TEXTILE SHOP MANAGEMENT SYSTEM

PROJECT REPORT

Submitted in partial fulfillment of requirement for the award of the Degree

Bachelor of Computer Science

In the faculty of Computer Science of Bharathiar University, Coimbatore

Submitted by

S.MUTHAMILSELVI

(Reg.No.2022K0141)

Under the guidance of

Dr.A.FINNY BELWIN,MCA., M.Sc.,Ph.D

Guest lecturer Department of Computer Science

Department of Computer Science

L.R.G GOVERNMENT ARTS COLLEGE FOR WOMEN

(Affiliated To Bharathiar University)

TIRUPUR-4

APRIL-2023

CERTIFICATE

CERTIFICATE

This is to certify that the project work entitled **" TEXTILE SHOP MANAGEMENT SYSTEM"** Submitted to Bharathiar University in partial fulfilled of the requirement for the award of the Degree of Bachelor of computer science is a record of the original work done by **Ms.S.MUTHAMILSELVI (Reg.No.2022K0141)** Under my supervisor and that project work has not formed the basis for the any Degree /Diploma /Association /Fellowship or similar title to any candidate of any university.

Internal Guide	**Head of the Department**
Dr.A.FINNY BELWIN,MCA.,MSc.,Ph.D	**Dr.R.PARIMALA,MSc.,M.Phil.,Ph.D**

Viva-voce examination is held on _____L.R.G Government Arts College for Women, Tirupur-641604.

Internal Examiner **External Examiner**

DECLARATION

DECLARATION

I hereby declare that the project work submitted to the **UG Department of the Computer Science, L.R.G. Government Arts College for Women, Tirupur** , affiliated to Bharathiar University, Coimbatore in the partial fulfillment of the required for the award of Bachelor of Computer Science is an original work done by me during the sixth semester.

Place: **Signature of the Candidate**

Date: **(S.MUTHAMILSELVI)**

(Reg.no:2022K0141)

ACKNOWLEDGEMENT

ACKNOWLEDGEMENT

As first and foremost I would like to external my thankfulness to the almighty for blessing the work of my hands, I am grateful to my parents for continued encouragement that they had given to me.

I would like to external my profound gratitude and sincere thanks to **Dr.M.R.YEZHINI MA.,M.Phil.,Ph.D** Principal, L.R.G Government Arts College For Women , for the encouragement rendered to me during this project.

It's my privilege to thank **DR.R.PARIMALA M.Sc.,M.Phil.,Ph.D** Incharge Head of the department of computer science for her valuable guidance and support throughout the project development work.

I express my deep sense of gratitude and sincere thanks to my guide **Dr.A.FINNY BELWIN MCA., M.Sc.,Ph.D** Lecturer, Department of computer science, for providing all sorts of facilities to complete my project work successfully.

I also extend my sincere thanks to all the other faculty member of the department and technical assistance for their co-operation and valuable guidance.

I thank our college library for providing me with many informative books that help me to enrich my knowledge to bring out the project successfully.

SYNOPSIS

SYNOPSIS

The Textile management system application is developed for managing the textile shop in a good manner. How the thing is proper in the shopping mall ,what is input in the shopping mall and what is the output hoe to track the goods are available there or which is sort.

All this is auto track by the application from which there will be no any difficulties facing by the management after all there are certain report generation based on the shopping mall daily turnover, monthly turnover etc..

CONTENTS

INTRODUCTION

1.INTRODUCTION

1.1 ABSTRACT

The textile shop management system is a report that highlights the importance of maintaining the textile industry. This report can easily focus on how the working of the textile shop takes place easily.

The textile shop management can help the users to keep track of the maintanance activities that take place at the textile shop.

The details of the members can also be maintainable easily.

The customer's details are also easily maintainable through the application. The report can also emphasize the same.

Textile Management System is end user application software.

The textile system we are designing that helps the textile company to handle those task that had been handled manually.

The tasks like:

- Keeping record of stock available.
- Maintaining record of customer details.
- Generates billing information.

The textile shop management system report can easily emphasize how the textile shop works. The report also gets the details of how the textile industry works easily.

Most of the works at the textile shop is easily automated through the use of some automation that is being done easily.

All the discounts that are available at the textile shops are easily highlighted through this report without any issue. The people can easily get the information of the textile industry which can emphasize the same easily.

1.2 SYSTEM SPECIFICATION

System Requirements Specification also known as Software Requirements Specification, is a document or set of documentation that describes the features and behavior of a software application.

WINDOWS OS

Windows is a graphical operating system developed by Microsoft. It allows users to view and store files, run the software, play games, watch videos, and provides a way to connect to the internet. It was released for both home computing and professional works.

Microsoft introduced the first version as 1.0

It was released for both home computing and professional functions of Windows on 10 November 1983. Later, it was released on many versions of Windows as well as the current version, Windows 10.

In 1993, the first business-oriented version of Windows was released, which is known as Windows NT 3.1. Then it introduced the next versions, Windows 3.5, 4/0, and Windows 2000. When the XP Windows was released by Microsoft in 2001, the company designed its various versions for a personal and business environment. It was designed based on standard x86 hardware, like Intel and AMD processor. Accordingly, it can run on different brands of hardware, such as HP, Dell, and Sony computers, including home-built PCs.
Play Video.
Editions of Windows
Microsoft has produced several editions of Windows, starting with Windows XP. These versions have the same core operating system, but some versions included advance features with an additional cost. There are two most common editions of Windows:

- Windows Home
- Windows Professional

Windows Home is basic edition of Windows. It offers all the fundamental functions of Windows, such as browsing the web, connecting to the Internet, playing video games, using office software, watching videos. Furthermore, it is less expensive and comes pre-installed with many new computers.

JAVA

Java is a high-level programming language developed by Sun Microsystems. It was originally designed for developing programs for set-top boxes and handheld devices, but later became a popular choice for creating web applications.

The Java syntax is similar to C++, but is strictly an object-oriented programming language. For example, most Java programs contain classes, which are used to define objects, and methods, which are assigned to individual classes. Java is also known for being stricter than C++, meaning variables and functions must be explicitly defined. This means Java source code may produce errors or "exceptions" more easily than other languages, but it also limits other types of errors that may be caused by undefined variables or unassigned types.

Unlike Windows executable (.EXE files) or Macintosh applications (.APP files), Java programs are not run directly by the operating system. Instead, Java programs are interpreted by the Java Virtual Machine, or JVM, which runs on multiple platforms. This means all Java programs are multiplatform and can run on different platforms, including Macintosh, Windows, and Unix computers. However, the JVM must be installed for Java applications or applets to run at all. Fortunately, the JVM is included as part of the Java Runtime Environment (JRE),

MYSQL

MySQL is an Oracle-backed open source relational database management system (RDBMS) based on Structured Query Language (SQL). MySQL runs on virtually all platforms, including Linux, UNIX and Windows. Although it can be used in a wide range of applications, MySQL is most often associated with web applications and online publishing.

MySQL is an important component of an open source enterprise stack called LAMP. LAMP is a web development platform that uses Linux as the operating system, Apache as the web server, MySQL as the relational database management system and PHP as the object-oriented scripting language. (Sometimes Perl or Python is used instead of PHP.)

Originally conceived by the Swedish company MySQL AB, MySQL was acquired by Sun Microsystems in 2008 and then by Oracle when it bought Sun in 2010. Developers can use MySQL under the GNU General Public License (GPL), but enterprises must obtain a commercial license from Oracle.

Relational database management systems use structured query language (SQL) to store and manage data. The system stores multiple database tables that relate to each other. MS SQL Server, MySQL, or MS Access are examples of relational database management systems. The

following are the components of such a system.

A SQL table is the basic element of a relational database. The SQL database table consists of rows and columns. Database engineers create relationships between multiple database tables to optimize data storage space.

SQL statements, or SQL queries, are valid instructions that relational database management systems understand. Software developers build SQL statements by using different SQL language elements. SQL language elements are components such as identifiers, variables, and search conditions that form a correct SQL statement.

1.2.1 HARDWARE REQUIREMENT

- Processor : Intel core I3
- RAM : 4 GB RAM
- Hard Disk Drive : 500 GB
- Keyboard : Optical
- Mouse : Optical

1.2.2 SOFTWARE SPECIFICATION

- Operating System : Windows 7
- Front End : JAVA
- Back End : MYSQL

SYSTEM STUDY

2.SYSTEM STUDY

2.1 EXISTING SYSTEM

Users very hard to maintain the textile shop. We can't calculate the amount of dress or material details manually. Sometimes it makes confusing to find the dress rates it may cause big issue.

2.1.1 DRAWBACKS

- All works are manually implemented.
- Changes to make mistakes in calculations.

2.2 PROPOSED SYSTEM

The purpose of the project is to develop a 'Textile management system', which will be used by the company through which all purchase details of textile can be managed by the company. The system deal with very popular interface tool retrieval of the record is which faster than the present system. Hence it cause to saving time for the further work.

2.2.1 FEATURES

- Searching features is quite faster.
- Attractive user interface.
- Billing details very easy to handle.

SYSTEM DESIGN AND DEVELOPMENT

3.SYSTEM DESIGN AND DEVELOPMENT

3.1 FILE DESIGN

The selection of the file system design approach is done according to the needs of the developers what are the needed requirements and specifications for the new design. It allowed us to identify where our proposal fitted in with relation to current and past file system development. Our experience with file system development is limited so the research served to identify the different techniques that can be used. The variety of file systems encountered show what an active area of research file system development is. The file systems may be from one of the two fundamental categories. In one category, the file system is developed in user space and runs as a user process. Another file system may be developed in the kernel space and runs as a privileged process. Another one is the mixed approach in which we can take the advantages of both aforesaid approaches. Each development option has its own pros and cons. In this article, these design approaches are discussed.

A file system is the data structure designed to support the abstraction of the data blocks as an archive and collection of files. This data structure is unique because it is stored on secondary storage (usually the disk), which is a very slow device.

The file system structure is the most basic level of organization in an operating system. Almost all of the ways an operating system interacts with its users, applications, and security model are dependent upon the way it organizes files on storage devices.

File Design Information systems in business are file and database oriented. Data are accumulated into files that are processed or maintained by the system. The systems analyst is responsible for designing files, determining their contents and selecting a method for organizing the data.

The most important purpose of a file system is to manage user data. This includes storing, retrieving and updating data. Some file systems accept data for storage as a stream of bytes which are collected and stored in a manner efficient for the media.

3.2 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:'

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

- Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
- It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
- When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user
- will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

3.3 OUTPUT DESIGN

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

External Outputs

Manufacturers create and design external outputs for printers. External outputs enable the system to leave the trigger actions on the part of their recipients or confirm actions to their recipients.

Some of the external outputs are designed as turnaround outputs, which are implemented as a form and re-enter the system as an input.

Internal outputs

Internal outputs are present inside the system, and used by end-users and managers. They support the management in decision making and reporting.

Output Integrity Controls

Output integrity controls include routing codes to identify the receiving system, and verification messages to confirm successful receipt of messages that are handled by network protocol.

Printed or screen-format reports should include a date/time for report printing and the data. Multipage reports contain report title or description, and pagination. Pre-printed forms usually include a version number and effective date.

3.4 DATABASE DESIGN

Today's businesses depend on their databases to provide information essential for day-to-day operations, especially in case of electronic commerce businesses who has a definite advantage with up-to-date database access. Good design forms the foundation of any database, and experienced hands are required in the automation process to design for optimum and stable performance.

Software Solutions have been constantly working on these platforms and have attained a level of expertise. We apply proven methodologies to design, develop, integrate and implement database systems to attain its optimum level of performance and maximize security to meet the client's business model.

Business needs addressed:

- Determine the basic objects about which the information is stored
- Determine the relationships between these groups of information and the objects
- Effectively manage data and create intelligent information
- Remote database administration or on site administrative support
- Database creation, management, and maintenance
- Information retrieval efficiency, remove data redundancy and ensure data security

The most important consideration in designing the database is how the information will be used. The main objective of designing a database is Data Integration, Data Integrity and Data Independence.

Data Integration

In a database, information from several files is coordinated, accessed and operated upon as through it is in a single file. Logically, the information is centralized, physically; the data may be located on different devices, connected through data communication facilities.

Data Integrity

Data integrity means storing all data in one place only and how each application accesses it. This approach results in more consistent information, one update being sufficient to achieve a new record status for all applications. This leads to less data redundancy that is data items need not be duplicated.

Data Independence

Data in dependence is the insulation of application programs from changing aspects of physical data organization. This objective seeks to allow changes in the content and organization of physical data without reprogramming of application and allow modifications to application programs without reorganizing the physical data.

3.5 SYSTEM DEVELOPMENT

Systems development is the process of defining, designing, testing, and implementing a new software application or program. It could include the internal development of customized systems, the creation of database systems, or the acquisition of third party developed software.

Systems development life cycle phases include planning, system analysis, system design, development, implementation, integration and testing, and operations and maintenance.

3.5.1 DESCRIPTION OF MODULES

Admin module

Managing the whole work of our textile. He is responsible person for maintaining the customer details, purchase order details, sales and billing.

Customer module

A Customer module is used to store the customer's details in this module. We can search the customer details immediately. We can manage the history of customer's details which is helpful to find our regular customer.

Purchase module

This modules contains are user purchase the item details maintained. It includes are item name, purchasing date, no of quantity, rate details and so on.

Sales module

This module has been used when the manager submit the billing after collecting the money. This module helps to calculate the stock and billing details. This module has contain dress and customer details.

Billing module

Billing module is an output of our application also it's a main module. This will generate the date wise bill details. Also it will show the customer details who is purchasing and how much they were purchasing..

Stock module

Stock module has details of available stock. With item, batch price and quantity details. It also has details of sold stock details. Using this details we can put more purchase order if we want.

TESTING AND IMPLEMENTATION

4.TESTING AND IMPLEMENTATION

TESTING METHODOLOGIES

System testing is state of implementation, which is aimed at ensuring that the system works accurately and efficiently as expect before live operation commences. It certifies that the whole set of programs hang together.

System testing requires a test plan that consists of several key activities and step for run program, string, system and user acceptance testing. The implementation of newly designed package is important in adopting a successful new system

Testing is the important stage in software development. the system test in implementation stage in software development process. The system testing implementation should be confirmation that all is correct and an opportunity to show the users that the system works as expected. It accounts the largest percentage of technical effort in the software development process.

Testing phase in the development cycle validates the code against the functional specification testing is vital to achievement of the system goals. The objective of the testing is to discover errors to fulfill this objective a series of test step unit, integration. Validation and system tests were planned and executed the test steps are:

SYSTEM TESTING

Testing is an integral part of any system development life cycle. Insufficient and untested applications may tend to crash and the result is loss of economic and manpower investment besides user's dissatisfaction and downfall of reputation. Software testing can be looked upon as one among many processes, an organization performs, and that provides the lost opportunity to correct any flaws in the developed system. Software testing includes selecting test data that have more probability of giving errors.

The first step in system testing is to develop a plan that tests all aspects of the system. Completeness, correctness, reliability and maintainability of the software are to be tested for the best quality assurance that the system meets the specification and requirements for its intended use and performance. System testing is the most useful practical process of executing a program with the implicit intention of finding errors that make the program fails. System testing is done in three phases.

- Unit Testing
- Integration Testing
- Validation Testing

UNIT TESTING

Unit testing focuses verification effort on the smallest unit of software the module. Using the detailed design and the process specification testing is done to registration by the user with in the boundary of the Login module. The login form receives the username and password details and validates the value with the database. If valid, the home page is displayed.

INTEGRATION TESTING

Integration Testing is the process of this activity can be considered as testing the design and hence module interaction. The primary objective of integration testing is to discover errors in the interfaces between the components. Login form and registration form are integrated and tested together. If the user is newly registered, the received details will be stored in the registration table. While logging in, the application will check for valid user name and password in the registration table and if valid the user is prompted for submitting complaints.

Data can be lost across an interface, one module can have adverse effect on another sub function when combined it may not produce the desired major functions. Integration testing is a systematic testing for constructing test to uncover errors associated within an interface.

The objectives taken from unit tested modules and a program structure is built for integrated testing. All the modules are combined and the test is made.

A correction made in this testing is difficult because the vast expenses of the entire program complicated the isolation of causes. In this integration testing step, all the errors are corrected for next testing process.

VALIDATION TESTING

Validation are independent procedures that are used together for checking that a product, service, or system meets requirements and specifications and that it fulfills its in purpose the actual result from the expected result for the complaint process. Select the complaint category of the complaint by user. The input given to various forms fields are validated effectively. Each module is tested independently. It is tested that the complaint module fields receive the correct input for the necessary details such as complaint category, complaint id, reference name, complaint description, and email for further process.

After the completion of the integrated testing, software is completely assembled as a package; interfacing error has been uncovered and corrected and a final series of software test validation begins.

Validation testing can be defined in many ways but a simple definition is that validation succeeds when the software function in a manner that can be reasonably expected by the customer. After validation test has been conducted, one of two possible conditions exists.

OUTPUT TESTING

The next process of validation testing, is output testing of the proposed system, since no system could be successful if it does not produce the required output in the specified format. Asking the user about the format required, list the output to be generated or displayed by the system under considerations.

Output testing is a different test whose primary purpose is to fully exercise the computer based system although each test has a different purpose all the work should verify that all system elements have been properly integrated and perform allocated functions.

The output format on the screen is found to be corrected as the format was designed in the system design phase according to the user needs for the hard copy also; the output testing has not resulted in any correction in the system.

SYSTEM IMPLEMENTATION

When the initial design was done for the system, the client was consulted for the acceptance of the design so that further proceedings of the system development can be carried on. After the development of the system a demonstration was given to them about the working of the system. The aim of the system illustration was to identify any malfunction of the system.

After the management of the system was approved the system implemented in the concern, initially the system was run parallel with existing manual system. The system has been tested with live data and has proved to be error free and user friendly.

Implementation is the process of converting a new or revised system design into an operational one when the initial design was done by the system; a demonstration was given to the end user about the working system.

This process is uses to verify and identify any logical mess working of the system by feeding various combinations of test data. After the approval of the system by both end user and management the system was implemented.

System implementation is made up of many activities. The six major activities are as follows.

CODING

Coding is the process of whereby the physical design specifications created by the analysis team turned into working computer code by the programming team. A design code may be a tool

which helps ensure that the aspiration for quality and quantity for customers and their requirements..

INSTALLATION

Installation is the process during which the current system is replaced by the new system. This includes conversion of existing data, software, and documentation and work procedures to those consistent with the new system.

DOCUMENTATION

Documentation is descriptive information that describes the use and operation of the system.

The admin could verify the authentication of the users, users requirements and need to take delivery process, thus the documentation is made of full view of project thus it gives the guideline to study the project and how to execute also.

USER TRAINING AND SUPPORT

The goal of an end user training program is to produce a motivated user who has the skills needed to apply what has been to apply what has been learned to perform the job related task. The following are the instruction which is specified the handling and un-handling events in the application,

- The authenticated user of admin and office workers only login in the application with authorized username and password.
- It can easily track through online by the user.
- Very user friendliness software.

IMPLEMENTATION PROCEDURES

Implementation includes all the activities that take place to convert the old system to the new one. Proper implementation is essential to provide a reliable system to meet the organization requirements. Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & giving the user confidence in that the new system will work efficiently & effectively in the implementation state.

PILOT RUNNING

Processing the current data by only one user at a time called the pilot running process. When one user is accessing the data at one system, the system is sets to be engaged and connected in network. This process is useful only in system where more than one user is restricted.

PARALLEL RUNNING:

Processing the current data by more than one user at a time simultaneously is said to be parallel running process. This same system can be viewed and accessed by more than one user at the time. Hence the implementation method used in the system is a pilot type of implementation.

Implementation is the stage in the project where the theoretical design is turned into a working system. The most crucial stage is achieving a successful new system & giving the user confidence in that the new system will work efficiently & effectively in the implementation state.

The stage consists of,

- Testing the developed program with sample data.
- Detection's and correction of error.
- Creating whether the system meets user requirements.
- Making necessary changes as desired by the user.
- Training user personnel.

USER TRAINING

User Training is designed to prepare the user for testing &consenting the system. .

- User Manual.
- Help Screens.
- Training Demonstration.

USER MANUAL

The summary of important functions about the system and software can be provided as a document to the user.

HELP SCREENS

This features now available in every software package, especially when it is used with a menu. The user selects the “Help” option from the menu. The system accesses the necessary description or information for user reference.

TRAINING DEMONSTRATION:

Another User Training element is a Training Demonstration. Live demonstrations with personal contact are extremely effective for Training Users.

SYSTEM MAINTENANCE

Maintenance is actually the implementation of the review plan. As important as it is, many programmers and analysts are to perform or identify themselves with the maintenance effort. There are psychological, personality and professional reasons for this. Analysts and programmers spend far more time maintaining programs than they do writing them. Maintenance accounts for 50-80 percent of total system development.

Maintenance is expensive. One way to reduce the maintenance costs are through maintenance management and software modification audits**.**

- Maintenance is not as rewarding as exciting as developing systems. It is perceived as requiring neither skill not experience.
- Users are not fully cognizant of the maintenance problem or its high cost.
- Few tools and techniques are available for maintenance.
- A good test plan is lacking.
- Standards, procedures, and guidelines are poorly defined and enforced.
- Programs are often maintained without care for structure and documentation.
- There are minimal standards for maintenance.
- Programmers expect that they will not be in their current commitment by time their programs go into the maintenance cycle.

Corrective Maintenance

It means repairing, processing or performance failure or making changes because of previously uncovered problems or false assumptions. Task performed to identify, isolate, and rectify a fault so that the failed equipment, machine, or system can be restored to an operational condition within the tolerances or limits established for in-service operations.

Corrective maintenance can be subdivided into "immediate corrective maintenance" (in which work starts immediately after a failure) and "deferred corrective maintenance" (in which work is delayed in conformance to a given set of maintenance rules).

CONCLUSION

5. CONCLUSION

Textile Management Systems (TMS) are essential tools for managing and optimizing the production, distribution, and sales of textiles. These systems provide several benefits, including improved efficiency, increased visibility, and reduced costs.

TMS ensures improved efficiency by enabling textile managers to streamline processes, optimize inventory levels, and reduce lead times. This helps to improve the overall productivity of textile operations, reduce manufacturing and distribution costs, and enhance customer satisfaction.

Moreover, TMS provides increased visibility by providing real-time information on the status of orders, inventory levels, and production schedules. This helps to improve decision-making, enhance transparency, and improve customer service.

TMS also reduces costs by optimizing resource allocation, reducing waste, and minimizing stockouts. This leads to improved financial sustainability and the ability to invest in better infrastructure and services.

In conclusion, Textile Management Systems are essential tools for managing and optimizing the production, distribution, and sales of textiles. They offer several benefits, including improved efficiency, increased visibility, and reduced costs. By leveraging these benefits, textile managers can improve their services, enhance customer satisfaction, and promote sustainable development.

BIBLIOGRAPHY

BIBLIOGRAPHY

Book Reference:

- Bloch, Joshua. Effective Java: Programming Language Guide. Addison-Wesley, 2017.
- Eckel, Bruce. Thinking in Java. Prentice Hall, 2006.
- Freeman, Eric, and Elisabeth Robson. Head First Java. O'Reilly Media, 2005.
- Horstmann, Cay S. Core Java Volume I--Fundamentals. Prentice Hall, 2018.
- Lea, Doug. Concurrent Programming in Java: Design Principles and Patterns. Addison-Wesley, 2000.

Website:

- Baeldung. "Java Tutorials and Articles." Baeldung, 2023, https://www.baeldung.com/java-tutorials.
- GeeksforGeeks. "Java Programming Language." GeeksforGeeks, 2023, https://www.geeksforgeeks.org/java/.
- Stack Overflow. "Questions tagged [java]." Stack Overflow, https://stackoverflow.com/questions/tagged/java.
- Tutorials Point. "Java Tutorial." Tutorials Point, 2023, https://www.tutorialspoint.com/java/index.htm.
- Vogella. "Java Tutorials." Vogella, 2023, https://www.vogella.com/tutorials/java.html

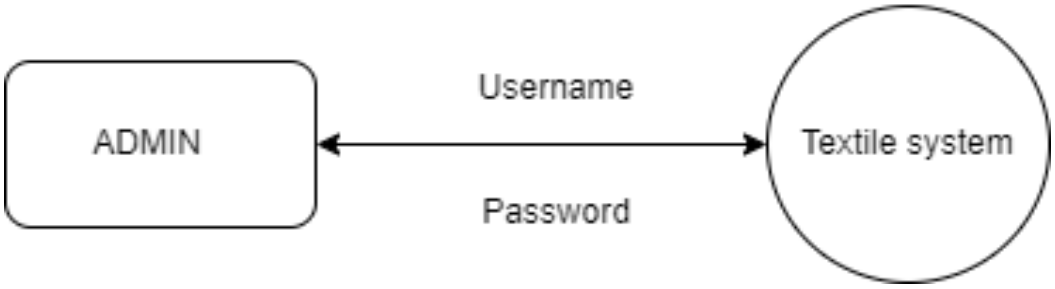
APPENDICES

APPENDICES

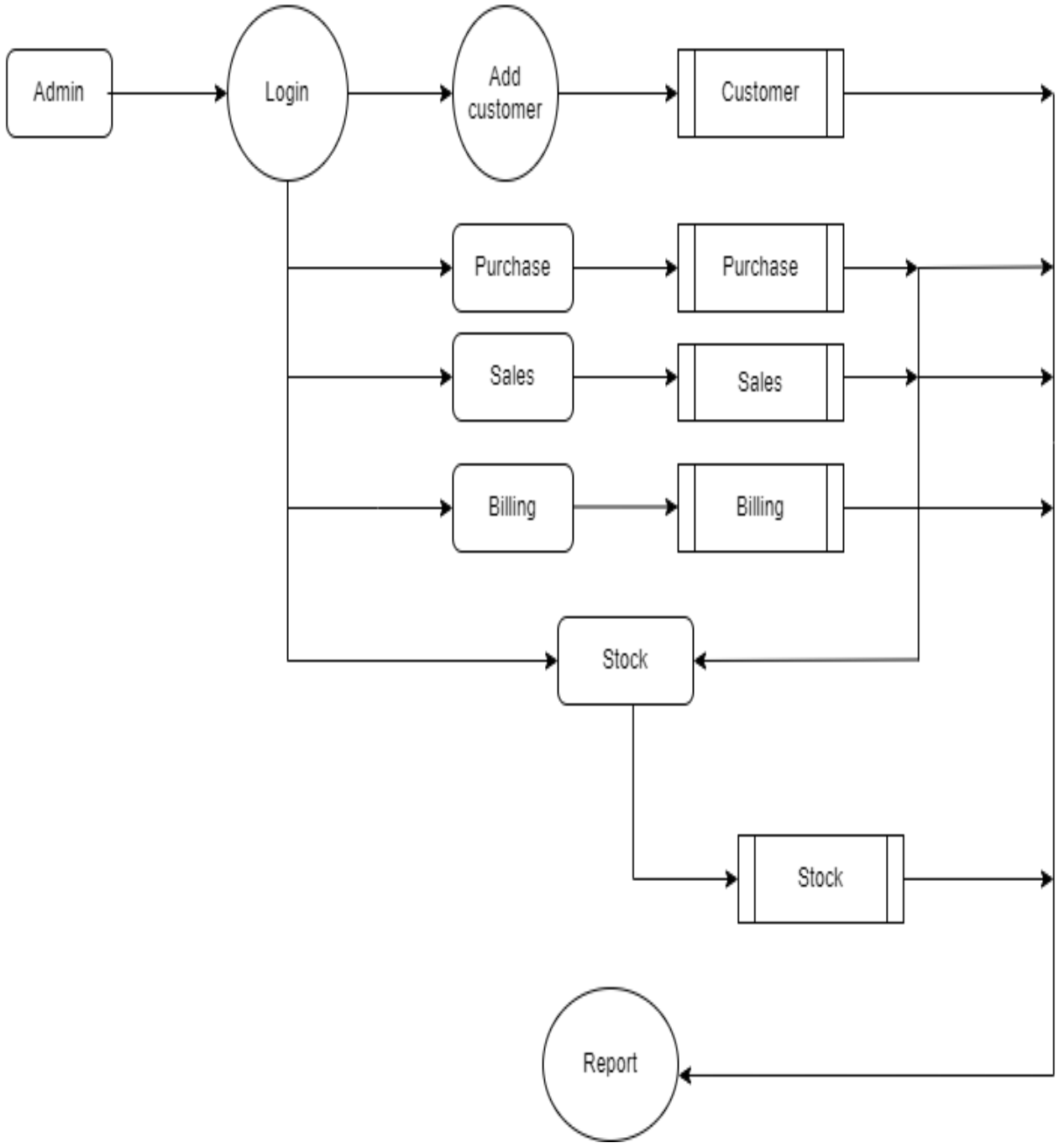
A. DATA FLOW DIAGRAM

A data-flow diagram (DFD)is a way of representing a flow of a data of a process or system. The DFD also provides information about the outputs and inputs of each entity and process itself. A data-flow diagram is a part of structured-analysis modeling tools.

LEVEL 0



LEVEL 1:


Admin
Login
Add customer
Customer
Purchase
Purchase
Sales
Sales
Billing
Billing
Stock
Stock
Report

B. TABLE STRUCTURE

The table needed for each module was designed and the specification of each and every column was given based on the records and details collected during record specification of the system study.

TABLE NAME: ADMIN

FIELD	DATA TYPE	SIZE	CONSTRAINT
Adminid	Int	10	Primary key
Username	Varchar	20	Not null
password	Varchar	20	Not null

TABLE NAME: CUSTOMER

FIELD	DATA TYPE	SIZE	CONSTRAINT
Customer id	Int	10	Primary key
Customer Name	Varchar	20	Not null
Mobile number	Int	10	Not null
Address	Varchar	30	Not null
Pin code	Int	6	Not null

TABLE NAME: PRODUCT

FIELD	DATA TYPE	SIZE	CONSTAINT
Product id	Int	10	Primary key
Company name	Varchar	20	Not null
Brand	Varchar	20	Not null
Product name	Varchar	20	Not null
Price	Int	7	Not null

TABLE NAME: PURCHASE

FIELD	DATA TYPE	SIZE	CONSTRAINT
Purchase id	Int	10	Primary key
Product name	Varchar	20	Not null
Quantity	Int	10	Not null
Company name	Varchar	20	Not null
Brand	Varchar	20	Not null
Price	Int	7	Not null

TABLE NAME: SALES

FIELD	DATA TYPE	SIZE	CONSTRAINT
Sales id	Int	10	Primary key
Customer id	Int	10	Foreign key
Mobile number	Int	10	Not null
Customer name	Varchar	20	Not null
Address	Varchar	20	Not null
Pin code	Int	6	Not null
Product name	Varchar	20	Not null
Quantity	Int	10	Not null
Company name	Varchar	20	Not null
Brand	Varchar	20	Not null
Price	Int	7	Not null
Total amount	Int	7	Not null

TABLE NAME: BILLING

FIELD	DATA TYPE	SIZE	CONSTRAINT
Date	Date/time	DD/MM/YY	Not null
Customer name	varchar	20	Not null
Company name	varchar	20	Not null
Brand	Varchar	20	Not null
Product name	Varchar	20	Not null
Price	Int	7	Not null
Quantity	Int	10	Not null
Total amount	Int	7	Not null
Billing id	Int	30	Not null

TABLE NAME: STOCK

FIELD	DATA TYPE	SIZE	CONSTRAINT
Stock id	Int	10	Foreign key
Company name	varchar	20	Not null
Brand name	varchar	10	Not null
Price	Int	7	Not null
Stock	Int	20	Not null

C.SAMPLE CODEING

```
import org.springframework.web.bind.annotation.*;
import org.springframework.beans.factory.annotation.*;
import org.springframework.jdbc.core.*;
import org.springframework.jdbc.core.namedparam.*;
import javax.sql.*;
import java.util.*;

@RestController
Public class MyController{

   @Autowired
   private NamedParameterJdbcTemplate jdbcTemplate;

   @PostMapping("/addCustomer")
   public String addCustomer(@RequestBody Map<String, String> req) {
      String name = req.get("name");
      String mobile = req.get("mobile");
      String address = req.get("address");
      String pincode = req.get("pincode");
```

```
        String insert_customer = "INSERT INTO `customer` (`id`, `name`, `mobile`, `address`, `pincode`) VALUES (NULL, :name, :mobile, :address, :pincode)";
        Map<String, Object> paramMap = new HashMap<>();
        paramMap.put("name", name);
        paramMap.put("mobile", mobile);
        paramMap.put("address", address);
        paramMap.put("pincode", pincode);
        jdbcTemplate.update(insert_customer, paramMap);

        return "Customer has been created";
    }

    @GetMapping("/viewCustomer")
    public List<Map<String, Object>> viewCustomer() {
        String select_customer = "SELECT * FROM customer";
        List<Map<String, Object>> customers = jdbcTemplate.queryForList(select_customer);
        return customers;
    }

    @PostMapping("/addProduct")
```

```
    public String addProduct(@RequestBody Map<String, String> req) {
        String company = req.get("company");
        String brand = req.get("brand");
        String product = req.get("product");
        String price = req.get("price");

        String insert_product = "INSERT INTO `product` (`id`, `company`, `brand`, `product`, `price`) VALUES (NULL, :company, :brand, :product, :price)";
        Map<String, Object> paramMap = new HashMap<>();
        paramMap.put("company", company);
        paramMap.put("brand", brand);
        paramMap.put("product", product);
        paramMap.put("price", price);
        jdbcTemplate.update(insert_product, paramMap);

        return "Product has been created";
    }

    @GetMapping("/viewProduct")
    public List<Map<String, Object>> viewProduct() {
        String select_product = "SELECT * FROM product";
```

```
    List<Map<String, Object>> products = jdbcTemplate.queryForList(select_product);
    return products;
  }

  @GetMapping("/productDetails/{id}")
  public Map<String, Object> productDetails(@PathVariable int id) {
    String select_product = "SELECT * FROM product WHERE id = :id";
    Map<String, Object> paramMap = new HashMap<>();
    paramMap.put("id", id);
    Map<String, Object> product = jdbcTemplate.queryForMap(select_product, paramMap);
    return product;
  }

  @PostMapping("/addPurchase")
  public String addPurchase(@RequestBody Map<String, String> req) {
    String pid = req.get("pid");
    String quantity = req.get("quantity");

    String query = "INSERT INTO `purchase` (`id`, `pid`, `quantity`) VALUES (NULL, :pid, :quantity)";
    Map<String, Object> paramMap = new HashMap<>();
```

```
        paramMap.put("pid", pid);
        paramMap.put("quantity", quantity);
        jdbcTemplate.update(query, paramMap);

        String query1 = "SELECT COUNT(*) AS count, quantity FROM stock WHERE pid = :pid";
        Map<String, Object> paramMap1 = new HashMap<>();
        paramMap1.put("pid", pid);
        List<Map<String, Object>> result = jdbcTemplate.queryForList(query1, paramMap1);

        String query3;
        if (result.get(0).get("count").
}

// Material Design example
$(document).ready(function () {
    update();
});
function update() {
    showCustomerList();
    showProductList();
    showBillings();
```

```
    showStock();
}
function addCustomer() {
    var name = $("#name1").val();
    var mobile = $("#mobile1").val();
    var address = $("#address1").val();
    var pincode = $("#pincode1").val();
    $.ajax({
        type: 'POST',
        url: '/api/addCustomer',
        data: {
            name: name,
            mobile: mobile,
            address: address,
            pincode: pincode,
        }
    }).done(function (datas) {
        alert(datas);
        $("#name1").val("");
        $("#mobile1").val("");
        $("#address1").val("");
```

```
        $("#pincode1").val("");
        update();
    });
}
function showCustomerList() {
    $.ajax({
        type: 'GET',
        url: '/api/viewCustomer',
    }).done(function (datas) {
        var html = ``;
        for (var i in datas) {
            var data = datas[i];
            html += ` <tr>
            <th scope="row">` + (++i) + `</th>
                <td>` + data.name + `</td>
                <td>` + data.mobile + `</td>
            <td>` + data.address + `</td>
            <td>` + data.pincode + `</td>
            </tr>`;
        }
        $("#customerbody").html(html)
```

```
    });
}
function addProduct() {
    var company = $("#company2").val();
    var brand = $("#brand2").val();
    var product = $("#productname2").val();
    var price = $("#price2").val();
    $.ajax({
        type: 'POST',
        url: '/api/addProduct',
        data: {
            company: company,
            brand: brand,
            product: product,
            price: price,
        }
    }).done(function (datas) {
        alert(datas);
        $("#company2").val("");
        $("#brand2").val("");
        $("#productname2").val("");
```

```
        $("#price2").val("");
        update();
    });
}
function showProductList() {
    $.ajax({
        type: 'GET',
        url: '/api/viewProduct',
    }).done(function (datas) {
        var html = ``;
        var opt = ``;
        for (var i in datas) {
            var data = datas[i];
            opt += `<option id='` + data.id + `'>` + data.product + `</option>`;
            html += ` <tr>
            <th scope="row">` + data.id + `</th>
                <td>` + data.company + `</td>
                <td>` + data.brand + `</td>
            <td>` + data.product + `</td>
            <td>` + data.price + `</td>
            </tr>`;
```

```
        }
        $("#productbody").html(html)
        $("#product3").html(opt)
        $("#product4").html(opt)
    });
}
function loadProductDetails() {
    var id = $("#product3").children(":selected").attr("id");
    $.ajax({
        type: 'GET',
        url: '/api/productDetails/' + id,
    }).done(function (datas) {
        var data = datas[0];
        $("#company3").val(data.company)
        $("#brand3").val(data.brand)
        $("#price3").val(data.price)
    });
}
function addPurchase() {
    var id = $("#product3").children(":selected").attr("id");
    var quantity = $("#quantity3").val();
```

```
    $.ajax({
        type: 'POST',
        url: '/api/addPurchase',
        data: {
            pid: id,
            quantity: quantity,
        }
    }).done(function (datas) {
        alert(JSON.stringify(datas));
        $("#company2").val("");
        $("#brand2").val("");
        $("#productname2").val("");
        $("#price2").val("");
        update();
    });
}
function getCustomerDetails() {
    var mobile = $("#mobile4").val();
    $.ajax({
        type: 'GET',
        url: '/api/getCustomer/' + mobile,
```

```
    }).done(function (datas) {
        var data = datas[0];
        $("#name4").val(data.name);
        $("#address4").val(data.address);
        $("#pincode4").val(data.pincode);
    });
}
function loadProductDetails4() {
    var id = $("#product4").children(":selected").attr("id");
    $.ajax({
        type: 'GET',
        url: '/api/productDetails/' + id,
    }).done(function (datas) {
        var data = datas[0];
        $("#company4").val(data.company)
        $("#brand4").val(data.brand)
        $("#price4").val(data.price)
        $("#quantity4").val("")
        $("#total4").val("")
    });
}
```

```
function totalamount() {
    var price = $("#price4").val();
    var quantity = $("#quantity4").val();
    $("#total4").val((quantity - 0) * (price - 0))
}
function addSales() {
    var mobile = $("#mobile4").val();
    var pid = $("#product4").children(":selected").attr("id");
    var quantity = $("#quantity4").val();
    var total = $("#total4").val();
    $.ajax({
        type: 'POST',
        url: '/api/addSales',
        data: {
            mobile: mobile,
            pid: pid,
            quantity: quantity,
            total: total,
        }
    }).done(function (datas) {
        alert(JSON.stringify(datas));
```

```
        $("#company2").val("");
        $("#brand2").val("");
        $("#productname2").val("");
        $("#price2").val("");
        update();
    });
}
function showBillings() {
    $.ajax({
        type: 'GET',
        url: '/api/getBillings',
    }).done(function (datas) {
        var html = ``;
        for (var i in datas) {
            var data = datas[i];
            html += ` <tr>
                        <th scope="row">` + (++i) + `</th>
                        <td>` + data.name + `</td>
                        <td>` + data.company + `</td>
                        <td>` + data.brand + `</td>
                        <td>` + data.product + `</td>
```

```
                        <td>$` + data.price + `</td>
                        <td>` + data.quantity + `</td>
                        <td>$` + data.total + `</td>
                    </tr>`;
        }
        $("#billingBody").html(html)
    });
}
function showStock() {
    $.ajax({
        type: 'GET',
        url: '/api/getStocks',
    }).done(function (datas) {
        var html = ``;
        for (var i in datas) {
            var data = datas[i];
            html += `<tr>
                        <th scope="row">`+(++i)+`</th>
                        <td>`+data.company+`</td>
                        <td>`+data.brand+`</td>
                        <td>`+data.product+`</td>
```

```
                    <td>$`+data.price+`</td>
                    <td>`+data.quantity+`</td>
                </tr>
                `;
        }
        $("#stockBody").html(html)
    });
}
import org.springframework.web.bind.annotation.*;
import org.springframework.beans.factory.annotation.*;
import org.springframework.jdbc.core.*;
import org.springframework.jdbc.core.namedparam.*;
import javax.sql.*;
import java.util.*;

@RestController
public class MyController {

    @Autowired
    private NamedParameterJdbcTemplate jdbcTemplate;
```

```
@PostMapping("/addCustomer")
public String addCustomer(@RequestBody Map<String, String> req) {
    String name = req.get("name");
    String mobile = req.get("mobile");
    String address = req.get("address");
    String pincode = req.get("pincode");

    String insert_customer = "INSERT INTO `customer` (`id`, `name`, `mobile`, `address`, `pincode`) VALUES (NULL, :name, :mobile, :address, :pincode)";
    Map<String, Object> paramMap = new HashMap<>();
    paramMap.put("name", name);
    paramMap.put("mobile", mobile);
    paramMap.put("address", address);
    paramMap.put("pincode", pincode);
    jdbcTemplate.update(insert_customer, paramMap);

    return "Customer has been created";
}

@GetMapping("/viewCustomer")
public List<Map<String, Object>> viewCustomer() {
```

```
    String select_customer = "SELECT * FROM customer";
    List<Map<String, Object>> customers = jdbcTemplate.queryForList(select_customer);
    return customers;
  }

  @PostMapping("/addProduct")
  public String addProduct(@RequestBody Map<String, String> req) {
    String company = req.get("company");
    String brand = req.get("brand");
    String product = req.get("product");
    String price = req.get("price");

    String insert_product = "INSERT INTO `product` (`id`, `company`, `brand`, `product`, `price`) VALUES (NULL, :company, :brand, :product, :price)";
    Map<String, Object> paramMap = new HashMap<>();
    paramMap.put("company", company);
    paramMap.put("brand", brand);
    paramMap.put("product", product);
    paramMap.put("price", price);
    jdbcTemplate.update(insert_product, paramMap);
```

```
    return "Product has been created";
}

@GetMapping("/viewProduct")
public List<Map<String, Object>> viewProduct() {
    String select_product = "SELECT * FROM product";
    List<Map<String, Object>> products = jdbcTemplate.queryForList(select_product);
    return products;
}

@GetMapping("/productDetails/{id}")
public Map<String, Object> productDetails(@PathVariable int id) {
    String select_product = "SELECT * FROM product WHERE id = :id";
    Map<String, Object> paramMap = new HashMap<>();
    paramMap.put("id", id);
    Map<String, Object> product = jdbcTemplate.queryForMap(select_product, paramMap);
    return product;
}

@PostMapping("/addPurchase")
public String addPurchase(@RequestBody Map<String, String> req) {
```

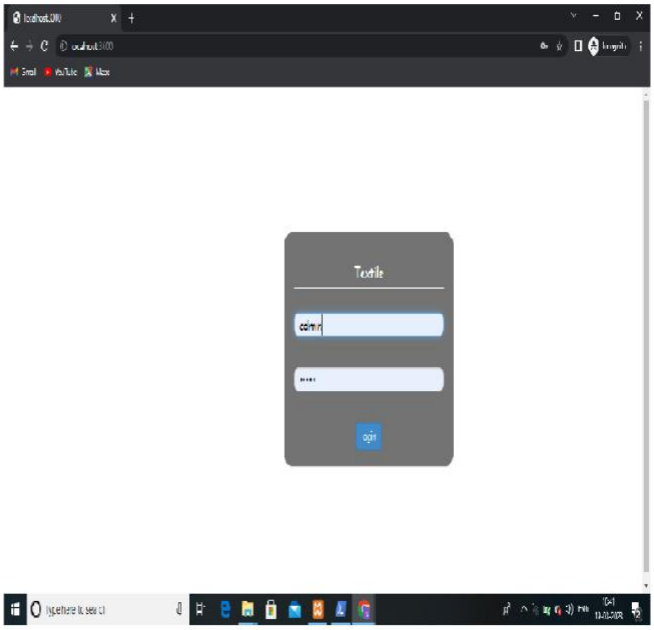
```
        String pid = req.get("pid");
        String quantity = req.get("quantity");

        String query = "INSERT INTO `purchase` (`id`, `pid`, `quantity`) VALUES (NULL, :pid, :quantity)";
        Map<String, Object> paramMap = new HashMap<>();
        paramMap.put("pid", pid);
        paramMap.put("quantity", quantity);
        jdbcTemplate.update(query, paramMap);

        String query1 = "SELECT COUNT(*) AS count, quantity FROM stock WHERE pid = :pid";
        Map<String, Object> paramMap1 = new HashMap<>();
        paramMap1.put("pid", pid);
        List<Map<String, Object>> result = jdbcTemplate.queryForList(query1, paramMap1);

        String query3;
        if (result.get(0).get("count").
}
@RestController
public class PurchaseController {
```

```
    @Autowired
    private JdbcTemplate jdbcTemplate;

    @PostMapping("/addPurchase")
    public ResponseEntity<String> addPurchase(@RequestBody PurchaseRequest purchaseRequest) {
        String pid = purchaseRequest.getPid();
        String quantity = purchaseRequest.getQuantity();

        String insertQuery = "INSERT INTO purchase (pid, quantity) VALUES (?, ?)";
        jdbcTemplate.update(insertQuery, pid, quantity);

        String countQuery = "SELECT COUNT(*) AS count, quantity FROM stock WHERE pid=?";
        List<Map<String, Object>> result = jdbcTemplate.queryForList(countQuery, pid);
        String query3 = "";

        if (result.get(0).get("count").equals(0L)) {
            query3 = "INSERT INTO stock (pid, quantity) VALUES (?, ?)";
            jdbcTemplate.update(query3, pid, quantity);
        } else {
            Integer oldQuantity = (Integer) result.get(0).get("quantity");
```

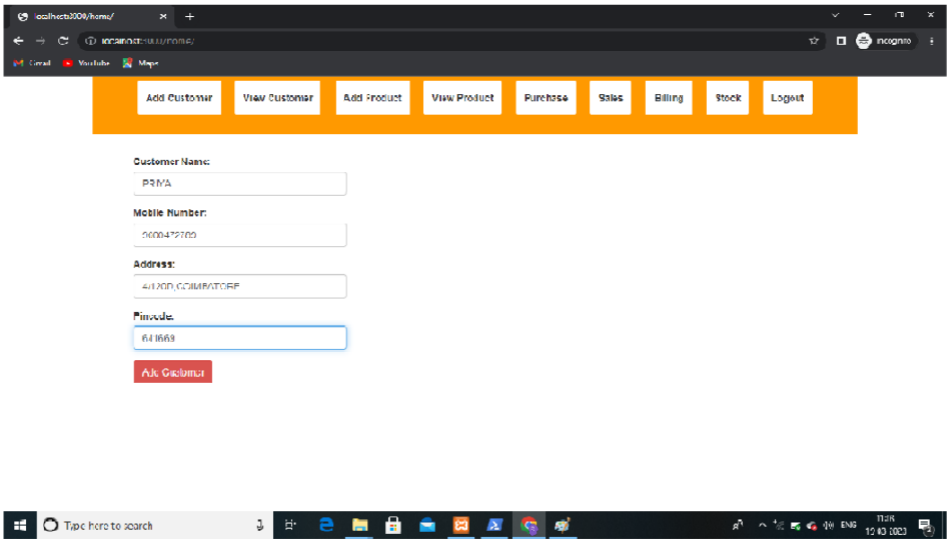
```
        Integer newQuantity = oldQuantity + Integer.parseInt(quantity);
        query3 = "UPDATE stock SET quantity=? WHERE pid=?";
        jdbcTemplate.update(query3, newQuantity, pid);
    }
    return ResponseEntity.ok("Purchase successfully added");
}

@GetMapping("/getCustomer/{mobile}")
public ResponseEntity<Customer> getCustomer(@PathVariable String mobile) {
    String query = "SELECT * FROM customer WHERE mobile=?";
    List<Customer> customers = jdbcTemplate.query(query, new Object[]{mobile},
            (rs, rowNum) -> new Customer(rs.getInt("id"), rs.getString("name"),
                    rs.getString("mobile"), rs.getString("email")));
    if (customers.isEmpty()) {
        return ResponseEntity.notFound().build();
    }
    return ResponseEntity.ok(customers.get(0));
}
```

D.SAMPLE INPUT & E. OUTPUT DESIGN

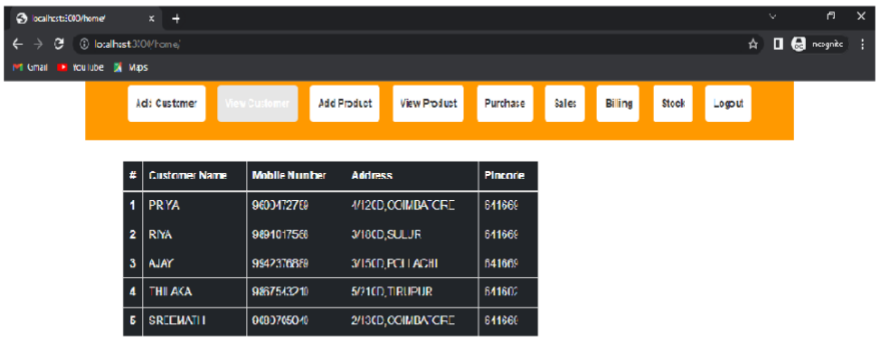
ADMIN REGISTRATION :



ADD CUSTOMER:

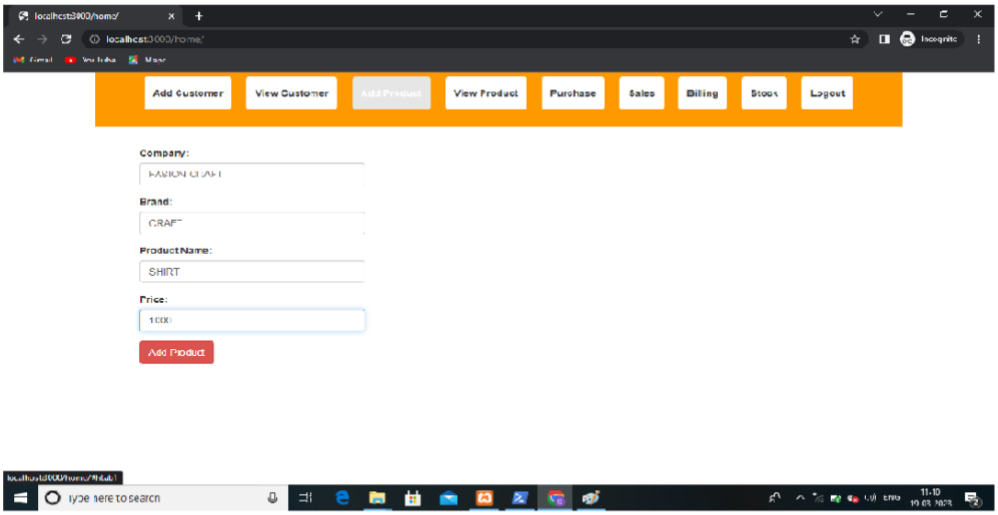


VIEW CUSTOMER:

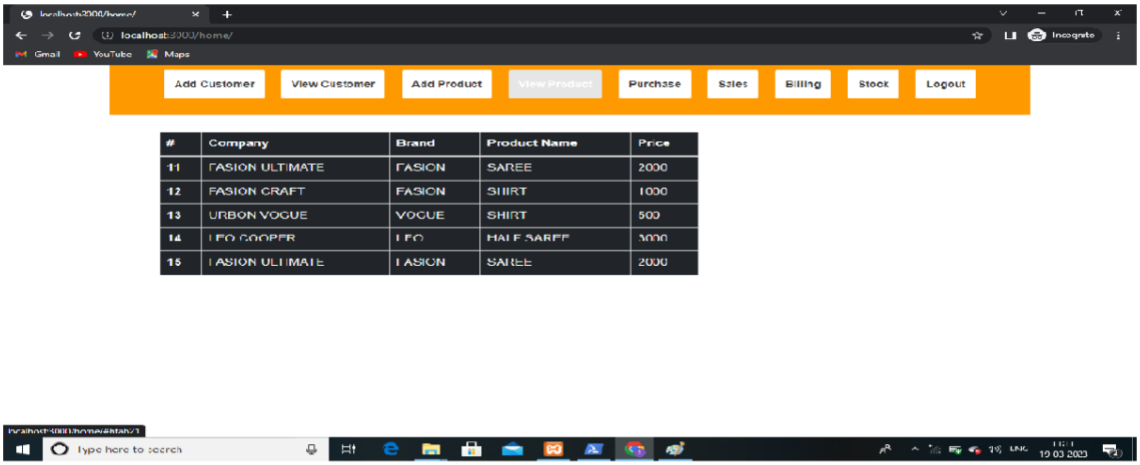


#	Customer Name	Mobile Number	Address	Pincode
1	PRIYA	9600472709	4/12CD,CCIM/BATCRE	641669
2	RIYA	9894017568	3/10CD,SULUR	641669
3	AJAY	9942376859	3/15CD,PCI I ACHI	641669
4	THILAKA	9865543210	5/21CD,TIRUPUR	641602
5	SREEMATI	9600766040	2/13CD,CCIM/BATCRE	641669

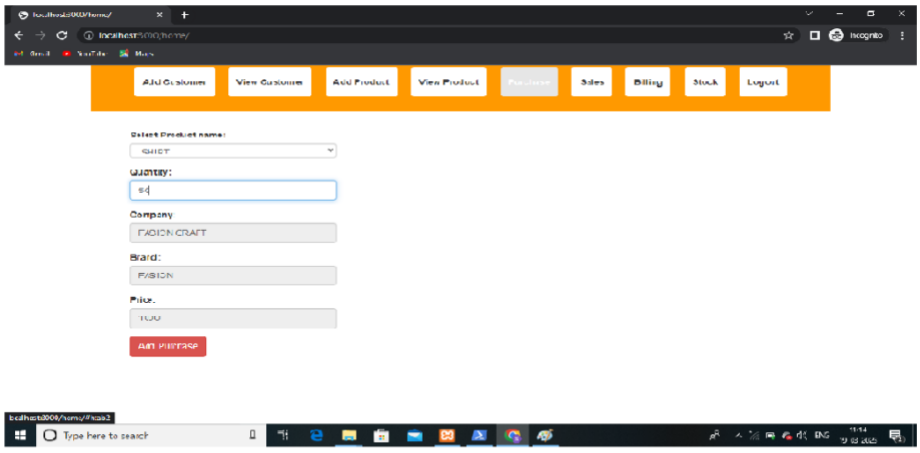
ADD PRODUCT:


Add Customer
View Customer
Add Product
View Product
Purchase
Sales
Billing
Stock
Logout
Company:
Brand:
CRAFT
Product Name:
SHIRT
Price:
1000
Add Product

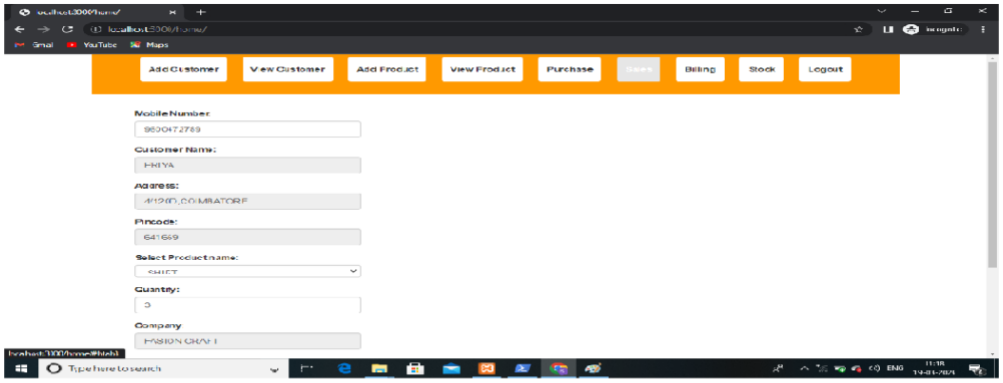
VIEW PRODUCT:

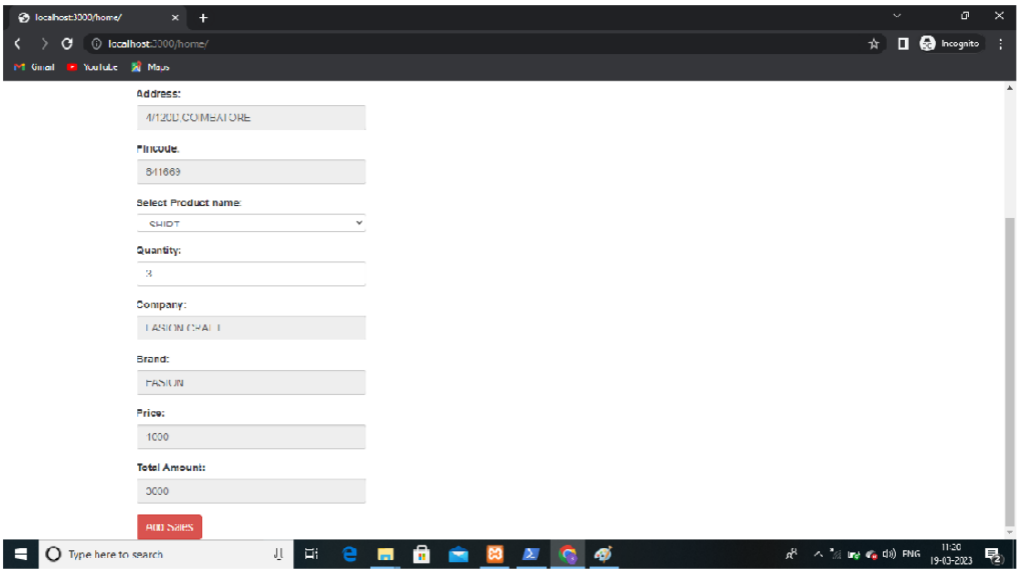


PURCHASE:

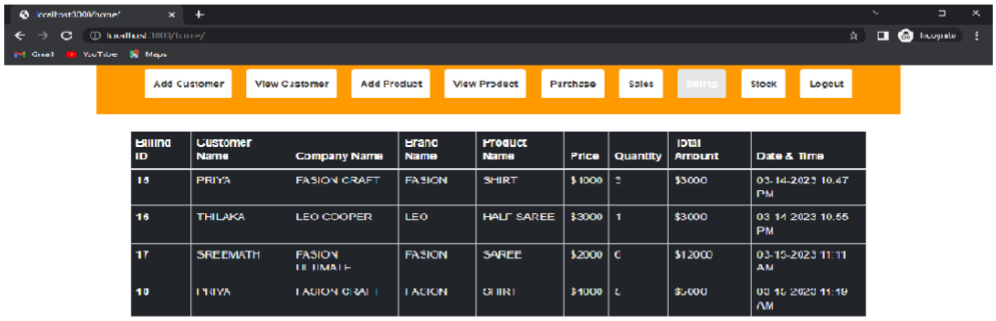


SALES:


Add Customer
View Customer
Add Product
View Product
Purchase
Sales
Billing
Stock
Logout
Mobile Number:
Customer Name:
Address:
Pincode:
Select Product name:
Quantity:
Company:

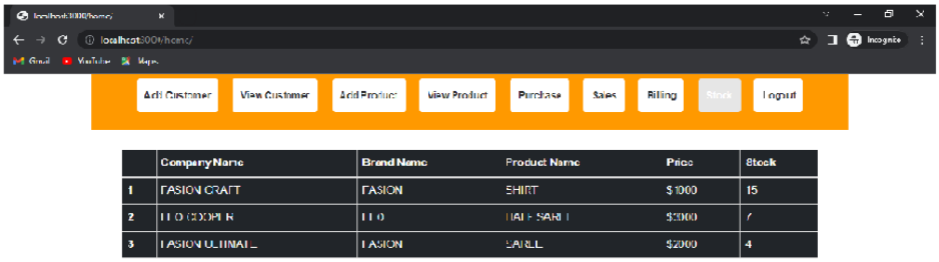

localhost:3000/home/
Incognito
Address:
Pincode:
641669
Select Product name:
SHIRT
Quantity:
3
Company:
Brand:
FASHION
Price:
1000
Total Amount:
3000
ADD Sales
Type here to search
ENG
11:30
19-03-2023

BILLING:



Billing ID	Customer Name	Company Name	Brand Name	Product Name	Price	Quantity	Total Amount	Date & Time
15	PRIYA	FASION CRAFT	FASION	SHIRT	$ 1000	3	$3000	03-14-2023 10:47 PM
16	THILAKA	LEO COOPER	LEO	HALF SAREE	$3000	1	$3000	03-14-2023 10:55 PM
17	SREEMATHI	FASION	FASION	SAREE	$2000	6	$12000	03-15-2023 11:11 AM
18	PRIYA	FASION CRAFT	FASION	SHIRT	$1000	2	$2000	03-15-2023 11:19 AM

STOCK:


Add Customer
View Customer
Add Product
View Product
Purchase
Sales
Billing
Stock
Logout
Company Name
Brand Name
Product Name
Price
Stock
1
FASION CRAFT
FASION
SHIRT
$1000
15
2
3
FASION ULTIMATE
FASION
SAREE
$2000
4


Type here to search