**A**

**PROJECT REPORT**

**ON**

**“LOGISTICS SYSTEM”**

*Submitted in partial fulfillment of the requirements for the award of*

**DIPLOMA IN ENGINEERING**

**In**

**COMPUTER SCIENCE & ENGINEERING**

Submitted to

**RAJIV GANDHI PROUDYOGIKI VISHWAVIDHYALAYA**

**BHOPAL (M.P)**

By

**DEEPAK KUMAR CHOUHAN (16014C04013)**

Under the guidance of

**Mr. Rishi Yadav**

****

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**GOVT. POLYTECHNIC COLLEGE DAMOH (M.P.)**

**Session 2020-2021**

**GOVT POLYtechnic COLLEGE DAMOH (M.P.)**

**CERTIFICATE**

***This is to certify that MAHENDRA BURMAN, DEEPAK KUMAR , PANKAJ VISHWAKARMA , RAHUL VISHWAKARMA, RAJU SAHU*** ***students of Final year(*Computer Science& Engineering*)of this institute have completed project of* “(LOGISTICS SYSTEM)”*which has been submitted in partial fulfillment of the requirements for the award of Diploma in Engineering (*Computer Science & Engineering*) from Rajiv Gandhi Proudyogiki vishwavidhalaya Bhopal (M.P.).***

**Mr. Rishi Yadav Mr. Rishi Yadav Dr. Lata Tripathi**

**Project Guide Head of Department CSE Principal**

**GPC DAMOH (MP) GPC DAMOH (MP) GPC DAMOH(MP)**

**GOVT POLYtechnic COLLEGE DAMOH (M.P.)**

**CERTIFICATE**

**CERTIFICATE**

***This is to certify that* DEEPAK KUMAR** ***students of Final year (Computer Science& Engineering)of this institute have completed project of* “(**LOGISTICS SYSTEM)**” *which has been submitted in partial fulfillment of the requirements for the award of Diploma in Engineering (*Computer Science & Engineering*) from Rajiv Gandhi Proudyogiki vishwavidhalaya Bhopal (M.P.).***

**InternalExaminer External Examiner**

**GOVT POLYtechnic COLLEGE DAMOH (M.P.)**

****

**COPUTER SCIENCE & ENGINEERING**

**DECLARATION**

*We,* DEEPAK KUMAR ***students of diploma in Engineering,* Computer Science & Engineering *Branch* Govt. Polytechnic College Damoh*hereby declare that the work presented in the Major project entitled* “ (LOGISTICS SYSTEM ) ” *is outcome of our own work, is corrected to the best of our knowledge and this work has been carried out taking care of Engineering Ethics. The work presented does not infringe any patented work and has not been submitted to any University for the award of professional diploma.***

**DEEPAK KUMAR CHOUHAN (16014C04013)**

**\**

**ACKNOWLEDGEMENT**

***It gives us immense pleasure to express our hearty gratitude to our project guide and Head of the Department*(Mr. Rishi Yadav) *of* Computer Science & Engineering Govt. Polytechnic College Damoh (M.P.) *for providing invaluable guidance and constant encouragement throughout the period of this work.***

***It’s our privilege to have worked under his esteemed guidance without his all round support it would not have been possible to complete the work within the specified time. His insight in to the subject and critical suggestion always gave us simpler ways of dealing with problems.***

***We are also thankful to* Dr. Lata Tripathi*, Principal, Govt. Polytechnic College Damoh providing all necessary facilities during the course of this project.***

***We are also thankful to* Mr. B. D. Aharwal (Technical Support) and all faculties of CS department *Govt. Polytechnic College Damoh providing all necessary facilities during the course of this project.***

**DEEPAK KUMAR CHOUHAN (16014C04013)**

**Abstract**

For logistics systems automation of logistics activities, the change in planning scope and planning capabilities are relevant drivers. Automation especially can be found within warehouses; this reduces variable storage costs but increases the necessary investments. The tendency for warehouse structures can be different. An example of changes in planning scope is growing logistics systems like the replacement of national by European distribution systems. This may lead to new centres of gravity of demand and supply within the logistics system in space and thus to new locations. Also, a better usage of the logistic system could be the consequence which would result in fewer warehouses. The planning capabilities today change through more powerful IT systems and the availability of data. This can increase the efficiency of all processes and may influence all costs, transport costs can be lower because of better usage or handling costs can be reduced through improved processes in the warehouse, the overall tendency for the distribution system is difficult to predict.

Third-party logistics (3PL), a relatively new industry, has gained momentum since the emergence of global market and the Internet, in particular electronic commerce (e-commerce). Global competitive-ness places more pressure on companies to improve their delivery performance of products and services to customers. In an effort to improve the quality of delivery service, companies have outsourced their logistics services, including packing, warehousing (inventory management) and shipping of goods to customers. Communication plays an important role in integrating the activities along the logistics value chain. Information technologies such as electronic data interchange (EDI), the Internet, World Wide Web (WWW) and e-commerce have contributed greatly to improving communication with partners in the logistics chain. In particular, real-time information systems such as web-based logistics information systems help to improve 3PL services. In this paper, a case study of e-logistics is used to illustrate the implications of information technology, in particular the Internet, WWW and EDI, on the performance of the logistics value chain.

**TABLE OF CONTENT**

|  |  |  |  |
| --- | --- | --- | --- |
| **CHAPTERhhhv** Chapterzz  **S.NO.** | **Context** | | **PAGE NO** |
| 1. | **INTRODUCTION 8** | | |
| **1.1 OBJECTIVE** | **8** | |
| 2. | **TECHNOLOGY USED 9** | | |
| **2.1FRONT END** | **9** | |
| **2.2BACK END** | **12** | |
| 3. | **SYSTEM REQUIREMENT &SPECIFICATION 16** | | |
| **3.1HARDWARE** | **16** | |
| **3.2SOFTWARE** | **16** | |
| 4. | **SOFTWARE DEVELOPMENT LIFE CYCLE** | **17** | |
| 5. | **SYSTEM DESIGN 17** | | |
| **DATAFLOW DIAGRAM** | **20** | |
| **TABLE DEFINITION** | **22** | |
| 6. | **CODING** | **24** | |
| 7. | **TESTING** | **29** | |
| 8. | **SCREEN SHOT** | **32** | |
| 9. | **CONCLUSION** | **35** | |
| 10. | **FUTURE ENHANCEMENT** | **36** | |
| 11. | **BIBLIOGRAPHY & REFERENCE** | **37** | |

1. **INTRODUCTION**

A logistics system (LS) is a network of organizations, people, activities, information, and resources involved in the physical flow of products from supplier to customer. An LS may consist of three main networks or subsystems:

1. Procurement: The acquisition of raw material and parts from suppliers and their transportation to the manufacturing plants.
2. Production: The transformation of the raw materials into finished products.
3. Distribution: The transportation of finished products from plants to a network of stocking locations (warehouses) and from there to end users.

In a digital world, to manage these processes in both ways, businesses use logistics management systems – a combination of software tools that optimize all processes from making an order and delivering it to a customer’s door. Inventory management is a vital part of the supply chain responsible for controlling and documenting the amount of product for sale. Receiving, storing, and tracking inventory, while dealing with its rapid and constant changes, requires highly accurate product information management.

* **1.1 Objectives**

In the modern era, the technology boom and the complexity of logistics processes have spawned logistics management software and specialized logistics-focused firms that expedite the movement of resources along the supply chain. One reason large online retailers like Amazon have come to dominate the retail landscape is the overall innovation and efficiency of their logistics along every link of the supply chain.

Manufacturing companies may choose to outsource the management of their logistics to specialists or manage logistics internally if it is cost-effective to do so.

The system that delivers goods quickly and on time from the location of production to the consumer is the logistics flow shown below. Logistics also includes elements such as storing and packaging goods in addition to carrying and moving those goods.

Information systems that perform centralized management of the logistics flow from ordering to picking, shipping, and delivery are now essential to the world of logistics that handles thousands of items each day. This is because information systems are deeply involved in the traceability of goods that allows one to know where products that have been shipped from a production location or factory currently are and how long they will take to reach the assigned destination. IT devices such as handheld computers that link to the internet and intranet are used in these systems.

**2.TECHNOLOGY USED**

**2.1 Frontend design**:

 **HyperText Markup language (HTML)-**

HyperText Markup Language is the backbone of any website development process, without which a web page doesn't exist. It is the HTML code that provides an overall framework of how the site will look. The latest version of HTML is called HTML5 which has new and efficient way of handling elements such as video and audio files.

Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. With Cascading Style Sheets (CSS) and JavaScript, it forms a triad of cornerstone technologies for the World Wide Web.[4] Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML is the standard markup language for creating Web pages.

* HTML stands for Hyper Text Markup Language
* HTML describes the structure of Web pages using markup
* HTML elements are the building blocks of HTML pages
* HTML elements are represented by tags
* HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
* Browsers do not display the HTML tags, but use them to render the content of the page

**Sass CSS (SCSS**)-

Cascading Style Sheets (CSS) controls the presentation aspect of the site and allows your site to have its own unique look.

It does this by maintaining style sheets which sit on top of other style rules and are triggered based on other inputs, such as device screen size and resolution.

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS on its own can be fun, but stylesheets are getting larger, more complex, and harder to maintain. This is where a preprocessor can help. Sass has features that don't exist in CSS yet like nesting, mixins, inheritance, and other nifty goodies that help you write robust, maintainable CSS.

**SCSS vs CSS SYNTAX:-**

****

**A SCSS has-**

* **Partials**
* **Mixins**
* **Modules**
* **Inheritance**
* **Operators**

The selector points to the HTML element you want to style. The declaration block contains one or more declarations separated by semicolons. Each declaration includes a CSS property name and a value, separated by a colon. A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

In the following example all <p> elements will be center-aligned, with a red text color.

** JavaScript-**

JavaScript is an event-based imperative programming language (as opposed to HTML's declarative language model) that is used to transform a static HTML page into a dynamic interface. JavaScript code can use the Document Object Model (DOM), provided by the HTML standard, to manipulate a web page in response to events, like user input.

**Angular (Typescript)-**

Angular is an application design framework and development platform for creating efficient and sophisticated single-page apps. Angular is a development platform, built on TypeScript. As a platform, Angular includes:

A component-based framework for building scalable web applications

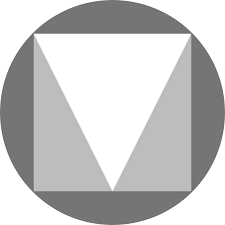
A collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more

A suite of developer tools to help you develop, build, test, and update your code

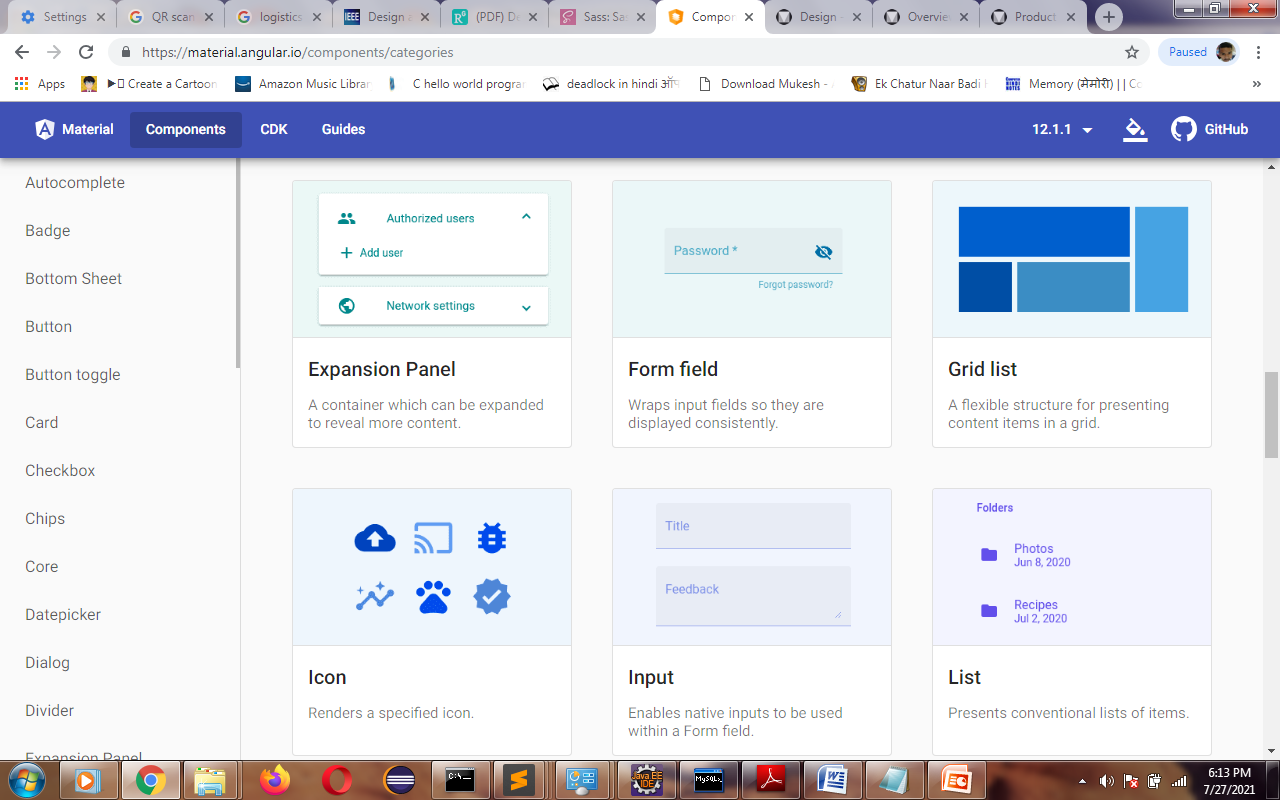
With Angular, you're taking advantage of a platform that can scale from single-developer projects to enterprise-level applications. Angular is designed to make updating as straightforward as possible, so take advantage of the latest developments with a minimum of effort. Best of all, the Angular ecosystem consists of a diverse group of over 1.7 million developers, library authors, and content creators.

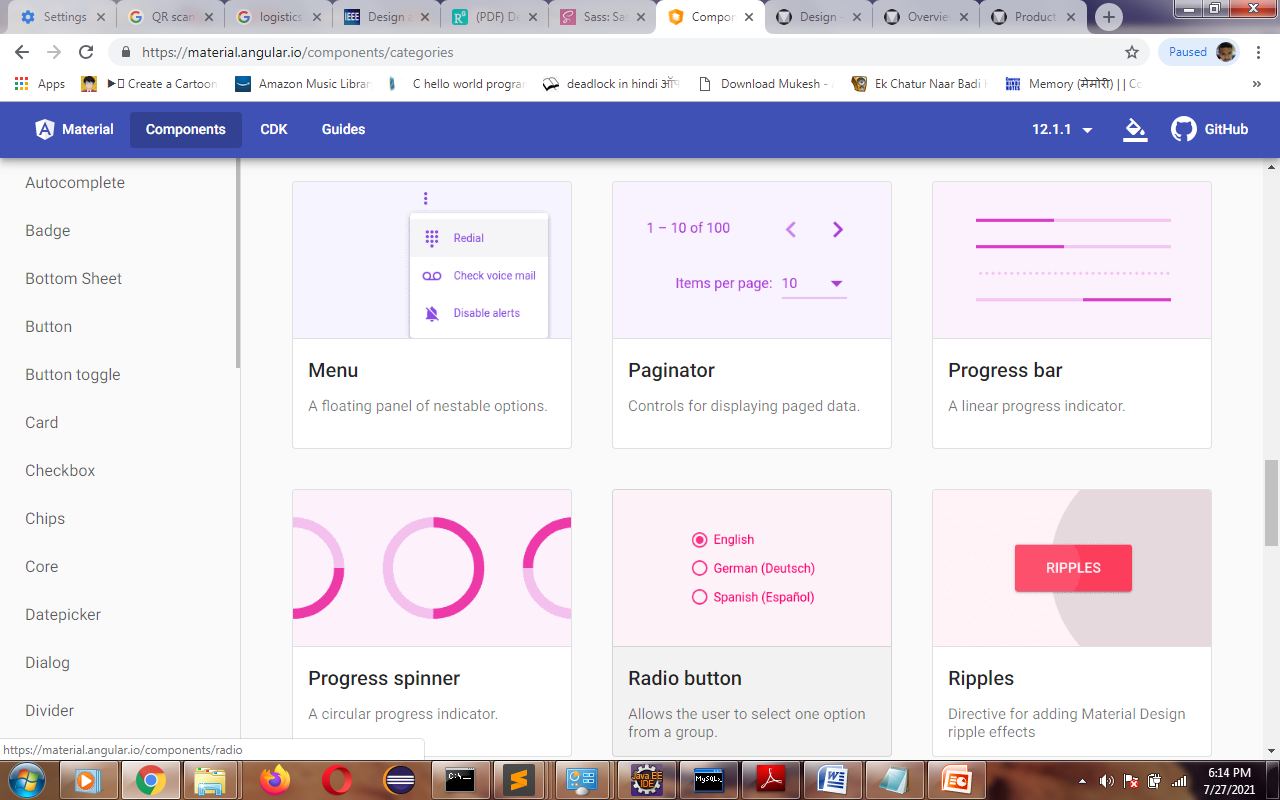
Angular applications Modules-

* Component
* Template
* Service
* Dependency Injection

**Material Design(Angular Material UI)-**

Material is an adaptable system of guidelines, components, and tools that support the best practices of user interface design. Backed by open-source code, Material streamlines collaboration between designers and developers, and helps teams quickly build beautiful products.Angular Material offers a wide variety of UI components based on the Material Design specification-





** NodeJS-**

As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications.

This is in contrast to today's more common concurrency model, in which OS threads are employed. Thread-based networking is relatively inefficient and very difficult to use. Furthermore, users of Node.js are free from worries of dead-locking the process, since there are no locks. Almost no function in Node.js directly performs I/O, so the process never blocks. Because nothing blocks, scalable systems are very reasonable to develop in Node.js.

**2.2 Backend design:**

**Java -**

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA),meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub. particularly for client-server web applications, with a reported 9 million developers.

Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle) and released in 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GPL-2.0-only license. Oracle offers its own HotSpot Java Virtual Machine, however the official reference implementation is the OpenJDK JVM which is free open source software and used by most developers and is the default JVM for almost all Linux distributions.

Java JVM and bytecode

One design goal of Java is portability, which means that programs written for the Java platform must run similarly on any combination of hardware and operating system with adequate run time support. This is achieved by compiling the Java language code to an intermediate representation called Java bytecode, instead of directly to architecture-specific machine code. Java bytecode instructions are analogous to machine code, but they are intended to be executed by a virtual machine (VM) written specifically for the host hardware. End users commonly use a Java Runtime Environment (JRE) installed on their machine for standalone Java applications, or in a web browser for Java applets.

Standard libraries provide a generic way to access host-specific features such as graphics, threading, and networking.

The use of universal bytecode makes porting simple. However, the overhead of interpreting bytecode into machine instructions made interpreted programs almost always run more slowly than native executables. Just-in-time (JIT) compilers that compile byte-codes to machine code during runtime were introduced from an early stage. Java itself is platform-independent and is adapted to the particular platform it is to run on by a Java virtual machine (JVM) for it, which translates the Java bytecode into the platform's machine language.

**Spring Framework-**

The Spring Framework provides a comprehensive programming and configuration model for modern Java-based enterprise applications - on any kind of deployment platform.

The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE (Enterprise Edition) platform. Although the framework does not impose any specific programming model, it has become popular in the Java community as an addition to the Enterprise JavaBeans (EJB) model. The Spring Framework is open source.

** MySQL database-**

**MySQL** is an open-sourcerelational database management system (RDBMS) in July 2013, it was the world's second mostwidely used RDBMS, and the most widely used open-source client–server model RDBMS. It is named after co-founder Michael Widenius's daughter, My. The SQLacronym stands for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a singlefor-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

more distinct APIs for creating, accessing, managing, searching and replicating the data it holds. Other A database is a separate application that stores a collection of data. Each database has one or kinds of data stores can also be used, such as files on the file system or large hash tables in memory, but data fetching and writing would not be so fast and easy with those type of systems. Nowadays, we use relational database management systems (RDBMS) to store and manage huge volume of data. This is called relational database because all the data is stored into different tables and relations are established using primary keys or other keys known as Foreign Keys.

A Relational DataBase Management System (RDBMS) is a software that:

* Enables you to implement a database with tables, columns and indexes.
* Guarantees the Referential Integrity between rows of various tables.
* Updates the indexes automatically.
* Interprets an SQL Query and combines information from various tables

package, unzip it anywhere and run the setup.exe file. The default installer setup.exe will walk you through the trivial process and by default will install everything under C:\mysql. Test the server by firing it up from the command prompt the first time. Go to the location of the mysqld server which is probably C:\mysql\bin, and type:

**MySQL – Datatypes**

Properly defining the fields in a table is important to the overall optimization of your database. You should use only the type and size of field you really need to use. For example, do not define a field 10 characters wide, if you know you are only going to use 2 characters. These type of fields (or columns) are also referred to as data types, after the type of data you will be storing in those fields.

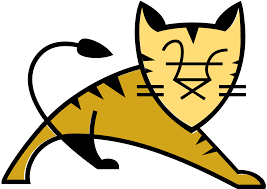
***MySQL uses many different data types broken into three categories:***

* Numeric
* Date and Time
* String Types

***Let us now discuss them in detail.***

**Numeric Data Types**

MySQL uses all the standard ANSI SQL numeric data types, so if you're coming to MySQL from a different database system, these definitions will look familiar to you. The following list shows the common numeric data types and their descriptions.

**Tomcat server-**

Tomcat is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. Most commonly used on a Unix-like system (usually Linux), the software is available for a wide variety of operating systems, besides Unix and GNU+Linux, including eComStation, Microsoft Windows, NetWare, OpenVMS, OS/2, and TPF. Released under the Apache License, Apache isfree and open-source software***.***

**API**

An application programming interface (API) is a connection between computers or between computer programs. It is a type of software interface, offering a service to other pieces of software. A document or standard that describes how to build such a connection or interface is called an API specification. A computer system that meets this standard is said to implement or expose an API. The term API may refer either to the specification or to the implementation.

In contrast to a user interface, which connects a computer to a person, an application programming interface connects computers or pieces of software to each other. It is not intended to be used directly by a person (the end user) other than a computer programmer who is incorporating it into software. An API is often made up of different parts which act as tools or services that are available to the programmer. A program or a programmer that uses one of these parts is said to call that portion of the API. The calls that make up the API are also known as subroutines, methods, requests, or endpoints. An API specification defines these calls, meaning that it explains how to use or implement them.

One purpose of APIs is to hide the internal details of how a system works, exposing only those parts a programmer will find useful and keeping them consistent even if the internal details later change. An API may be custom-built for a particular pair of systems, or it may be a shared standard allowing interoperability among many systems.

**3.SYSTEM REQUIREMENT &SPECIFICATYION**

**3.1 HARDWARE REQUIREMENTS:**

|  |  |  |
| --- | --- | --- |
| **SNO** | **SECTION** | **REQUIREMENTS** |
| 1. | OPERATING SYSTEM | Windows, Linux, Mac or Unix OS |
| 2. | Processor | Any x86 architeture 32bit or 64bit Processor |
| 3. | Memory | 512MB |
| 4. | Hard disk required | 200MB |

**3.2 SOFTWARE REQUIREMENTS:**

|  |  |  |
| --- | --- | --- |
| **SNO** | **SECTION** | **REQUIREMENT** |
| 1. | IDE/Text editor | Eclipse IDE with Notepad or sublime |
| 2. | Web browser | Any Browser with ES5 or later compiling Engine |
| 3. | Web server | Tomcat/JBoss, Node, MySQL |

**4. SOFTWARE DEVELOPMENT LIFE CYCLE**

The intent of a SDLC process it to help produce a product that is cost-efficient, effective, and of high quality. Software life cycle models describe phases of the software cycle and the order in which those phases are executed. Each phase produces deliverables required by the next phase in the life cycle. The following are the six steps include planning, analysis, design, development & implementation, testing & deployment and maintenance. Let’s study each of these steps to know how the perfect software is developed.

****

**FIG 4.1 : SOFTWARE DEVELOPMENT LIFE CYCLE**

* **1.Planning**:

 First and important phase planning of SDLC for the success of software. This phase includes communication between project stakeholders, end users and project team, as requirements (both Functional & non functional) are being gathered from clients. Without the perfect plan, calculating the strengths and weaknesses of the project, development of software is meaningless. Planning kicks off a project flawlessly and affects its progress positively.

* **2.Analysis:**

This step is about analyzing the performance of the software at various stages and making notes on additional requirements. Analysis is very important to proceed further to the next step

* **3.Design:**

 Once the analysis is complete, the step of designing takes over, which is basically building the architecture of the project. This step helps remove possible flaws by setting a standard and attempting to stick to it.

* **4.Development & Implementation:**

The actual task of developing the software starts here with data recording going on in the background. Once the software is developed, the stage of implementation comes in where the product goes through a pilot study to see if it’s functioning properly.

* **5.Testing:**

The testing stage assesses the software for errors and documents bugs if there are any.system testing is the executing software in a controlling manner , in order to answer the question “does the software behave as specified”.

* **6.Deployment and Maintenance:**

Once the software passes through all the stages without any issues, it is to undergo a maintenance process wherein it will be maintained and upgraded from time to time to adapt to changes. Almost every software development Indian company follows all the six steps, leading to the reputation that the country enjoys in the software market today.

**5. SYSTEM DESIGN & ANALYSIS:**

Design is the first step into the development phase for any engineered product or system. Design is a creative process. A good design is the key to effective system. The term “design” is defined as “the process of applying various techniques and principles for the purpose of defining a process or a system in sufficient detail to permit its physical realization”. It may be defined as a process of applying various techniques and principles for the purpose of defining a device, a process or a system in sufficient detail to permit its physical realization.

* **LOGICAL DESIGN:**

The logical flow of a system and define the boundaries of a system. It includes the following steps:

* Reviews the current physical system – its data flows, file content, volumes, Frequencies etc.
* Prepares output specifications – that is, determines the format, content and Frequency of reports.
* **PHYSICAL DESIGN:**

Physical system produces the working systems by define the design specifications that tell the programmers exactly what the candidate system must do. It includes the following steps.

* Design the physical system.
* Specify input and output media.
* Design the database and specify backup procedures.
* Design physical information flow through the system and a physical design.
* **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.
* **OUTPUT DESIGN:**
* A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user.
* Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively.
* **DATA FLOW DIAGRAM :**

A Data Flow Diagram (DFD) is a structured analysis and design tool that can be used for flowcharting. A DFD is a network that describes the flow of data and the processes that change or transform the data throughout a system. This network is constructed by using a set of symbols that do not imply any physical implementation. It has the purpose of clarifying system

requirements and identifying major transformations. So it is the starting point of the design phase that functionally decomposes the requirements specifications down to the lowest level of detail. DFD can be considered to an abstraction of the logic of an information-oriented or a process-oriented system flow-chart. For these reasons DFD’s are often referred to as logical data flow diagrams.

* **EXTERNAL ENTITY**

An external entity is a source or destination of a data flow. Only those entities which originate or receive data are represented on a data flow diagram. The symbol used is a rectangular box.

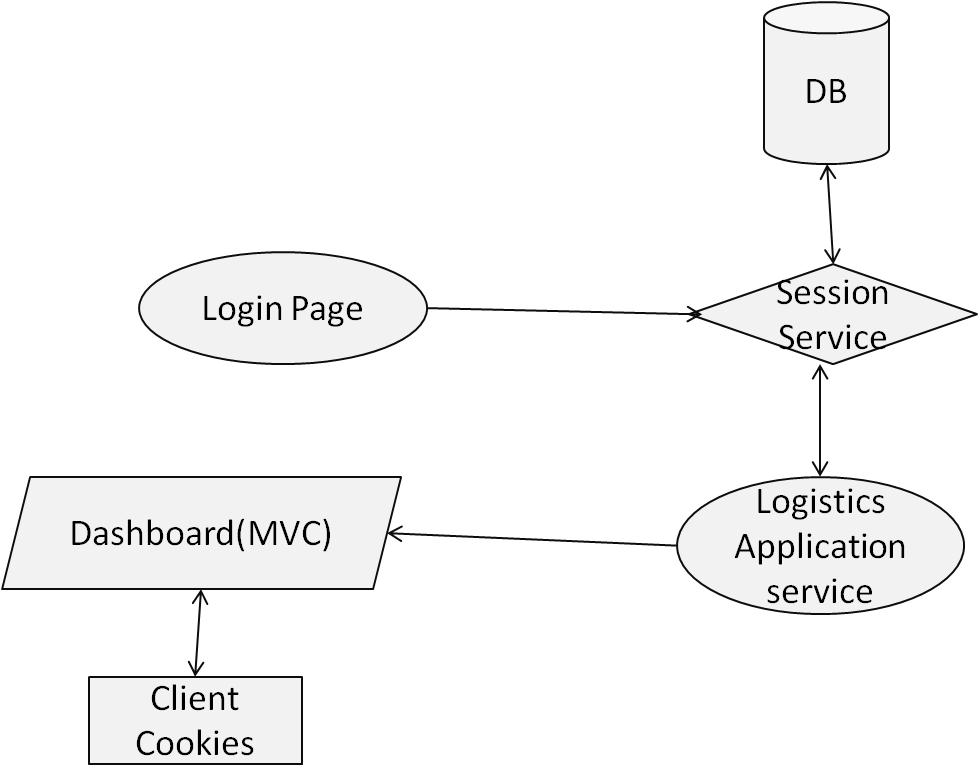
* **PROCESS**

A process shows a transformation or manipulation of data flow within the system. The symbol used is an oval shape.

The data flow shows the flow of information from a source to its destination. Data flow is represented by a line, with arrowheads showing the direction of flow. Information always flows to or from a process and may be written, verbal or electronic. Each data flow may be referenced by the processes or data stores at its head and tail, or by a description of its contents.

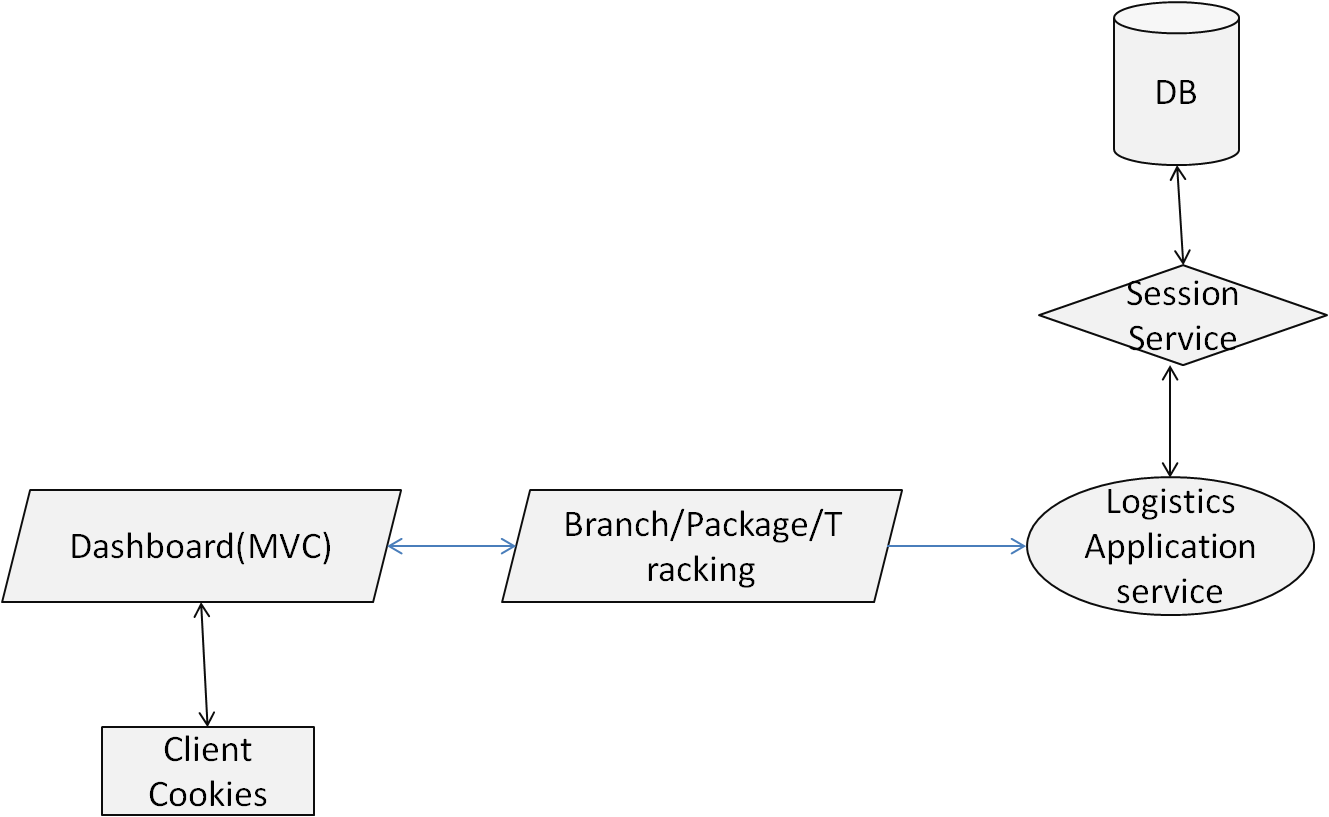
* **DATA STORE**

A data store is a holding place for information within the system: It is represented by an open ended narrow rectangle. Data stores may be long-term files such as sales ledgers, or may be short-term accumulations: for example batches of documents that are waiting to be processed. Each data store should be given a reference followed by an arbitrary number.

* **LOGIN DFD:** 

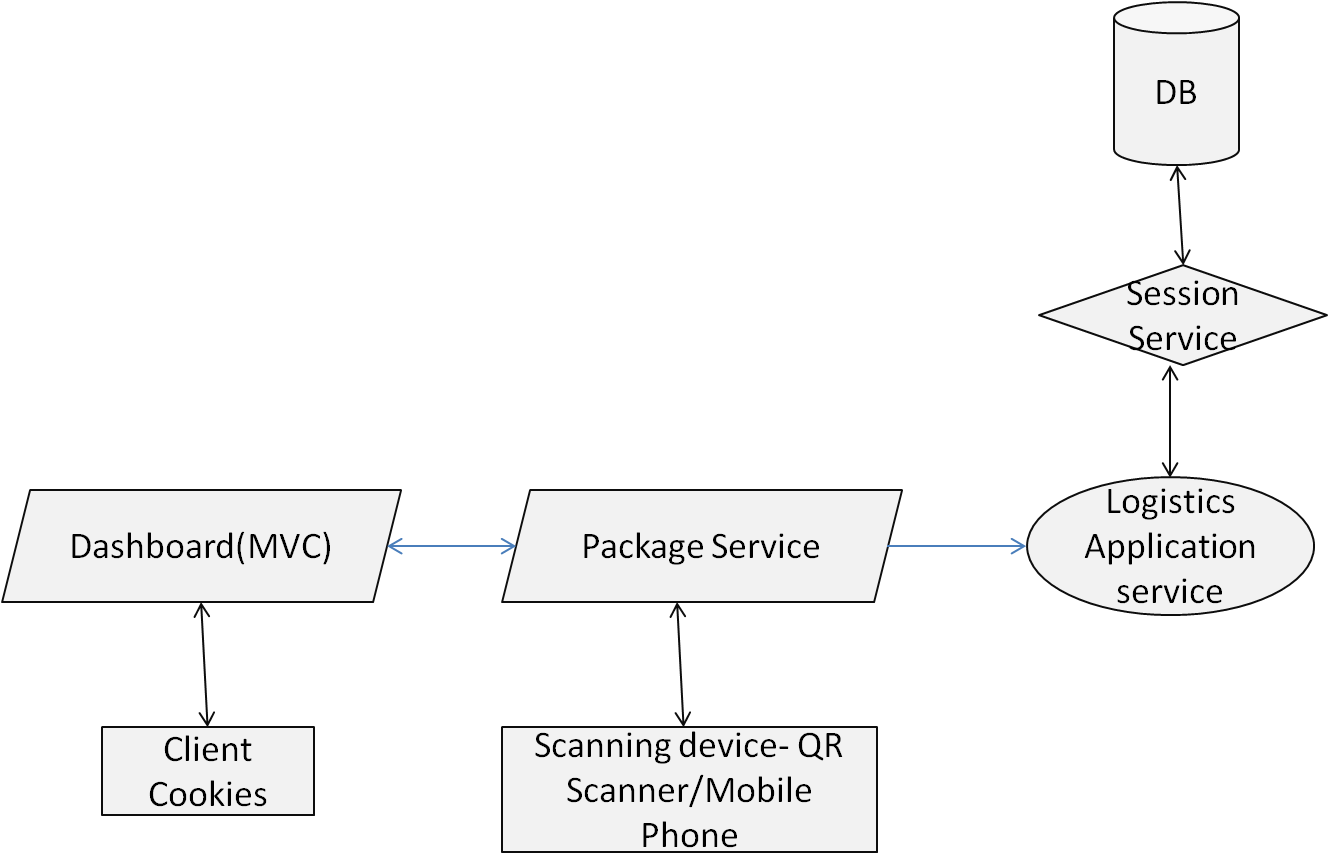
**FIG 5.1 : DFD For Login Page**

* **MODULES(Branch/Package/Tracking) DFD:**

****

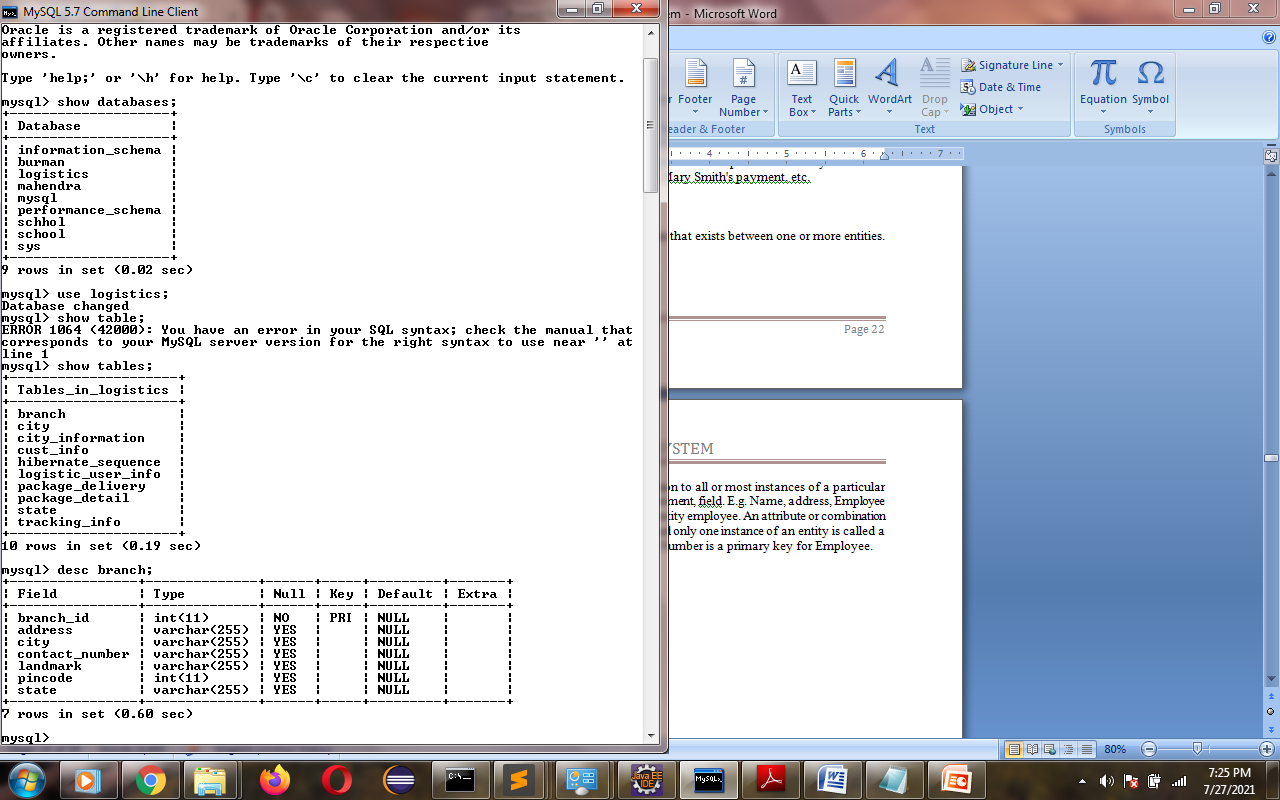
**FIG 5.2 : DFD For Modules pages**

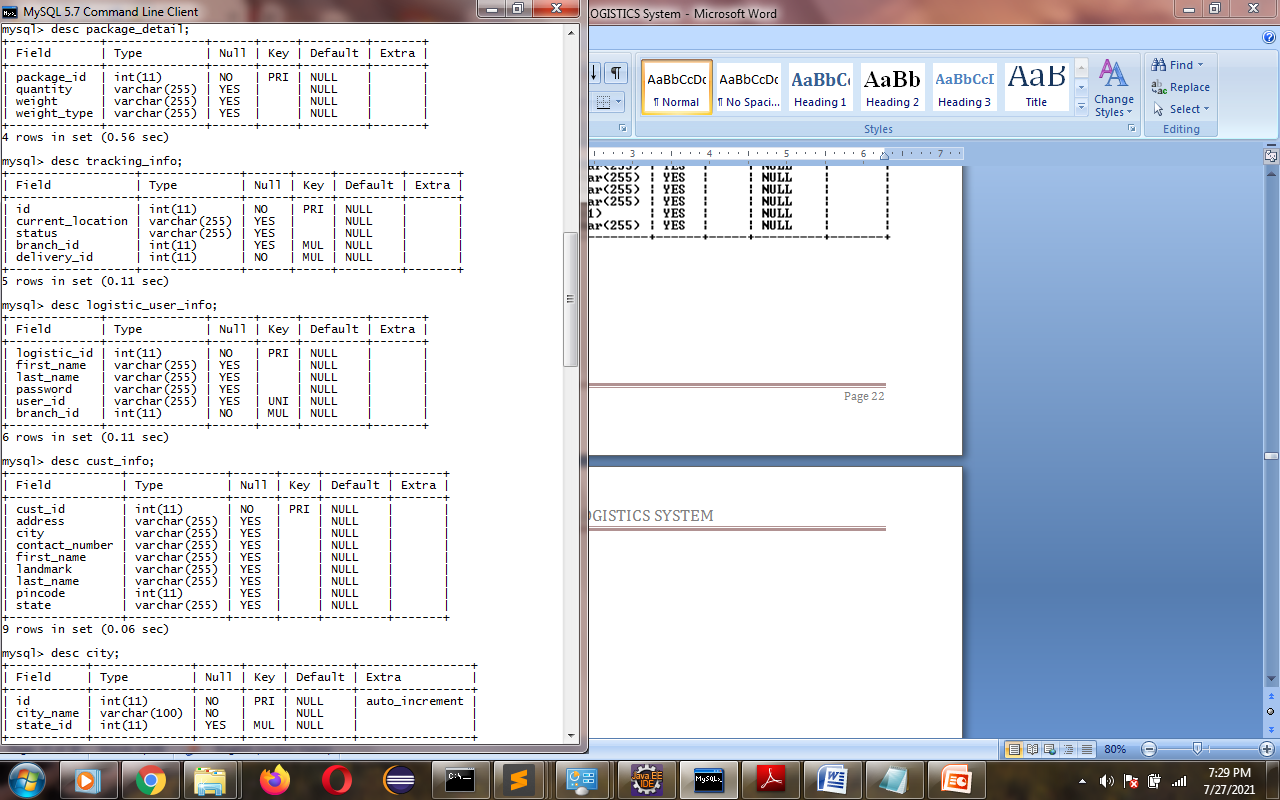
* **QR Generation and Scanning DFD:**

****

**FIG 5.3 : DFD For QR Scanning and Generation**

**Tables-**

****

****

**6.CODING:**

**DTO Class-**

package com.mahendra.ppt.logistic.dto;

import java.io.Serializable;

import java.util.Date;

import java.util.Set;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.OneToMany;

import javax.persistence.OneToOne;

import javax.persistence.Table;

@Entity

@Table(name = "package\_delivery")

public class PackageDelivery implements Serializable{

@Id

@GeneratedValue(strategy = GenerationType.AUTO)

@Column(name = "delivery\_id")

private Integer deliveryId;

@OneToOne(cascade = CascadeType.ALL)

@JoinColumn(name = "customer\_from", referencedColumnName = "cust\_id")

private CustInfo pkgAddressFrom;

@OneToOne(cascade = CascadeType.ALL)

@JoinColumn(name = "customer\_to", referencedColumnName = "cust\_id")

private CustInfo pkgAddressTo;

@OneToOne(cascade = CascadeType.ALL)

@JoinColumn(name = "logistic\_id", referencedColumnName = "logistic\_id")

private LogisticUserInfo logisticUserInfo;

@OneToOne(cascade = CascadeType.ALL)

@JoinColumn(name = "package\_id", referencedColumnName = "package\_id")

private PackageDetails packageDetails;

@Column(name = "package\_start\_date")

private Date startDate;

@Column(name = "package\_end\_date")

private Date endDate;

@OneToMany(mappedBy = "delivery", cascade = CascadeType.ALL)

private Set<Tracking> tracking;

public Integer getDeliveryId() {

return deliveryId;

}

public void setDeliveryId(Integer deliveryId) {

this.deliveryId = deliveryId;

}

public CustInfo getPkgAddressFrom() {

return pkgAddressFrom;

}

public void setPkgAddressFrom(CustInfo pkgAddressFrom) {

this.pkgAddressFrom = pkgAddressFrom;

}

public CustInfo getPkgAddressTo() {

return pkgAddressTo;

}

public void setPkgAddressTo(CustInfo pkgAddressTo) {

this.pkgAddressTo = pkgAddressTo;

}

public Date getStartDate() {

return startDate;

}

public void setStartDate(Date startDate) {

this.startDate = startDate;

}

public Date getEndDate() {

return endDate;

}

public void setEndDate(Date endDate) {

this.endDate = endDate;

}

public Set<Tracking> getTracking() {

return tracking;

}

public void setTracking(Set<Tracking> tracking) {

this.tracking = tracking;

for(Tracking t : tracking) {

t.setDelivery(this);

}

}

public LogisticUserInfo getLogisticUserInfo() {

return logisticUserInfo;

}

public void setLogisticUserInfo(LogisticUserInfo logisticUserInfo) {

this.logisticUserInfo = logisticUserInfo;

}

public PackageDetails getPackageDetails() {

return packageDetails;

}

public void setPackageDetails(PackageDetails packageDetails) {

this.packageDetails = packageDetails;

}

@Override

public String toString() {

return "PackageDelivery [deliveryId=" + deliveryId

+ ", pkgAddressFrom=" + pkgAddressFrom.toString() + ", pkgAddressTo="

+ pkgAddressTo.toString() + ", logicticUserInfo=" + logisticUserInfo.toString()

+ ", packageDetails=" + packageDetails + ", startDate="

+ startDate + ", endDate=" + endDate + ", tracking=" + tracking.toString()

+ "]";

}

}

**JPA Repository Class-**

package com.mahendra.ppt.logistic.repository;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.data.repository.query.Param;

import org.springframework.stereotype.Repository;

import com.mahendra.ppt.logistic.common.dto.City;

@Repository

public interface CityRepository extends JpaRepository<City, Integer>{

@Query("SELECT c FROM City c WHERE c.stateId = :stateId")

List<City> getCities(

@Param("stateId") Integer stateId);

}

**Controller-**

package com.mahendra.ppt.logistic.controller;

import java.util.List;

import javax.validation.Valid;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.CrossOrigin;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

import com.mahendra.ppt.logistic.dto.Branch;

import com.mahendra.ppt.logistic.dto.LogisticUserInfo;

import com.mahendra.ppt.logistic.error.ResourceNotFoundException;

import com.mahendra.ppt.logistic.error.errors;

import com.mahendra.ppt.logistic.repository.BranchRepository;

import com.mahendra.ppt.logistic.repository.LogisticUserInfoRepository;

@CrossOrigin("\*")

@RestController

@RequestMapping("/api/logisticuser")

public class LogisticUserInfoController {

@Autowired

private LogisticUserInfoRepository logisticUserInfoRepository;

@Autowired

private BranchRepository branchRepository;

@GetMapping("/listall")

public List<LogisticUserInfo> getLogisticUserInfo() {

return logisticUserInfoRepository.findAll();

}

@GetMapping("/find/{id}")

public ResponseEntity<LogisticUserInfo> getLogisticUserInfoById(

@PathVariable(value = "id") Integer LogisticId)

throws ResourceNotFoundException {

LogisticUserInfo logisticUserInfo = logisticUserInfoRepository.findById(LogisticId)

.orElseThrow(

() -> new ResourceNotFoundException(errors.BRANCH

.getMessage()));

return ResponseEntity.ok().body(logisticUserInfo);

}

@PostMapping(consumes="application/json",produces="application/json",path="/add")

public LogisticUserInfo createLogisticUser(@Valid @RequestBody LogisticUserInfo logisticUserInfo) throws ResourceNotFoundException {

System.out.println("BranchId:"+logisticUserInfo.toString());

if(logisticUserInfo.getBranch().getBranchId()!=null){

Branch branch =branchRepository.findById(logisticUserInfo.getBranch().getBranchId()).orElseThrow(() -> new ResourceNotFoundException(errors.BRANCH

.getMessage()));

logisticUserInfo.setBranch(branch);

}

return logisticUserInfoRepository.save(logisticUserInfo);

}

@PostMapping(consumes="application/json",produces="application/json",path="/authorize")

public LogisticUserInfo authorize(@Valid @RequestBody LogisticUserInfo logisticUserInfo) throws ResourceNotFoundException {

System.out.println("logisticuserid:"+logisticUserInfo.getUserId());

LogisticUserInfo detail =logisticUserInfoRepository.authUser(logisticUserInfo.getUserId(),logisticUserInfo.getPassword());

return detail;

}

}

**7.TESTING:**

* **The box approach:-**

Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

* **White box testing:**

White box testing is when the tester has access to the internal data structures and algorithms including the code that implement these.

**Types of white box testing:-**

The following types of white box testing exist:

* API testing (application programming interface) - testing of the application using public and private APIs.
* Code coverage - creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once) .
* Fault injection methods - improving the coverage of a test by introducing faults to test code paths .
* Mutation testing methods .
* Static testing - White box testing includes all static testing .

**Test coverage:-**

White box testing methods can also be used to evaluate the completeness of a test suite that was created with black box testing methods. This allows the software team to examine parts of a system that are rarely tested and ensures that the most important function points have been tested.

Two common forms of code coverage are:

* Function coverage, which reports on functions executed
* Statement coverage, which reports on the number of lines executed to complete the test
* **Black box testing**

Black box testing treats the software as a "black box"—without any knowledge of internal implementation. Black box testing methods include: equivalence partitioning, boundary value analysis, all-pairs testing, fuzz testing, model-based testing, exploratory testing and specification-based testing.

* **Specification-based testing:**

Specification-based testing aims to test the functionality of software according to the applicable requirements. Thus, the tester inputs data into, and only sees the output from, the test object. This level of testing usually requires thorough test cases to be provided to the tester, who then can simply verify that for a given input, the output value (or behavior), either "is" or "is not" the same as the expected value specified in the test case.

Specification-based testing is necessary, but it is insufficient to guard against certain risks.

* **Advantages and disadvantages:**

The black box tester has no "bonds" with the code, and a tester's perception is very simple: a code must have bugs. Using the principle, "Ask and you shall receive," black box testers find bugs where programmers do not. On the other hand, black box testing has been said to be "like a walk in a dark labyrinth without a flashlight," because the tester doesn't know how the software being tested was actually constructed. As a result, there are situations when (1) a tester writes many test cases to check something that could have been tested by only one test case, and/or (2) some parts of the back-end are not tested at all.

* **Grey box testing**

**Grey box testing** (American spelling: **gray box testing**) involves having knowledge of internal data structures and algorithms for purposes of designing the test cases, but testing at the user, or black-box level. Manipulating input data and formatting output do not qualify as grey box, because the input and output are clearly outside of the "black-box" that we are calling the system under test.

* **Testing levels:-**

Tests are frequently grouped by where they are added in the software development process, or by the level of specificity of the test. The main levels during the development process as defined by the SWEBOK guide are unit-, integration-, and system testing that are distinguished by the test target without implying a specific process model. Other test levels are classified by the testing objective.

* **Unit testing**

**Unit testing** refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to assure that the building blocks the software uses work independently of each other.

Unit testing is also called component testing.

* **Integration testing**

**Integration testing** is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be localized more quickly and fixed.

* **System testing**

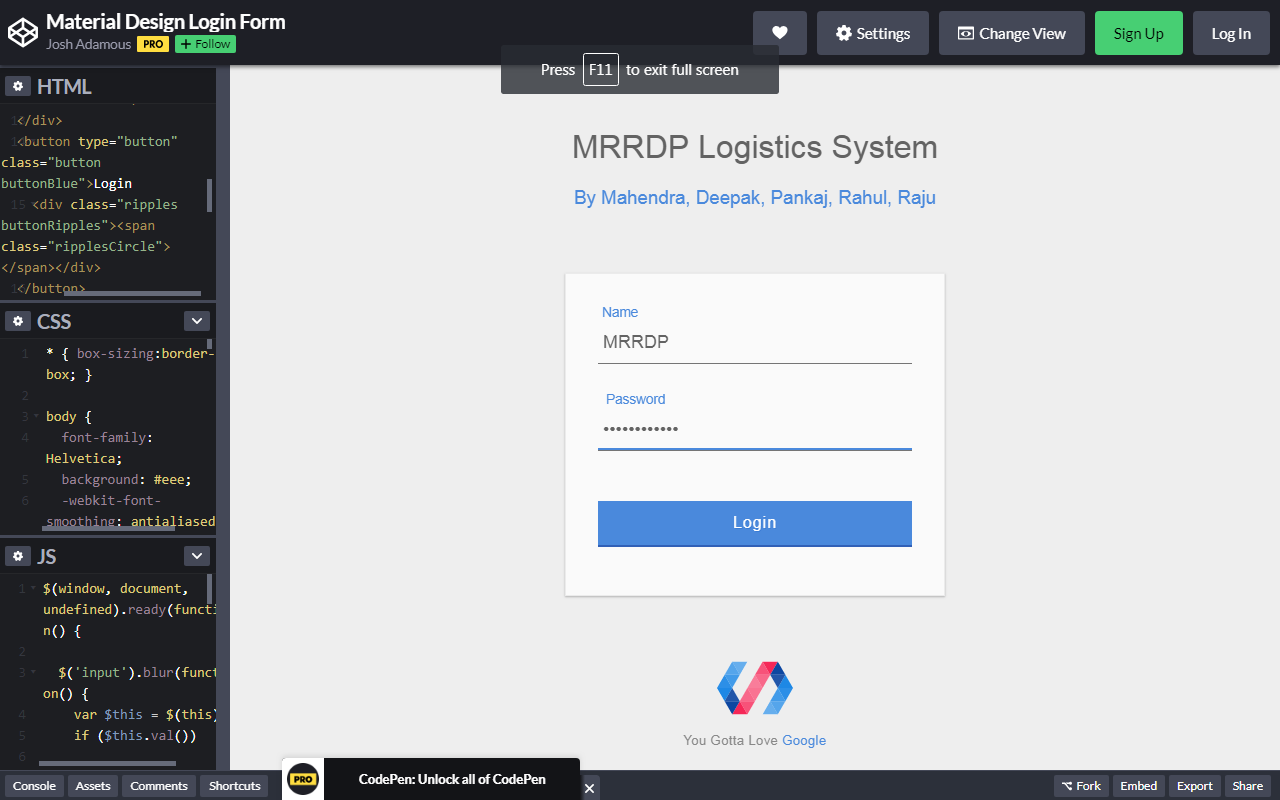
**System testing** of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

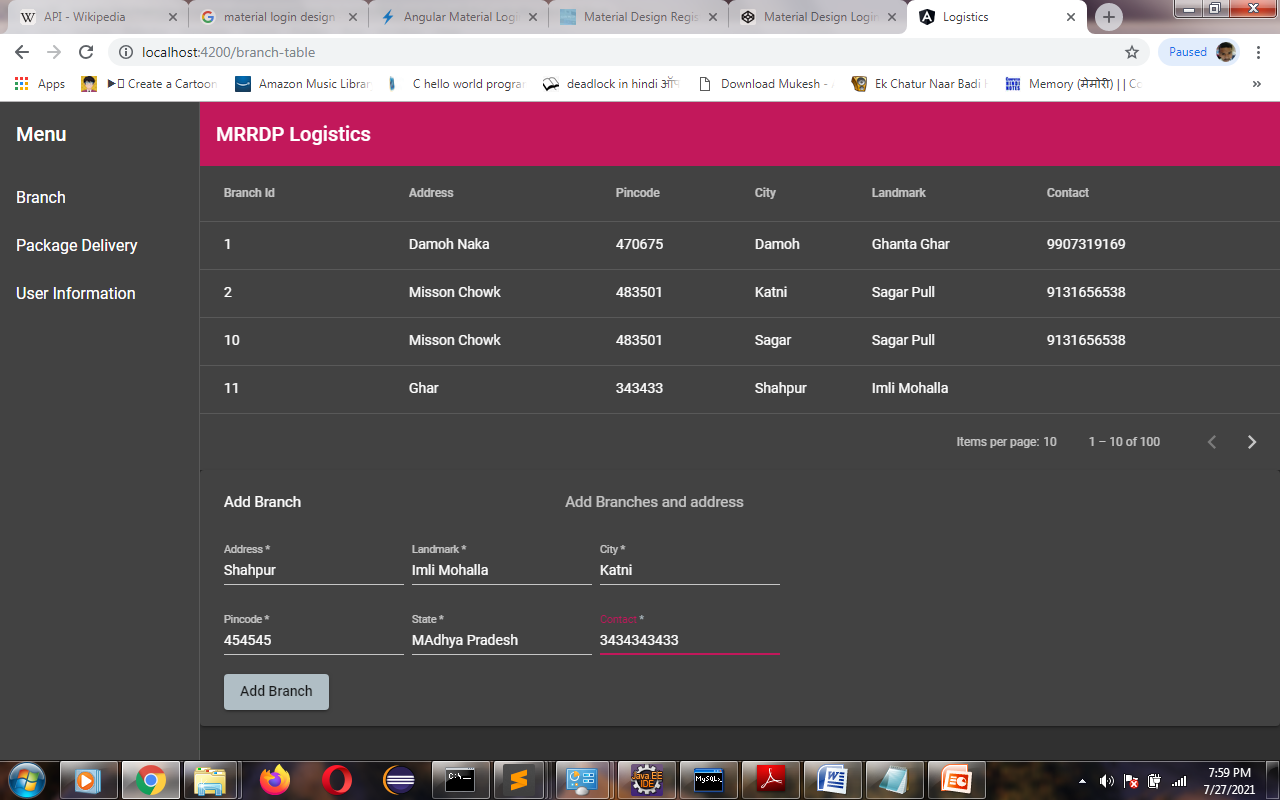
The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together (called assemblages) or between any of the assemblages and the hardware. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

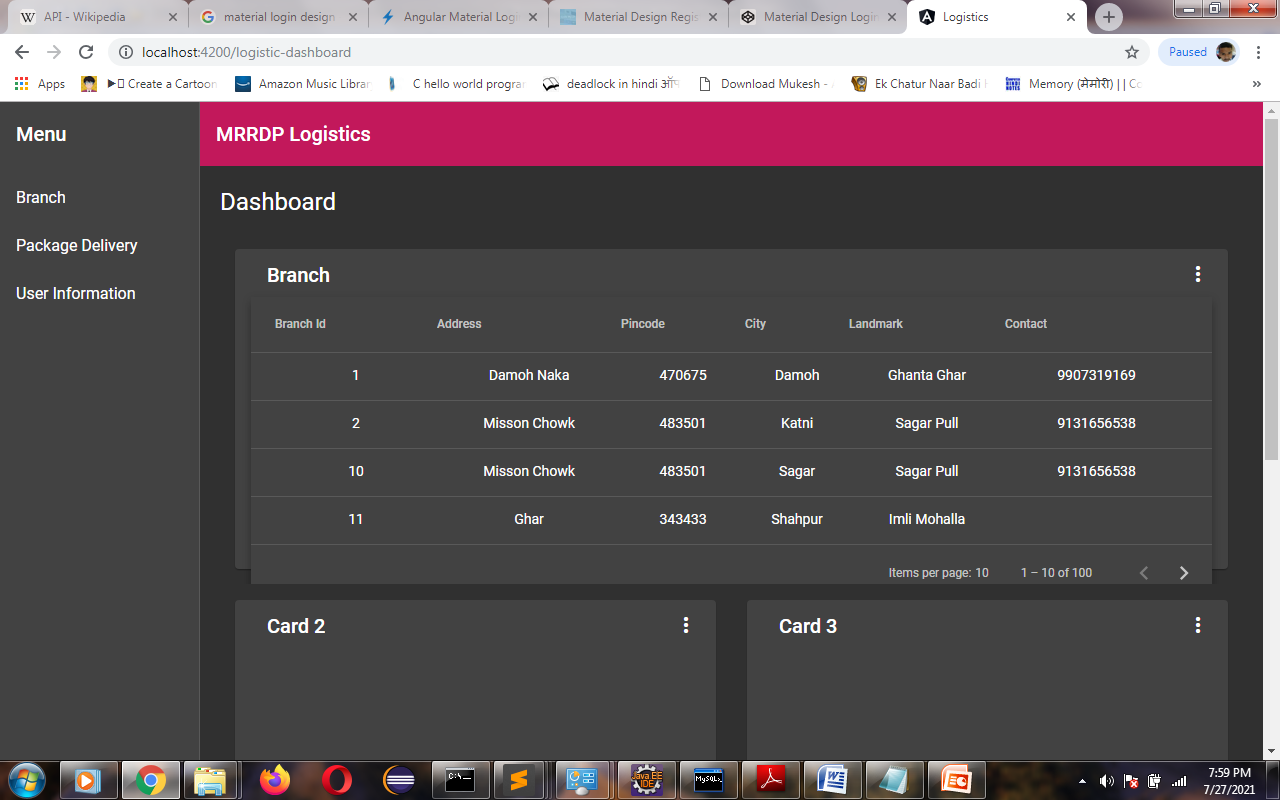
* **Regression testing**

**Regression testing** focuses on finding defects after a major code change has occurred. Specifically, it seeks to uncover software regressions, or old bugs that have come back. Such regressions occur whenever software functionality that was previously working correctly stops working as intended. Typically, regressions occur as an unintended consequence of program changes, when the newly developed part of the software collides with the previously existing code. Common methods of regression testing include re-running previously run tests and checking whether previously fixed faults have re-emerged. The depth of testing depends on the phase in the release process and the risk of the added features.

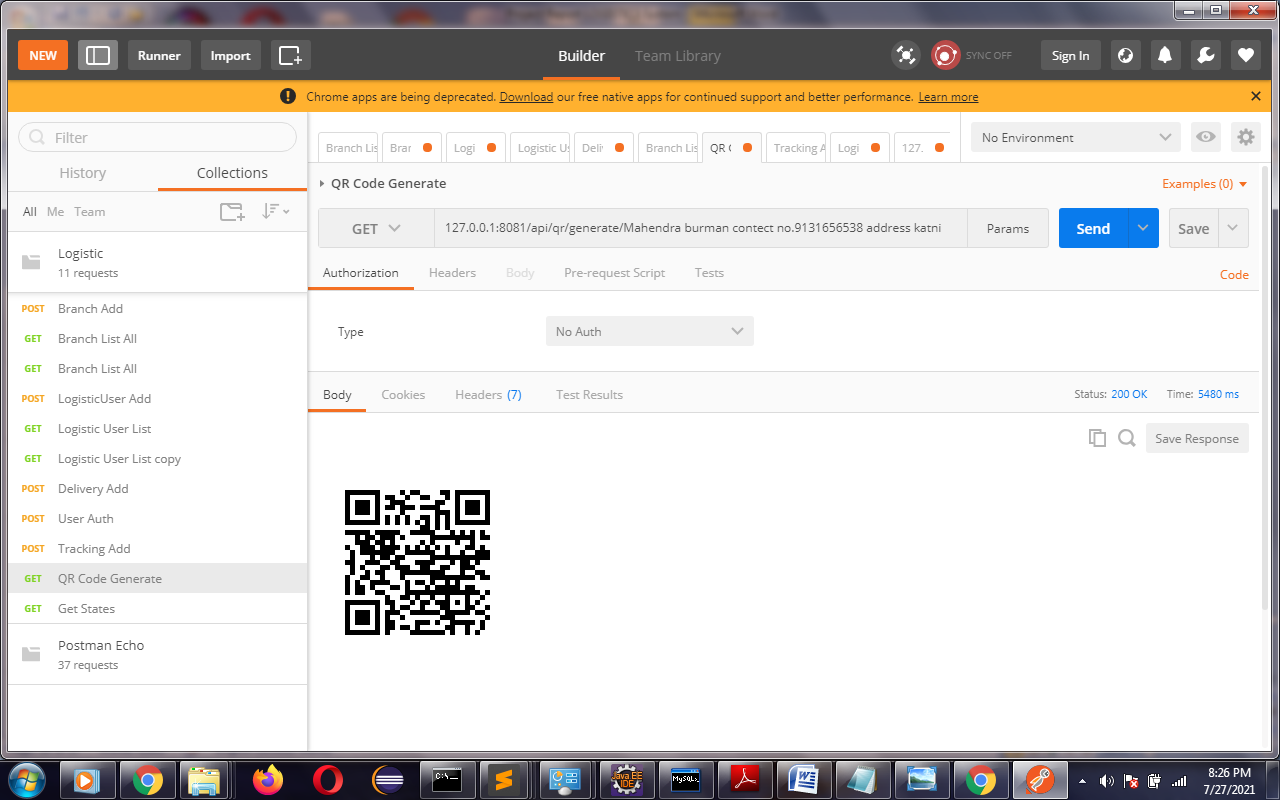
**8.SCREEN SHOT:**

****

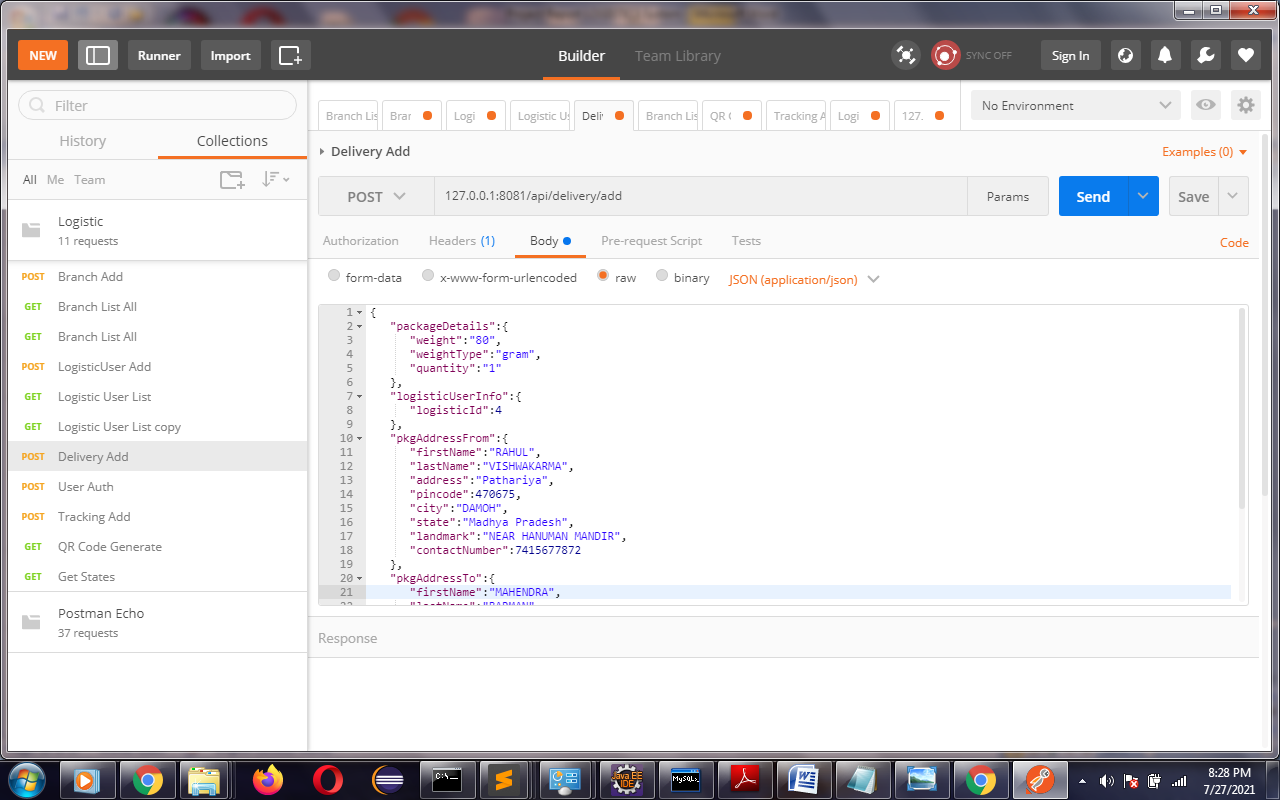
****

****

**APIs-**

****

**QR Code API**

****

**Delivery API**

**9.CONCLUSION:**

A Logistics System is the system of physical- and technology-based records and reports that supply

chain workers and managers use to collect, organize, present and use logistics data gathered

across all levels of the system. An effective Logistics System depends on the right combination of people,

processes, and technology. Skilled people must record, analyze, manage, and use supply chain

data at every level. The Logistics System must enable efficient business processes and workflows (see annex

3.8 at the end of this chapter for an example)—forecasting, inventory management, distribution

planning, reporting and ordering, order fulfillment, temperature monitoring, equipment

maintenance, performance monitoring, etc.—and incorporate routine data management

processes. And the Logistics System must leverage appropriate technology that is feasible to deploy and

sustain, and is embraced by users at each level

Technology is changing how health supply chains are managed. Paper-based Logistics System are being replaced by digital applications used on cell phones, tablets, and computers, often linked to central databases and online dashboards that provide supply chain managers easy access to data.

**10.FUTURE ENHANCEMENT:**

In most systems, the transition from paper to digital technology starts with a limited number of uses, such as SMS reporting of stock balances from community health workers,

and expands over time to capture more data from different levels of the supply chain to handle a variety of business processes. These digital tools include:

• **Dispensing or point-of-service (POS) systems**

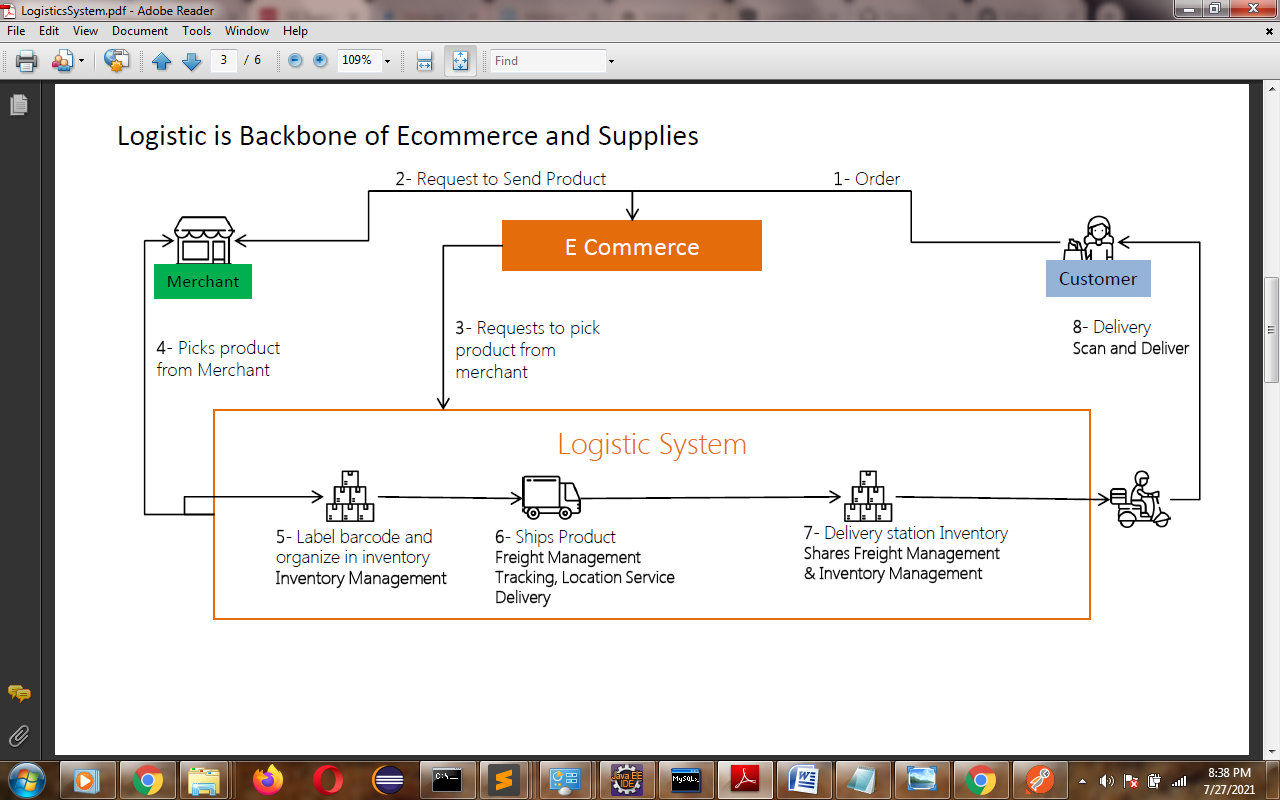
**• Warehouse management systems (WMS) for inventory control**

**• Fleeting management systems for transport and load planning**

**• Distribution planning systems for load and route planning**

But also include finances, human resources, procurement, sales, and other business

functions Barcode technology and remote temperature monitoring devices are also increasingly used in combination with these systems. GS1 is the emerging barcode standard for pharmaceuticals. Some countries use a mixture of technologies that are linked to each other. An eLMIS links to a distributor’s ERP to automatically transform a customer’s requisition into a sales order for the distributor’s finance unit and a picking list for the warehouse unit. Dashboards draw data from these systems and display critical operational and performance data for supply chain managers.

****

**11.BIBLIOGRAPHY & REFERENCES:**

|  |  |  |
| --- | --- | --- |
| **S.NO.** | **TECHNOLOGY** | **Books/Web** |
| 1 | Javascript | Book-[**You Don't Know JS**](https://github.com/getify/You-Dont-Know-JS) |
| 2 | Angular | <https://angular.io/> |
| 3 | Angular Material Design | <https://material.io/design> |
| 4 | Spring | <https://spring.io/> |
| 5 | SCSS | <https://sass-lang.com/> |
| 6 | MySQL | <https://www.mysql.com/> |
| 7 | HTML | <https://developer.mozilla.org/en-US/docs/Web/HTML> |
| 8 | NodeJS | <https://nodejs.org/> |
| 9 | GitHub | <https://github.com/> |
| 10 | Java | <https://www.hackerrank.com/domains/java> |