INVENTORY MANAGEMENT SYSTEM FOR DEPARTMENTAL STORE

CONTENTS OF SYNOPSIS

S. No.	Heading	Page No.
1.	Introduction	5
2.	Objectives	6
3.	Category of Project	7
4.	Tools Requirement Specification	9
5.	Data Flow Diagram	11
6. .	Entity Relationship Diagram (Erd)	13
7.	Context Level DFD	14
8.	Data Modeling	19
9.	Modules	21
10.	Project Planning & Scheduling	24
11.	Security and Validation Checks	25
12.	Scope of Future Application	28
13.	Limitation of the Project	29
14.	Bibliography	29

TITLE OF THE PROJECT

INVENTORY MANAGEMENT SYSTEM FOR DEPARTMENTAL STORE

(Dynamic Site using JSP and MY SQL)

1. INTRODUCTION

INVENTORY MANAGEMENT SYSTEM FOR DEPARTMENTAL STORE integrates internal and external management information across an entire organization, embracing finance/accounting, manufacturing, sales and service, customer relationship management, etc. ERP systems automate this activity with an integrated software application. Its purpose is to facilitate the flow of information between all business functions inside the boundaries of the organization and manage the connections to outside stakeholders.

This project is related to sales and inventory. Inventory takes turns under all types of economic systems. In a free market economy, manufacturing is usually directed toward the mass production of products for sale to consumers at a profit. In a collectivist economy, inventory system is more frequently directed by the state to supply a centrally planned economy. In free market economies, manufacturing occurs under some degree of government regulation.

An inventory control system may be used to automate a sales order fulfillment process. Such a system contains a list of order to be filled, and then prompts workers to pick the necessary items, and provides them with packaging and shipping information.

An inventory system also manages in and outwards material of hardware.

Real-time inventory control systems may use wireless, mobile terminals to record inventory transactions at the moment they occur.

IMSDS systems typically include the following characteristics:

- An integrated system that operates in real time (or next to real time), without relying on periodic updates.
- A common database, which supports all applications.
- A consistent look and feel throughout each module.

- Installation of the system without elaborate application/data integration by the Information Technology (IT) department.

2. OBJECTIVES

1. The web-site is to be accessed from any where anytime.

2. As the project is web-based so the software should be platform independent.

3. The data is very important asset for corporation so strong authentication method is to be used to ensure security of information from malicious user.

4. As the valuable data is being transferred through internet so the security of data is to be maintained at any cost.

5. Easy to be customized in future. As the client demand some other additional features. The complexity of customer's company may be different or if mode of business changes then the system has capability to make appropriate modification to suite that change. Customization is key factor of designing this software.

6. The web-site manipulates data in an accurate way.

7. Accuracy of all type of calculations are important and to be achieved at any cost.

8. This should always be observed during the development of the software that the user is not from technical background, so the software should be user friendly and can be handled easily.

9. Storage of data should be easily accessible.

10. The data retrieval and other manipulation related task which is done at the database level should be fast enough. The database should support multiple users at a time.

11. The product should fulfill user's requirement.

3. CATEGORY OF PROJECT

I. The User’s layer: tools and languages used □ JSP, HTML, and DHTML.

Editors: - NetBeans IDE

There is a front end which is web-pages which are displayed before the user and user directly interact with the software. Web pages are developed by using various design tools like Flash, Photo Shop etc and documentation languages like HTML and DHTML.

II. The Presentation layer: tools and languages used □ JSP, SERVLET

Editor: NetBeans IDE
At this layer all the services related with presentations like format check, mailing interface, parsing, deployment descriptor etc. this layer is developed in java using servlet, JSP.

III. The Integration Level: tool/technology used □JDBC Type 4

JDBC Type 4 Driver (Pure Java implementation) is used.

This tier is dedicated for database connectivity and connection management with the database. This is tier is developed in java and the components to be used JDBC 2.0.

IV. The Data Resources level: [MySQl]

At the back end, there is a data base which maintains data. This is to be developed in MySQL. In this server tables are created and all primary key –foreign key constraints are implemented. The data base follows all the integrity constraints.

Business Intelligence

Gain deeper insight into your business with integrated, comprehensive analysis and reporting for enhanced decision making.

High Availability

Ensure business continuity with the highest levels of system availability through technologies that protect your data against costly human errors and minimize disaster recovery downtime.

Performance and Scalability

Deliver an infrastructure that can grow with your business and has a proven record in handling today's large amounts of data and most critical enterprise workloads.

Security

Provide a secure environment to address privacy and compliance requirements with built-in features that protect your data against unauthorized access.

Manageability

Manage your infrastructure with automated diagnostics, tuning, and configuration to reduce operational costs while reducing maintenance and easily managing very large amounts of data.

Developer Productivity

Build and deploy critical business-ready applications more quickly by improving developer productivity and reducing project life cycle times.

4. TOOLS / PLATFORM, HARDWARE AND SOFTWARE REQUIREMENT SPECIFICATION

HARDWARE:

Processor	:	Pentium 2.4 GHz or above
Memory	:	256 MB RAM or above
Cache Memory	:	128 KB or above
Hard Disk	:	3 GB or above [at least 3 MB free space required]
Pen Drive	:	2 GB
Printer	:	Laser Printer

SOFTWARE:

Operating System	:	Windows 95/98, Win NT, Win 2000,window xp.
Font-End Tool	:	JSP, SERVLETS
Back End Tool	:	MY SQL

FRONT END:

We have implemented JavaScript for all the Client side validations. Client side JavaScript is designed to reside inside HTML document & ensure they run properly. It is object based, event driven, platform independent. These are important parts of any Web application to implement Client side Validations and the invalid data is not submitted. The form is not submitted until user fills in correct data. It is extremely useful to restrict mistakes by user.

BACK END:

We have used My Sql. My Sql provides efficient/effective solution for major database tech.

- Large database and space management.
- Many concurrent database users.
- High transaction processing requirement
- High Availability
- Industry accepted standards
- Manageable security
- Portability

5. DATA FLOW DIAGRAM

Data Flow Diagram

Data flow diagrams are the most commonly used way of documenting the processing of the candidate system. As their name suggest they are a pictorial way of representing the flow of data into, around, and out of the system. They are easily understandable and are less prone to misinterpretation than textual description. A complete set of DFDs provides a compact top - down representation of the system, which makes it easier for the user and the analyst to envisage the system as a whole.

DFDs are constructed using four major components:

• **External entities** - represents the sources of the data that enter the system or the recipients of the system that leave the system.

for example - passenger is the usual receiver of information and supplier of data during form filling.

• **Data stores -** represent the stores of the data within the system example: computer files, databases or in the manual system files, etc. data stores can not be linked directly by data flows either to each other or to external entities without an intervening process to transform them.

• **Processes** - represent activities in which data is manipulated by being stored or retrieved or transformed in some way. Process names are generally unambiguous and convey as much meaning as possible without being too long. Example: verify data, acquired time schedule etc.

Data flows - represents the movement of data between other components.

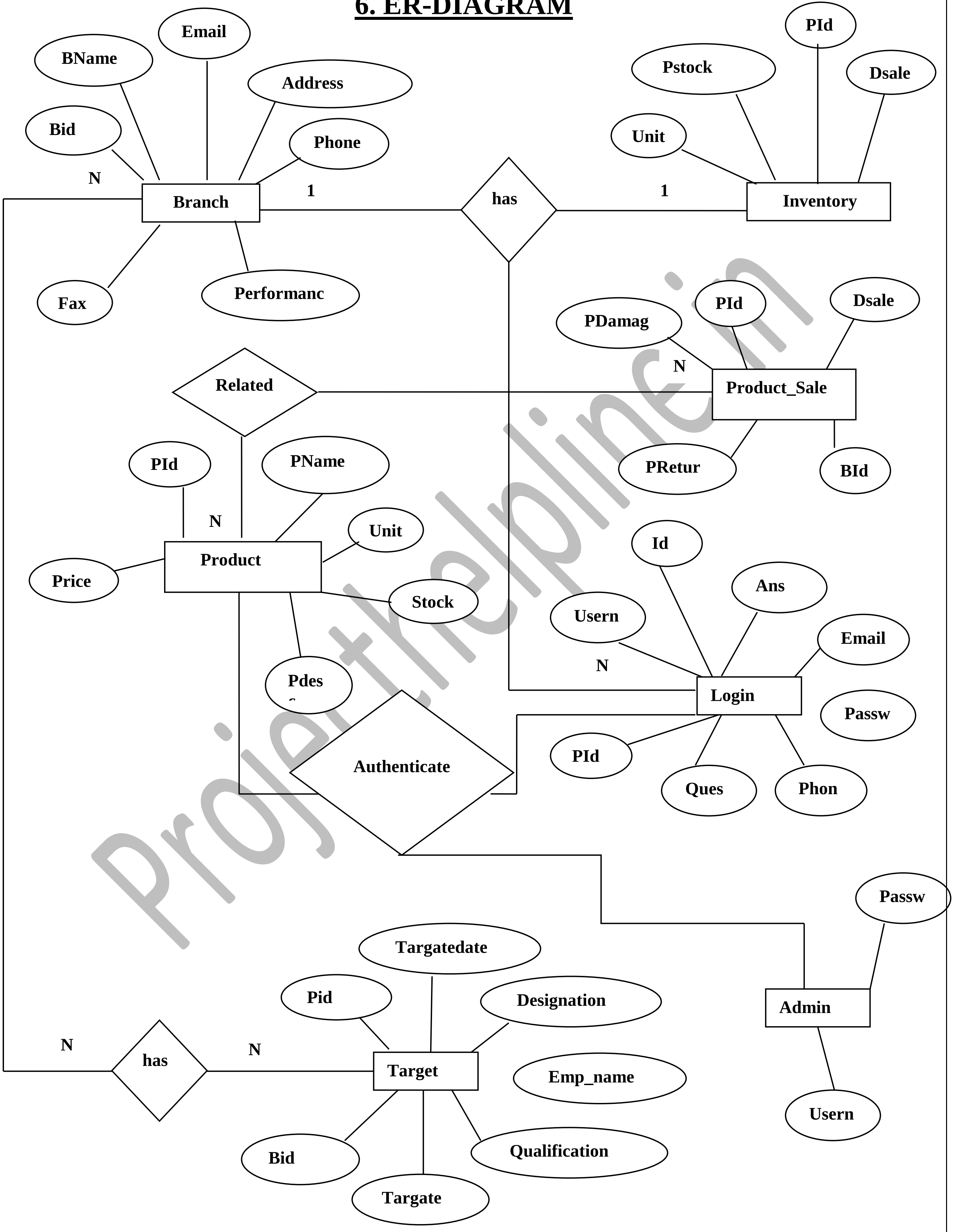
Data flow diagrams are used to describe how the system transforms information. They define how information is processed and stored and identify how the information flows through the processes.

i. Bubbles: - A circle or bubble represents that transform data from once form to another by performing some tasks with the data.

ii. Data store:- A data store is place where data is held temporarily from one transaction to next or is stored permanently.

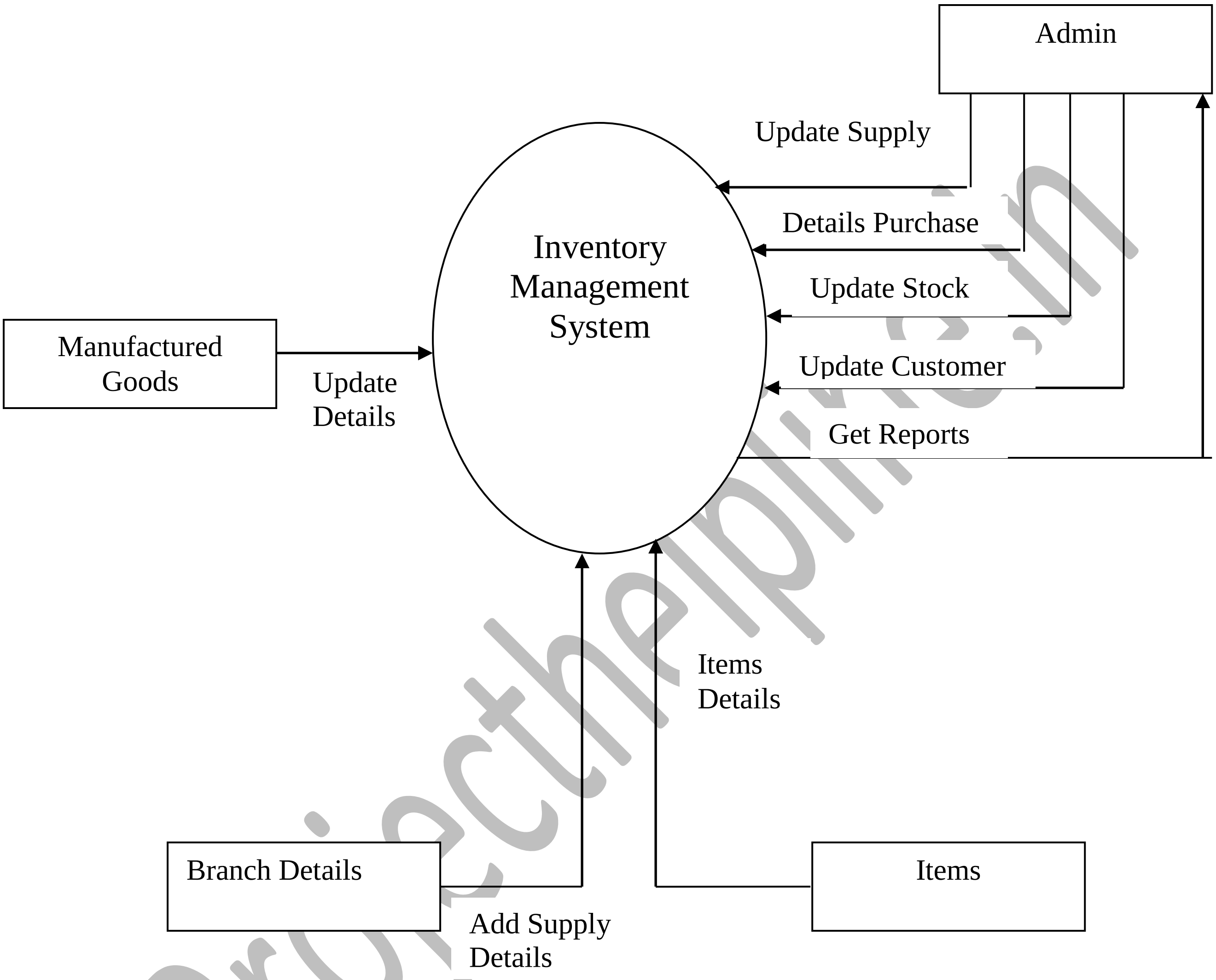
iii. External Entity:- Which defines a source or destination of system data also called an external entity Based on the working process of the proposed system Data Flow Diagram (DFD) is a model, Which gives the insight into the information domain and functional domain at the same time can be drown using OMT symbols, DFD is refined into different levels. The more refined DFD is details of the system are incorporated. In the process of creating a DFD, we decompose the system into different functional subsystems. The DFD refinement results in a corresponding refinement of data.

6. ER-DIAGRAM

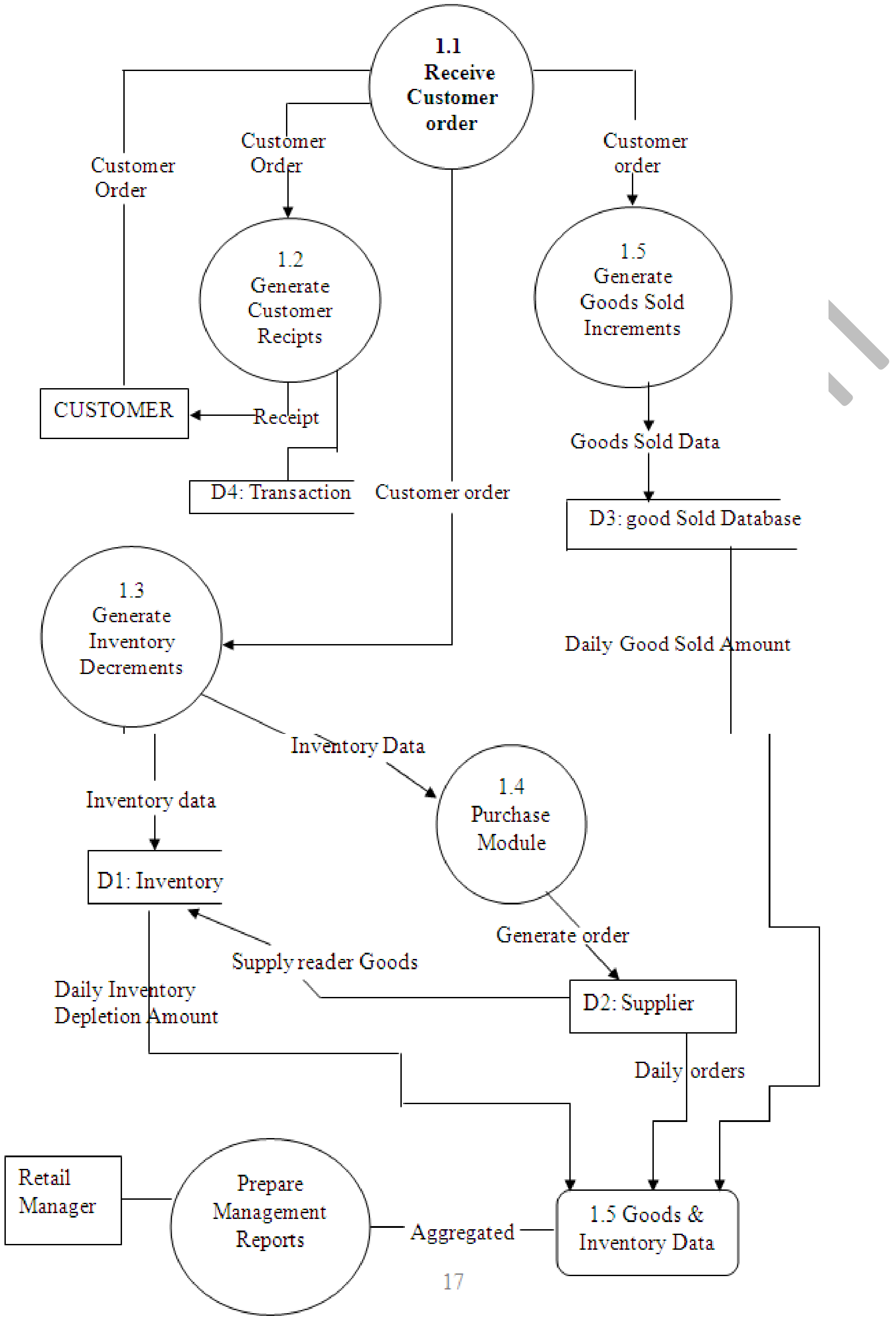


Email
BName
Address
Bid
Phone
N
Branch
1
has
1
Inventory
PId
Pstock
Dsale
Unit
Fax
Performanc
PDamag
PId
Dsale
N
Related
Product_Sale
PRetur
BId
PId
PName
N
Unit
Product
Price
Stock
Id
Ans
Usern
Email
N
Login
Pdes
Passw
Authenticate
PId
Ques
Phon
Passw
Targatedate
Pid
Designation
Admin
N
has
N
Target
Emp_name
Usern
Bid
Qualification
Targate

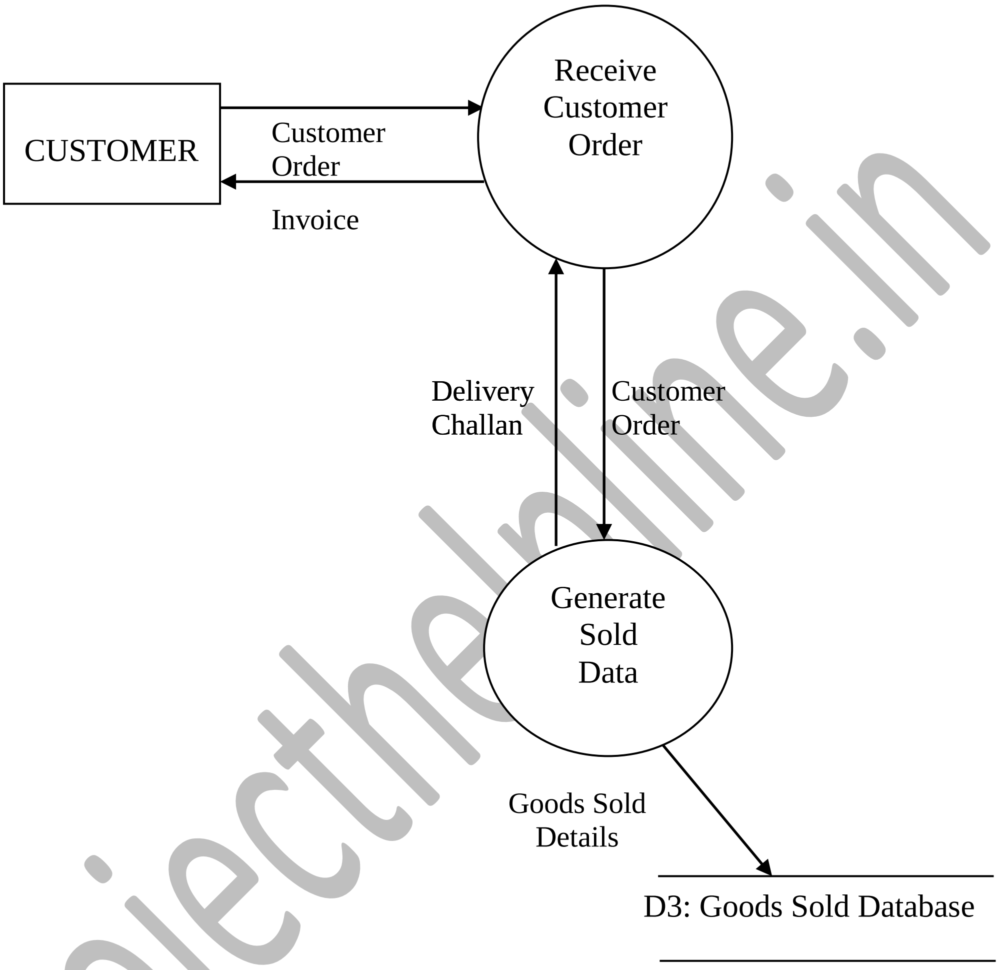
7. CONTEXT LEVEL DFD



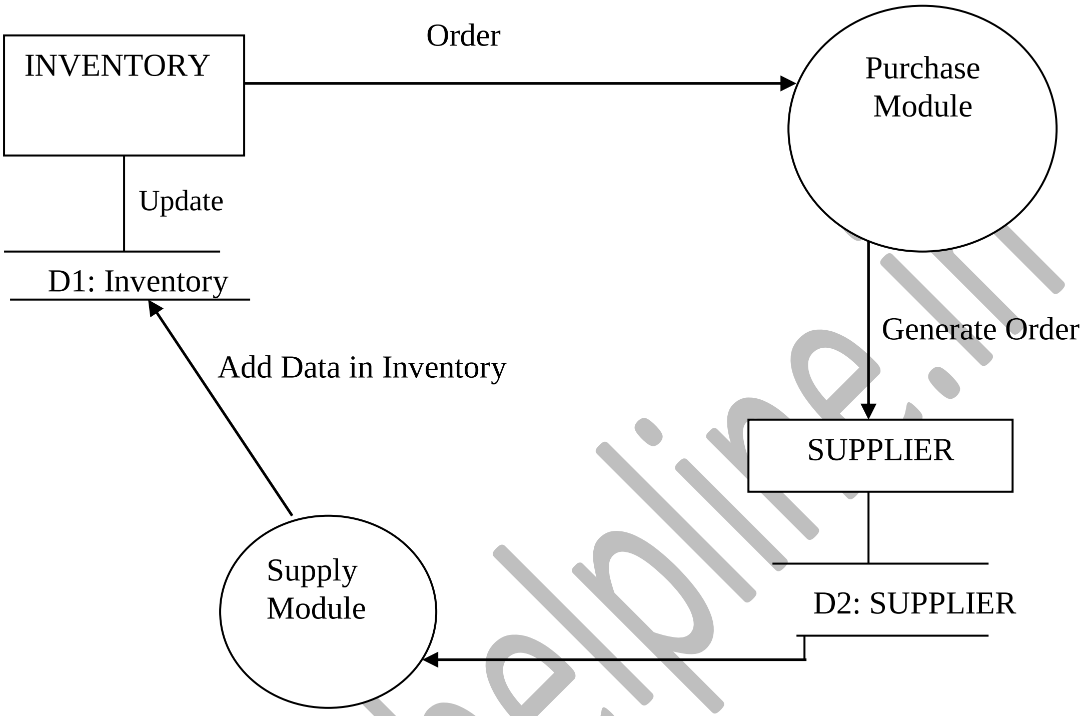
Ist Level DFD



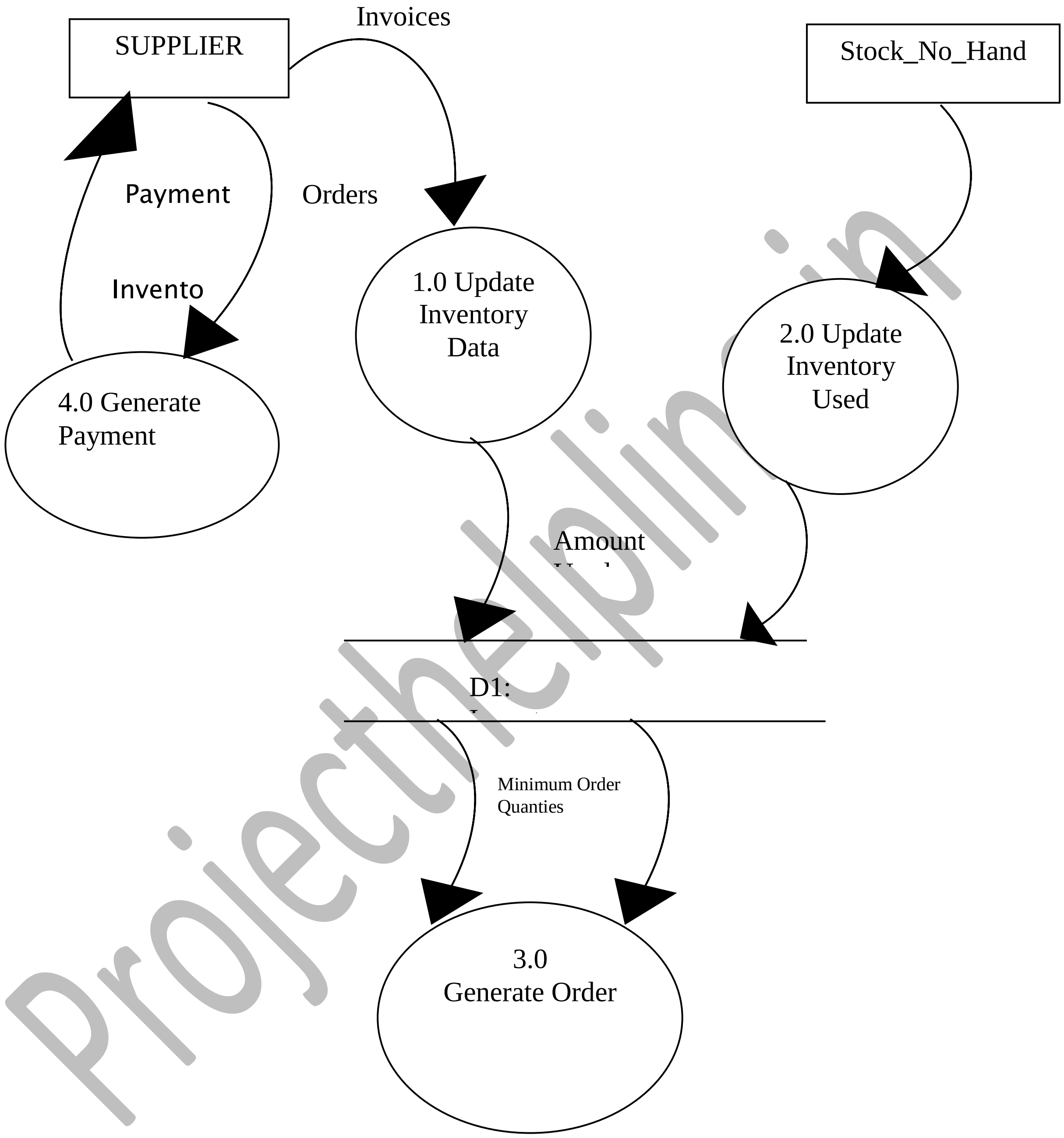
II Level DFD for Goods Sold Inventory



II LEVEL DFD FOR PURCHASE MODULE



II LEVEL DFD FOR INVENTORY DECREMENT



8. DATA MODELING

Admin Table

Field Name	Type	Constraint	Description
User Name	Char(30)	Primary Key	User Name
Password	Char(30)	Not Null	Password

Branch Table

Field Name	Type	Constraint	Description
B_Id	Char(30)	Primary Key	Branch ID
B_Name	Char(30)	Not Null	Branch Name
Performance	Char(100)	Not Null	Performance
Address	Char(100)	Not Null	Address
Phone	Number(10)	Not Null	Phone
Fax	Number(10)	Not Null	Fax
E mail	Char(50)	Not Null	E mail

Feedback Table

Field Name	Type	Constraint	Description
Name	Char(30)	Not Null	Name
E mail	Char(60)	Not Null	E mail Id
Phone	Number(10)	Not Null	Phone No
Comment	Char(70)	Not Null	Comment

Inventory Table

Field Name	Type	Constraint	Description
P_Id	Char(30)	Not Null	Product Id
P_Stock	Char(30)	Not Null	Product in Stock
Unit	Char(30)	Not Null	Measuring Unit
Date_Sale	Date	Not Null	Date of sale

Login Table

Field Name	Type	Constraint	Description
Id	Char(30)	Primary Key	Login Id
User Name	Char(30)	Not Null	User Name
Password	Char(30)	Not Null	Password
E mail	Char(70)	Not Null	E mail
Ques	Char(70)	Not Null	Security Question
Answer	Char(70)	Not Null	Answer
Phone	number	Not Null	Contact No

Product Table

Field Name	Type	Constraint	Description
P_id	Char(30)	Primary Key	Product Id
P_Name	Char(30)	Not Null	Product Name
P_desc	Char(250)	Not Null	Product Description

Price	Number	Not Null	Price
Unit	Char(30)	Not Null	Unit
Stock	number	Not Null	Stock

Sales Table

Field Name	Type	Constraint	Description
Sales_Id	Char(30)	Primary Key	Transaction Id
P_Id	Char(30)	Not Null	Product Id
P_Sales	Number	Not Null	Quantity of Product Sell
__Return	Number	Not Null	No. of Items returned
Damage	Number	Not Null	Damaged Product
Branch_Id	Char(30)	Not Null	Branch Id
Date_Of_Sale	Date	Not Null	Date Of Sale

Purchase Table

Field Name	Type	Constraint	Description
Purchase_Id	Char(30)	Primary Key	Purchase Id
Product Name	Char(50)	Not Null	Product Name
Qty	Number	Not Null	Quantity Purchased
Price	Number	Not Null	Price
Pur_Date	Date	Not Null	Purchase Date
Supplier Name	Char(50)	Not Null	Supplier Name
Address	Char(50)	Not Null	Supplier Address
City	Char(50)	Not Null	City
Phone	Char(30)	Not Null	Phone
Total	Char(50)	Not Null	Total

Target Table

Field Name	Type	Constraint	Description
P_Id	Char(30)	Not Null	Product_Id
B_id	Char(30)	Not Null	Branch_Id
Target	Char(30)	Not Null	Target
Targer_Date	Date	Not Null	Target Date

Supplier Table

Field Name	Type	Constraint	Description
Supplier_Id	Char(30)	Primary Key	Supplier Id
Supplier_Name	Char(30)	Not Null	Supplier Name
Address	Char(30)	Not Null	Address
City	Char(30)	Not Null	City
State	Char(30)	Not Null	State
Contact No	Number(10)	Not Null	Contact No.

Emp Table

Field Name	Type	Constraint	Description
Emp_Id	Char(30)	Primary Key	Employee Id
Name	Char(30)	Not Null	Employee Name

Designation	Char(30)	Not Null	Designation
Department	Char(30)	Not Null	Department
E mail	Char(70)	Not Null	E mail
Date_of_Joining	Date	Not Null	Date of Joining
Age	Number	Not Null	Age
Salary	Number	Not Null	Salary
Contact No.	Number	Not Null	Contact No

9. MODULES

"A modularization consists of well-defined manageable units with well defined interfaces among the units"

Desirable property of modular system include

i. Each module is a well defined sub-system.

ii. Single, well – defined purpose of each module.

iii. Modules can be separately compiled and stored in a library.

iv. Modules can use other module.

v. Modules should be easier to use than to build.

vi. Modules should be simpler from outside then from inside.

Design constraints to be observed during modularization are:-

Coupling: "Coupling is a measure of the degree of interdependence between software modules".

Two modules with high coupling are strongly inter connected and thus dependent on each other.

Two modules with low coupling are not dependent on one another.

Cohesion: "Cohesion is a measure of the degree to which the elements of a module are functionally related".

A strongly cohesive module implements functions, which is related to one feature of the solution and requires little interaction with other modules.

Module 1: Password Module

In this module, this project is for multiple users. If a User enters a password and the software checks is validity. If the password is valid then option is given to change the password, otherwise “Invalid User/Password” message is displayed. There is an option for change password, new users sign in.

Module 2: Creating new Entities (Users, Suppliers, Clients, Purchasers, Salesman) In this module, whenever a new entity is required to be added the corresponding forms are opened and the database is manipulated to check whether the data is already existing or not. If it already exists, then it prompts that “Entry already existing” and if not than the data is entered with the various validation checks.

Module 3: Modifying / Updating Existing Entities

In this module, whenever an existing entity is required to be modified the corresponding forms are opened and the database is manipulated and the data is fetched. Now the administrator can made the required changes and then accordingly, he updates the data. Again, the checks are followed in case there is any invalid entry.

Module 4: Searching

In this module, whenever an existing entity is required to be searched the corresponding forms are opened and the database is manipulated and the data is fetched. Again, the checks are followed in case there is any invalid entry.

Module 5: Validation of Data Entered by the User & Error Handling

In this module, the validity of data entered by the user during the various business processes is checked through various validation checks. For example, there should not be any characters entered in the numeric fields, likewise if there is any error occurs than it should handle that particular error and give the required messages.

Module 6: Sales Orders

This module keeps track sales orders. It keeps tracks of sales order number, order dates, client number, delivery, salesman number etc.

Module 7: Challan

This module this function is used to store, search or modify details of challans like challan number, challan date, order number etc.

Module 8: Stock

This module is used to display all the details regarding Stock currently available in the department, according to which purchase order is placed.

Module 9: Purchase

This module is related to purchase orders. Purchase details- purchase quantity, purchase order number, purchase rate, supplier name etc.

Module 10: Transactions

This module keeps track of all the transactions done by the departments for future records and income tax purposes.

Data Structure Requirements according to the modules:

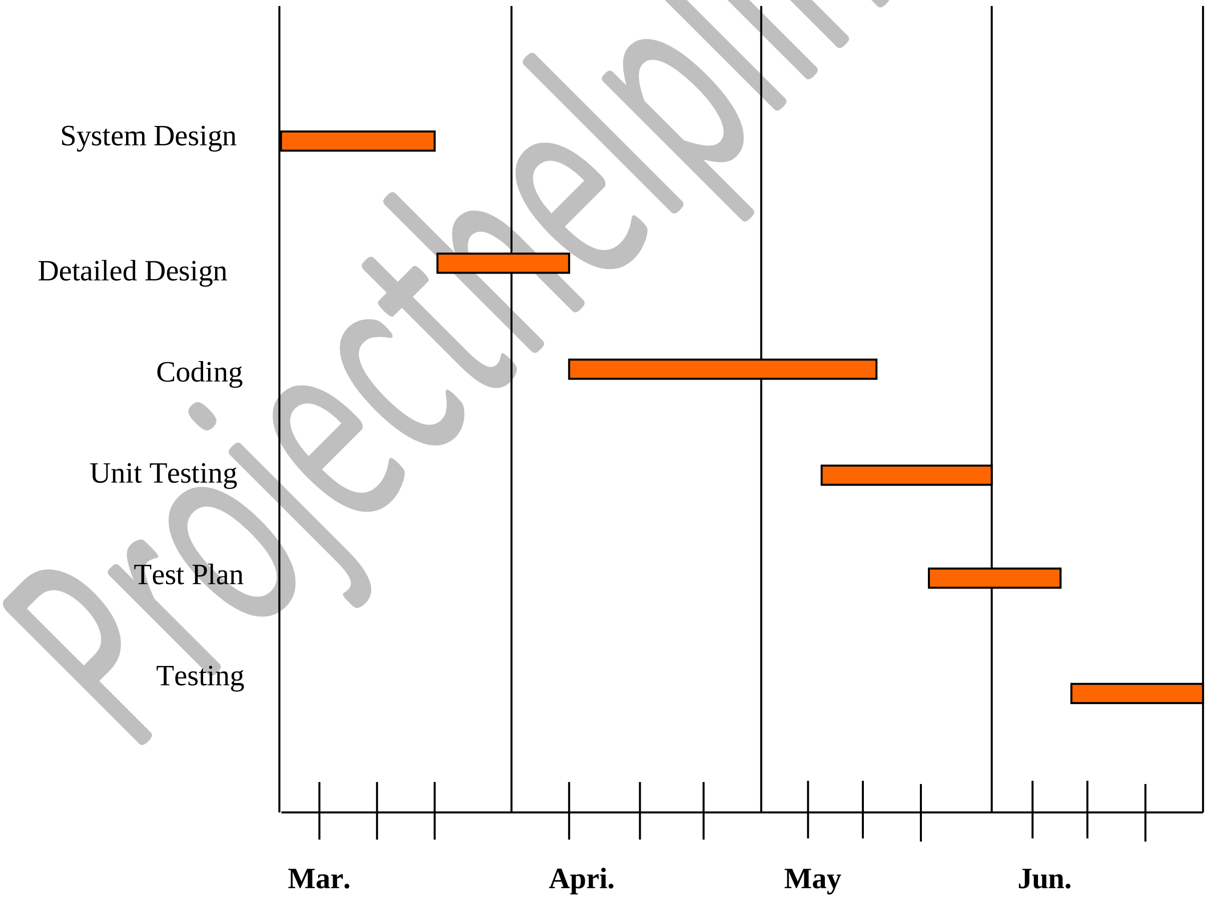
- Identify The Various Tables Required.
- Fields for These Tables.
- The Various Key Fields (for example Primary key and foreign key).
- Identify The Various Constraints like Not Null, Unique etc.

10. PROJECT PLANNING AND SCHEDULING

GANT CHART

Gantt charts mainly used to allocate resources to activities. The resources allocated to activities include staff, hardware, and software. Gantt charts (named after its developer Henry Gantt) are useful for resource planning. A Gantt chart is special type of bar chart where each bar represents an activity. The bars are drawn along a timeline. The length of each bar is proportional to the duration of the time planned for the corresponding activity.

Gantt chart is a project scheduling technique. Progress can be represented easily in a Gantt chart, by coloring each milestone when completed. The project will start in the month of March and end after 4 months at the end of June.



11. SECURITY AND VALIDATION CHECKS

Software's Vulnerability to Attack

Software development is not yet a science or a rigorous discipline, and the development process by and large is not controlled to minimize the vulnerabilities that attackers exploit.

The security of software is threatened at various points throughout its life cycle, both by inadvertent and intentional choices and actions taken by "insiders"—individuals closely affiliated with the organization that is producing, deploying, operating, or maintaining the software, and thus trusted by that organization—and by "outsiders" who have no affiliation with the organization. The software's security can be threatened

- during its development: A developer may corrupt the software—intentionally or unintentionally—in ways that will compromise the software's dependability and trustworthiness when it is operational.
- during its deployment (distribution and installation): If those responsible for distributing the software fail to tamperproof the software before shipping or uploading, or transmit it over easily intercepted communications channels, they leave the software vulnerable to intentional or unintentional corruption. Similarly, if the software's installer fails to "lock down" the host platform, or configures the software insecurely, the software is left vulnerable to access by attackers.
- during its operation: Once COTS and open source software has gone operational, vulnerabilities may be discovered and publicized; unless security patches and updates are applied and newer supported versions (from which the root causes of vulnerabilities have been eliminated) are adopted, such software will become increasingly vulnerable. Non-commercial software and open source software (OSS) may also be vulnerable, especially as it may manifest untrustworthy behaviors over time due to changes in its environment that stress the software in ways that were not anticipated and simulated during its testing.
- during its sustainment: If those responsible for addressing discovered vulnerabilities in released software fail to issue patches or updates in a timely manner, or fail to seek out and eliminate the root causes of the vulnerabilities to prevent their perpetuation in future releases of the software, the software will become increasingly vulnerable to

threats over time. Also, the software’s maintainer may prove to be a malicious insider, and may embed malicious code, exploitable flaws, etc., in updated versions of the code.

The Challenge of Building Secure Software

1. **Dependability:** Dependable software executes predictably and operates correctly under all conditions, including hostile conditions, including when the software comes under attack or runs on a malicious host.
2. **Trustworthiness:** Trustworthy software contains few if any vulnerabilities or weaknesses that can be intentionally exploited to subvert or sabotage the software’s dependability. In addition, to be considered trustworthy, the software must contain no malicious logic that causes it to behave in a malicious manner.
3. **Survivability (also referred to as “Resilience”):** Survivable—or resilient—software is software that is resilient enough to (1) either resist (i.e., protect itself against) or tolerate (i.e., continue operating dependably in spite of) most known attacks plus as many novel attacks as possible, and (2) recover as quickly as possible, and with as little damage as possible, from those attacks that it can neither resist nor tolerate.

Software Assurance

The main objective of software assurance is to ensure that the processes, procedures, and products used to produce and sustain the software conform to all requirements and standards specified to govern those processes, procedures, and products. Software security and secure software are often discussed in the context of software assurance. Software assurance in its broader sense refers to the assurance of any required property of software.

An increasingly agreed-upon approach for assuring the security of software is the software security assurance case, which is intended to provide justifiable confidence that the software under consideration (1) is free of vulnerabilities; (2) functions in the “intended manner,” and this “intended manner” does not compromise the security or any other required properties of the software, its environment, or the information it handles; and (3) can be trusted to continue operating dependably under all anticipated circumstances, including anomalous and hostile environmental and utilization circumstances—which means that those who build the software need to anticipate such circumstances and design and implement the software to be able to handle them gracefully. Such circumstances include

- the presence of unintentional faults in the software and its environment
- the exposure of the operational software to accidental events that threaten its security
- the exposure of the software to intentional choices or actions that threaten its security during its development, deployment, operation, or sustainment

Software is more likely to be assuredly secure when security is a key factor in the following aspects of its development and deployment:

- **development principles and practices:** The practices used to develop the software and the principles that governed its development are expressly intended to encourage and support the consideration and evaluation of security in every phase of the software's development life cycle. Some secure development principles and practices for software are suggested later in this article.
- **development tools:** The programming language(s), libraries, and development tools used to design and implement the software are evaluated and selected for their ability to avoid security vulnerabilities and to support secure development practices and principles.
- **testing practices and tools:** The software is expressly tested to verify its security, using tools that assist in such testing.
- **acquired components:** Commercial off-the-shelf (COTS) and OSS components are evaluated to determine whether they contain vulnerabilities, and if so whether the vulnerabilities can be remediate through integration to minimize the risk they pose to the software system.
- **deployment configuration:** The installation configuration of the software minimizes the exposure of any residual vulnerabilities it contains.
- **execution environment:** Protections are provided by the execution environment that can be leveraged to protect the higher level software that operates in that environment.

practitioner knowledge: The software's analysts, designers, developers, testers, andmaintainers are provided with the necessary information (e.g., through training and education) to give them sufficient security awareness and knowledge to understand, appreciate, and effectively adopt the principles and practices that will enable them.

12. SCOPE OF FUTURE APPLICATION

I am confident that this software package can be readily used by non-programming personal avoiding human handled chance of error.

The decision to automate generally depends on the needs to have accurate, consistent and timely data in a variety of reporting formats. But the most important factor that should be considered is –

- ✓ Be sure to take future growth into consideration and evaluate whether the software package could be modified if the organization expands in the future or you need to revise the system.

- ✓ The next generation of Windows-based computerized systems would be changing the way owners and management think about running their businesses. It would a powerful tool that gives them more ways to get their financial Information so that they can better manage and grow their business.

- ✓ The next generation of systems would take advantage of the latest technologies, including Microsoft Windows XP and SQL Server to offer access and integrate with all aspects of a business. Keeping this in view we could develop systems that would work efficiently and integrate seamlessly in virtually any industry setting, even up to mid-sized corporations and divisions of larger organizations.

- ✓ Consider another setting, where the users are not comfortable on computers, for such users we can develop systems where the users can have their own personalized menu setup in the accounting software so that they no longer have to search around to find what they need.

- ✓ Now consider the Internet. This wide open information exchange network comes with many benefits for businesses, including breaking down physical barriers as to where he/she can consult to the companies.

- ✓ Adding other capabilities can be added time to time. These are
    - Application may be installed client server.

- Application can upgraded according to user’s and administrator’s requirements with little changes made
- Application may be transferred to latest RDMBS like Oracle with little changes in current code. Use for Security purpose

13. CONCLUSION

This project is designed to meet the requirements of the enterprise. The software project is very big and includes almost all aspects of Enterprise application. It has been developed in JSP, SERVLETS, keeping in mind the specifications of the system.

For designing the system we have used simple data flow diagrams.

Overall the project teaches us the essential skills like:

- Using system analysis and design techniques like data flow diagram in designing the system.
- Understanding the database handling and query processing using My SQL .

14. BIBLIOGRAPHY

[1] Herbert Scheldt, Java Complete Reference, Fifth Edition, Tata McGraw Hill Edition.

[2] Phil Hanna, JSP 2.0: The Complete Reference, Tata McGraw Hill Edition, 2003.

[3] Elmarsi and Navathe, Fundamentals of Database System (Third Edition), Addision Wesley.

[4] Ian Somerville, Software Engineering, Third Edition, Pearson Education.

[5] Ali Bahrami, Object-Oriented System Development, Third Edition, Tata McGraw Hill Edition.

[6] Ivan Bayross, SQL, PL/SQL programming language of Oracle, Second Edition, BPB Publication.

WEB REFERENCES

[1] www.google.com

[2] www.htmlcodetutorial.com