.TextBox()
    Me.TextBox7 = New System.Windows.Forms.TextBox()
    Me.TextBox6 = New System.Windows.Forms.TextBox()
    Me.TextBox10 = New System.Windows.Forms.TextBox()
    Me.Label9 = New System.Windows.Forms.Label()
    Me.Label12 = New System.Windows.Forms.Label()
    Me.Label8 = New System.Windows.Forms.Label()
    Me.Label5 = New System.Windows.Forms.Label()
    Me.TextBox3 = New System.Windows.Forms.TextBox()
    Me.Label4 = New System.Windows.Forms.Label()
    Me.Label3 = New System.Windows.Forms.Label()
    Me.Label2 = New System.Windows.Forms.Label()
    Me.TextBox2 = New System.Windows.Forms.TextBox()
    Me.Label1 = New System.Windows.Forms.Label()
    Me.TextBox1 = New System.Windows.Forms.TextBox()
    Me.Label6 = New System.Windows.Forms.Label()
   End Class
```

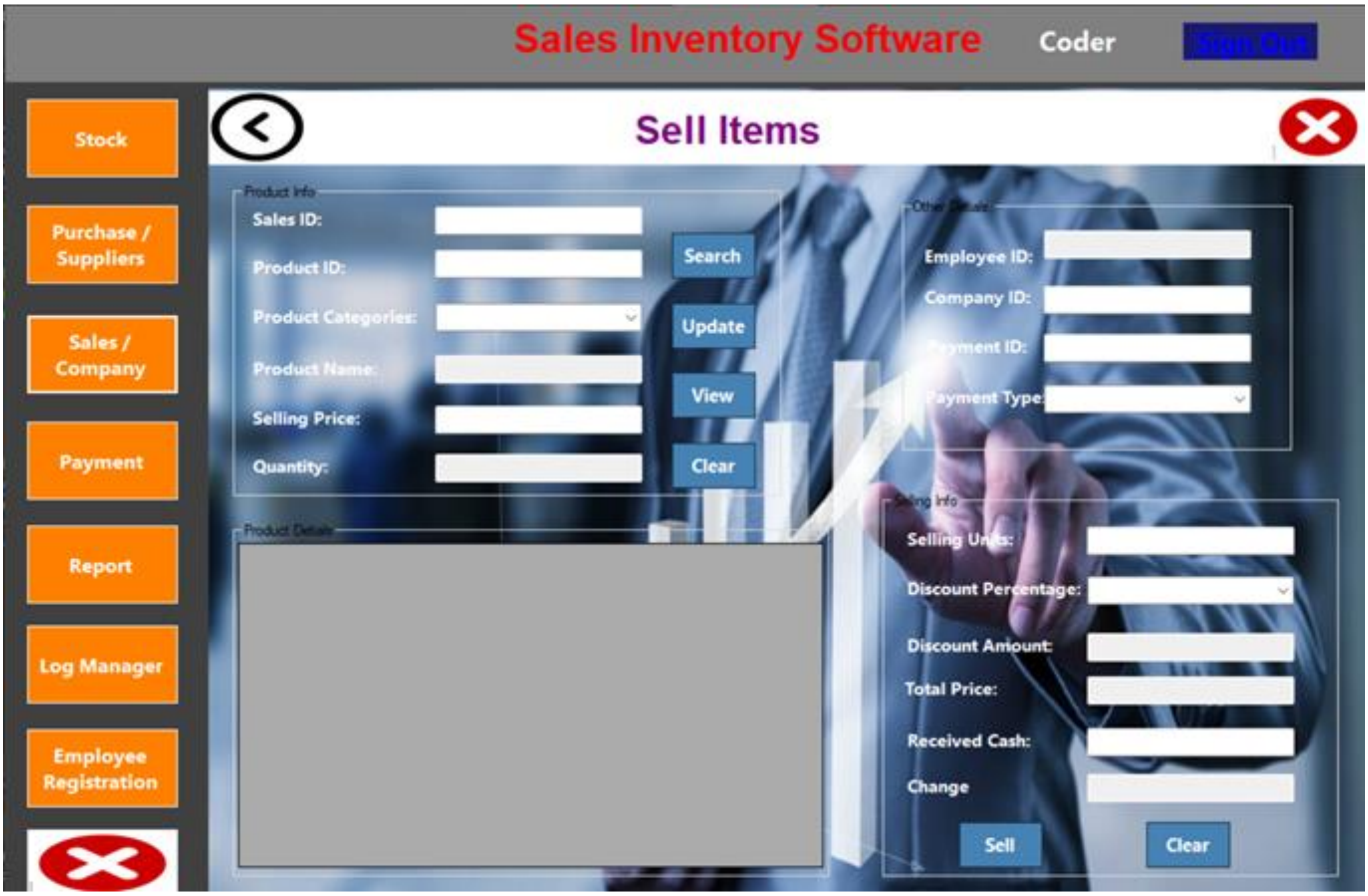


PATIENT REGISTRATION & UPDATE FORM:

```
<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()>
Partial Class SellItems
    Inherits System.Windows.Forms.Form

    'Form overrides dispose to clean up the component list.
    <System.Diagnostics.DebuggerNonUserCode()>
    Protected Overrides Sub Dispose(ByVal disposing As Boolean)
        Try
            If disposing AndAlso components IsNot Nothing Then
                components.Dispose()
            End If
        Finally
            MyBase.Dispose(disposing)
        End Try
    End Sub

    'Required by the Windows Form Designer
    Private components As System.ComponentModel.IContainer

    'NOTE: The following procedure is required by the Windows Form Designer
```

```
'It can be modified using the Windows Form Designer.
'Do not modify it using the code editor.
<System.Diagnostics.DebuggerStepThrough()>
Private Sub InitializeComponent()
    Me.Panel1 = New System.Windows.Forms.Panel()
    Me.PictureBox2 = New System.Windows.Forms.PictureBox()
    Me.PictureBox1 = New System.Windows.Forms.PictureBox()
    Me.Label7 = New System.Windows.Forms.Label()
    Me.GroupBox3 = New System.Windows.Forms.GroupBox()
    Me.TextBox13 = New System.Windows.Forms.TextBox()
    Me.Label16 = New System.Windows.Forms.Label()
    Me.Button5 = New System.Windows.Forms.Button()
    Me.Button4 = New System.Windows.Forms.Button()
    Me.Button3 = New System.Windows.Forms.Button()
    Me.TextBox2 = New System.Windows.Forms.TextBox()
    Me.ComboBox1 = New System.Windows.Forms.ComboBox()
    Me.Button2 = New System.Windows.Forms.Button()
    Me.TextBox5 = New System.Windows.Forms.TextBox()
    Me.Label5 = New System.Windows.Forms.Label()
    Me.TextBox4 = New System.Windows.Forms.TextBox()
    Me.Label12 = New System.Windows.Forms.Label()
    Me.Label4 = New System.Windows.Forms.Label()
    Me.Label8 = New System.Windows.Forms.Label()
    Me.TextBox3 = New System.Windows.Forms.TextBox()
    Me.Label3 = New System.Windows.Forms.Label()
    Me.GroupBox1 = New System.Windows.Forms.GroupBox()
    Me.TextBox12 = New System.Windows.Forms.TextBox()
    Me.Label15 = New System.Windows.Forms.Label()
    Me.TextBox6 = New System.Windows.Forms.TextBox()
    Me.Label2 = New System.Windows.Forms.Label()
    Me.TextBox1 = New System.Windows.Forms.TextBox()
    Me.Label1 = New System.Windows.Forms.Label()
    Me.GroupBox2 = New System.Windows.Forms.GroupBox()
    Me.TextBox11 = New System.Windows.Forms.TextBox()
    Me.Label14 = New System.Windows.Forms.Label()
    Me.ComboBox2 = New System.Windows.Forms.ComboBox()
    Me.Label13 = New System.Windows.Forms.Label()
    Me.TextBox10 = New System.Windows.Forms.TextBox()
    Me.Button1 = New System.Windows.Forms.Button()
    Me.btnAdd = New System.Windows.Forms.Button()
    Me.Label9 = New System.Windows.Forms.Label()
    Me.Label6 = New System.Windows.Forms.Label()
    Me.TextBox7 = New System.Windows.Forms.TextBox()
    Me.Label11 = New System.Windows.Forms.Label()
    Me.TextBox9 = New System.Windows.Forms.TextBox()
    Me.TextBox8 = New System.Windows.Forms.TextBox()
    Me.Label10 = New System.Windows.Forms.Label()
    Me.GroupBox4 = New System.Windows.Forms.GroupBox()
    Me.DataGridView1 = New System.Windows.Forms.DataGridView()
    Me.Label17 = New System.Windows.Forms.Label()
    Me.ComboBox3 = New System.Windows.Forms.ComboBox()
    Me.Panel1.SuspendLayout()
    CType(Me.PictureBox2, System.ComponentModel.ISupportInitialize).BeginInit()
```

```
        CType(Me.PictureBox1, System.ComponentModel.ISupportInitialize).BeginInit()
        Me.GroupBox3.SuspendLayout()
        Me.GroupBox1.SuspendLayout()
        Me.GroupBox2.SuspendLayout()
        Me.GroupBox4.SuspendLayout()
        CType(Me.DataGridView1, System.ComponentModel.ISupportInitialize).BeginInit()
        Me.SuspendLayout()
        '
 End Class
```



4. SYSTEM CONFIGURATION:

4.1 HARDWARE REQUIREMENTS:

PROCESSOR	Intel Core i5
SPEED	1.00 GHz
RAM	8GB

4.2 SOFTWARE REQUIREMENTS:

FRONTEND	VB.NET (VISUAL STUDIO) 2022
BACKEND	SQL SERVER 2022
DOCUMENTATION	MS WORD 2021
GANTT CHART	MS PROJECT 2021
OPERATING SYSTEM	WINDOWS 11 22H2

5. DETAILS OF S0FTWARE:

5.1 OVERVIEW OF FRONT-END:

Microsoft Visual Studio 2022 is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as web sites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

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

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

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

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

5.1.1 FEATURES:

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

5.1.2 ADVANTAGES:

- The structure of the Basic programming language is very simple, particularly as to the executable code.
- VB is not only a language but primarily an integrated, interactive development environment (“IDE”).
- The VB-IDE has been highly optimized to support rapid application development(“RAD”). It is particularly easy to develop graphical user interface and to connect them to handler functions provided by the application.
- The graphical user interface of the VB-IDE provides intuitively appealing views for the management of the program structure in the large and the various types of entities (classes, modules, procedures, forms, …).
- VB provides a comprehensive interactive and context-sensitive online help system.
- When editing program texts the “IntelliSense” technology informs you in a little popup window about the types of constructs that may be entered at the current cursor location.
- VB is a component integration language which is attuned to Microsoft’s Component Object Model (“COM”). COM components can be written in different languages and then integrated using VB.
- Interfaces of COM components can be easily called remotely via Distributed COM (“DCOM”), which makes it easy to construct distributed applications.
- COM components can be embedded in / linked to your application’s user interface and also in/to stored documents (Object Linking and Embedding “OLE”, “Compound Documents”).
- There is a wealth of readily available COM components for many different purposes.
- Visual Basic is built around the .NET environment used by all Microsoft.

5.2 OVERVIEW OF BACK-END:

SQL Server is a relational database management system developed by Microsoft. It is used to store, manipulate and analyze data using SQL (Structured Query Language), which is standard language for accessing database. SQL Server can run on-primases, on cloud platform like Azure, or on hybrid environment. SQL Server offers various editions for different purpose and workloads, such as Express, Developer, Standard and Enterprise. SQL server also provides features like high availability, stability, security and analytics. Server has many features that make it a powerful and versatile database system. Some of these features are

- Data Definition Language (DDL): SQL Server allows you to create, modify and delete database objects like tables, view, indexes and constraints using SQL commands.
- Data Manipulation Language (DML): SQL server enables you to insert, update, delete and query data stored in tables using SQL commands.

5.2.1 COMPONENTS OF SQL SERVER:

As SQL Server is a relational database management system developed by Microsoft. It consist of several components and services that work together to store, process, and secure data.

Some of the main components of SQL Server are:

- Database Engine: This component is responsible for storage, secure data, and rapid transaction processing. It also creates and executes database object such as stored procedures views and triggers.
- SQL Server Agent: This component is background services that automates administrative tasks such as backups, replication, maintenance plans, etc.
- SQL Server Analysis Services (SSAS): This components provides online analytical processing (OLAP) and data mining capabilities for business intelligence application.
- SQL Server Reporting Services (SSRS): This components provides a platform for creating managing and delivering based on data from various sources.
- SQL Server Integration Services (SSIS): This component provides a framework for building and running data integration workflow that can extract, transform, and load (ETL) data from various sources.

5.2.2 ADVANTAGES OF SQL SERVER:

- Stability: SQL Server can handle large amounts of data and grow with your business needs . It can support up to one terabyte of data per database.
- Data Processing: SQL Server allows you to process and analyze your data using SQL commands, which are fast, efficient and easy to use.
- Data Security: SQL Server ensures that security of you data by providing features like encryption, authentication and auding.
- Ease of configuration: SQL Servers is easy to install and configure, and it integrates well with other Microsoft products and services.
- Optimized Data Storage: SQL Server optimizes the storage of your data by using comprehension, partitioning and indexing techniques.

5.2.3 OTHER COMPONENTS OF SQL SERVER:

1. **Cross-Platform:** This components is a services that runs scheduled tasks or jobs on SQL Server instances. It can execute Transact-SQL statements, PowerShell scripts, SSIS packages, etc., at specified times or intervals.
2. **SQL Server Management Studio (SSMS):** This components is an integrated environment for accessing, configuring, managing, administering, and developing all components of SQL Server. It provides a graphical user interface (GUI) for performing various tasks such as creating database, tables, views, stored procedure, etc. executing queries and scripting, monitoring performance and activity, etc.
3. **SQL Server profiler:** This Component is a tool that captures and analyzes events that occur on SQL Server instances. It can help troubleshoot performance issues, identify errors and exception, audit user activity etc.
4. **SQL Server Configuration Manager:** This component is a tool that manages the services and network protocols used by SQL Server instances. It allows user to start or stop services, changes services accounts or password, enable or disable protocols or features TCP/IP or Named Pipes

5.2.4 SQL SERVER FORMAT SYSTEM:

SQL Server is supported in three file formats:

- **SQL Files**: These are plain text files that contain SQL Statement and commands. You can use them to create, modify or query database and objects.
- **Database Files:** These are binary files that store the data and object of SQL Server database. They have two types: primary data files (with .mdf extenstion) and secondary data files. Each database has one primary data files and zero or more secondary data files.
- **Transaction Files:** These are binary filese that store the log record of all transactions that modify the data in a database. They have .ldf extension and are essential for recovering a database in case of failure.

5.3 ABOUT THE PLATFORM:

Windows is a series of Operating Systems developed by Microsoft. Each version of Windows includes a Graphical User Interface, with a desktop that allows users to view files and folders in Windows. For the past two decades, Windows has been the most widely used operating system for personal computers PCs.

Microsoft Windows is designed for both home computing and professional purposes. Past versions of Windows home editions include Windows 3.0 (1990),Windows 3.1 (1992),Windows 95 (1995), Windows 98 (1998), WindowsMe (2000), WindowsXP (2001), and Windows Vista (2006),windows 7(2009),windows 8(2012),windows 8.1(2013),windows 10(2015).

Windows 11 is the latest major release of Microsoft's Windows NT operating system, released in October 2021. It is a free upgrade to its predecessor, Windows 10 (2015), and is available for any Windows 10 devices that meet the new Windows 11 system requirements.

Windows 11 features major changes to the Windows shell influenced by the cancelled Windows 10X, including a redesigned Start menu, the replacement of its "live tiles" with a separate "Widgets" panel on the taskbar, the ability to create tiled sets of windows that can be minimized and restored from the taskbar as a group, and new gaming technologies inherited from Xbox Series X and Series S such as Auto HDR and Direct Storage on compatible hardware. Internet Explorer (IE) has been replaced by the Chromium-based Microsoft Edge as the default web browser, like its predecessor, Windows 10, and Microsoft Teams is integrated into the Windows shell. Microsoft also announced plans to allow more flexibility in software that can be distributed via the Microsoft Store and to support Android apps on Windows 11 (including a partnership with Amazon to make its app store available for the function).

5.3.1 .NET FRAMEWORK:

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

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

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

Framework and other libraries.

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

Despite the standardization efforts, developers, mainly those in the free and open-source software communities, expressed their unease with the selected terms and the prospects of any free and open-source implementation, especially regarding software patents. Since then, Microsoft has changed .NET development to more closely follow a contemporary model of a community developed software project, including issuing an update to its patent promising to address the concerns.

6. TESTING:

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

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

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

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

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

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

In this light, two important principles become clear:

6.1 TESTS REPRESENT REQUIREMENTS:

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

6.2 WE ARE NOT DONE TILL THE TEST IS PASSED:

The only useful measure of completionis when tests have been performed successfully.

Those principles apply no matter how you develop your software.

6.3 SOFTWARE TESTING TYPES:

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

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

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

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

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

FUNCTIONAL TESTING: This type of testing ignores the internal parts and focus on the output isas per requirement or not. Black-box type testing geared to functional requirements of an application.

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

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

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

REGRESSION TESTING: Testing the application as a whole for the modification in any module or functionality. Difficult to cover all the system in regression testing so typically automation tools are used for these testing types.

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

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

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

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

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

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

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

SECURITY TESTING: Can system be penetrated by any hacking way. Testing how well the system protects against unauthorized internal or external access. Checked if system, database is safe from external attacks.

COMPABILITY TESTING: Testing how well software performs in a particular hardware/software/operating system/network environment and different combination s of above.

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

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

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

7. CONCLUSION AND FUTURE ENHANCEMENT:

7.1 CONCLUSION:

SALES INVENTORY SOFTWARE offers a platform for businessmen or the people who do business to maintain and keep a record of all the product purchased from the supplier and the products sold to the company and the stock available. Thus, through this payment details are also maintained so that it can be easily accessible. The majority of laborious paperwork is eliminated by this application, which also accurately and effectively automates the procedure.

7.2 FUTURE ENHANCEMENT:

SALES INVENTORY SOFTWARE has a lot of scope in future where various modules can be upgraded.

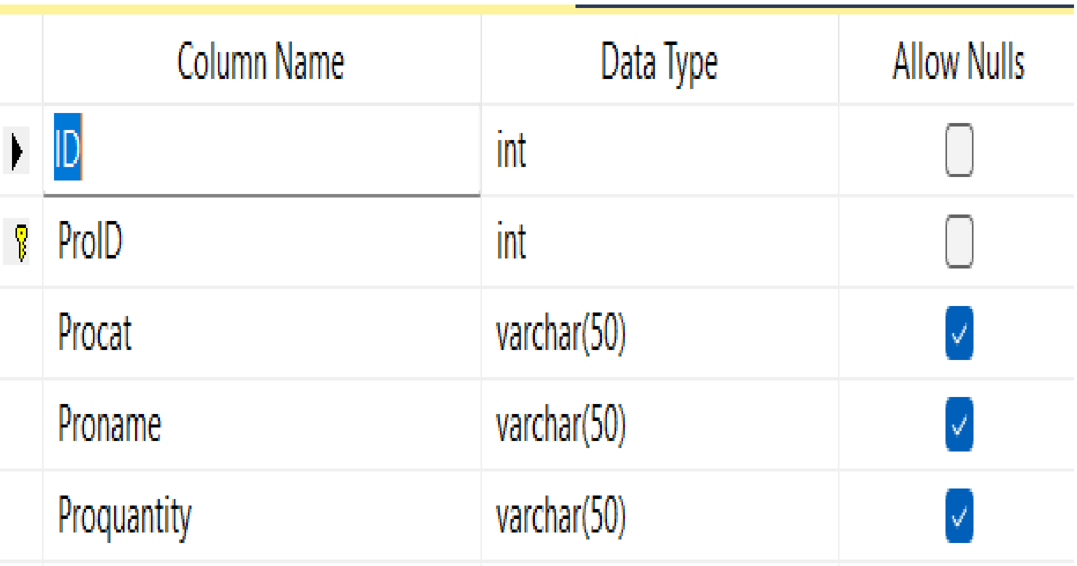
1. Online: SALES INVENTORY SOFTWARE can be turned into an internet website so that business can purchase the products online and sell online and sell online and maintain a record of online payment done as well.
2. Mobile Application: It is possible to create a SALES INVENTORY SOFTWARE mobile application that would also enables business to manage the stock (i.e., Phones, Laptop), purchase the stock from the supplier and pay the supplier for the stock, sell the product to the company and receive money for the stock that they sold to the company, and also view the report of purchase , sales and stock.
3. Open Source: We can maintain our source code in an open-source community so that more modules or features can be added by the open-source community, improving the application's effectiveness and being useful to both ourselves and others.

8. BIBLIOGRAPHY

- Amit Raj, (2016) VB.Net Tutorial, New Delhi.
- Jeffery R. Shipiro, (2017) VB.Net The Complete Reference, Mumbai
- Vikram Vaswani, (2017) MySQL The Complete Reference, Mumbai
- www.geeksforgeeks.org
- www.tutorialspoint.com
- www.javatpoint.com

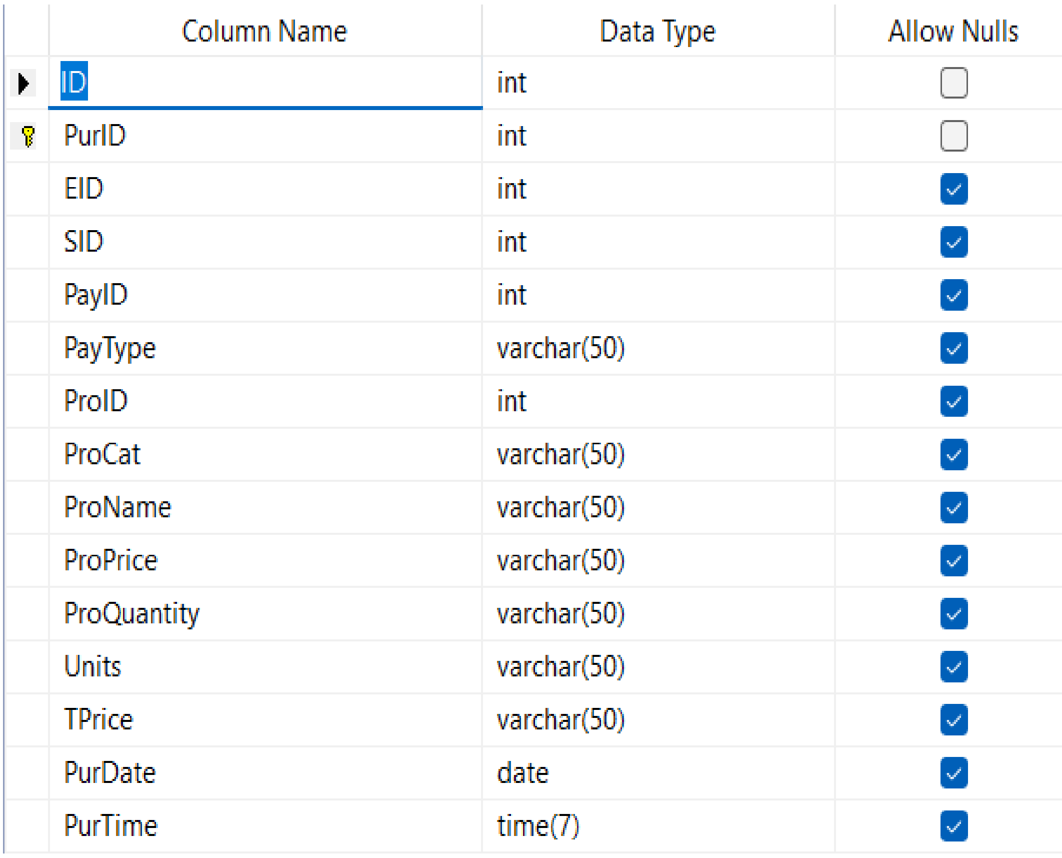
9. APPENDICES A-TABLE STRUCTURE:

9.1 PHONE / LAPTOP STOCK:



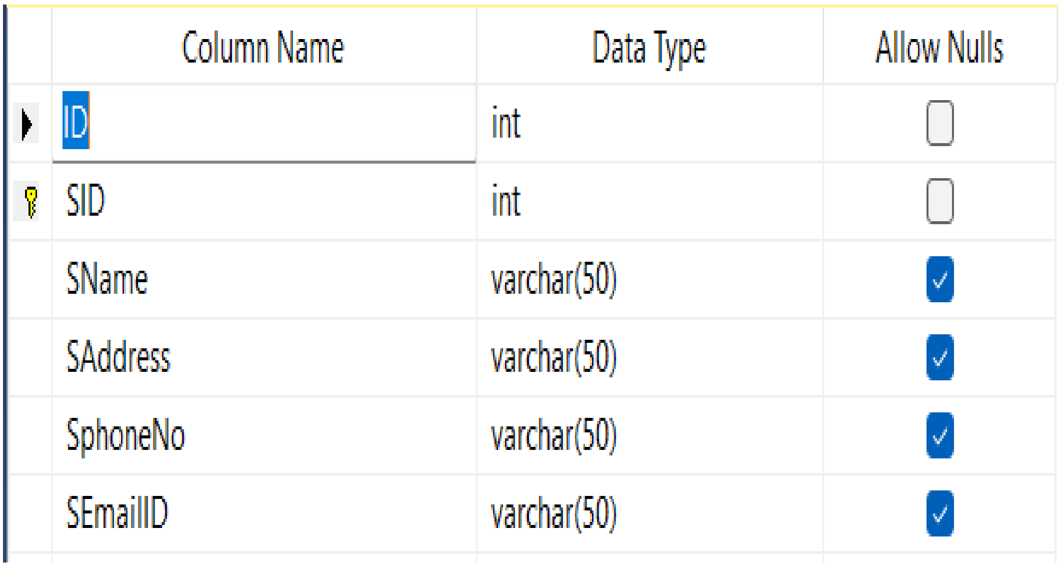
	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	ProID	int	□
	Procat	varchar(50)	☑
	Proname	varchar(50)	☑
	Proquantity	varchar(50)	☑

9.2 PURCHASE TABLE:



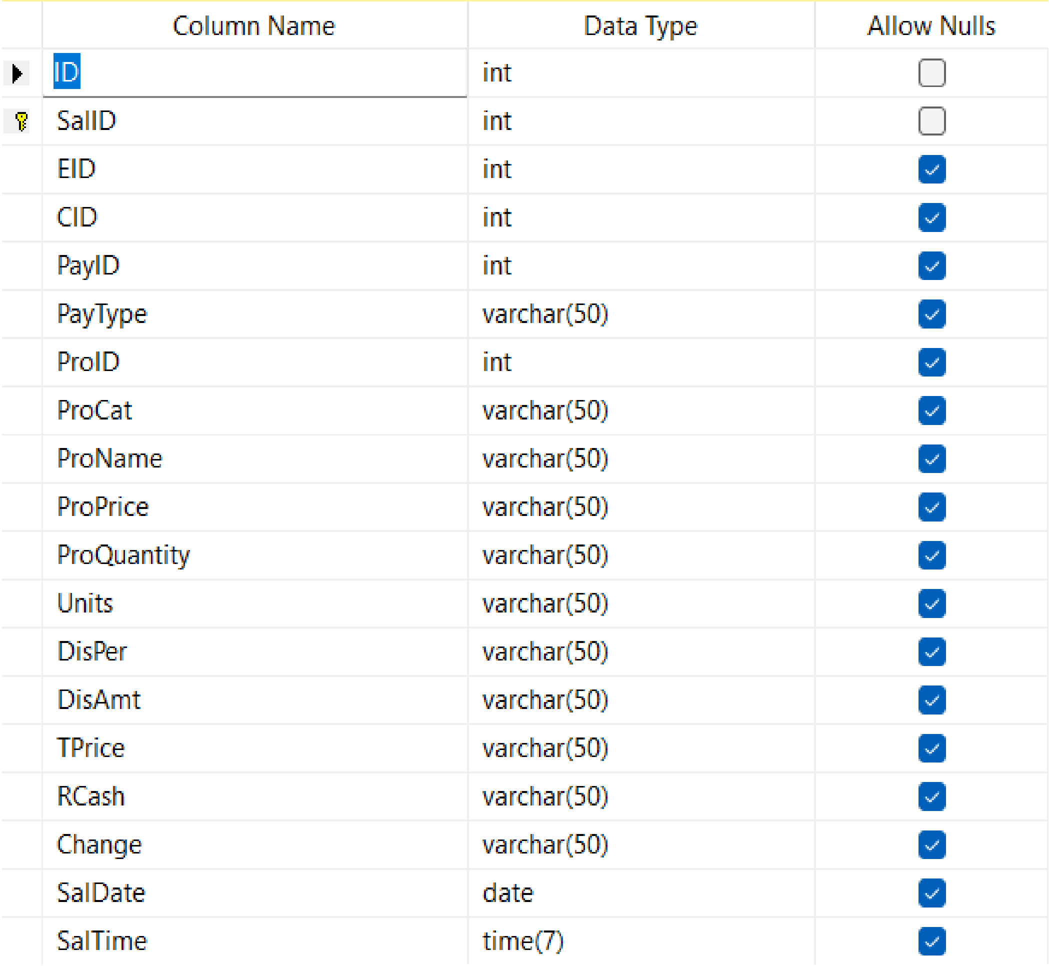
	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	PurID	int	□
	EID	int	☑
	SID	int	☑
	PayID	int	☑
	PayType	varchar(50)	☑
	ProID	int	☑
	ProCat	varchar(50)	☑
	ProName	varchar(50)	☑
	ProPrice	varchar(50)	☑
	ProQuantity	varchar(50)	☑
	Units	varchar(50)	☑
	TPrice	varchar(50)	☑
	PurDate	date	☑
	PurTime	time(7)	☑

9.3 SUPPLIER TABLE:



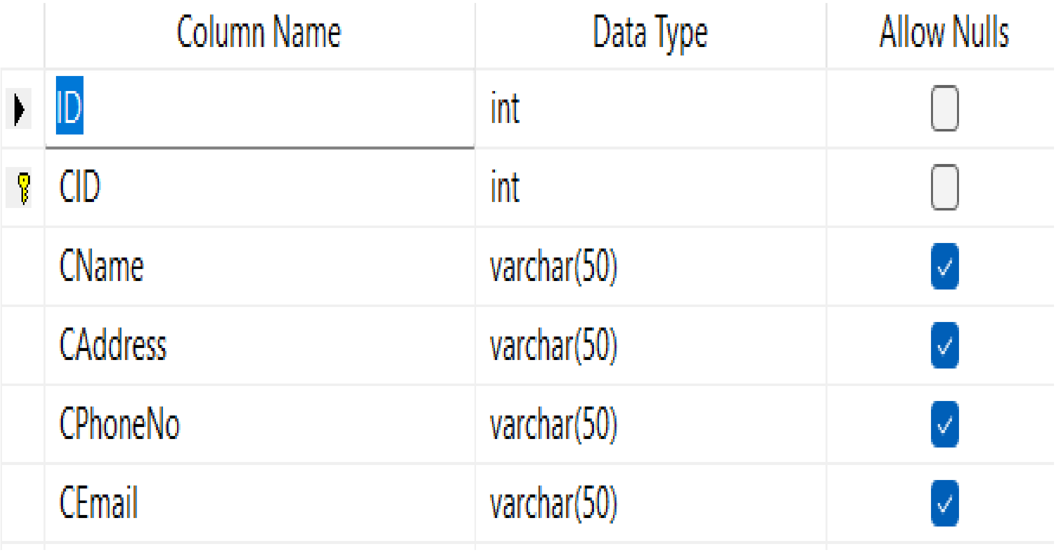
	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	SID	int	□
	SName	varchar(50)	☑
	SAddress	varchar(50)	☑
	SphoneNo	varchar(50)	☑
	SEmailID	varchar(50)	☑

9.4 SALES TABLE:



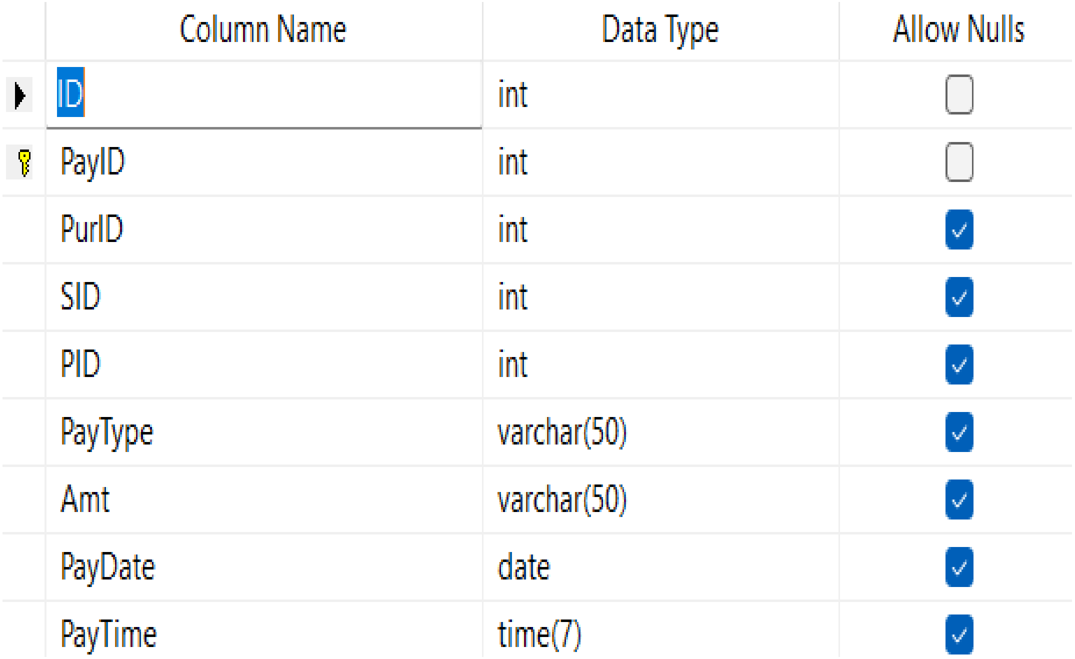
	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	SalID	int	□
	EID	int	☑
	CID	int	☑
	PayID	int	☑
	PayType	varchar(50)	☑
	ProID	int	☑
	ProCat	varchar(50)	☑
	ProName	varchar(50)	☑
	ProPrice	varchar(50)	☑
	ProQuantity	varchar(50)	☑
	Units	varchar(50)	☑
	DisPer	varchar(50)	☑
	DisAmt	varchar(50)	☑
	TPrice	varchar(50)	☑
	RCash	varchar(50)	☑
	Change	varchar(50)	☑
	SalDate	date	☑
	SalTime	time(7)	☑

9.5 COMPANY TABLE:



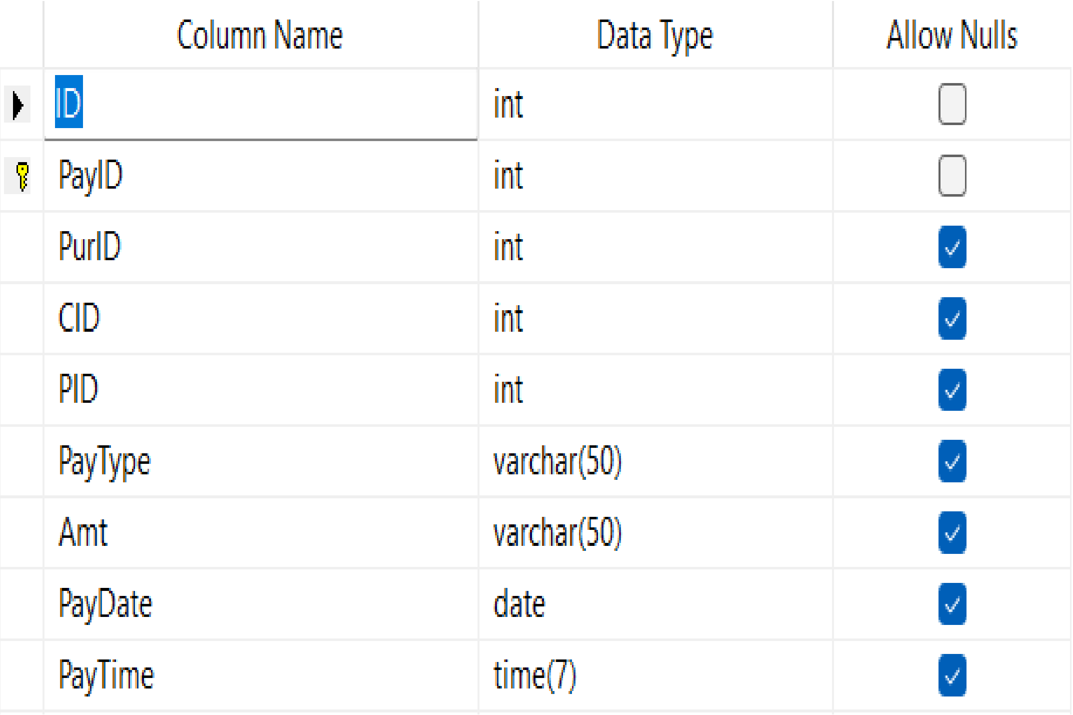
	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	CID	int	□
	CName	varchar(50)	☑
	CAddress	varchar(50)	☑
	CPhoneNo	varchar(50)	☑
	CEmail	varchar(50)	☑

9.6 PAYMENT TO SUPPLIER TABLE:



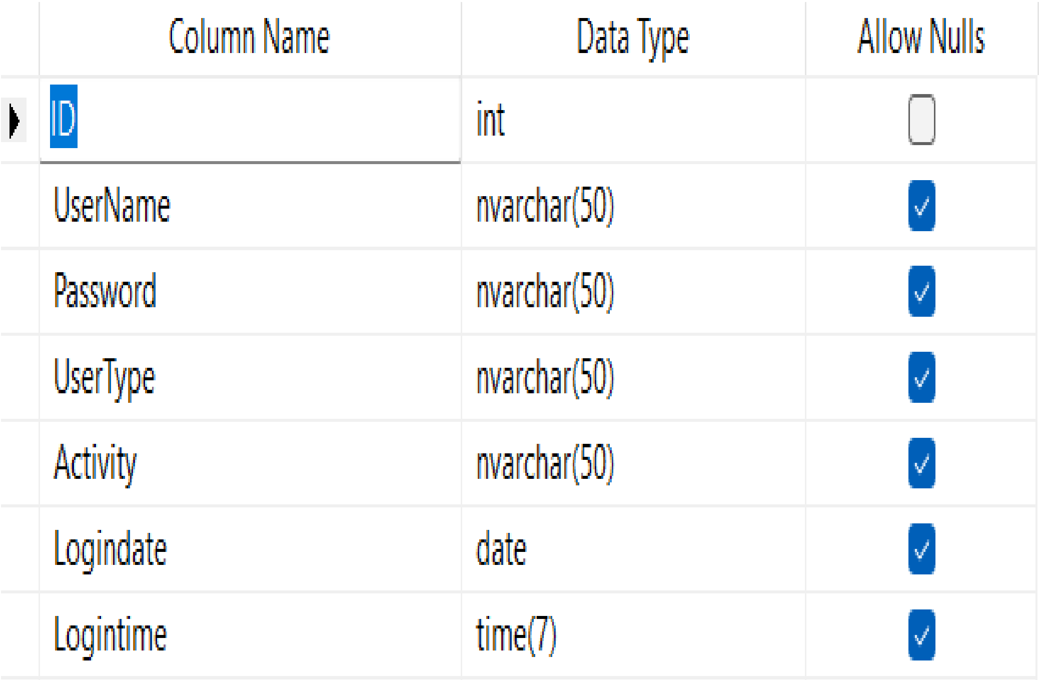
	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	PayID	int	□
	PurID	int	☑
	SID	int	☑
	PID	int	☑
	PayType	varchar(50)	☑
	Amt	varchar(50)	☑
	PayDate	date	☑
	PayTime	time(7)	☑

9.7 PAYMENT TO COMPANY



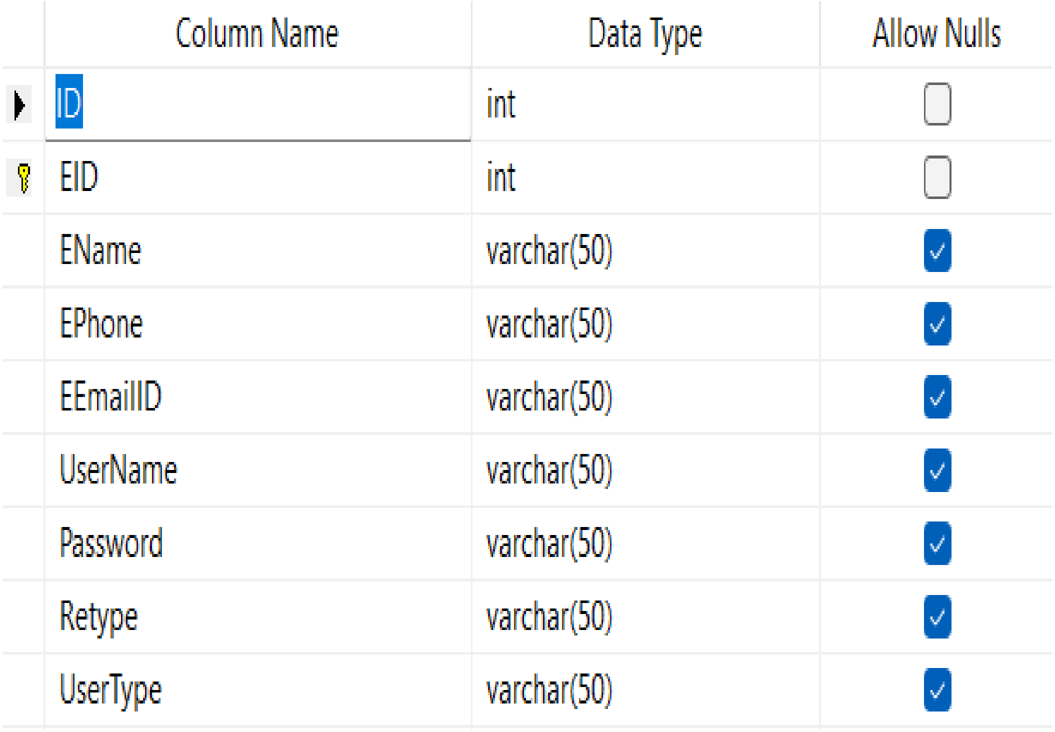
	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	PayID	int	□
	PurID	int	☑
	CID	int	☑
	PID	int	☑
	PayType	varchar(50)	☑
	Amt	varchar(50)	☑
	PayDate	date	☑
	PayTime	time(7)	☑

9.8 LOG MANAGER



	Column Name	Data Type	Allow Nulls
▶	ID	int	□
	UserName	nvarchar(50)	☑
	Password	nvarchar(50)	☑
	UserType	nvarchar(50)	☑
	Activity	nvarchar(50)	☑
	Logindate	date	☑
	Logintime	time(7)	☑

9.9 EMPLOYEE REGISTRATION



	Column Name	Data Type	Allow Nulls
▶	ID	int	□
🔑	EID	int	□
	EName	varchar(50)	☑
	EPhone	varchar(50)	☑
	EEmailID	varchar(50)	☑
	UserName	varchar(50)	☑
	Password	varchar(50)	☑
	Retype	varchar(50)	☑
	UserType	varchar(50)	☑

10. APPENDICES-B SOURCE CODE:

LOGIN FORM:

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

Public Class LoginForm
    Public LoginUser As String
    Public LoginPass As String
    Public UserType As String
    Public q As String
    Dim currentTime As DateTime = DateTime.Now
    Dim timeString As String = currentTime.ToString("hh:mm:ss tt")
    Private Sub btnLogin_Click(sender As Object, e As EventArgs) Handles btnLogin.Click
        If (IsFormValid()) Then
            qry = "Select *from Employee where UserName ='" & TextBox1.Text & "' and Password ='" &
TextBox2.Text & "' and UserType ='" & ComboBox1.Text & "'"
            ds = SearchData(qry)
            If (ds.Tables(0).Rows.Count > 0) Then
                LoginUser = TextBox1.Text
                LoginPass = TextBox2.Text
                UserType = ComboBox1.Text
                MadTitan()
                EditUser()
                DashboardForm.Show()
                Me.Close()
            Else
                MsgBox("Invalid Login", MsgBoxStyle.Critical)
            End If
        End If
    End Sub
    Private Sub MadTitan()
        Dim connectionString As String = "Data Source=THEDS482\SQLEXPRESS;Initial Catalog=SIS482;Integrated
Security=True"
        Dim connection As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial
Catalog=SIS482;Integrated Security=True")
        connection.Open()
        Dim query As String = "Alter DataBase Scoped Configuration Set Identity_Cache = off"
        Dim command As New SqlCommand(query, connection)
        connection.Close()
    End Sub
    Private Sub EditUser()
        If (IsFormValid()) Then
            q = "Insert into Audit values ('" & TextBox1.Text & "','" & TextBox2.Text & "','" & ComboBox1.Text &
"','IN','" & Date.Now & "','" & timeString & "')"
            Dim logincorrect As Boolean = Convert.ToBoolean(InsertData(q))
            If (logincorrect) Then
                MsgBox("Login Successful", MsgBoxStyle.Information)
            Else
                MsgBox("Login Failed", MsgBoxStyle.Critical)
            End If
        End If
    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(sender As Object, e As EventArgs) Handles
ComboBox1.SelectedIndexChanged
        btnLogin.Enabled = True
    End Sub

    Private Function IsFormValid() As Boolean
        If (TextBox1.Text.Trim() = String.Empty) Then
            MsgBox("User Name is Required", MsgBoxStyle.Critical)
            Clr()
            Return False
        ElseIf (TextBox2.Text.Trim() = String.Empty) Then
            MsgBox("Password is Required", MsgBoxStyle.Critical)
            Clr()
            Return False
        ElseIf (ComboBox1.SelectedIndex = -1) Then
            MsgBox("User Type is Required", MsgBoxStyle.Critical)
        End If
        Return True
    End Function
    Private Sub Clr()
        TextBox1.Clear()
        TextBox2.Clear()
    End Sub
    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles Button2.Click
        Application.Exit()
    End Sub

    Private Sub LoginForm_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        TextBox1.Text = "Coder"
        TextBox2.Text = "123"
        ComboBox1.Text = "Admin"
    End Sub
    End Class
```

STOCK:

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

    Public Class Phones
        Public executed As Boolean = False
        Private Sub btnAdd_Click(sender As Object, e As EventArgs) Handles btnAdd.Click
            If (IsFormValid()) Then
                qry = "Insert into Phones values ('" & TextBox6.Text & "','" & ComboBox1.Text & "','" & TextBox3.Text
    & "','" & TextBox5.Text & "')"
                Dim logincorrect As Boolean = Convert.ToBoolean(InsertData(qry))
                If (logincorrect) Then
                    MsgBox("Phones Added Sucessfully", MsgBoxStyle.Information)
                    Clr()
                Else
                    MsgBox("Record Not Saved", MsgBoxStyle.Critical)
                End If
            End If
        End Sub
        Private Function IsFormValid() As Boolean
            If (TextBox6.Text.Substring(0, 2) > 100) Then
                MsgBox("ProID is Required", MsgBoxStyle.Critical)
                Clr()
                Return False
            ElseIf (ComboBox1.SelectedIndex = -1) Then
                MsgBox("Product Catetory is Required", MsgBoxStyle.Critical)
                Clr()
                Return False
            ElseIf (TextBox3.Text.Trim() = String.Empty) Then
                MsgBox("Product Name is Required", MsgBoxStyle.Critical)
                Clr()
                Return False
            ElseIf (TextBox5.Text.Trim() = String.Empty) Then
                MsgBox("Product Quantity is Required", MsgBoxStyle.Critical)
                Clr()
                Return False
            End If
            Return True
        End Function
        Public Sub Clr()
            TextBox6.Text = "10"
            ComboBox1.Text = "Phones"
            TextBox3.Clear()
            TextBox5.Clear()
        End Sub
        Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
            If (IsFormValid()) Then
                qry = "Update Phones set ProID ='" & TextBox6.Text & "',ProCat ='" & ComboBox1.Text & "' ,ProName
    ='" & TextBox3.Text & "' ,ProQuantity ='" & TextBox5.Text & "'where ProID ='" &
    Convert.ToInt32(TextBox6.Text) & "'"
                Dim isupdatetrue As Boolean = Convert.ToBoolean(InsertData(qry))
                If (isupdatetrue) Then
                    MsgBox("Phones Updated Sucessfully", MsgBoxStyle.Information)
                    btnAdd.Enabled = True
                    Clr()
                Else
                    MsgBox("Record Not Updated", MsgBoxStyle.Critical)
                End If
            End If
        End Sub
        Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
            Dim result As Integer = MsgBox("Do you really want to delete the record. ?", MsgBoxStyle.YesNo +
    MsgBoxStyle.Question)
            If result = DialogResult.Yes Then
                If (IsFormValid2()) Then
                    qry = "Delete from Phones where ProID ='" & Convert.ToInt32(TextBox6.Text) & "'"
                    Dim wantToDelete As Boolean = Convert.ToBoolean(InsertData(qry))
                    If (wantToDelete) Then
                        MsgBox("Phones Deleted Sucessfully", MsgBoxStyle.Information)
                        btnAdd.Enabled = True
                        'MasterShifu()
                        Clr()
                    Else
                        MsgBox("Record Not Deleted", MsgBoxStyle.Critical)
                    End If
                End If
            End If
        End Sub
        Private Sub MasterShifu()
            If (IsFormValid2()) Then
                qry = "Delete from Buy where ProID ='" & Convert.ToInt32(TextBox6.Text) & "'"
                Dim wantToDelete As Boolean = Convert.ToBoolean(InsertData(qry))
                qry = "Delete from Supplier2 where PID ='" & Convert.ToInt32(TextBox6.Text) & "'"
                Dim BellaCiao As Boolean = Convert.ToBoolean(InsertData(qry))
                qry = "Delete from Sales where ProID ='" & Convert.ToInt32(TextBox6.Text) & "'"
                Dim orange7 As Boolean = Convert.ToBoolean(InsertData(qry))
                qry = "Delete from Company2 where PID ='" & Convert.ToInt32(TextBox6.Text) & "'"
                Dim kong As Boolean = Convert.ToInt32(InsertData(qry))
            End If
        End Sub
        Private Function IsFormValid2() As Boolean
            If (TextBox6.Text.Trim() = String.Empty) Then
                MsgBox("Product ID is Required", MsgBoxStyle.Critical)
                Return False
            End If
            Return True
        End Function

        Private Sub btnSearch_Click(sender As Object, e As EventArgs) Handles btnSearch.Click
            Dim con As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial
    Catalog=SIS482;Integrated Security=True")
            con.Open()
            Dim cmd As New SqlCommand("Select count(*) from Phones where ProID = @ProID", con)
            cmd.Parameters.AddWithValue("@ProID", TextBox1.Text)
            Dim count As Integer = Convert.ToInt32(cmd.ExecuteScalar())
            If count > 0 Then
                Dim ConnectionString As String = "Data Source=THEDS482\SQLEXPRESS;Initial
    Catalog=SIS482;Integrated Security=True"
                Dim sql As String = "Select ProID,ProCat,ProName,ProQuantity from Phones where ProID ='" &
    Convert.ToInt32(TextBox1.Text) & "'"
                Using connection As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial
    Catalog=SIS482;Integrated Security=True")
                    connection.Open()
                    Using command As New SqlCommand(sql, connection)
                        Using adapter As New SqlDataAdapter(command)
                            Dim dt As New DataTable("Phones")
                            adapter.Fill(dt)
                            DataGridView1.DataSource = dt
                            DataGridView1.Refresh()
                        End Using
                    End Using
                End Using
            Else
                MsgBox("ID Not Found", MsgBoxStyle.Information)
            End If
            con.Close()
        End Sub
    End Class
```

PURCHASE:

```
Imports System.Data.SqlClient
Imports System.Security.Cryptography.X509Certificates
Imports System.Windows.Forms.VisualStyles.VisualStyleElement
Imports System.Windows.Forms.VisualStyles.VisualStyleElement.Status
Public Class buy
    Public executed As Boolean = False
    Dim currentTime As DateTime = DateTime.Now
    Dim timeString As String = currentTime.ToString("hh:mm:ss tt")
    Private Sub btnadd_click(sender As Object, e As EventArgs) Handles btnAdd.Click
        If (isformvalid()) Then
            qry = "Insert into Buy values ('" & TextBox11.Text & "','" & TextBox1.Text & "','" & TextBox2.Text & "','" & TextBox5.Text & "','" & ComboBox2.Text & "','" & TextBox3.Text & "','" & ComboBox1.Text & "','" & TextBox4.Text & "','" & TextBox6.Text & "','" & TextBox7.Text & "','" & TextBox10.Text & "','" & TextBox8.Text & "','" & Date.Now & "','" & timeString & "')"
            Dim logincorrect As Boolean = Convert.ToBoolean(InsertData(qry))
            If (logincorrect) Then
                MsgBox("Purchase Added Sucessfully", MsgBoxStyle.Information)
                Exlir()
                WakandaForever()
                clr()
            Else
                MsgBox("Record not Saved", MsgBoxStyle.Critical)
            End If
        End If
    End Sub
    Private Sub WakandaForever()
        If (isformvalid()) Then
            qry = "Insert into Supplier2 values ('" & TextBox5.Text & "','" & TextBox11.Text & "','" & TextBox2.Text & "','" & TextBox3.Text & "','" & ComboBox2.Text & "','" & TextBox8.Text & "','" & Date.Now & "','" & timeString & "')"
            Dim logincorrect As Boolean = Convert.ToBoolean(InsertData(qry))
        End If
        Me.Close()
        Dim f As New Supplier2()
        f.TopLevel = False
        f.Visible = True
        My.Forms.DashboardForm.Panel3.Controls.Add(f)
        MsgBox("Payment Sucessfully Added", MsgBoxStyle.Information)
    End Sub
    Private Function Exlir() As Boolean
        Dim r1 As String = TextBox3.Text
        If r1.Length >= 2 Then
            Dim firstTwoDigits As String = r1.Substring(0, 2)
            If Integer.TryParse(10, Nothing) Then
                Dim number As Integer = Integer.Parse(firstTwoDigits)
                If number = 10 Then
                    Dim con As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial
Catalog=SIS482;Integrated Security=True")
                    con.Open()
                    Dim cmd As New SqlCommand("Select count(*) from Phones where ProID = @ProID", con)
                    cmd.Parameters.AddWithValue("@ProID", TextBox3.Text)
                    Dim count As Integer = Convert.ToInt32(cmd.ExecuteScalar())
                    If count > 0 Then
                        Hulk()
                        Return True
                    Else
                        qry = "Insert into Phones values ('" & TextBox3.Text & "','" & ComboBox1.Text & "','" &
TextBox4.Text & "','" & TextBox10.Text & "')"
                        Dim logincorrect As Boolean = Convert.ToBoolean(InsertData(qry))
                    End If
                    con.Close()
                ElseIf number = 20 Then
                    Dim con As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial
Catalog=SIS482;Integrated Security=True")
                    con.Open()
                    Dim cmd As New SqlCommand("Select count(*) from Laptops where ProID = @ProID", con)
                    cmd.Parameters.AddWithValue("@ProID", TextBox3.Text)
                    Dim count As Integer = Convert.ToInt32(cmd.ExecuteScalar())
                    If count > 0 Then
                        Hulk()
                        Return True
                    Else
                        qry = "Insert into Laptops values ('" & TextBox3.Text & "','" & ComboBox1.Text & "','" &
TextBox4.Text & "','" & TextBox10.Text & "')"
                        Dim logincorrect As Boolean = Convert.ToBoolean(InsertData(qry))
                    End If
                    con.Close()
                Else
                    MsgBox("Invalid Product ID", MsgBoxStyle.Information)
                    Return False
                End If
            End If
        Else
            MsgBox("Product ID is Required", MsgBoxStyle.Critical)
        End If
        Return True
        Return True
    End Function
    Private Sub Hulk()
        Dim r1 As String = TextBox3.Text
        If r1.Length >= 2 Then
            Dim firstTwoDigits As String = r1.Substring(0, 2)
            If Integer.TryParse(20, Nothing) Then
                Dim number As Integer = Integer.Parse(firstTwoDigits)
                If number = 10 Then
                    Dim a As String
                    a = Integer.Parse(TextBox7.Text) + TextBox10.Text
                    qry = "Update Phones set ProQuantity ='" & a & "'where ProID ='" &
Convert.ToInt32(TextBox3.Text) & "'"
                    Dim isupdatetrue As Boolean = Convert.ToBoolean(InsertData(qry))
                ElseIf number = 20 Then
                    Dim a As String
                    a = Integer.Parse(TextBox7.Text) + TextBox10.Text
                    qry = "Update Laptops set ProQuantity ='" & a & "'where ProID ='" &
Convert.ToInt32(TextBox3.Text) & "'"
                    Dim isupdatetrue As Boolean = Convert.ToBoolean(InsertData(qry))
                Else
                    MsgBox("Invalid Product ID", MsgBoxStyle.Information)
                End If
            End If
        Else
            MsgBox("Product ID is Required", MsgBoxStyle.Critical)
        End If
    End Sub
End Class
```

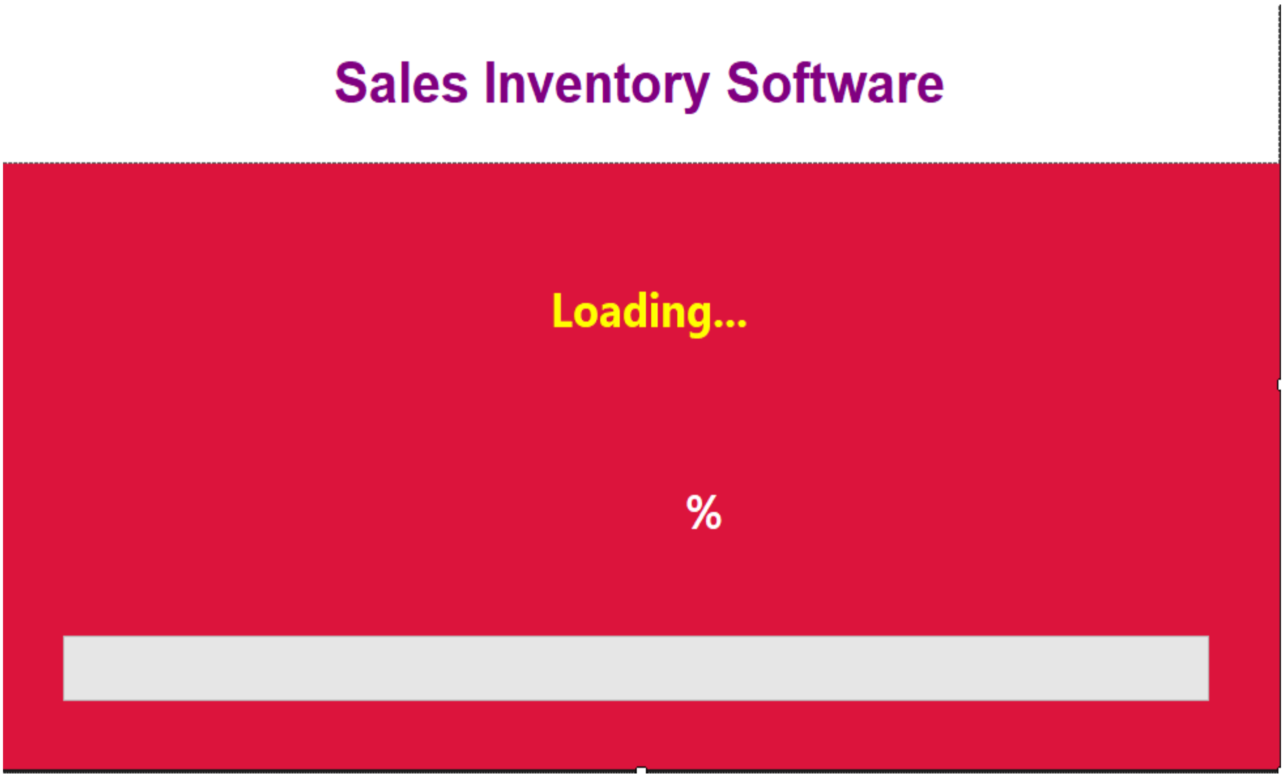
SALES:

```
Imports System.CodeDom.Compiler
Imports System.Data.SqlClient
Imports System.Runtime.CompilerServices
Imports System.Security.Cryptography.X509Certificates
Imports System.Windows.Forms.VisualStyles.VisualStyleElement
Public Class SellItems
    Public executed As Boolean = False
    Public executed1 As Boolean = False
    Dim currentTime As DateTime = DateTime.Now
    Dim timeString As String = currentTime.ToString("hh:mm:ss tt")
    Private Sub TextBox2_TextChanged(sender As Object, e As EventArgs)
        If TextBox2.Text.Length > 3 Then
            TextBox2.Text = TextBox2.Text.Substring(0, 3)
            TextBox2.SelectionStart = 3
            MsgBox("Only 3 Digits Allowed", MsgBoxStyle.Information)
        End If
    End Sub
    Private Function Thanos2() As Boolean
        Dim ak As String
        ak = TextBox2.Text
        Dim con As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial Catalog=SIS482;Integrated
Security=True")
        con.Open()
        Dim cmd As New SqlCommand("Select count(*) from Phones where ProID = @ProID", con)
        cmd.Parameters.AddWithValue("@ProID", TextBox2.Text)
        Dim count As Integer = Convert.ToInt32(cmd.ExecuteScalar())
        If count > 0 Then
        Else
            If executed = False Then
                MsgBox("Product ID Not Found", MsgBoxStyle.Information)
                Dim Result As DialogResult = MsgBox("Do you want to Purchase this Item ?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question)
                If Result = DialogResult.Yes Then
                    executed = True
                    executed1 = True
                    Me.Close()
                    Dim f As New buy()
                    f.TopLevel = False
                    f.Visible = True
                    My.Forms.DashboardForm.Panel3.Controls.Add(f)
                    MsgBox("Please Enter the Purchase Details", MsgBoxStyle.Information)
                    f.TextBox3.Text = ak
                Else
                    MsgBox("Invalid Product ID", MsgBoxStyle.Critical)
                    TextBox2.Clear()
                End If
            End If
            If executed1 = False Then
                Dim Result1 As DialogResult = MsgBox("Do you want to Add this Item to Phones?",
MsgBoxStyle.YesNo + MsgBoxStyle.Question)
```

```
            If Result1 = DialogResult.Yes Then
                executed1 = True
                Me.Close()
                Dim f As New Phones()
                f.TopLevel = False
                f.Visible = True
                My.Forms.DashboardForm.Panel3.Controls.Add(f)
                MsgBox("Please Enter the Phones Details", MsgBoxStyle.Information)
                f.TextBox6.Text = ak
            Else
                MsgBox("Invalid Product ID", MsgBoxStyle.Critical)
                TextBox2.Clear()
            End If
        End If
    End If
    Return True
    con.Close()
End Function
Private Function Thanos3() As Boolean
    Dim ak As String
    ak = TextBox2.Text
    Dim con As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial Catalog=SIS482;Integrated
Security=True")
    con.Open()
    Dim cmd As New SqlCommand("Select count(*) from Laptops where ProID = @ProID", con)
    cmd.Parameters.AddWithValue("@ProID", TextBox2.Text)
    Dim count As Integer = Convert.ToInt32(cmd.ExecuteScalar())
    If count > 0 Then
    Else
        If executed = False Then
            MsgBox("Product ID Not Found", MsgBoxStyle.Information)
            Dim result As DialogResult = MsgBox("Do you want to Purchase this Item ?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question)
            If result = DialogResult.Yes Then
                executed = True
                executed1 = True
                Me.Close()
                Dim f As New buy()
                f.TopLevel = False
                f.Visible = True
                My.Forms.DashboardForm.Panel3.Controls.Add(f)
                MsgBox("Please Enter the Purchase Details", MsgBoxStyle.Information)
                f.TextBox3.Text = ak
            Else
                MsgBox("Invalid Product ID", MsgBoxStyle.Critical)
                TextBox2.Clear()
            End If
            executed = True
        End If
        If executed1 = True Then
            Dim Result1 As DialogResult = MsgBox("Do you want to Add this Item to Laptops?",
MsgBoxStyle.YesNo + MsgBoxStyle.Question)
```

```
                If Result1 = DialogResult.Yes Then
                    executed1 = True
                    Me.Close()
                    Dim f As New Laptop()
                    f.TopLevel = False
                    f.Visible = True
                    My.Forms.DashboardForm.Panel3.Controls.Add(f)
                    MsgBox("Please Enter the Laptops Details", MsgBoxStyle.Information)
                    executed1 = True
                    f.TextBox6.Text = ak
                Else
                    MsgBox("Invalid Product ID", MsgBoxStyle.Critical)
                    TextBox2.Clear()
                End If
            End If
        End If
        Return True
        con.Close()
    End Function
    Private Sub TextBox1_KeyPress(sender As Object, e As KeyPressEventArgs) Handles TextBox1.KeyPress
        If Not Char.IsDigit(e.KeyChar) And Not Char.IsControl(e.KeyChar) Then
            e.Handled = True
        End If
    End Sub
    Private Sub TextBox1_TextChanged(sender As Object, e As EventArgs) Handles TextBox1.TextChanged
        If TextBox1.Text.Length > 3 Then
            TextBox1.Text = TextBox1.Text.Substring(0, 3)
            TextBox1.SelectionStart = 3
            MsgBox("Only 3 Digits Allowed", MsgBoxStyle.Information)
        End If
    End Sub
    Private Sub TextBox1_Leave(sender As Object, e As EventArgs) Handles TextBox1.Leave
        Sparky()
    End Sub
    Private Function Sparky() As Boolean
        Dim ak As String
        ak = TextBox1.Text
        Dim r1 As String = TextBox1.Text
        If r1.Length >= 2 Then
            Dim firstTwoDigits As String = r1.Substring(0, 2)
            If Integer.TryParse(40, Nothing) Then
                Dim number As Integer = Integer.Parse(firstTwoDigits)
                If number = 40 Then
                    Dim con As New SqlConnection("Data Source=THEDS482\SQLEXPRESS;Initial
 Catalog=SIS482;Integrated Security=True")
                    con.Open()
                    Dim cmd As New SqlCommand("Select count(*) from Company where CID = @CID", con)
                    cmd.Parameters.AddWithValue("@CID", TextBox1.Text)
                    Dim count As Integer = Convert.ToInt32(cmd.ExecuteScalar())
                    If count > 0 Then
                    Else
                        If executed = False Then
                    MsgBox("Company ID Not Found", MsgBoxStyle.Information)
                    Dim Result As DialogResult = MsgBox("Do you want to add this Item ?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question)
                    If Result = DialogResult.Yes Then
                        executed = True
                        Me.Close()
                        Dim f As New Company()
                        f.TopLevel = False
                        f.Visible = True
                        My.Forms.DashboardForm.Panel3.Controls.Add(f)
                        MsgBox("Please Enter the Company Details", MsgBoxStyle.Information)
                        f.TextBox1.Text = ak
                    Else
                        MsgBox("Invalid Comapny ID", MsgBoxStyle.Critical)
                        TextBox1.Text = "40"
                    End If
                End If
                Return False
            End If
            con.Close()
        Else
            MsgBox("Invalid Company ID", MsgBoxStyle.Information)
            TextBox1.Text = "40"
            Return False
        End If
    End If
End Class
```

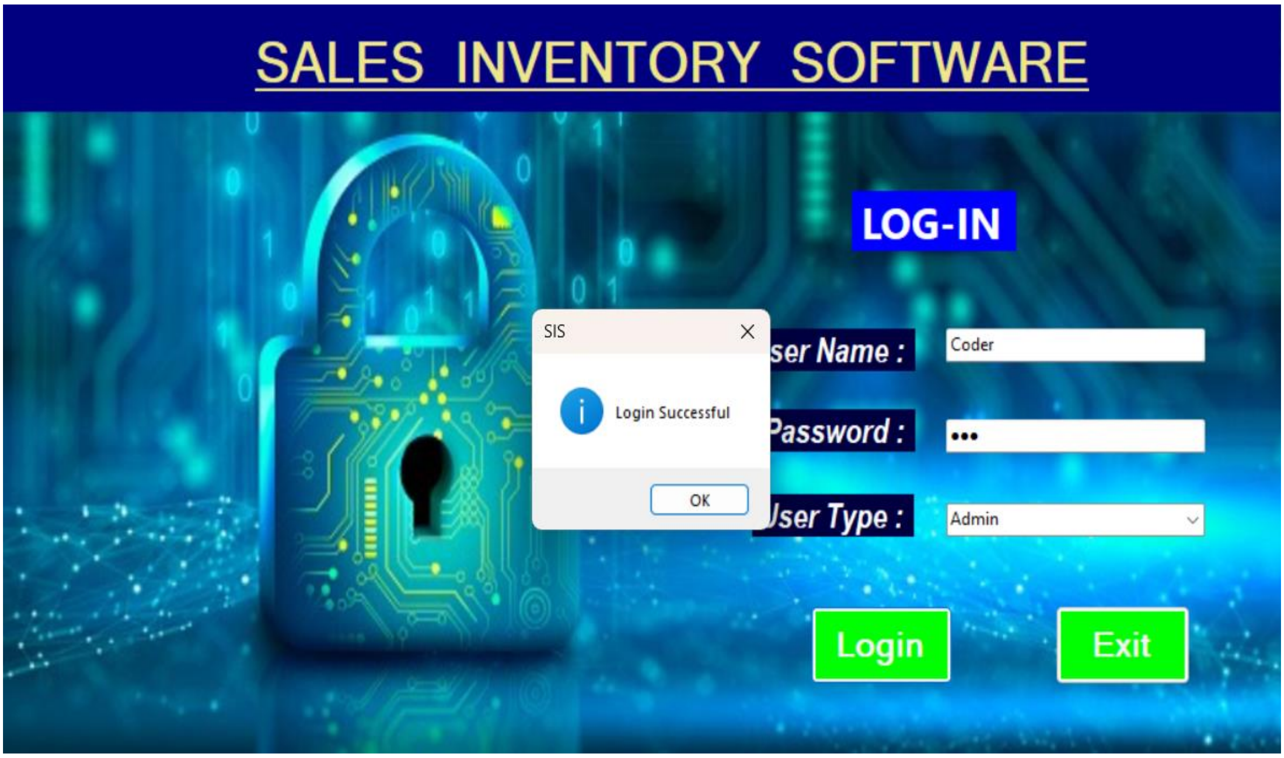
11. APPENDICES-C SCREENSHOTS:



Sales Inventory Software

Loading...

%



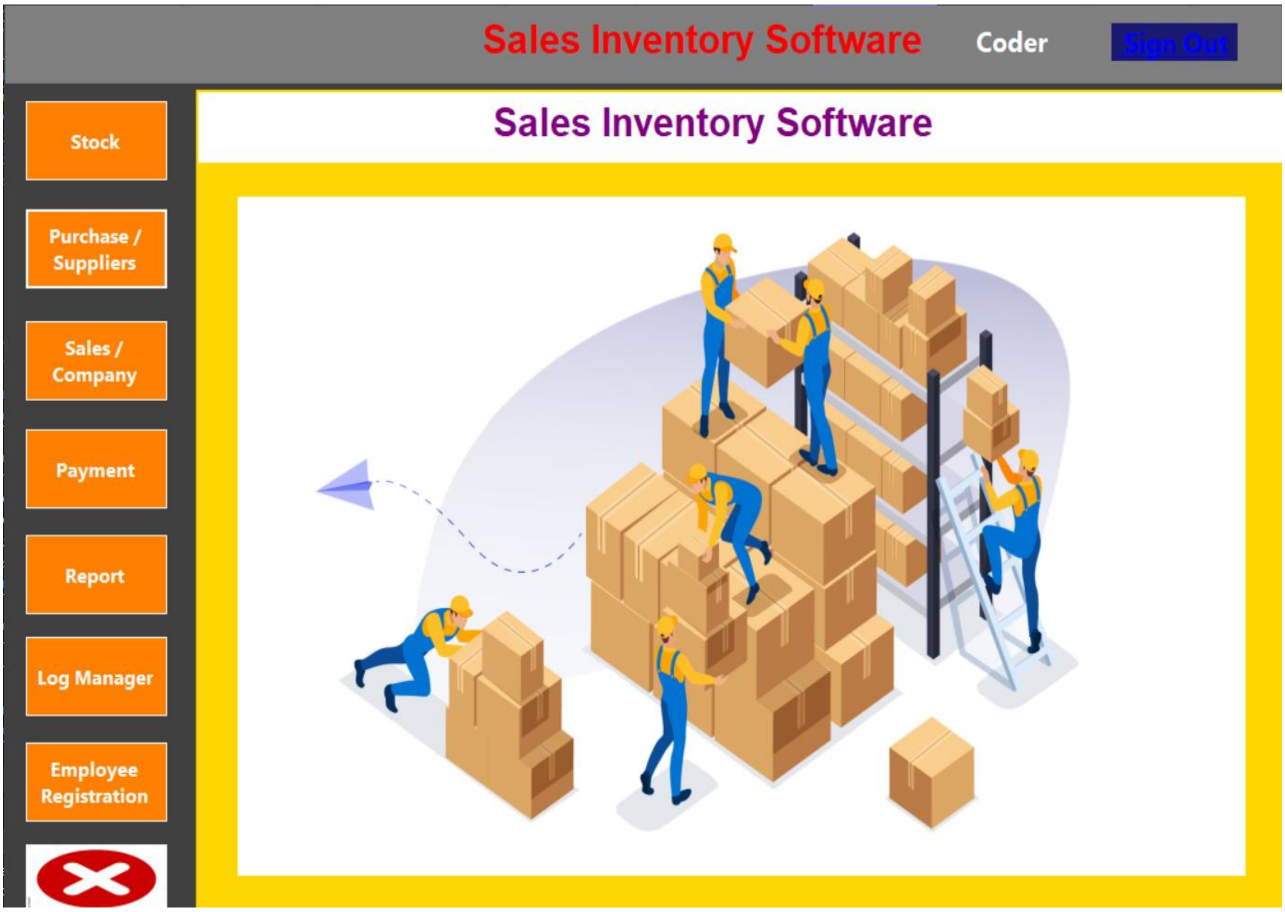
SIS

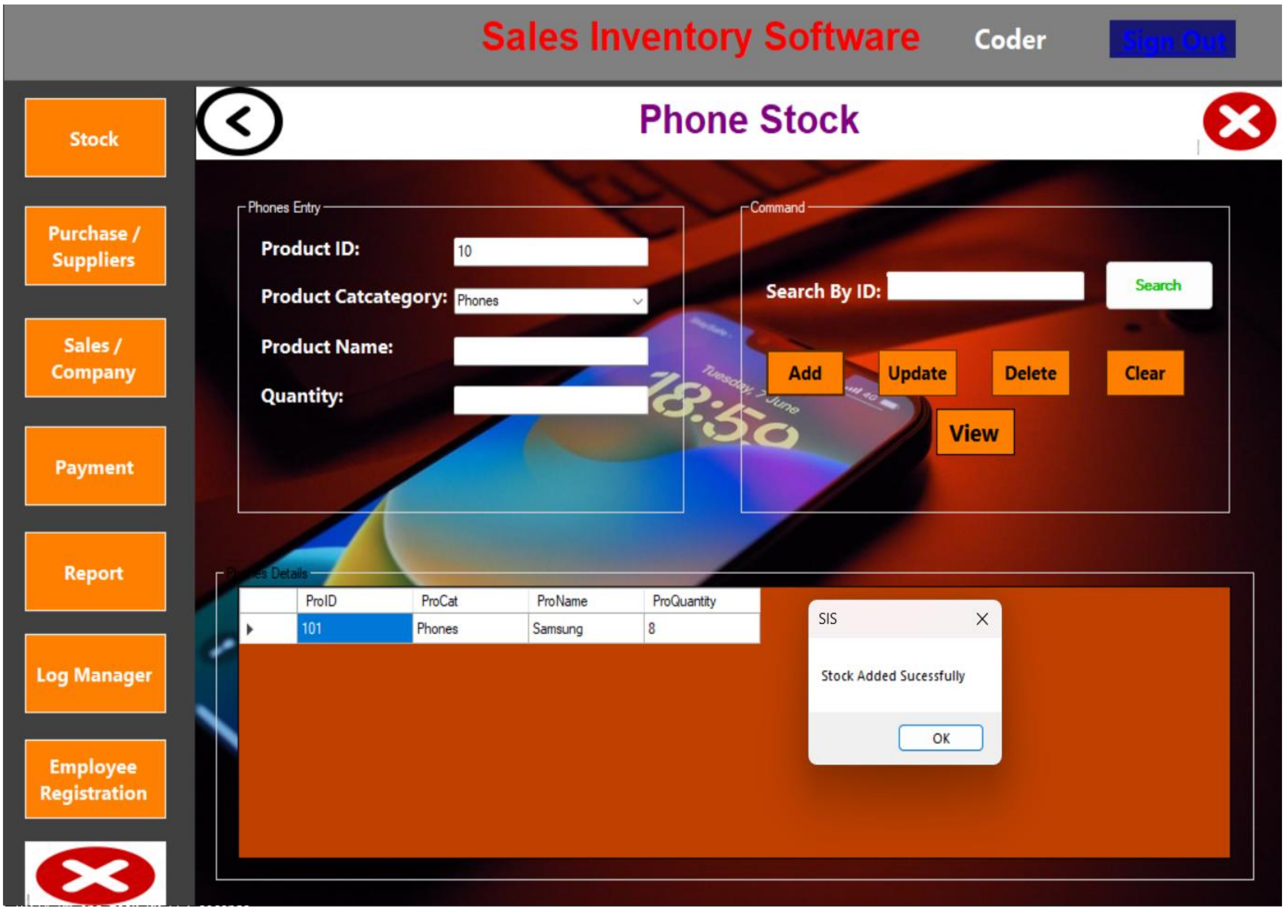
Login Successful

ser Name : Coder

Password : •••

User Type : Admin





Product ID: 10

Product Catcategory: Phones

Product Name:

Quantity:

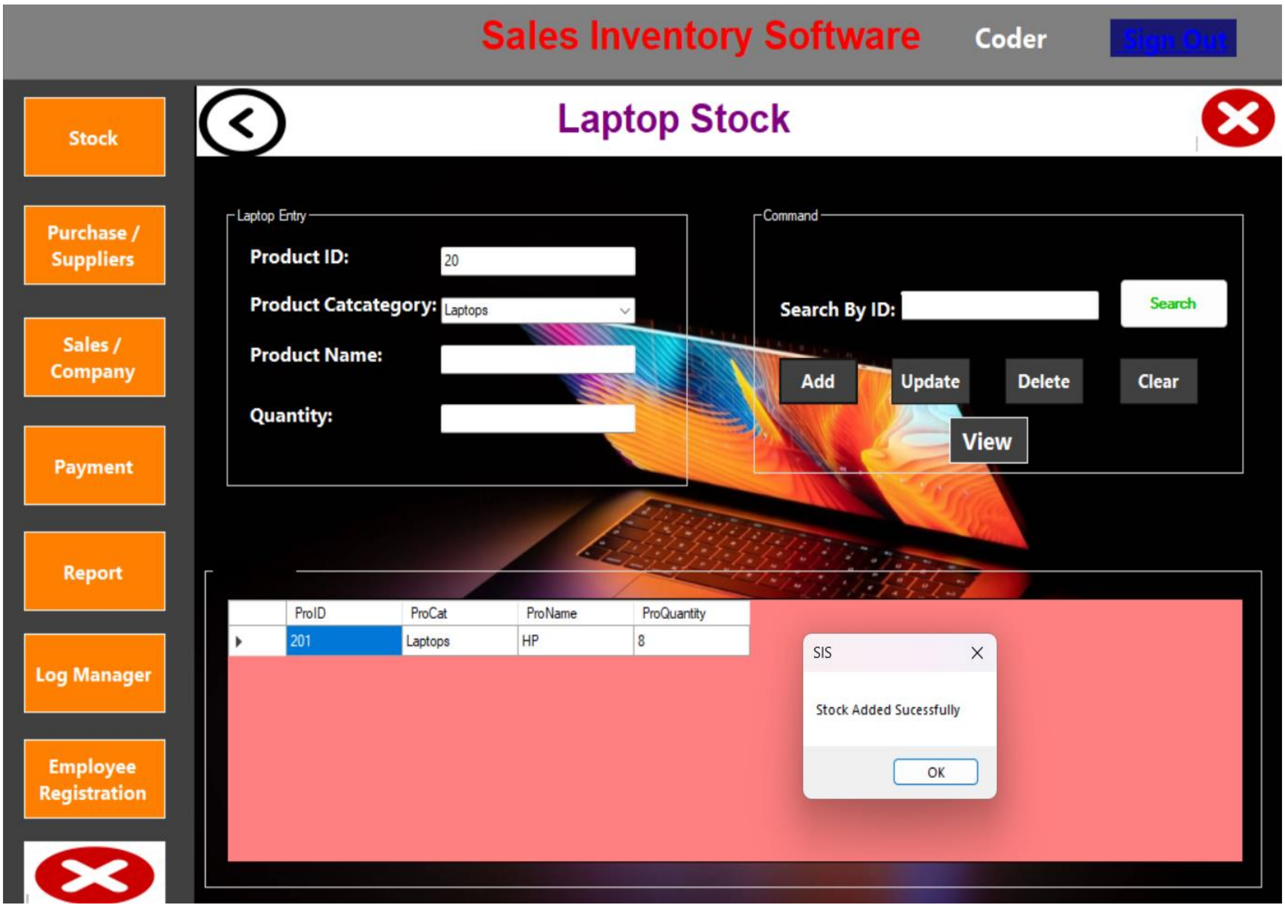
Search By ID:

	ProID	ProCat	ProName	ProQuantity
▸	101	Phones	Samsung	8

SIS

Stock Added Sucessfully

OK



Product ID: 20

Product Catcategory: Laptops

Product Name:

Quantity:

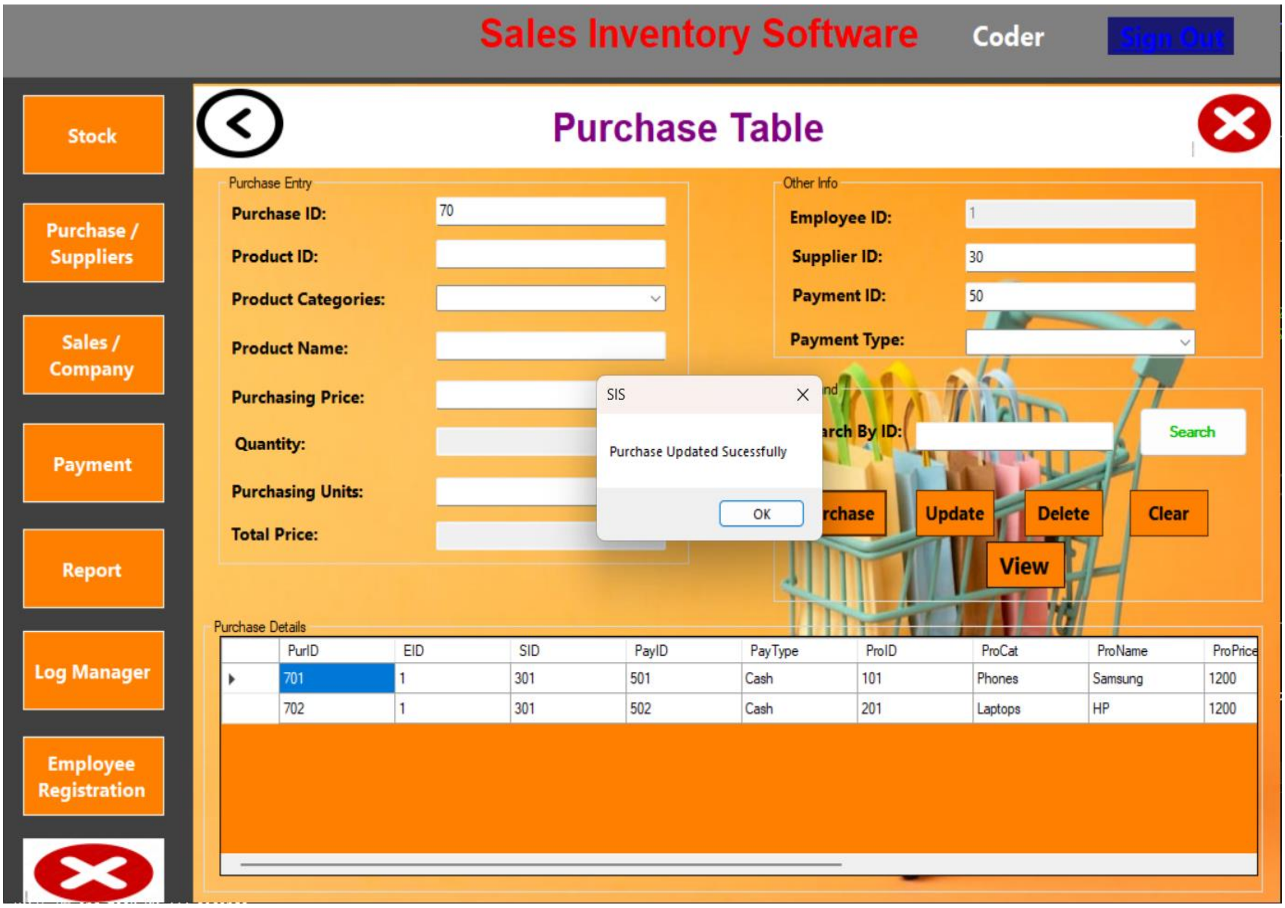
Search By ID:

	ProID	ProCat	ProName	ProQuantity
▸	201	Laptops	HP	8

SIS

Stock Added Sucessfully

OK



Purchase ID: 70

Product ID:

Product Categories:

Product Name:

Purchasing Price:

Quantity:

Purchasing Units:

Total Price:

Employee ID: 1

Supplier ID: 30

Payment ID: 50

Payment Type:

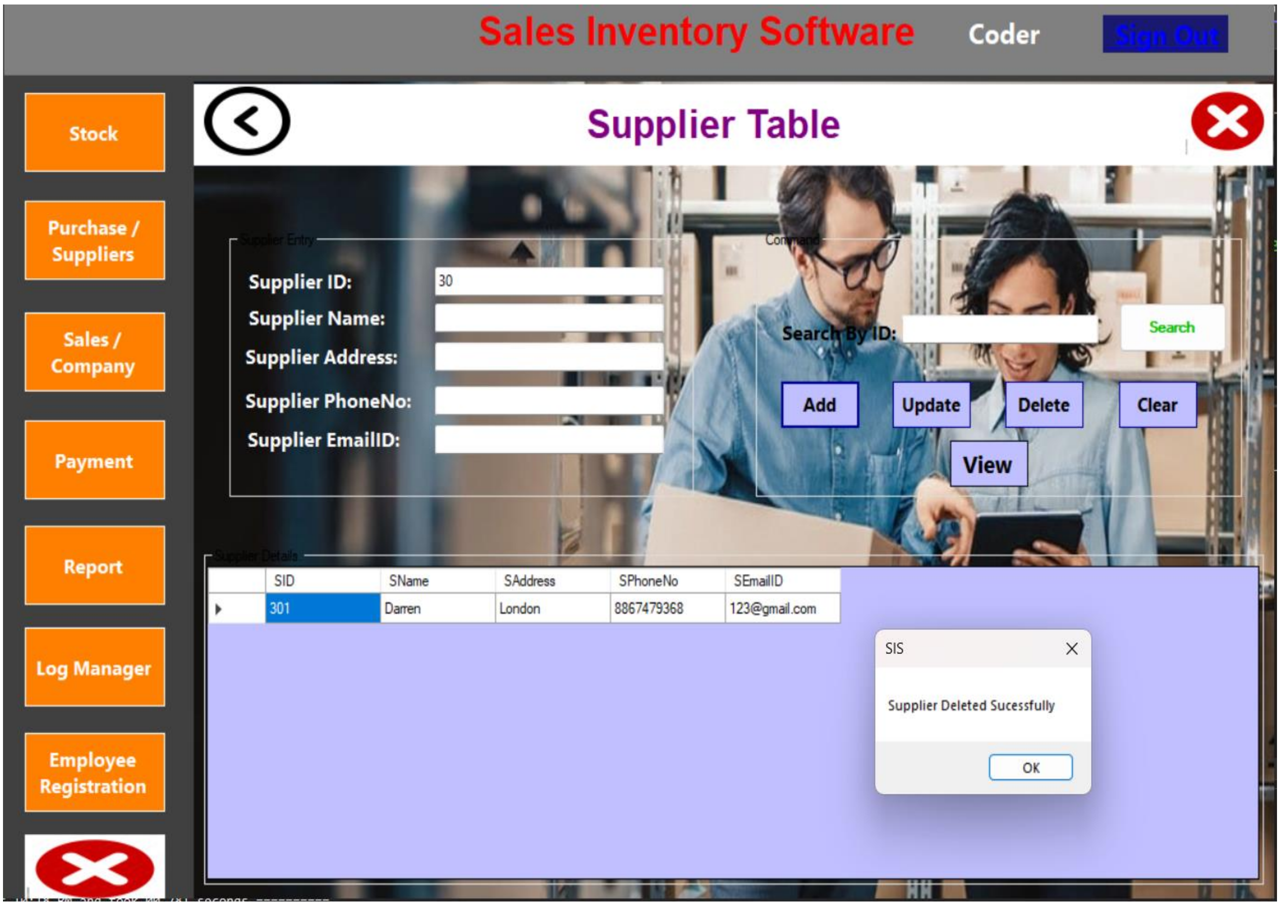
arch By ID:

SIS

Purchase Updated Sucessfully

OK

	PurID	EID	SID	PayID	PayType	ProID	ProCat	ProName	ProPrice
▸	701	1	301	501	Cash	101	Phones	Samsung	1200
	702	1	301	502	Cash	201	Laptops	HP	1200



Supplier ID: 30

Supplier Name:

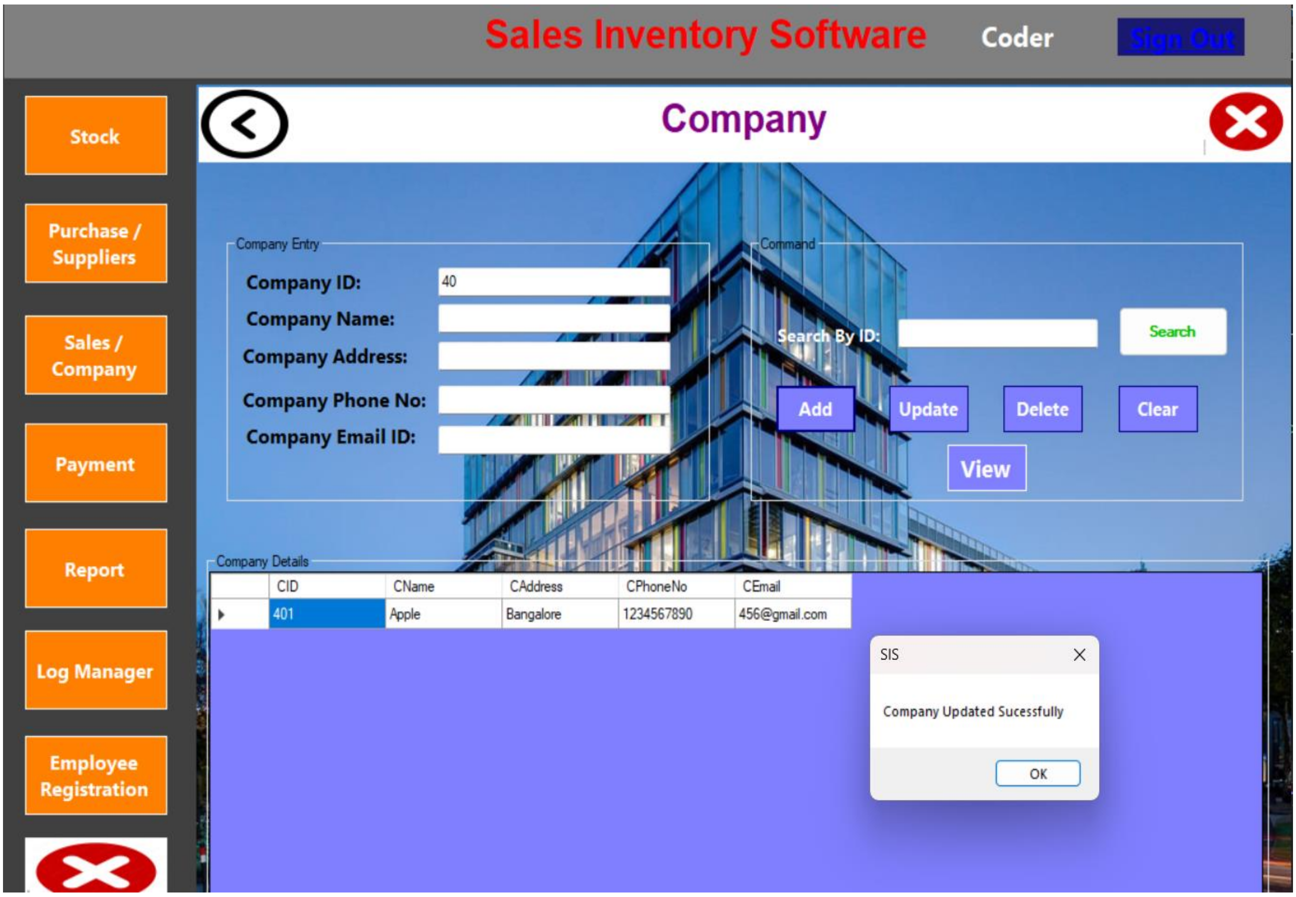
Supplier Address:

Supplier PhoneNo:

Supplier EmailID:

	SID	SName	SAddress	SPhoneNo	SEmailID
▸	301	Darren	London	8867479368	123@gmail.com

Supplier Deleted Sucessfully



Company ID: 40

Company Name:

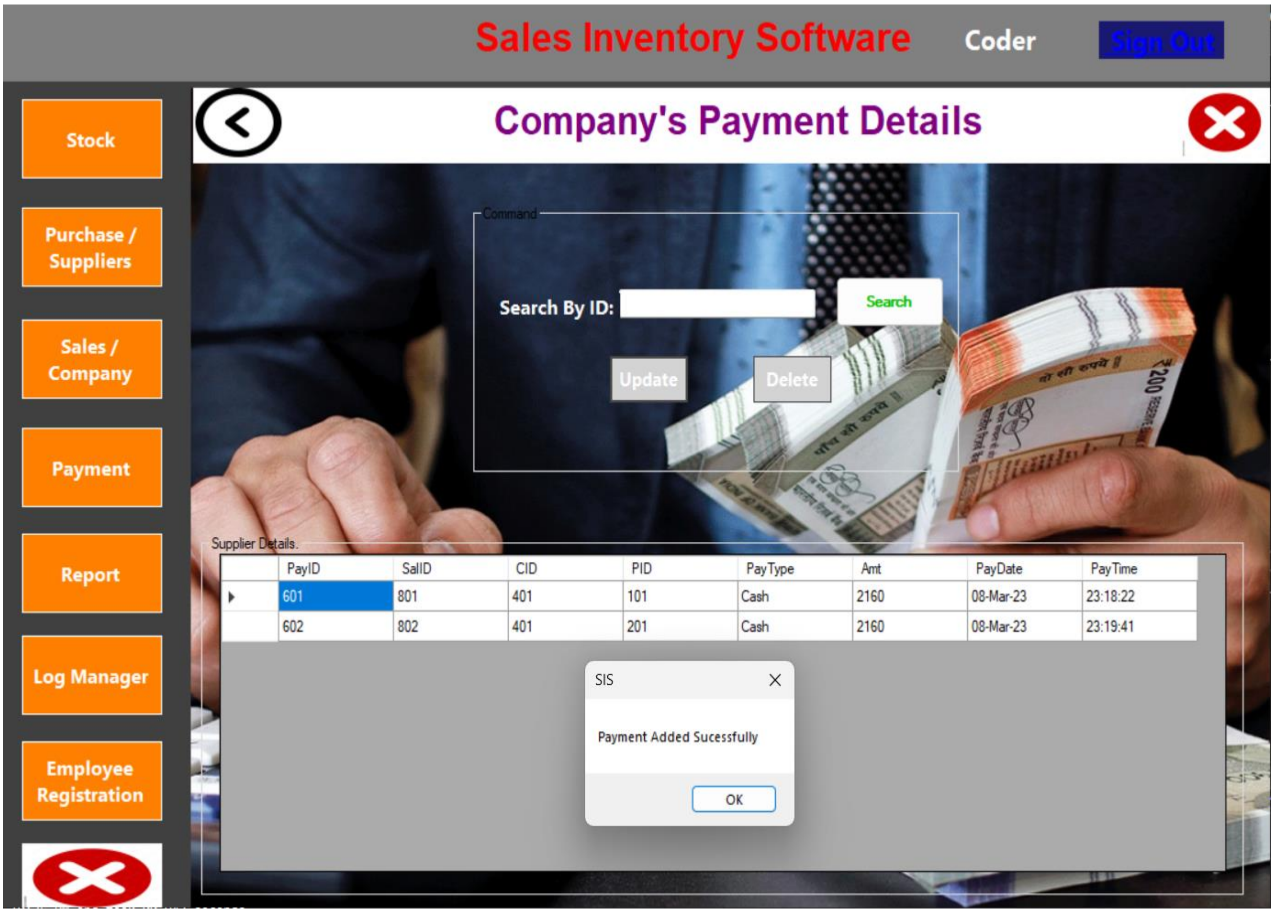
Company Address:

Company Phone No:

Company Email ID:

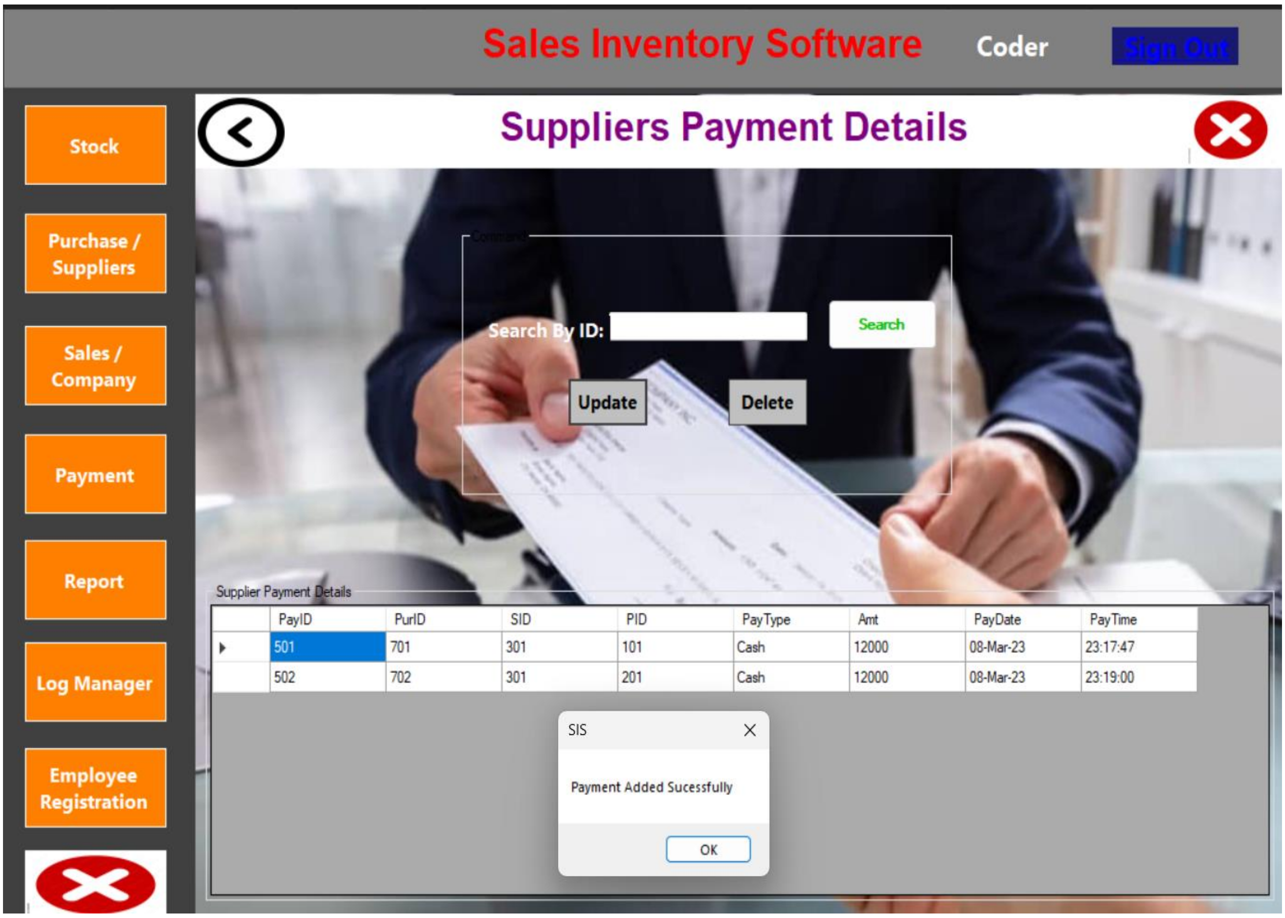
	CID	CName	CAddress	CPhoneNo	CEmail
▸	401	Apple	Bangalore	1234567890	456@gmail.com

Company Updated Sucessfully



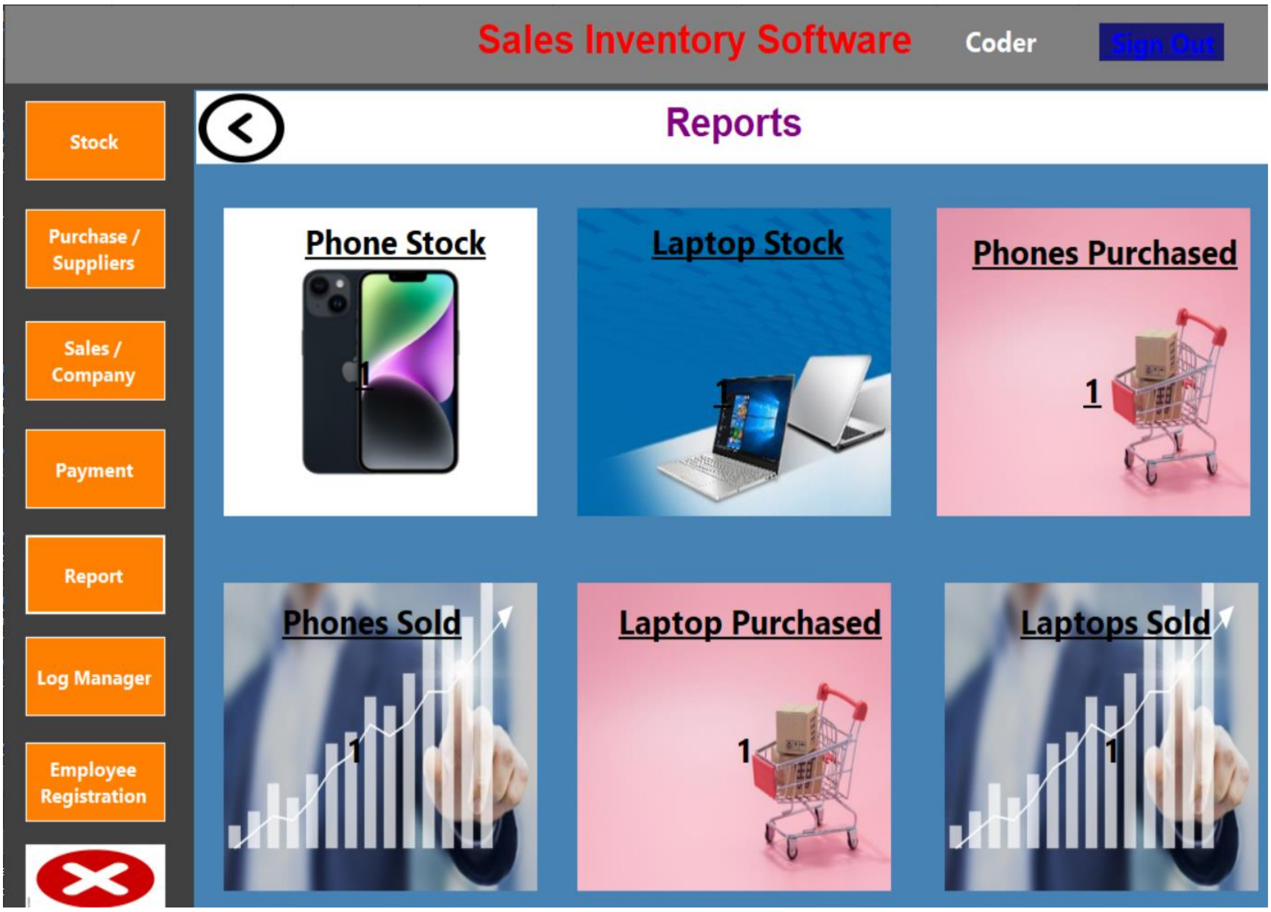
PayID	SalID	CID	PID	PayType	Amt	PayDate	PayTime
601	801	401	101	Cash	2160	08-Mar-23	23:18:22
602	802	401	201	Cash	2160	08-Mar-23	23:19:41

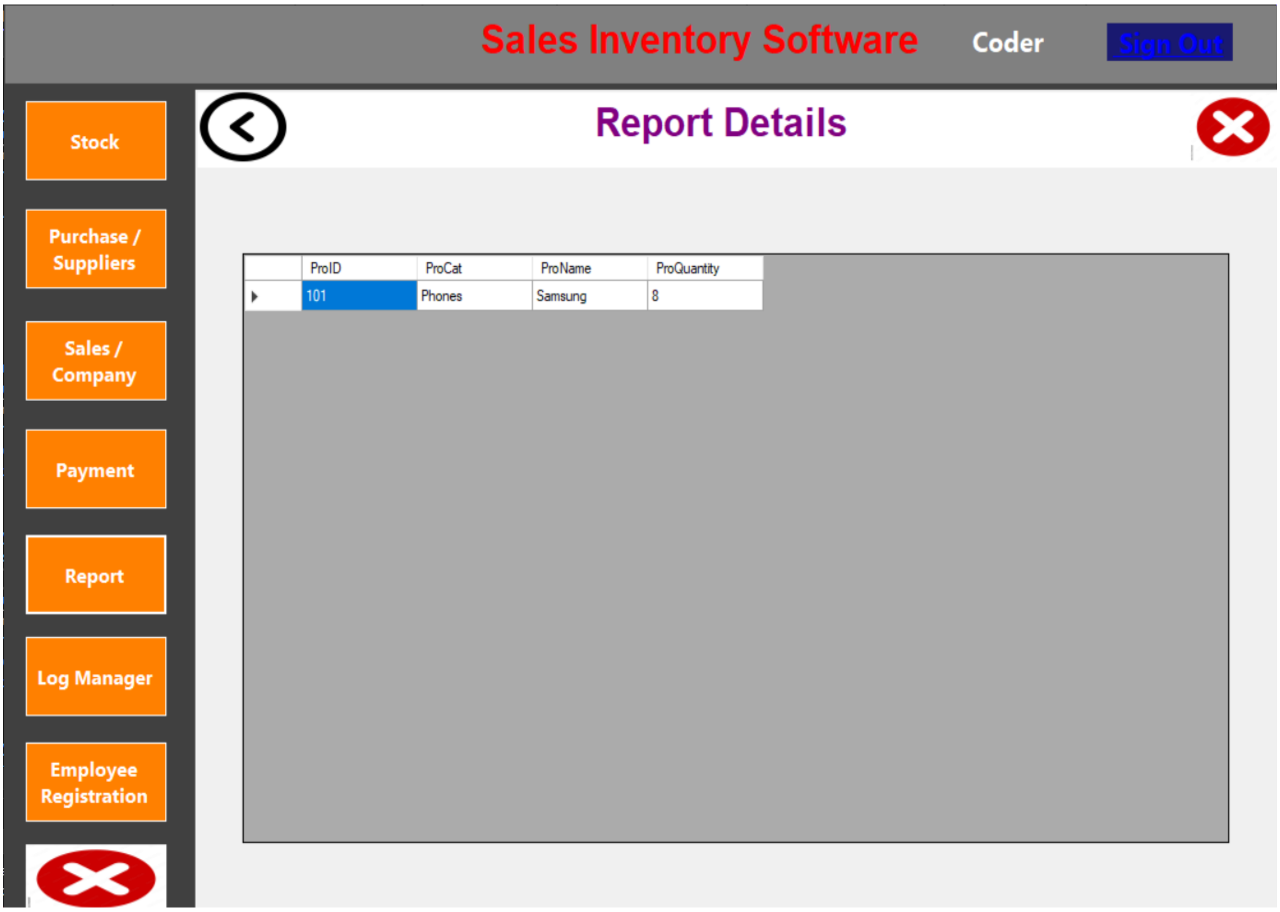
Payment Added Sucessfully



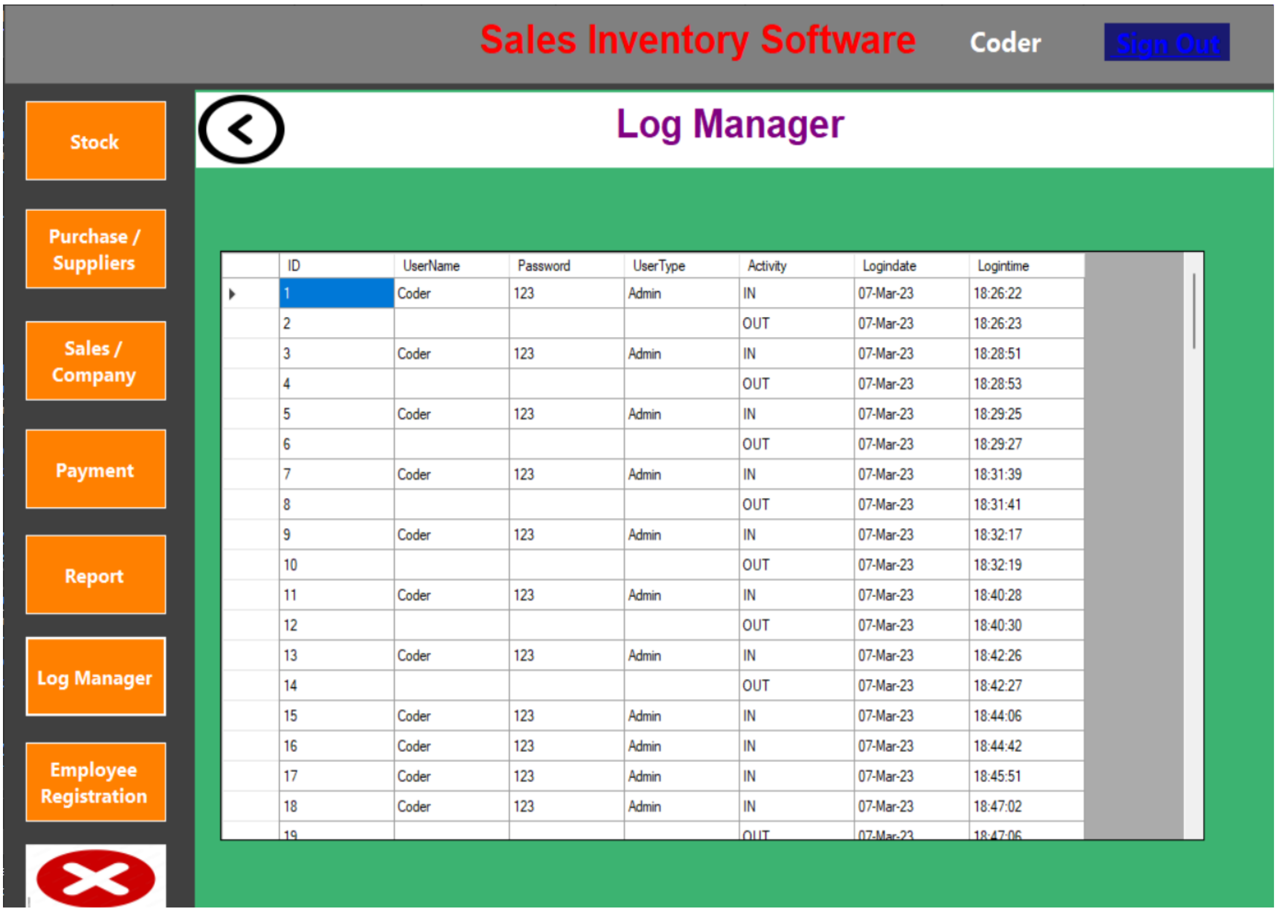
PayID	PurID	SID	PID	PayType	Amt	PayDate	PayTime
501	701	301	101	Cash	12000	08-Mar-23	23:17:47
502	702	301	201	Cash	12000	08-Mar-23	23:19:00

Payment Added Sucessfully



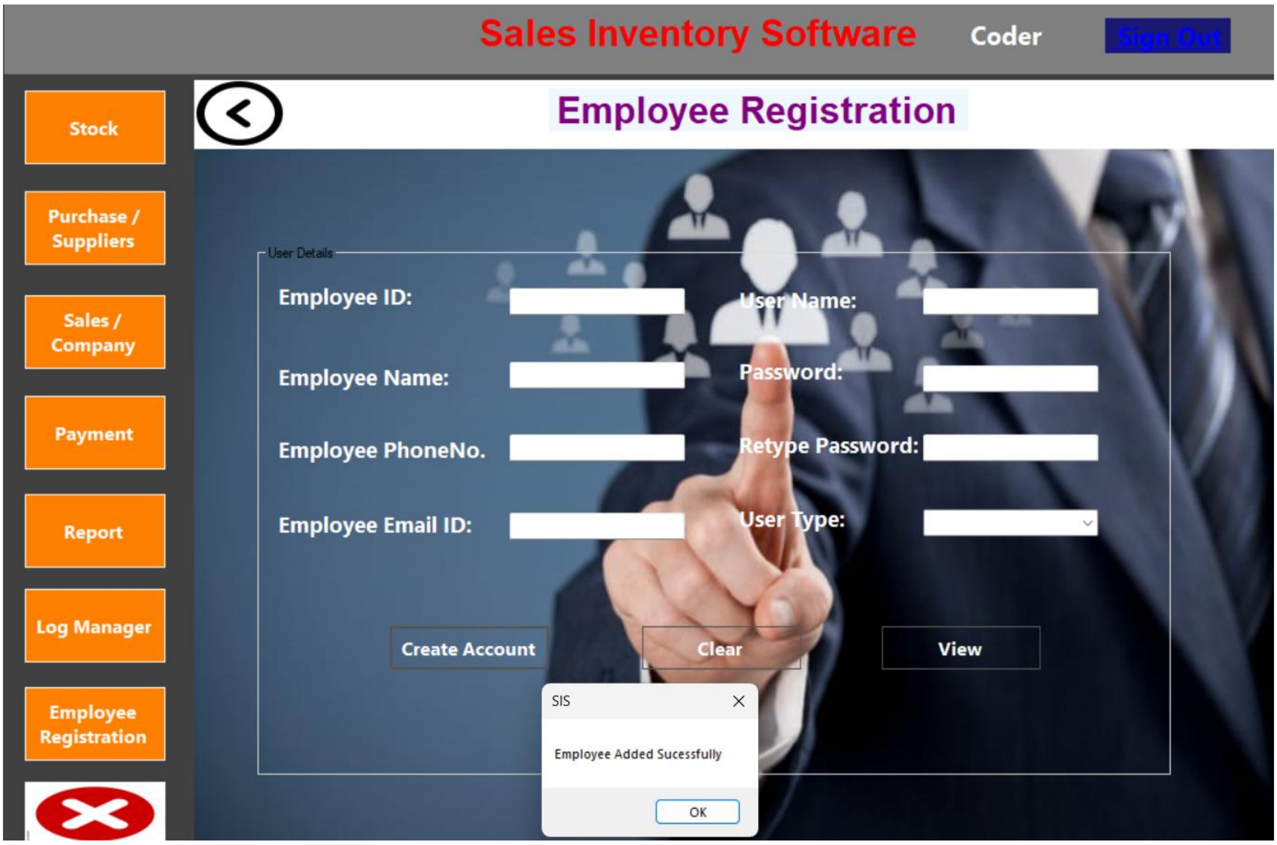


	ProID	ProCat	ProName	ProQuantity
▸	101	Phones	Samsung	8



Log Manager

ID	UserName	Password	UserType	Activity	Logindate	Logintime
1	Coder	123	Admin	IN	07-Mar-23	18:26:22
2				OUT	07-Mar-23	18:26:23
3	Coder	123	Admin	IN	07-Mar-23	18:28:51
4				OUT	07-Mar-23	18:28:53
5	Coder	123	Admin	IN	07-Mar-23	18:29:25
6				OUT	07-Mar-23	18:29:27
7	Coder	123	Admin	IN	07-Mar-23	18:31:39
8				OUT	07-Mar-23	18:31:41
9	Coder	123	Admin	IN	07-Mar-23	18:32:17
10				OUT	07-Mar-23	18:32:19
11	Coder	123	Admin	IN	07-Mar-23	18:40:28
12				OUT	07-Mar-23	18:40:30
13	Coder	123	Admin	IN	07-Mar-23	18:42:26
14				OUT	07-Mar-23	18:42:27
15	Coder	123	Admin	IN	07-Mar-23	18:44:06
16	Coder	123	Admin	IN	07-Mar-23	18:44:42
17	Coder	123	Admin	IN	07-Mar-23	18:45:51
18	Coder	123	Admin	IN	07-Mar-23	18:47:02
19				OUT	07-Mar-23	18:47:06



Employee Registration

User Details

Employee ID:

Employee Name:

Employee PhoneNo.

Employee Email ID:

User Name:

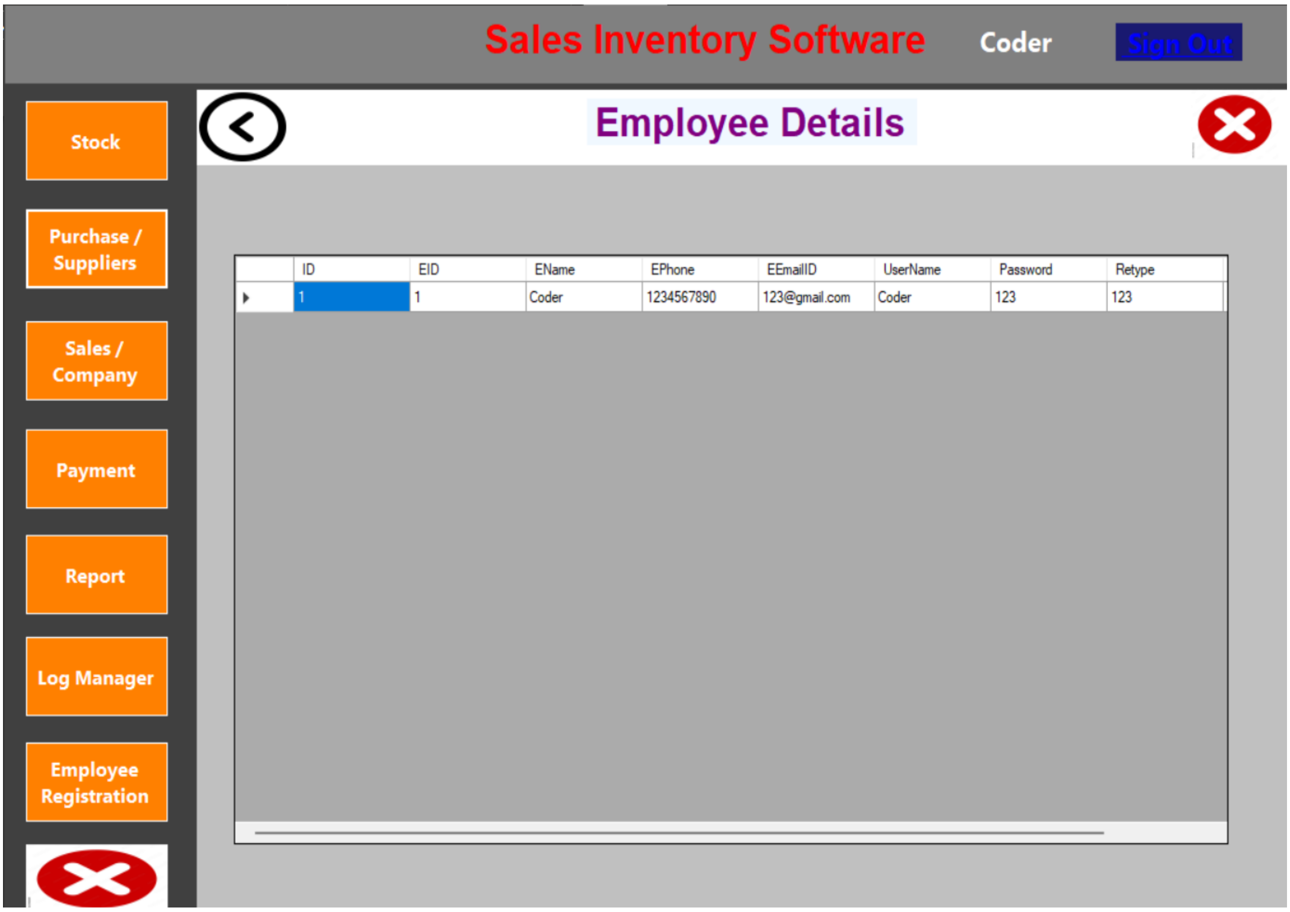
Password:

Retype Password:

User Type:

SIS

Employee Added Sucessfully



ID	EID	EName	EPhone	EEmailID	UserName	Password	Retype
1	1	Coder	1234567890	123@gmail.com	Coder	123	123

12. APPENDICES D-SAMPLE REPORT OF TEST CASES:

SL No.	Test Case Id	Scenario	Steps to Execute	Test Data	Expected Result	Actual Result	Status
1	Test Case Id 1	User Login	Start-enter the username, password, User Type login	Id: Invalid Pass: Invalid Type: Invalid	Error	Error	Pass
2	Test Case Id 2	User Login	Start-enter the username, password, User Type login	Id: Valid Pass: Invalid Type: Valid	Error	Error	Pass
3	Test Case Id 3	User Login	Start-enter the username, password, User Type login	Id: Invalid Pass: Valid Type: Invalid	Error	Error	Pass
4	Test Case Id 4	User Login	Start-enter the username, password, User Type login	Id: Valid Pass: Valid Type: Valid	Login Successful	Login Successful	Pass
5	Test Case Id 5	Pressing Search button in Stock form	Start-login, Enter the Stock ID and click search	Enter a valid Stock ID	Displays Search result	Displays Search result	Pass
6	Test Case Id 6	Pressing Search button in Stock form	Start-login, Enter the Stock ID and click search	Enter an invalid Stock ID	No results found	No results found	Pass

7	Test Case Id 7	Pressing Register button in Employee Registration form	Start-Login, Enter all details and click register	Enter all of the details	Employee registration successful	Employee registration successful	Pass
8	Test Case Id 8	Pressing Register button in Employee Registration form	Start-Login, Enter all details and click register	Enter some of the details	Enter all the Details	Enter all the Details	Pass
9	Test Case Id 9	Pressing Update button in Stock form	Start-Login Click the row you want to update	Click the stock row and enter the details what you want to update	Stock updated successfully	Stock updated Sucessfully	Pass
10	Test Case Id 10	Pressing Add button in Purchase form	Start-Login Enter all the details while purchasing	Enter all the details and click add	Purchase successful	Purchase Successful	Pass
11	Test Case Id 11	Pressing Add button in Purchase Form	Start-Login Enter all the details while purchasing	Enter some of the details and click add	Purchase Unsuccessfu l	Purchase Unsuccessful	Pass
12	Test Case Id 12	Pressing Add button in Stock Form	Start-Login Enter all the details of the product available	Enter all the details and click add	Stock added Successful	Stock added Successful	Pass
13	Test Case Id 13	Pressing Add button in Stock Form	Start-Login Enter all the details of the product available	Enter some of the details and click add	Stock Not added	Stock Not added	Pass
14	Test Case Id 14	Pressing Delete button in Stock Form	Start-Login Enter the row you want to delete from stock	Select the product ID or row and click delete	Stock deleted Successful	Stock deleted Sucessful	Pass

15	Test Case Id 15	Pressing add button in Sales form	Start-Login Enter all the details while selling	Enter all the details and click ad	Sold successful	Sold successful	Pass
16	Test Case Id 16	Pressing add button in Sales form	Start-Login Enter all the details while selling	Enter some the details and click add	Sold unsuccessful	Sold unsuccessful	Pass
17	Test Case Id 17	Pressing Add button in Supplier form	Start-Login Enter all the details of the Supplier	Enter all the details and click add	Supplier not added	Supplier not added	Pass
18	Test Case Id 18	Pressing Add button in Company form	Start-Login Enter all the details of the Supplier	Enter some the details and click add	Supplier not added	Supplier not added	Pass
19	Test Case Id 19	Pressing Add button in Company form	Start-Login Enter all the details of the Company	Enter all the details and click add	Company added successful	Company added successful	Pass
20	Test Case Id 20	Pressing Add button in Company form	Start-Login Enter all the details of the company	Enter some the details and click add	Company not added	Company not added	Pass