SMALL INVENTORY PLANNING (SIP)

Project under the guidance of

Manas Ghosh(RCCIIT), Munmun Dhali (HR) of ENTRANSCEND CONSULTING PRIVATE LIMITED.

REPORT OF MAJOR PROJECT SUBMITTED FOR THE FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF MASTER OF COMPUTER APPLICATION.

SAGARIKA CHANANI
Registration No: 151170510036 of 2015-2016
University Roll No: 11701015036

PRIYOJIT PAL
Registration No: 151170510033 of 2015-2016
University Roll No: 11701015033

SUMAN GHOSH
Registration No: 151170510050 of 2015-2016
University Roll No: 11701015049

SOMNATH MAITY
Registration No: 151170510042 of 2015-2016
University Roll No: 11701015041

श्रमम् बिना न किमपि साध्यम्

RCC INSTITUTE OF INFORMATION TECHNOLOGY

Approved by AICTE, New Delhi and Affiliated to MAKAUT, W.B.
An ISO 9001 - 2008 & ISO 14001 - 2004 Certified Institute
Canal South Road, Beliaghata, Kolkata, West Bengal 700015

RCC INSTITUTE OF INFORMATION TECHNOLOGY

श्रमम् बिना न किमपि साध्यम्

Certificate

The report of the project titled – 'SMALL INVENTORY PLANNING' submitted by Sagarika Chanani(Roll No: 11701015036), Priyojit Pal(Roll No: 11701015033), Suman Ghosh(Roll No: 11701015049), & Somnath Maity(Roll No: 11701015041) of MCA 6th Semester of 3rd Year, has been prepared under my supervision for the partial fulfillment of the requirements for MCA degree in Maulana Abul Kalam Azad University Of Technology. The report is hereby forwarded.

OPTIONAL IN CASE:

____

[Name of guide]

(**EXTERNAL SUPERVISOR)**

Counter Signed By -

____

Name of HOD
Department of Computer Application
RCC INSTITUE OF INFORMATION TECHNOLOGY
KOLKATA: 700015, India.

____

HR - Munmun Dhali
EnTranscend Consulting Private Limited

RCC INSTITUTE OF INFORMATION TECHNOLOGY

श्रमम् बिना न किमपि साध्यम्

Certificate Of Acceptance

The report of the project titled – 'SMALL INVENTORY PLANNING' submitted by Sagarika Chanani(Roll No: 11701015036), Priyojit Pal(Roll No: 11701015033), Suman Ghosh(Roll No: 11701015049), & Somnath Maity(Roll No: 11701015041) of MCA 6th Semester of 3rd Year is hereby recommended to be accepted for the fulfillment of the requirements for MCA degree in Maulana Abul Kamlam Azad University Of Technology.

NAME OF THE EXAMINERS : **SIGNATURE WITH DATE**

PLAGIARISM DECLARATION

1. I know that plagiarism means taking and using the ideas, writings, works or inventions of another as if they were one's own. I know that plagiarism not only includes verbatim copying, but also the extensive use of another person's ideas without proper acknowledgement (which includes the proper use of quotation marks). I know that plagiarism covers this sort of use of material found in textual sources and from the Internet.

2. I acknowledge and understand that plagiarism is wrong.

3. I understand that my research must be accurately referenced. I have followed the rules and conventions concerning referencing, citation and the use of quotations as set out in the Departmental Guide.

4. This assignment is my own work, or my group's own unique group assignment. I acknowledge that copying someone else's assignment, or part of it, is wrong, and that submitting identical work to others constitutes a form of plagiarism.

5. I have not allowed, nor will I in the future allow, anyone to copy my work with the intention of passing it off as their own work.

Name: …………………………………………… Roll No.: ....

Signature: ……………………………………………. Date: …………………………………

TABLE OF CONTENT

CONTENTS :

ACKNOWLEDGEMENT

We express our sincere gratitude to Mr. Manas Ghosh (Professor of Department of CA) and Munmun Dhali (HR) of 'EnTranscend Consulting Private Limited' for extending their valuable time for us and guide us for this project.

I am also indebted to the other teachers for their unconditional help and inspiration.

Last but not least I would like to express my gratitude to my HOD - Mr. AKB of our department (Department of CA) who helped me in his own way whenever needed.

(Signature of Student)
Reg. No:151170510036
Roll. No: 11701015036
MCA – 6th Sem
Session – 2015-2018,RCCIIT

(Signature of Student)
Reg. No:151170510033
Roll. No: 11701015033
MCA – 6th Sem
Session – 2015-2018,RCCIIT

(Signature of Student)
Reg. No:151170510050
Roll. No: 11701015049
MCA – 6th Sem
Session – 2015-2018,RCCIIT

(Signature of Student)
Reg. No:151170510042
Roll. No: 11701015041
MCA – 6th Sem
Session – 2015-2018,RCCIIT

INTRODUCTION

In retail shops, the traditional way of tracking inventory is used, which is very time consuming and labour-intensive. This software is designed and developed for the shopkeepers who want to have an integrated mini ERP system on their machines.

The primary objective of this project is to satisfy the needs of customers. Whether the customer is a direct consumer, a distributor or another department within the company, SIP should provide the best-quality service.

This project will control the inventory for raw materials to the direct consumer. Getting product to customers on time and as inexpensively as possible and real time inventory management are the main goals of our system.

One objective of an inventory planning is to ensure that the inventory is stocked at all times to suit the needs of customers. This includes inventory systems that keep track of products ready for sale and inventories that track supplies or raw materials. The company can lose sales and customers if products are not readily available when customers need them.

SIP (Small Inventory Planning) is a hybrid application with smart interface. SIP can be used across industries to manage customers, products, inventory, billing, and analytical reporting. It can be accessed in mobile, tabs, and desktops.

PROBLEM ANALYSIS:-

The application is designed and developed to collect, store, manage and interpret data from customers, products, inventory, and billing activities. ERP provides an integrated and continuously updated view of core business processes using common databases maintained by a database management system.

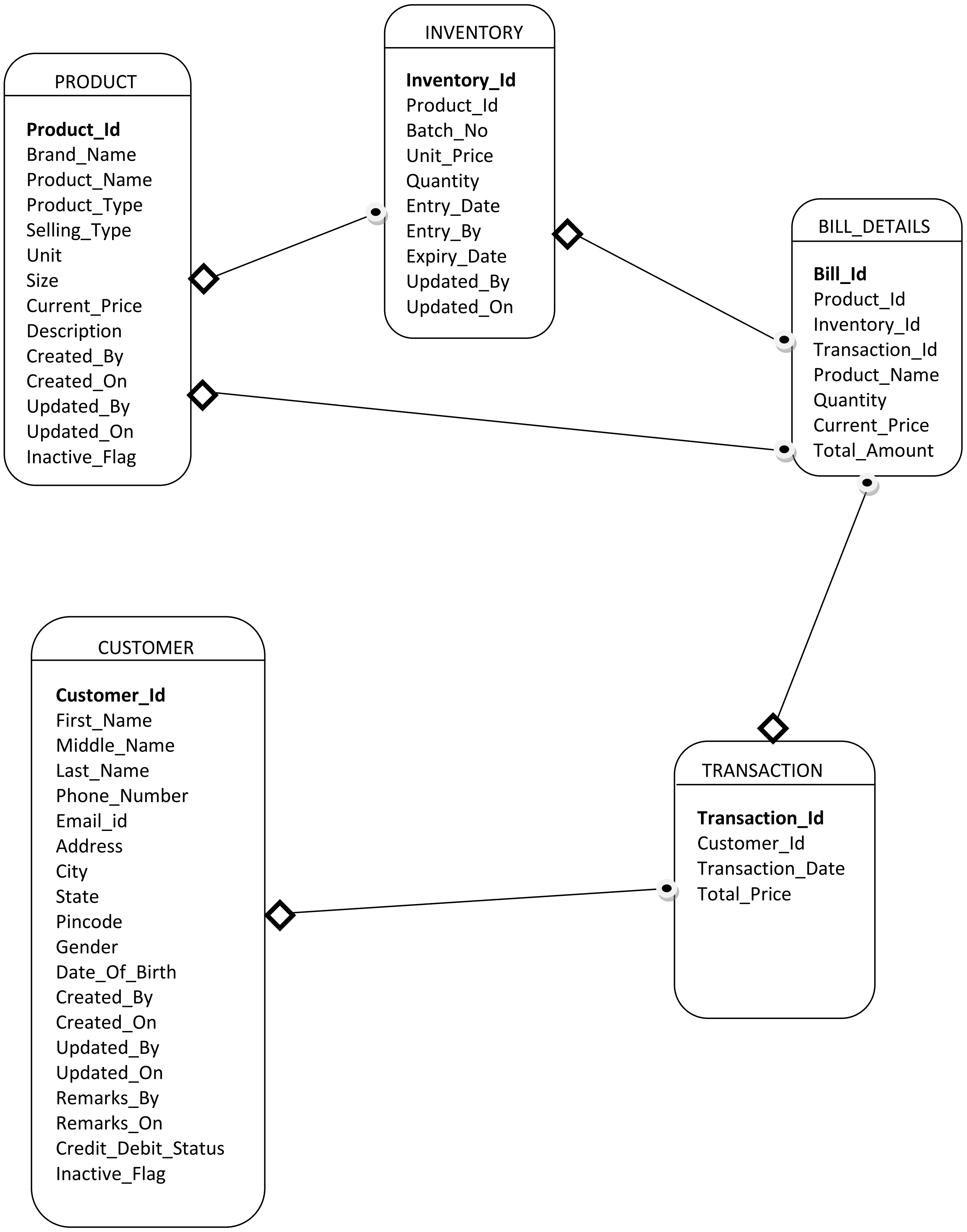
The objectives of this project are to:

- Reduce billing errors.
- Reduce time in searching for an item.
- Faster service to the customers to be attended.
- Automatic stock approximation.

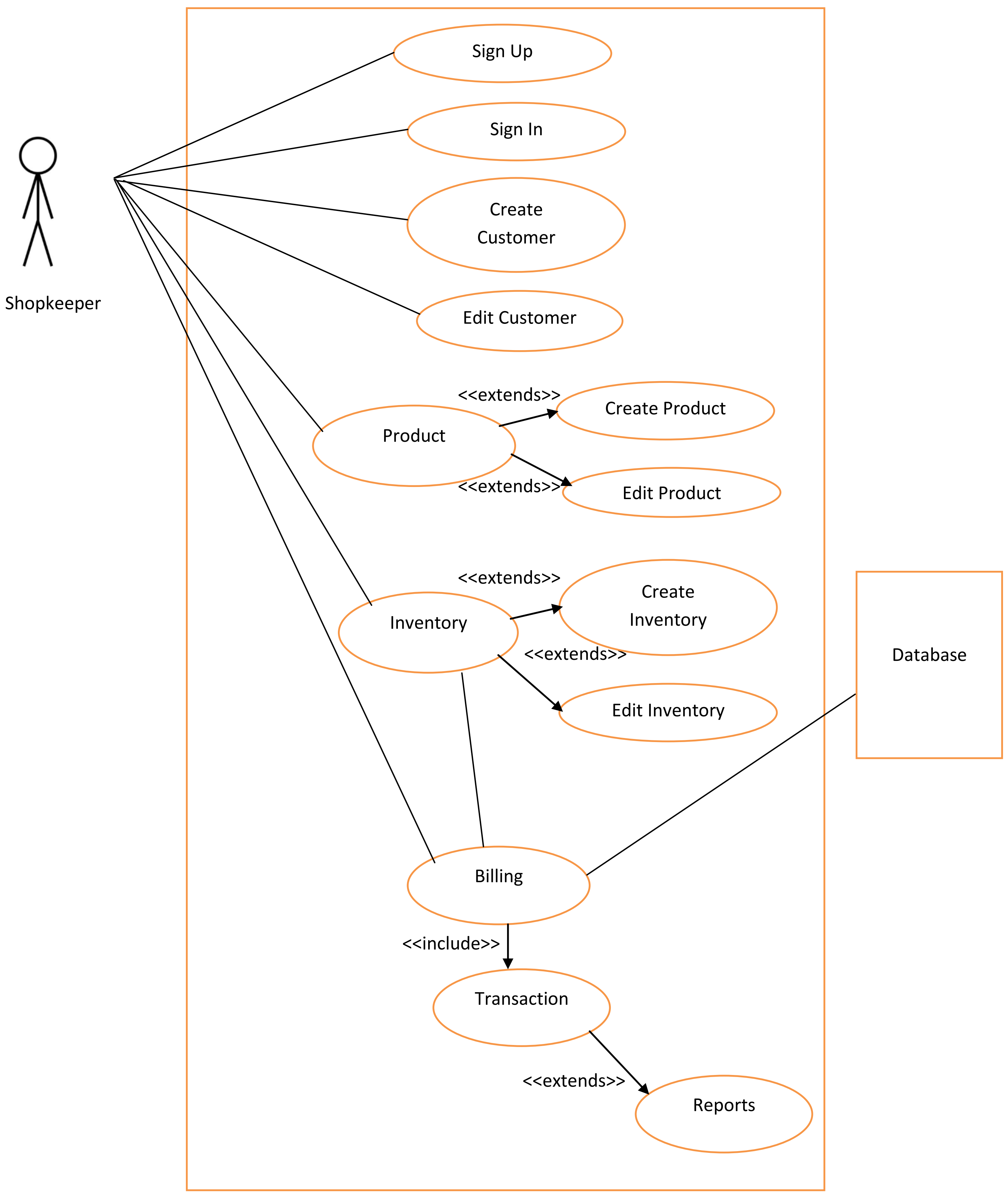
Another important job of this tool is to include the user/admin verification to login into the system, which has been done with the help of a database.

FORMULATION:-

The following diagram contains the application's **Entity-Relationship diagram**:-



The following diagram contains the application's **Use Case diagram**:-



TOOLS AND PLATFORMS:-

The project is based on a web application that runs in a web browser in the front end using Angular, and the entire backend operation is developed using JAVA programming language and its concepts(Collection classes, OOPS concept etc.), Java SE (Standard Edition) features are used to build the project.

The tools that were used to build the project were:

- VISUAL STUDIO CODE- (A code editor for building and debugging modern web and cloud applications) ,
- PGADMIN/MYSQL- (Database)

Platform & Operating System used:

- ECLIPSE- (Tool for creating Java EE and Web applications),
- WINDOWS O/S

Development Components:

- Client Side:

  HTML 5, CSS, Bootstrap, Angular 4, Node server

- Server Side:

  Rest Web Service, Spring Boot, Speedment (ORM), Postgres (DataBase)

IMPLEMENTATION DETAILS:-

Our modules in this application are:

USER, CUSTOMER, PRODUCT, INVENTORY, BILL_DETAILS, TRANSACTION

HOME:

It is a main module which consists of all other modules and renders the components as and when required. It consists of one Navbar and one Sidebar where all the routerLinks of the other components are specified to be rendered.

LOGIN:

In this application, there are two users, i.e.,

i) ADMIN

ii) USER

This is a module which is used by the admin to login into the system.

- **SIGN IN:** The admin of the system is already defined in the database so that he can login to the system. In this, USERNAME and PASSWORD are the mandatory fields.
- **SIGN UP:** It is used for creating new users. In this, FIRST NAME, PHONE NUMBER, PASSWORD are the mandatory fields. On clicking the Sign Up button, all the values from the form is taken and send to the database, via HTTP service, to the USER table. This will show an alert and automatically reset the form in this page. If the user already exists in the database, then it will show an alert.
- **FORGET PASSWORD:** It is used to set new passwords for the existing users. In this, FIRST NAME, PHONE NUMBER, NEW PASSWORD are the mandatory fields. On clicking the Submit button, the value of the NEW PASSWORD is taken and updated for the corresponding user in the database, via HTTP service, in the USER table. This will show an alert and automatically reset the form in this page. If the user does not exist in the database, then it will show an alert.
- **LOG OUT:** It is used to log out of the system. It takes the user back to the SIGN IN page.

CUSTOMER:

It is a module which helps us to generate new customers and also used to keep the customer details in a proper order.

- **CREATE CUSTOMER:** It is used for creating new customers. In this, FIRST NAME, GENDER, PHONE NUMBER, PINCODE are the mandatory fields. On clicking the Submit button, all the values from the form is taken and send to the database, via HTTP service, to the CUSTOMER table. This will show an alert and automatically reset the form in this page. If the customer already exists in the database, then it will show an alert. If a user wants to cancel this page or want to view the customer details, he can go back to the UPDATE CUSTOMER page by clicking on View button.
- **UPDATE CUSTOMER:** It is used to update or view existing customers. In this, all the details of each

customer are shown, via HTTP service, on initiating this page. Then, the user searches a particular customer name (FIRSTNAME) via using FILTERPIPE typescript file. After selecting a particular customer, the user clicks on Edit button which takes all the data of that selected customer, via DATA service, and it automatically renders the EDIT CUSTOMER page to edit the existing customer. If a user wants to create a new customer, he can click on the Add(+) button which takes it to the CREATE CUSTOMER page.

- **EDIT CUSTOMER:** It is used to edit the existing customer details. When a user selects a particular customer in the UPDATE CUSTOMER page, all the details of that particular customer is send to this page, via DATA service, and are placed in the corresponding positions, via using two-way binding (ngModel). On clicking the Confirm button, all the details from the form is taken and send to the database to update the details, via HTTP service. If a user wants to cancel this page or want to view the customer details, he can click on the Cancel button which will take it back to the UPDATE CUSTOMER page.

PRODUCT:

It is a module which helps us to generate new products and also used to keep the product details in a proper order.

- **CREATE PRODUCT:** It is used for creating new products. In this, BRAND NAME, PRODUCT NAME, SELLING TYPE, CURRENT PRICE, UNIT,SIZE are the mandatory fields. On clicking the Add button, all the values from the form is taken and send to the database, via HTTP service, to the PRODUCT table. This will show an alert and automatically reset the form in this page. If the product already exists in the database, then it will show an alert. If a user wants to cancel this page or want to view the product details, he can go back to the UPDATE PRODUCT page by clicking on View button.

- **UPDATE PRODUCT:** It is used to update or view existing products. In this, all the details of each product is shown, via HTTP service, on initiating this page. Then, the user searches a particular product name (PRODUCT NAME), via using FILTERPIPE typescript file. After selecting a particular product, the user clicks on Edit button which takes all the data of that selected product, via DATA service, and it automatically renders the EDIT PRODUCT page to edit the existing product. If a user wants to create a new product, he can click on the Add (+) button which takes it to the CREATE PRODUCT page.

- **EDIT PRODUCT:** It is used to edit the product details. When a user selects a particular product in the UPDATE PRODUCT page, all the details of that particular product is send to this page, via DATA service, and are placed in the corresponding positions, via using two-way binding (ngModel). On clicking the Confirm button, all the details from the form is taken and send to the database to update the details, via HTTP service. If a user wants to cancel this page or want to view the product details, he can click on the Cancel button which will take it back to the UPDATE PRODUCT page.

INVENTORY:

It is a module which helps us to add stocks (of products) and also used to keep the inventory details in a proper order.

- **CREATE INVENTORY:** It is used for adding product lots. In this, PRODUCT NAME, BATCH NUMBER, QUANTITY, EXPIRY DATE, UNIT PRICE, are the mandatory fields. When this page renders, all the product names (BRAND_NAME-PRODUCT_NAME) ,via HTTP service, from the

PRODUCT table will placed in the drop down box. Thus, whenever the user searches a particular product name(PRODUCT NAME), via using FILTERPIPE typescript file, the drop down box will act as an auto-complete box and will display(select) that particular product name. On selecting the product, it will automatically place the corresponding SIZE, UNIT and CURRENT PRICE of the selected product in the boxes in read only mode. If a user wants to edit the price (UNIT PRICE) of the product, he can click on the Edit button which will enable the price box to edit. On clicking the Add to Inventory button, all the values from the form is taken and send to the database, via HTTP service, to the INVENTORY table. This will automatically reset the form in this page. If a new product arrived which is not present in the PRODUCT table, then the user can create it by clicking on Create Product button, which will show an alert and take it to the CREATE PRODUCT page. If a user want to cancel this page or want to view the inventory details, he can go back to the UPDATE INVENTORY page by clicking on View button.

- **UPDATE INVENTORY:** It is used to update or view existing product stocks. In this, all the details of inventory is shown, via HTTP service, on initiating this page. Then, the user searches a particular batch number (BATCH NUMBER), via using FILTERPIPE typescript file. After selecting a particular batch number, the user clicks on Edit button which takes all the data of that selected product id with the corresponding batch number, via DATA service, and it automatically renders the EDIT INVENTORY page to edit the existing stock of that product. If a user wants to add a new stocks to the inventory, he can click on the Add (+) button which takes it to the CREATE INVENTORY page.

- **EDIT INVENTORY:** It is used to edit the inventory details. When a user selects a particular lot of the product in the UPDATE INVENTORY page, all the details of that particular product is send to this page, via DATA service, and are placed in the corresponding positions, via using two-way binding (ngModel). On clicking the Confirm button, all the details from the form is taken and send to the database to update the details, via HTTP service. If a user wants to cancel this page or want to view the inventory details, he can click on the Cancel button which will take it back to the UPDATE INVENTORY page.

BILLING:

It is a module which helps us to generate a bill for the customer and also used to keep the bill details as well as the transaction details in a proper order.

- **BILLING CART:** It is used for creating bills for the customers. When this page renders, all the customer details, via HTTP service, from the CUSTOMER table will placed in a table format. Thus, the user searches a particular customer name (FIRST NAME), via using FILTERPIPE typescript file, and selects that particular customer name. If the customer is not present, then he can create new customer by clicking on the Add (+) button, which will be redirected to the CREATE CUSTOMER page. On selecting a particular customer, the bill cart appears on the screen with the selected customer's PHONE NUMBER, NAME and SYSTEM DATE on the top of the cart. If a user wants to cancel this cart, he can go back to the search customer table by clicking on Cross (X) button.

When the billing cart renders, all the product names (BRAND_NAME-PRODUCT_NAME), via HTTP service, from the PRODUCT table will placed in the drop down box. Thus, whenever the user searches a particular product name(PRODUCT NAME), via using FILTERPIPE typescript file, the drop down box will act as an auto-complete box and will display(select) that particular product name. The corresponding batch numbers of the selected product will come in the drop down box, which are present in the inventory with quantity>0. On selecting a particular batch number, it will automatically place the corresponding EXPIRY DATE,UNIT PRICE in read only mode, and corresponding QUANTITY in the editable mode, as well as the AMOUNT(by calculating as price*quantity) in read only mode of the selected product in the boxes.

To add more rows, the user needs to click on the Add button. Similarly, to delete rows, the user needs to click on the delete button beside a row. If a user wants to edit the number of quantity for a particular row, he can click on the edit button which will prompt a box to change the quantity. For all the rows generated by the user, the TOTAL AMOUNT gets calculated automatically at the same time. On clicking the Make Payment button, all the rows from the cart is taken ans send to the TRANSACTION table.

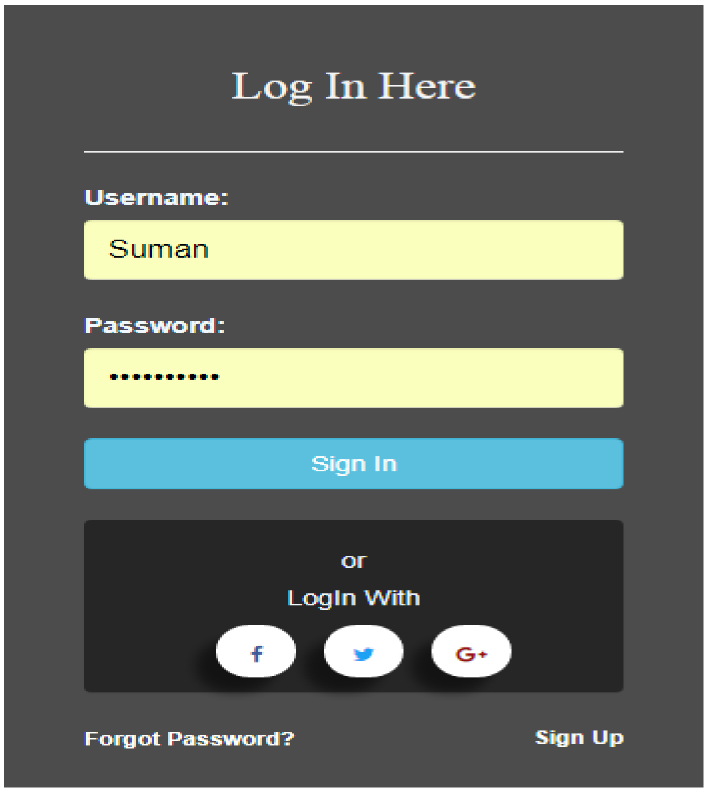
For **TRANSACTION** details, CUSTOMER ID (using ngModel) and TRANSACTION DATE (system date generated using DatePipeFilter(built-in) Service) and only the TOTAL AMOUNT from this cart will be send to the database, via HTTP service, to the TRANSACTION table and that particular row in which the total amount has been inserted will be returned from the database to get the TRANSACTION ID (auto-generated). Now, this TRANSACTION ID and all the other rows of the cart will be send to the database, via HTTP service, to the BILL DETAILS table.

This will automatically closes the cart and will show the search customer table again for the next billing process to continue.

REPORTS:

It is used to view transaction details and bill details for the customers.

TESTING



Log In Here

Username:

Suman

Password:

••••••••••

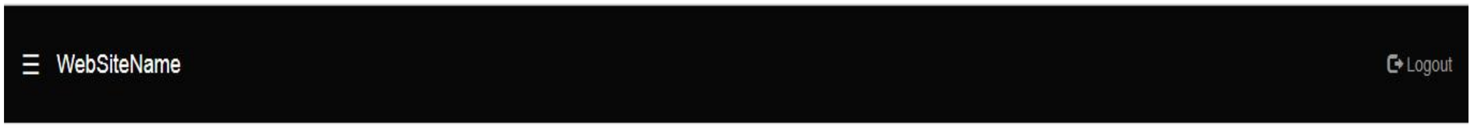
Sign In

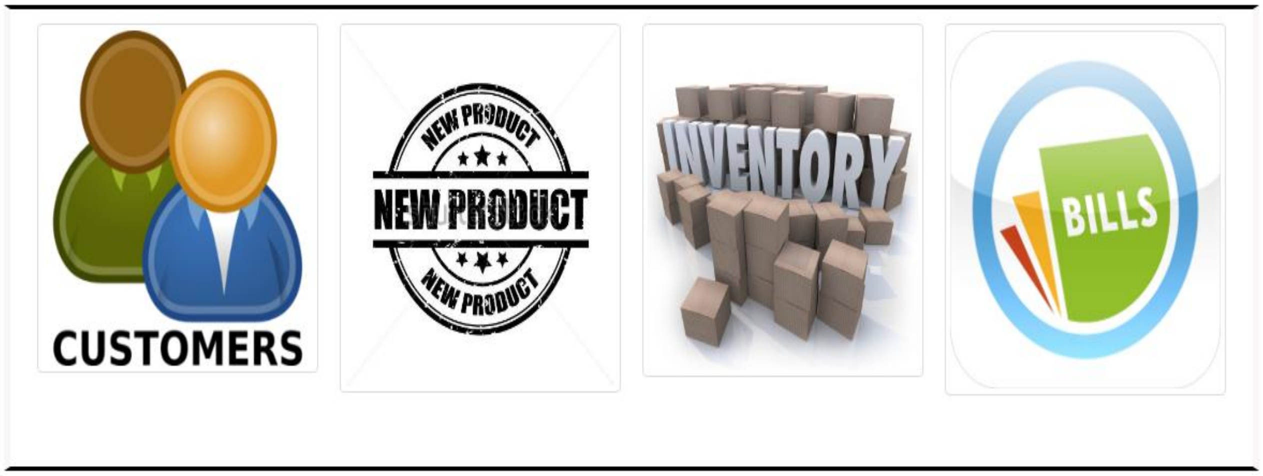
or

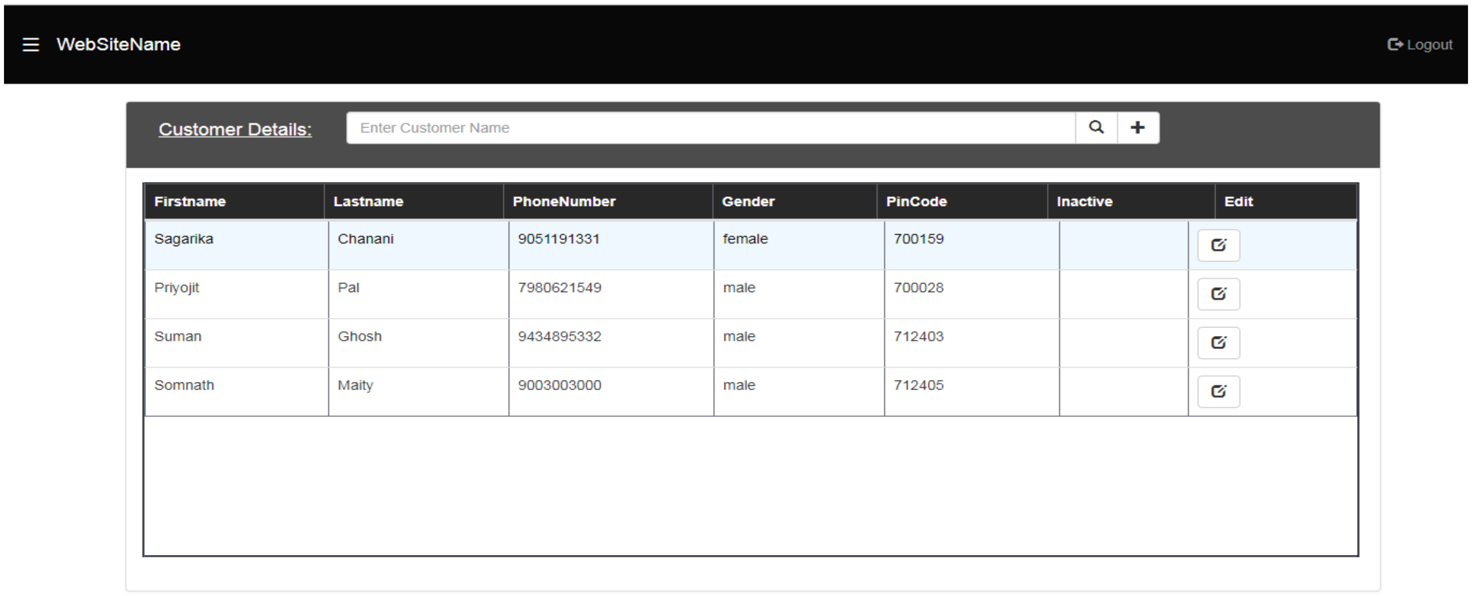
LogIn With

Forgot Password?

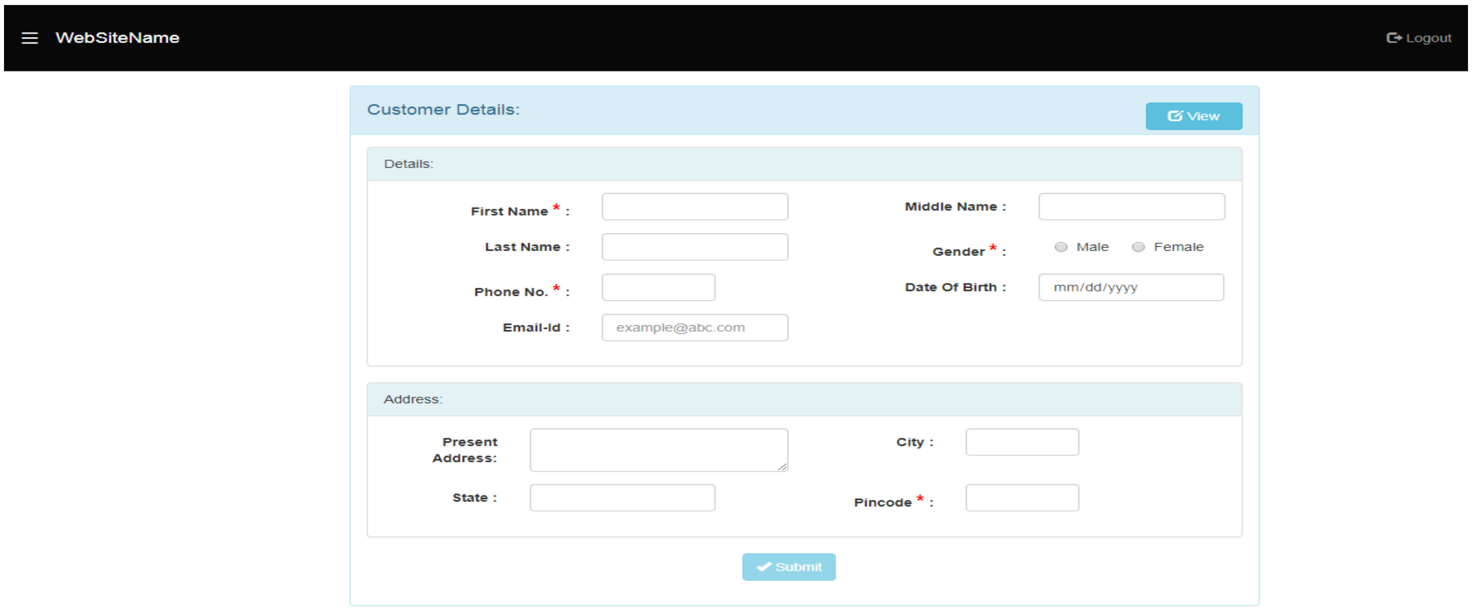
Sign Up



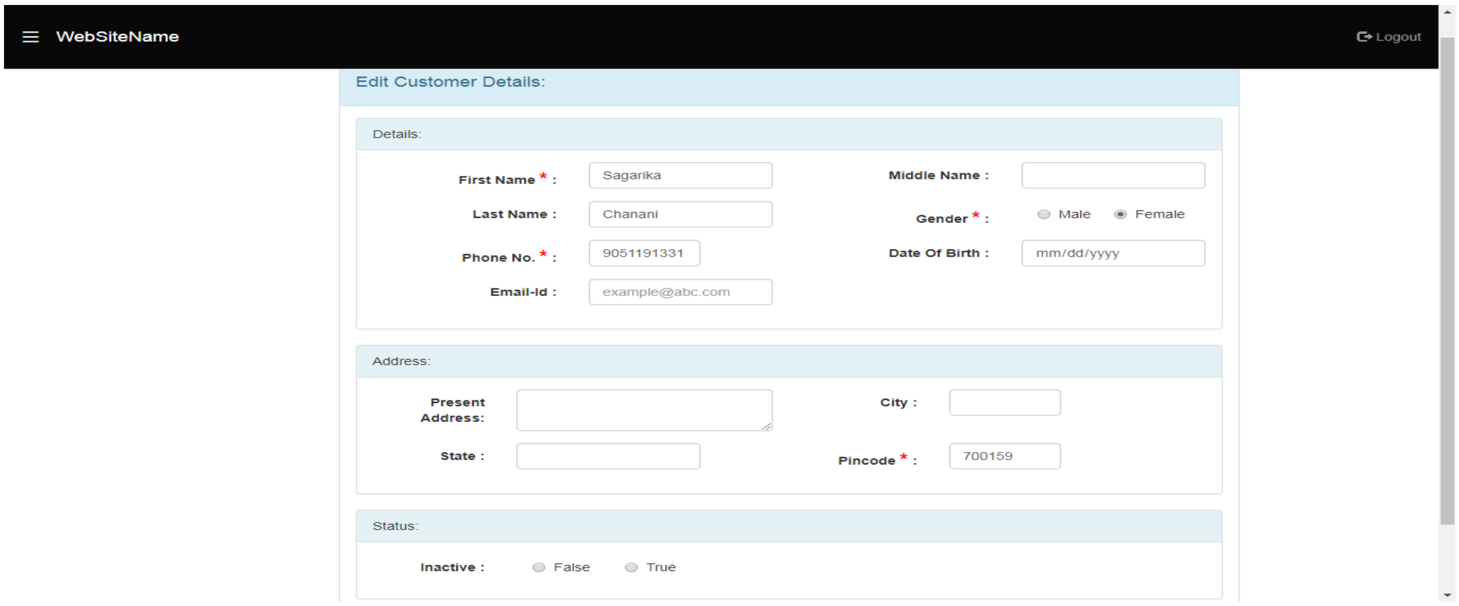




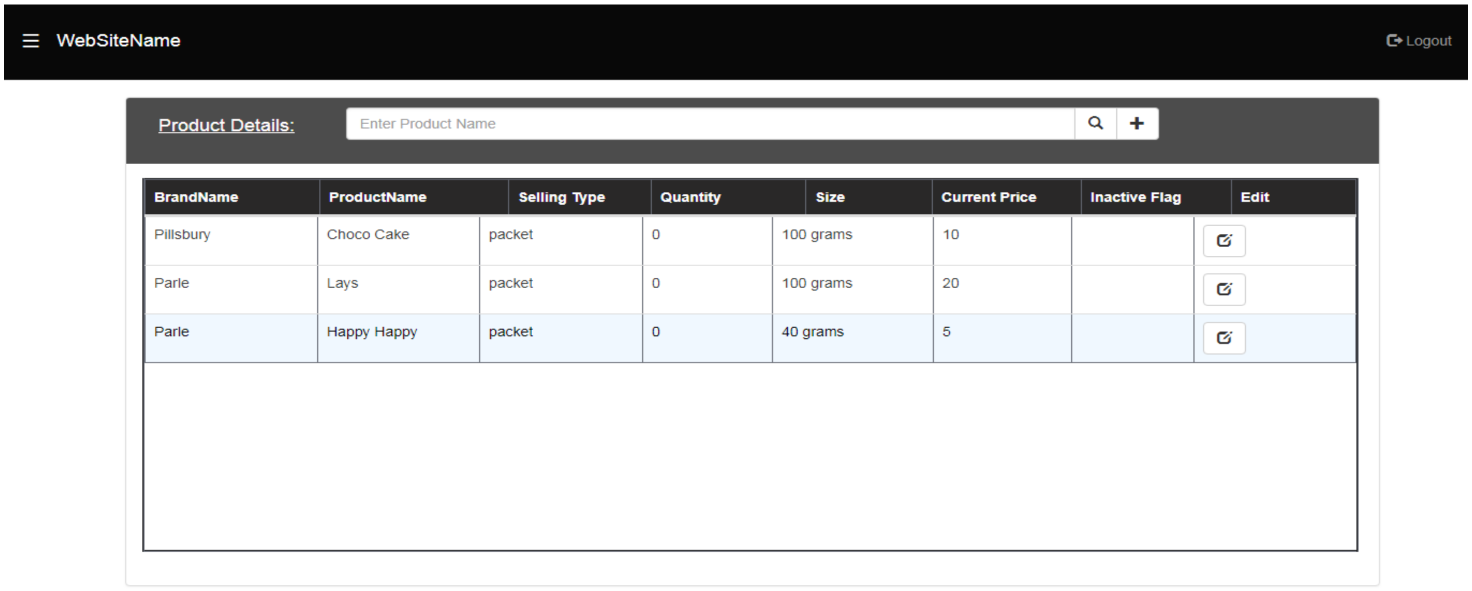
Firstname	Lastname	PhoneNumber	Gender	PinCode	Inactive	Edit
Sagarika	Chanani	9051191331	female	700159		
Priyojit	Pal	7980621549	male	700028		
Suman	Ghosh	9434895332	male	712403		
Somnath	Maity	9003003000	male	712405		



First Name * :		Middle Name :	
Last Name :		Gender * :	○ Male ○ Female
Phone No. * :		Date Of Birth :	mm/dd/yyyy
Email-Id :	example@abc.com		
Present Address:		City :	
State :		Pincode * :	



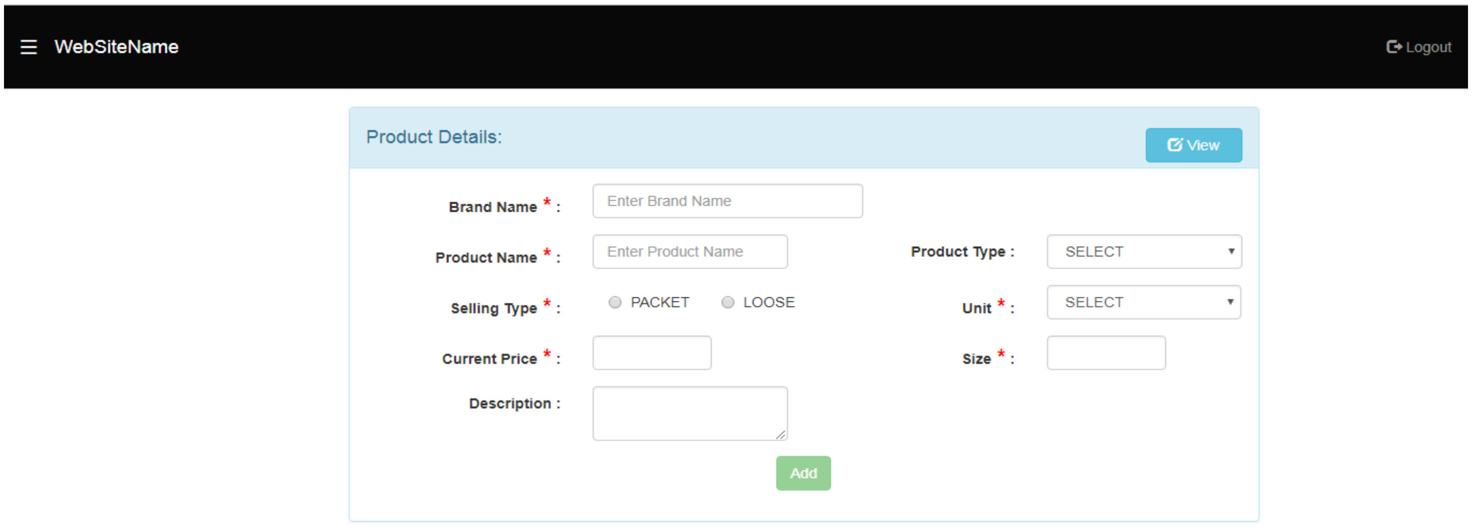
First Name * :	Sagarika	Middle Name :	
Last Name :	Chanani	Gender * :	○ Male [✓ Female]
Phone No. * :	9051191331	Date Of Birth :	mm/dd/yyyy
Email-Id :	example@abc.com		
Present Address:		City :	
State :		Pincode * :	700159
Inactive :	○ False ○ True		



WebSiteName Logout

Product Details: Enter Product Name

BrandName	ProductName	Selling Type	Quantity	Size	Current Price	Inactive Flag	Edit
Pillsbury	Choco Cake	packet	0	100 grams	10		
Parle	Lays	packet	0	100 grams	20		
Parle	Happy Happy	packet	0	40 grams	5		

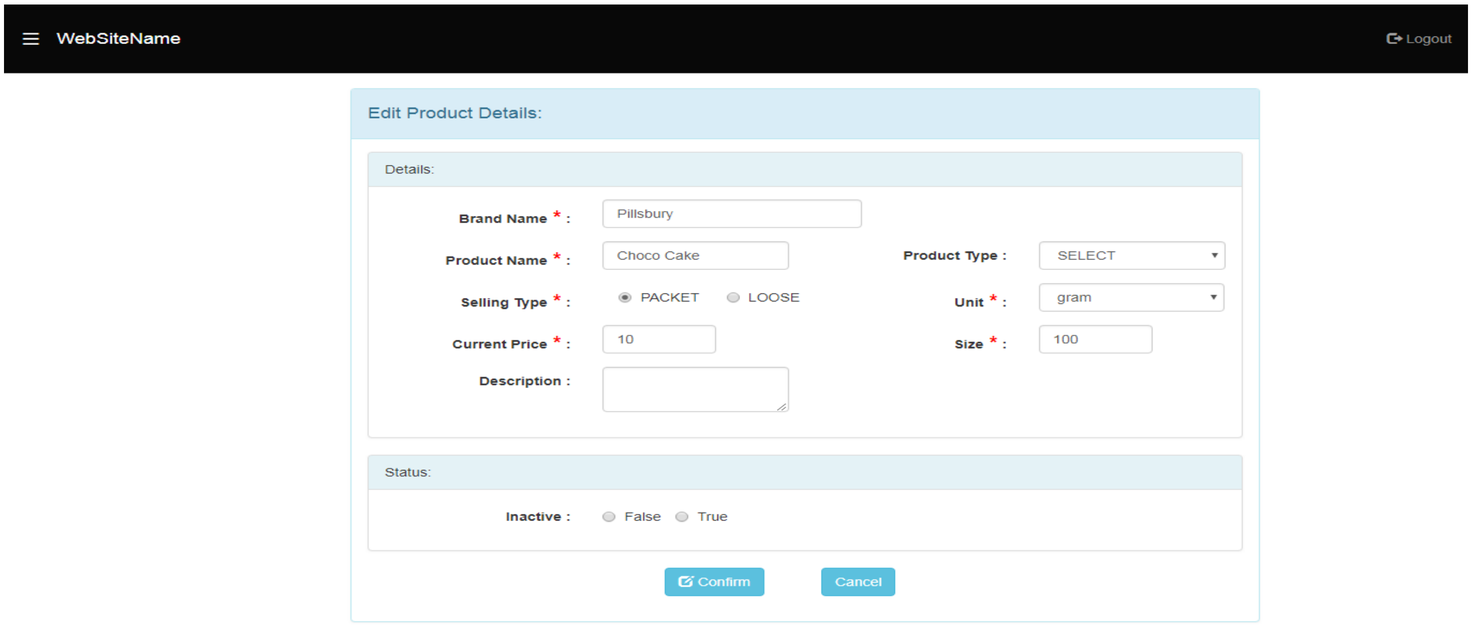


WebSiteName Logout

Product Details: View

Brand Name * :	Enter Brand Name		
Product Name * :	Enter Product Name	Product Type :	SELECT
Selling Type * :	PACKET LOOSE	Unit * :	SELECT
Current Price * :		Size * :	
Description :			

Add



WebSiteName Logout

Edit Product Details:

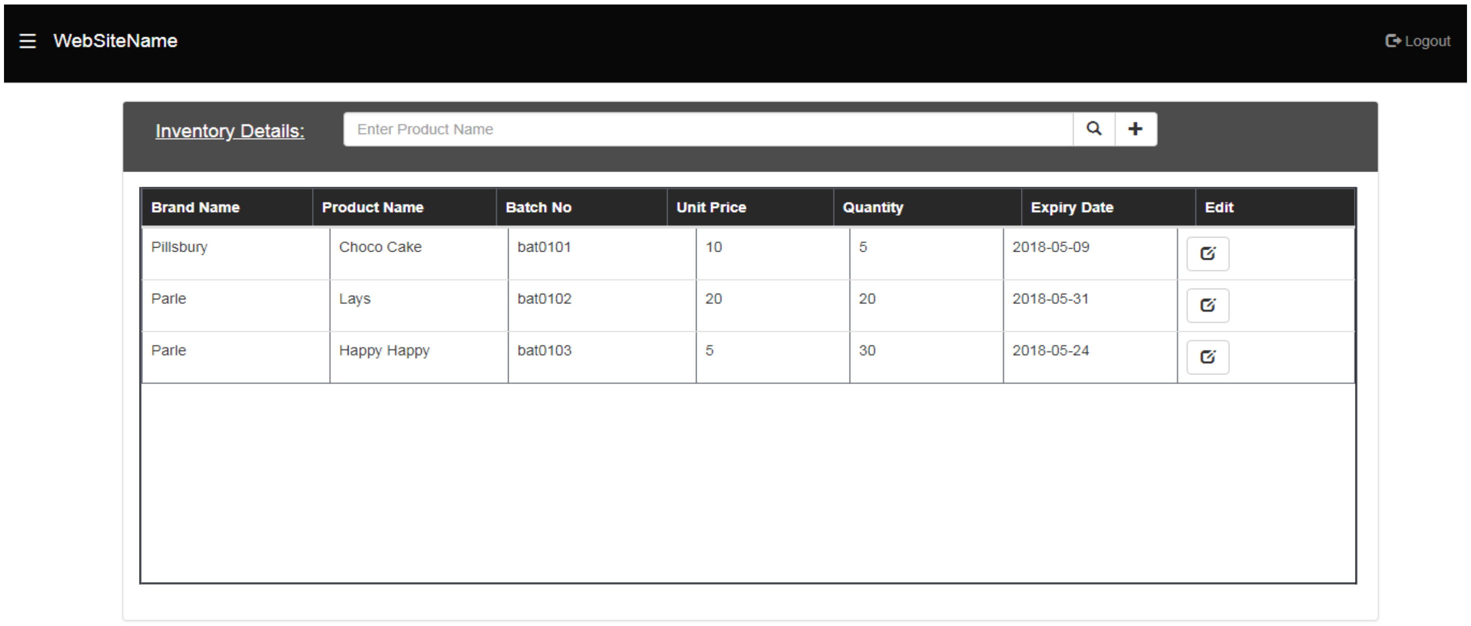
Details:

Brand Name * :	Pillsbury		
Product Name * :	Choco Cake	Product Type :	SELECT
Selling Type * :	[✓ PACKET] LOOSE	Unit * :	gram
Current Price * :	10	Size * :	100
Description :			

Status:

Inactive : False True

Confirm Cancel

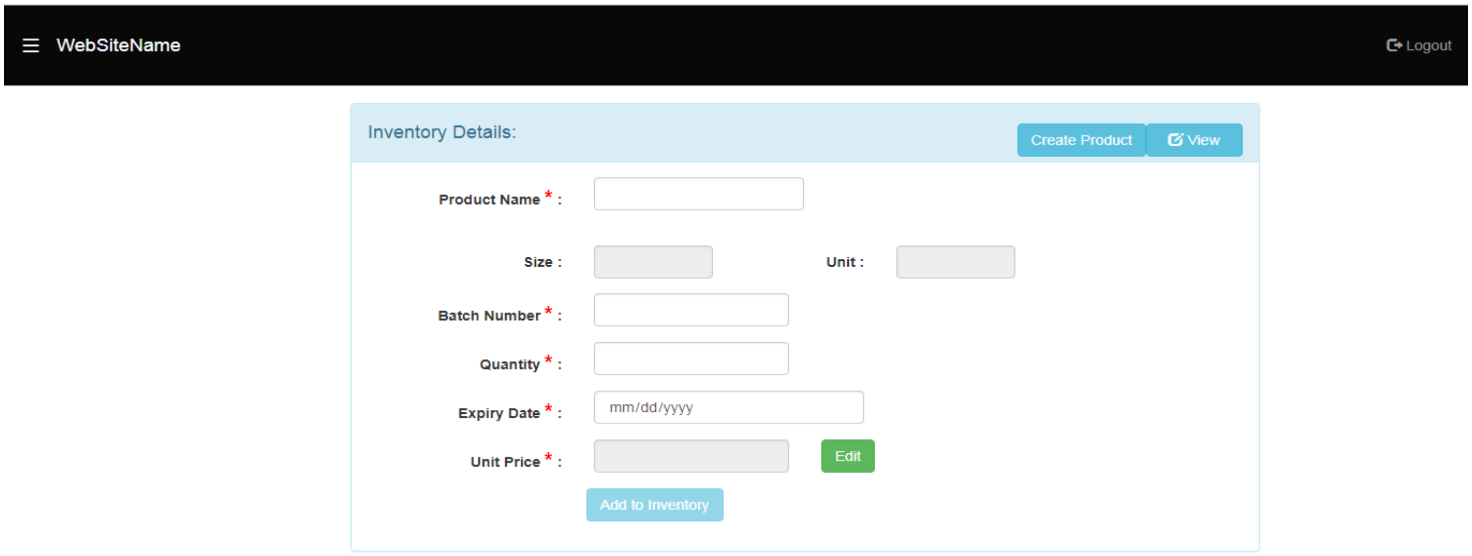

WebSiteName

Logout

Inventory Details:

Enter Product Name

Brand Name	Product Name	Batch No	Unit Price	Quantity	Expiry Date	Edit
Pillsbury	Choco Cake	bat0101	10	5	2018-05-09	
Parle	Lays	bat0102	20	20	2018-05-31	
Parle	Happy Happy	bat0103	5	30	2018-05-24	


WebSiteName

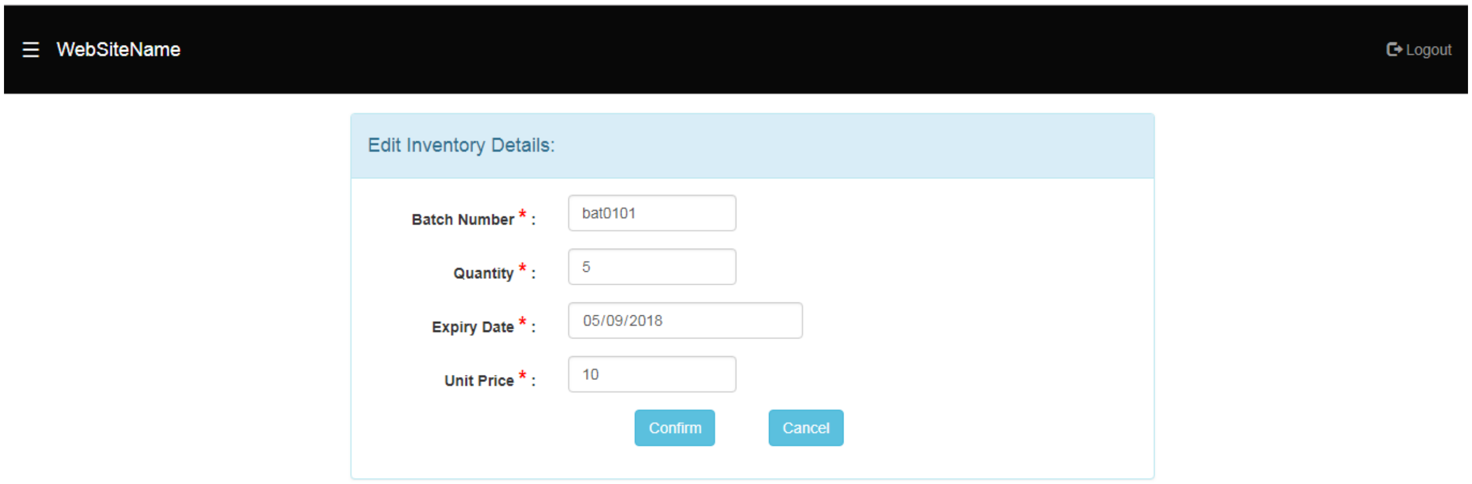
Logout

Inventory Details:

Create Product | View

Product Name * :			
Size :		Unit :	
Batch Number * :			
Quantity * :			
Expiry Date * :	mm/dd/yyyy		
Unit Price * :		Edit	

Add to Inventory

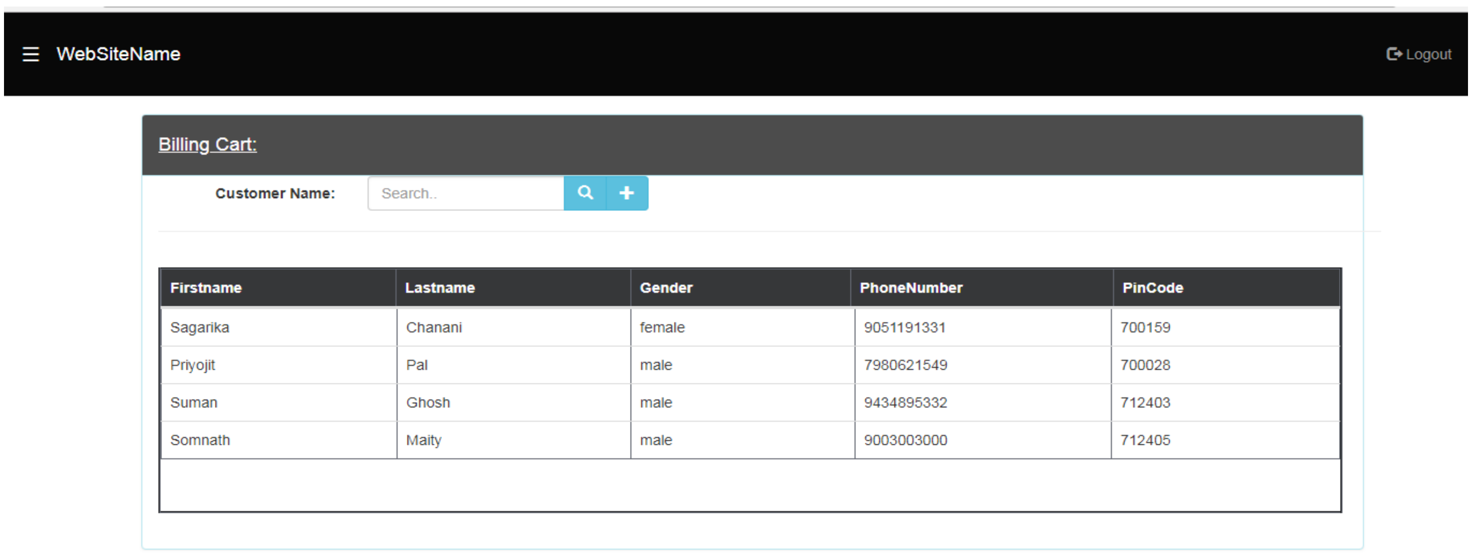

WebSiteName

Logout

Edit Inventory Details:

Batch Number * :	bat0101
Quantity * :	5
Expiry Date * :	05/09/2018
Unit Price * :	10

Confirm | Cancel

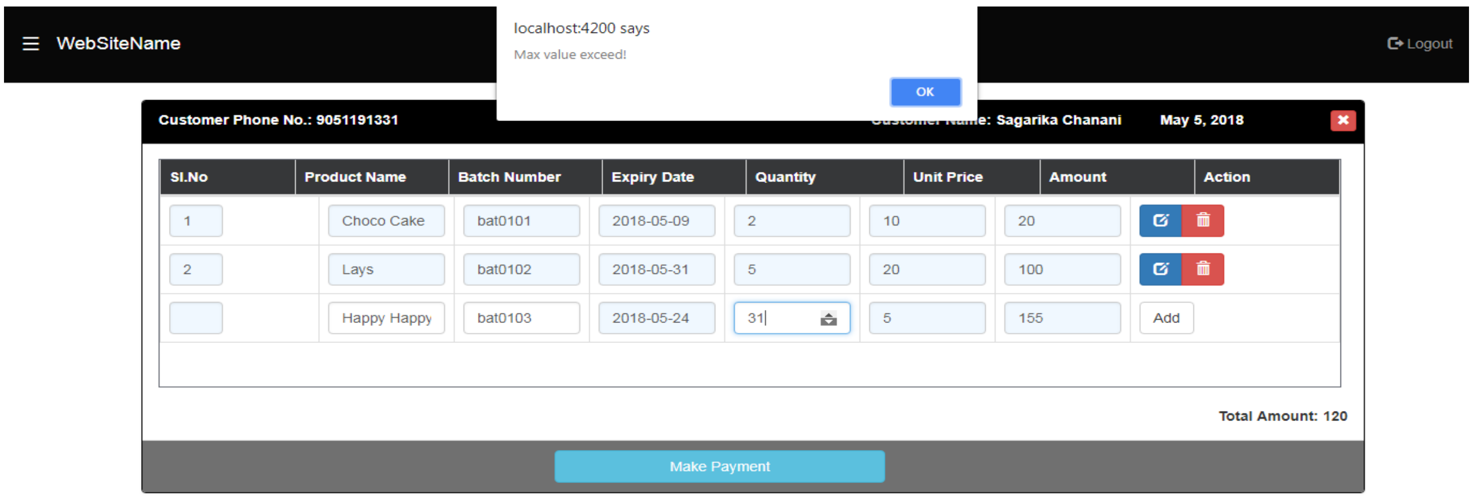


Billing Cart:

Customer Name: Search..

Firstname	Lastname	Gender	PhoneNumber	PinCode
Sagarika	Chanani	female	9051191331	700159
Priyojit	Pal	male	7980621549	700028
Suman	Ghosh	male	9434895332	712403
Somnath	Maity	male	9003003000	712405

i



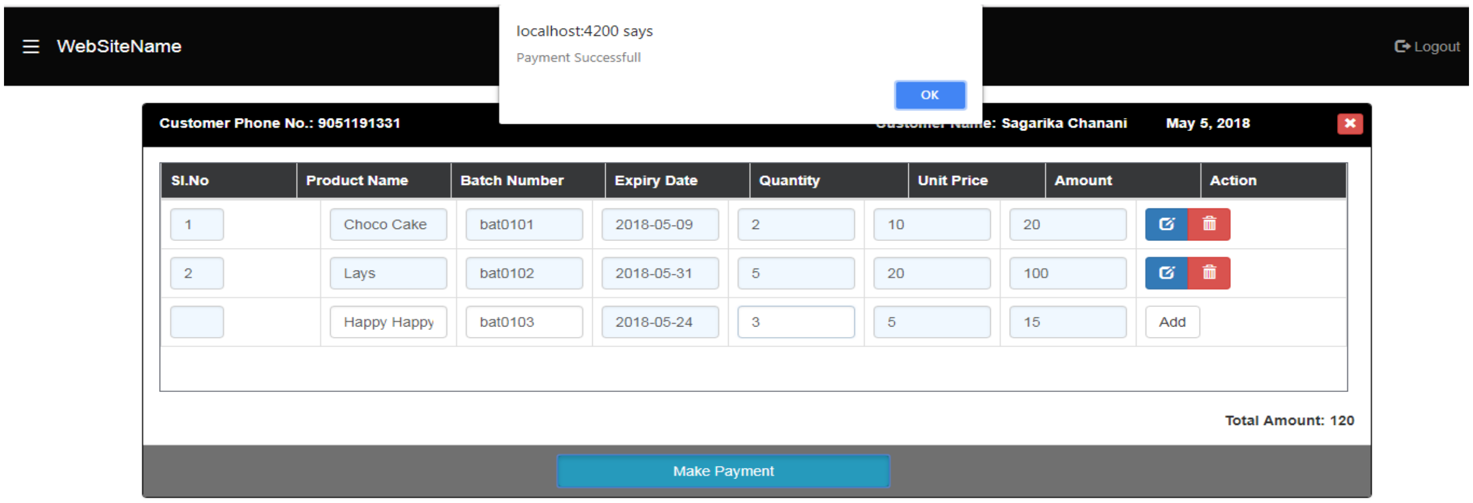
localhost:4200 says

Max value exceed!

Customer Phone No.: 9051191331 Customer Name: Sagarika Chanani May 5, 2018

Sl.No	Product Name	Batch Number	Expiry Date	Quantity	Unit Price	Amount	Action
1	Choco Cake	bat0101	2018-05-09	2	10	20	
2	Lays	bat0102	2018-05-31	5	20	100	
	Happy Happy	bat0103	2018-05-24	31	5	155	Add

Total Amount: 120



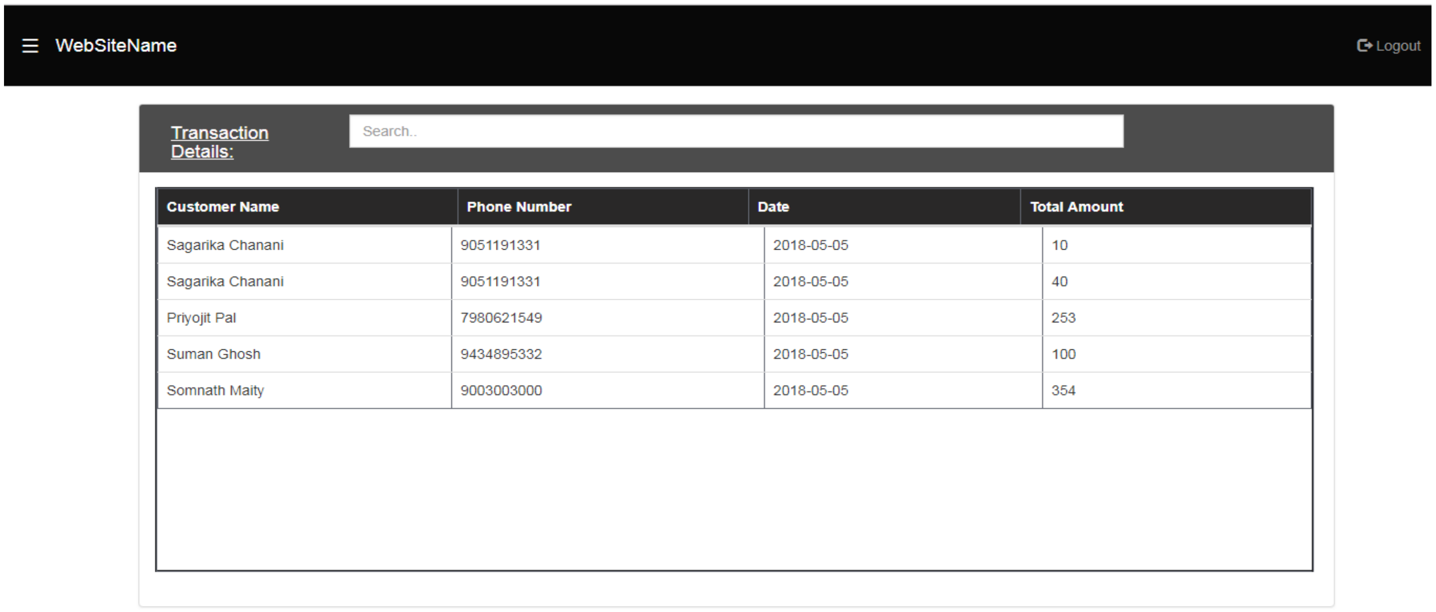
localhost:4200 says

Payment Successfull

Customer Phone No.: 9051191331 Customer Name: Sagarika Chanani May 5, 2018

Sl.No	Product Name	Batch Number	Expiry Date	Quantity	Unit Price	Amount	Action
1	Choco Cake	bat0101	2018-05-09	2	10	20	
2	Lays	bat0102	2018-05-31	5	20	100	
	Happy Happy	bat0103	2018-05-24	3	5	15	Add

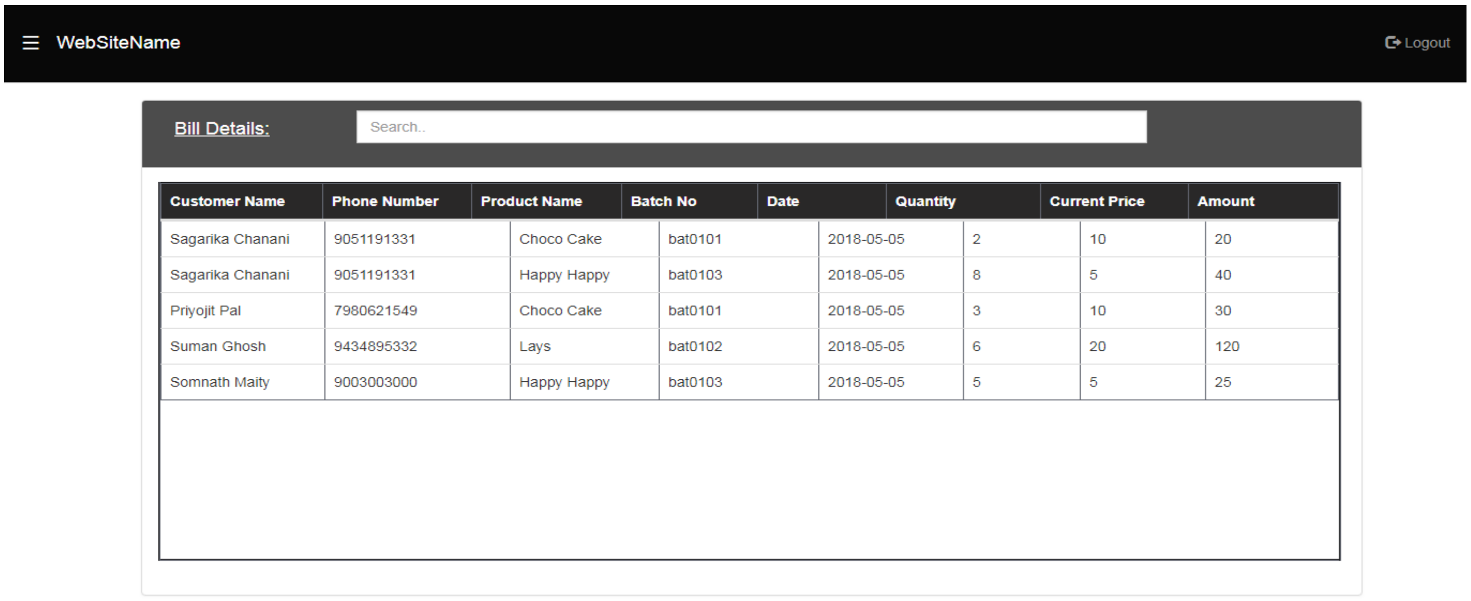
Total Amount: 120


☰ WebSiteName
Logout

Transaction Details:

Search..

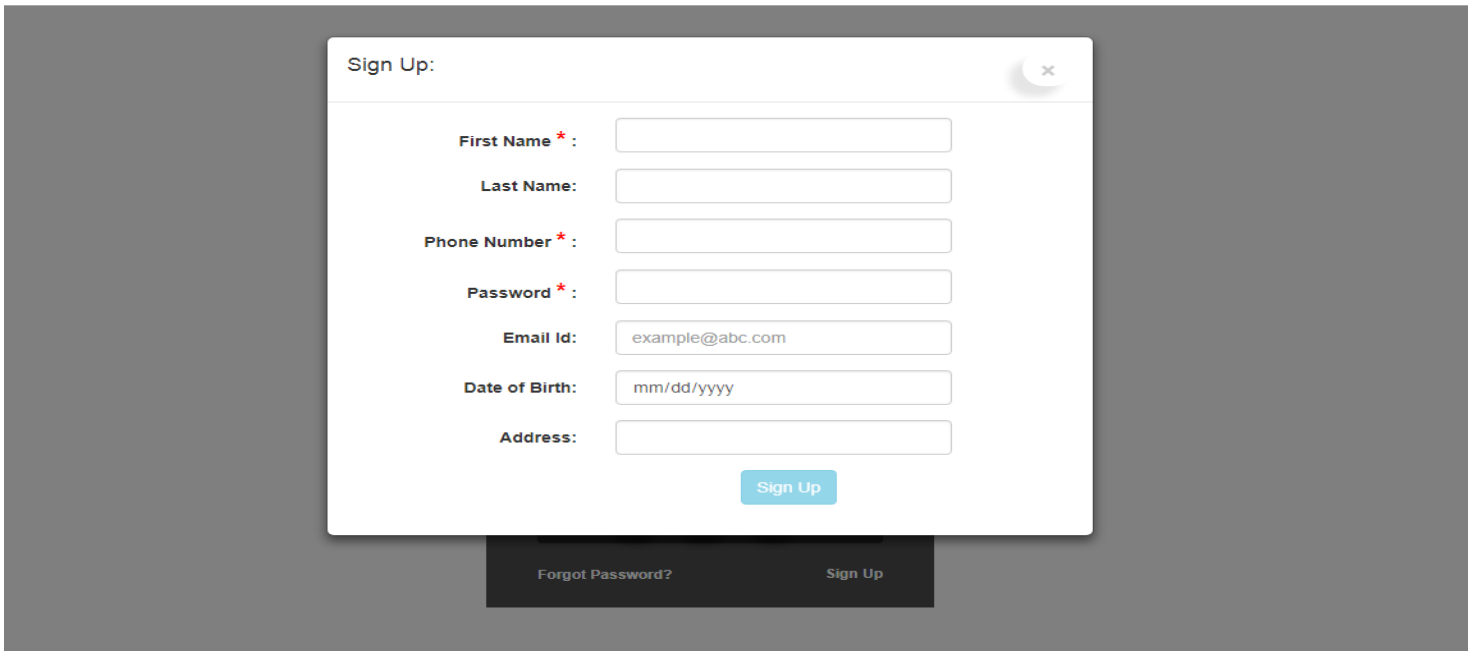
Customer Name	Phone Number	Date	Total Amount
Sagarika Chanani	9051191331	2018-05-05	10
Sagarika Chanani	9051191331	2018-05-05	40
Priyojit Pal	7980621549	2018-05-05	253
Suman Ghosh	9434895332	2018-05-05	100
Somnath Maity	9003003000	2018-05-05	354


☰ WebSiteName
Logout

Bill Details:

Search..

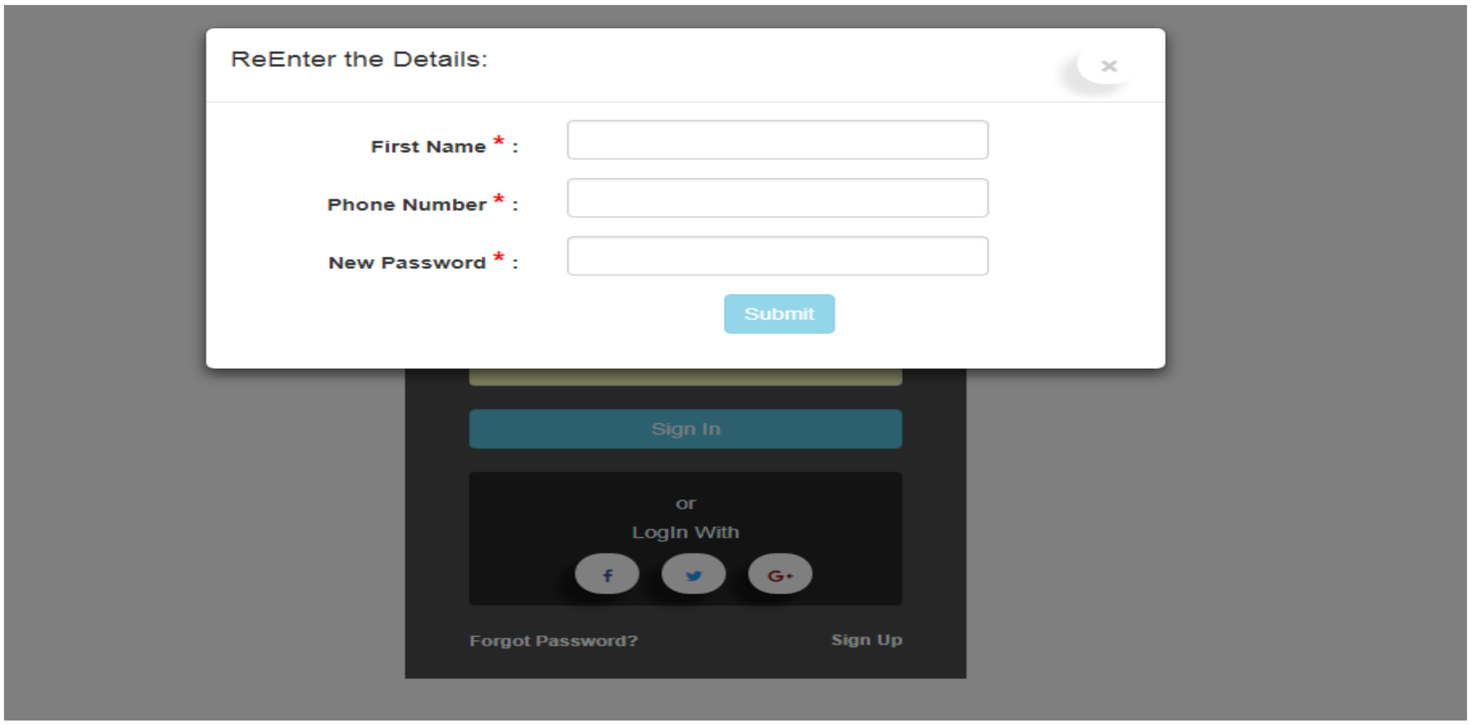
Customer Name	Phone Number	Product Name	Batch No	Date	Quantity	Current Price	Amount
Sagarika Chanani	9051191331	Choco Cake	bat0101	2018-05-05	2	10	20
Sagarika Chanani	9051191331	Happy Happy	bat0103	2018-05-05	8	5	40
Priyojit Pal	7980621549	Choco Cake	bat0101	2018-05-05	3	10	30
Suman Ghosh	9434895332	Lays	bat0102	2018-05-05	6	20	120
Somnath Maity	9003003000	Happy Happy	bat0103	2018-05-05	5	5	25


Sign Up:

First Name * :	
Last Name:	
Phone Number * :	
Password * :	
Email Id:	example@abc.com
Date of Birth:	mm/dd/yyyy
Address:	

Sign Up

Forgot Password? Sign Up



ReEnter the Details:

First Name * :

Phone Number * :

New Password * :

CLASS DESCRIPTION

Front End (Visual Studio Code):

Modules:

1. Home: It is used to render the main page of the site.
2. Customer: It is used to render the customer details.
3. Product: It is used to render the product details.
4. Inventory: It is used to render the inventory details.
5. Bill_Details: It is used to process the payment system.
6. Transaction: It is used to view the transaction details of the specified customer.
7. Admin-Login: It is used to render the login details.
8. Billing: It is used to view the bill details of the specified customer.
9. About-Us: It is used to render the company details.
10. Forget-Password: It is used to render the new password details.
11. Sign-Up: It is used to render the new user details.
12. Services: It is used to get and send the data between different modules.
13. Shared Classes: It is used to store the fields and its data types of the different modules.

Back End (Eclipse):

Services: - This is used to get or send the data from or to the database.

1. CustomerService.java
2. ProductService.java
3. InventoryService.java
4. Bill_DetailsService.java
5. TransactionService.java
6. UserService.java

Controllers: - This is used to get or send the data from or to the front end, i.e., the browser.

1. CustomerController.java
2. ProductController.java
3. InventoryController.java
4. Bill_DetailsController.java
5. TransactionController.java
6. UserController.java

Classes: - This is used to join the tables of the database to view in the front end.

1. BillInventTransCust.java
2. ProductInventory.java
3. ProductQuantity.java
4. TransactionBill.java
5. TransactionCustomer.java

Main Class: - This is the main class which is used to run the program.

1. MainApplication.java

SAMPLE CODE

```
package com.entranscend.erp.services;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.company.minierp.MinierpApplication;
import com.company.minierp.minierp.minierp.user.User;
import com.company.minierp.minierp.minierp.user.UserManager;

@Service
public class UserService {

        @Autowired
        MinierpApplication minierpApplication;
        @Autowired
        UserManager userManager;

        public long getUser(User user)//Used to get the Users for Logging in the syatem
        {
                long c = userManager.stream().filter(user.FIRST_NAME.equalIgnoreCase(user.getFirstName()))

        .filter(user.PASSWORD.equalIgnoreCase(user.getPassword())).count();
                return c;
        }

        public void putUser(User user)//Used to set(create) the Users
        {
                userManager.persist(user);
        }

        public long checkUser(User user)//Used to check for the existing Users
        {
                long c = userManager.stream().filter(user.FIRST_NAME.equalIgnoreCase(user.getFirstName()))

        .filter(user.PHONE_NUMBER.equal(user.getPhoneNumber())).count();
                return c;
        }

        public void updatePassword(User user)//Used to update the existing Users password
        {
                 userManager.stream()
                                        .filter(user.FIRST_NAME.equalIgnoreCase(user.getFirstName()))
                                        .filter(user.PHONE_NUMBER.equal(user.getPhoneNumber()))
                                        .map(user.PASSWORD.setTo(user.getPassword()))
                                        .forEach(userManager.updater());
        }
}
```

```java
package com.entranscend.erp.controller;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RestController;
import com.company.minierp.minierp.minierp.user.User;
import com.entranscend.erp.services.UserService;

@RestController
@CrossOrigin("*")
public class UserController {

	@Autowired
	UserService userService;

	@PostMapping("/getUser")
	public long getUser(@RequestBody User user)//get the users to login
	{
		long c=userService.getUser(user);
	   if(c==0)
			return 0;
	   return 1;
	}

	@PostMapping("/putUser")
	public String putUser(@RequestBody User user)//set(create) the user
	{
		long c=userService.checkUser(user);//Checking for existing users
		if(c==0)
			userService.putUser(user);
		else
			return "User already exist!";
		return "User Created Succesfully";
	}

	@PostMapping("/updatePassword")
	public String updatePassword(@RequestBody User user)//update the User's password
	{
		long c=  userService.checkUser(user);//Checking for existing users
		if(c==0)
			return "User or Password does not match!";
		else
			userService.updatePassword(user);
		return "New Password Created!";
	}
}
```

FUTURE SCOPE

Developments in software technology are continuing dynamically. This has forced developers to look for new approaches to design and development. In order to face this situation, the modules in a package should be upgraded any time. The modules in this package can be subjected to further enhancements.

Some of the other activities that can be implemented in this application are:-

- Hosting the application on the cloud server.
- Integration with GST module.
- Integration with HRMS (Human Resource Management System) module, which will have all the details about the employees and their respective departments.
- Can also have an accounting, payroll, receivables and other payments system which will generate invoices and receipts for the customers and employees.
- Can be implemented as a mobile application.
- More user friendly system.
- Intelligence Reporting.
- A new page for taking orders from the customer for reference.
- Customer self service for online ordering and accessing product information will encourage greater level of customer satisfaction.
- Use of bar-code-scanner to record an inventory transaction.

CONCLUSION

Small Inventory Planning (SIP) is a web-based application. The key concept is to manage the customers, products, inventory for a Retail Shop. Another advantage of going digital will reduce billing errors, and provide automatic inventory management. To manage inventory is one of the most time-consuming and labour-intensive tasks every store owner faces, SIP can control the inventory digitally, which will be very easy to manage inventory rather to have a labour-intensive task.

By working in this project, we have gathered knowledge about how to develop a web based software for the Small Inventory Planning using coding standards followed by Angular. This project when efficiently implemented will save the effort, time and human resources spent in the traditional way of labour-intensive tasks of a Retail shop.

We tried our best to make the project a useful one; however all the requirements could not be covered, and thus they are kept as a future scope.

REFERENCES

The following links were used by us for learning:-

- https://stackoverflow.com/
- https://www.google.com/
- https://angular.io/guide/architecture
- https://www.speedment.com/resources/user-guide/
- https://spring.io/guides/gs/spring-boot/

And other books like:

- Core Java - Vol. I – Fundamentals by Cay S. Horstmann
- Beginning Angular with Typescript – by Greg Lim