**INVENTORY MANAGEMENT SYSTEM**

**Team D –**

**Surya Kamal (18MIS0090 VIT VELLORE)**

**Srikar Kotra (18MIS0369 VIT VELLORE)**

**Prateek Bapat (18BEE0190 VIT Vellore)**

**Ashish Kumar (18BEC1294 VIT CHENNAI)**

**Introduction:**

Inventory Management System is a Spring software system to create and keep track of different inventories and to manage them. Here, with our Inventory Management System customer is the vendor. He can register an Inventory with its price and number of items. They have the feasibility to update the number of items and prices or to delete an inventory. They need to register and login to access their inventory.

**Literature Survey:**

Depending on the type of business or product, various inventory management methods are used. Some of these management methods include Material Requirement Planning (MRP), Just in Time (JIT), etc.

Material Requirement Planning (MRP)–

It is a computer-based inventory management system designed to improve productivity for businesses. It is used to track estimate of raw materials and to schedule their deliveries. This is the earliest computer-based inventory management system. Advantages of MRP include the assurance of availability of materials when needed, reduced customer lead times, improved customer satisfaction. Disadvantage to MRP process is its heavy reliance on estimate data, high cost to implement, and a lack of flexibility.

Just In Time (JIT)–

JIT inventory refers to an inventory management system with objective of having the number of items and its inventory readily available to meet demand, but not to a point of excess where an inventory is stockpiled. This strategy is generally used to increase efficiency and to decrease waste by receiving goods as they are needed, hence reducing the inventory costs.

There are two main types of Inventory control systems Perpetual Inventory System and Periodic Inventory System.

Perpetual Inventory System –

In this type of Inventory Control System, it continually updates inventory records and accounts of addition subtraction of inventory items as and as changed. Hence it also has its disadvantages as it requires a specialized equipment or software to continually update records that results in higher cost of implementation.

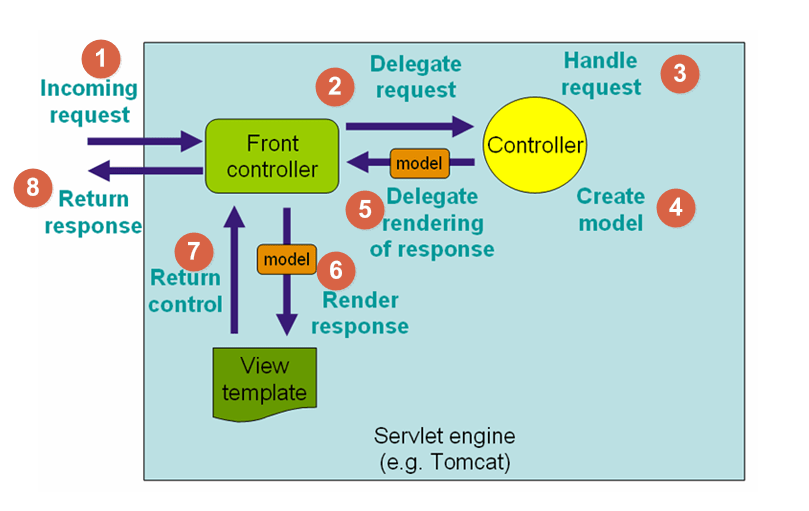
Periodic Inventory System –

Periodic inventory systems do not track inventory on a daily basis; instead, they allow businesses to know the beginning and ending inventory levels over a specific time period. Physical inventory counts are used to track inventory in these sorts of inventory control systems. When physical inventory is finished, the balance in the purchasing account is transferred to the inventory account and the cost of the ending inventory is modified.

Our Inventory Management System Project uses the Perpetual Inventory System. Here the special software used is Java Spring Software to update and record inventory as and when the registered customer (here the vendor) modifies.

Java Spring Boot is an open-source Java-based framework used to create a micro service dedicated to do a specialized task. Hence, used to build stand alone and production ready spring applications.

**Theoretical Analysis:**



**Hardware Requirements**:

Intel core i5 2nd generation is used as the processor because it is a quite fast processor providing reliability and stability to run the pc without any hassle. By using this processor, we can keep on developing our project without any issues.

1 GB of RAM is required as it provides with quick reading and writing of data which in turn leads to smooth processing of the application.

**Software Requirements:**

* Spring MVC
* Thymeleaf
* H2 Database
* JDBC API
* Spring Security
* JPA
* MAVEN
* HEROKU
* DOCKER

**Experimental Investigations –**

We started and created a new project on Spring Boot Tools.

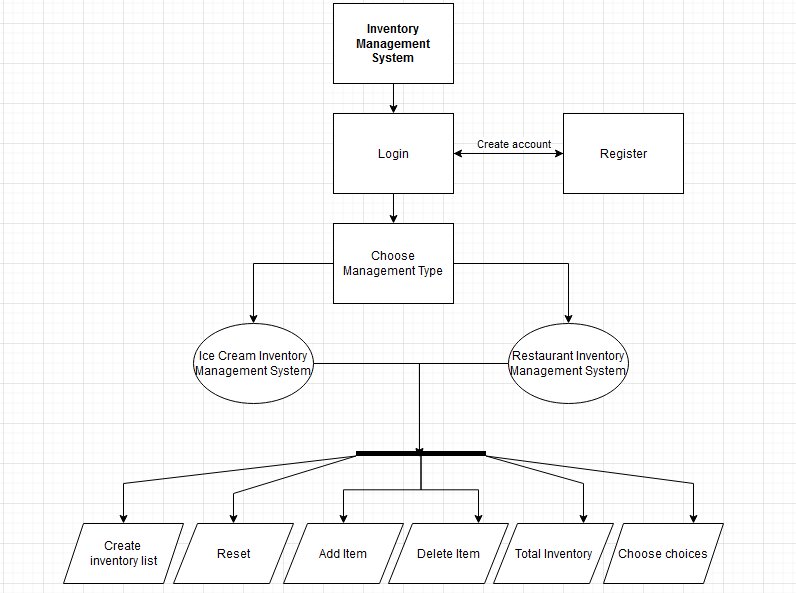
Software-based systems offer the benefits of having the most effective control with the least amount of work.

These advancements have resulted in new solutions for inventory management systems, and a new Inventory Management System (IMS) has been built and implemented as a result of these improvements.

We coded the system using MVC Architecture. It gives ability to program faster with less effort.

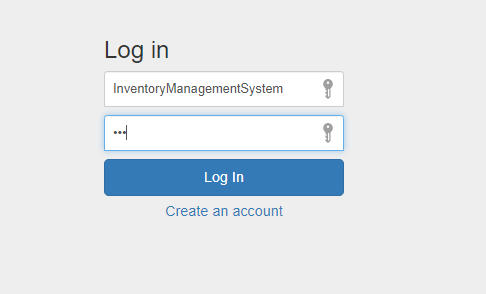
MVC is a software architecture which provides to implement the visual, data and processing code parts independent. Model unit is a collection of classes in communication with the other parts. It is the process unit, processes the task ordered by the Control Unit. View unit is the place to deal with the presentation of the data to the end user. It can get the data from both Model and Controller unit. Also it can send interactive data to both units. Controller part is the main part of the structure.

**Flow Chart –**

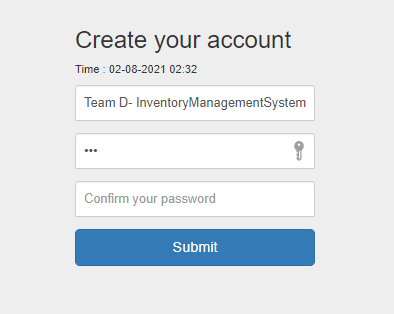


**Result:**

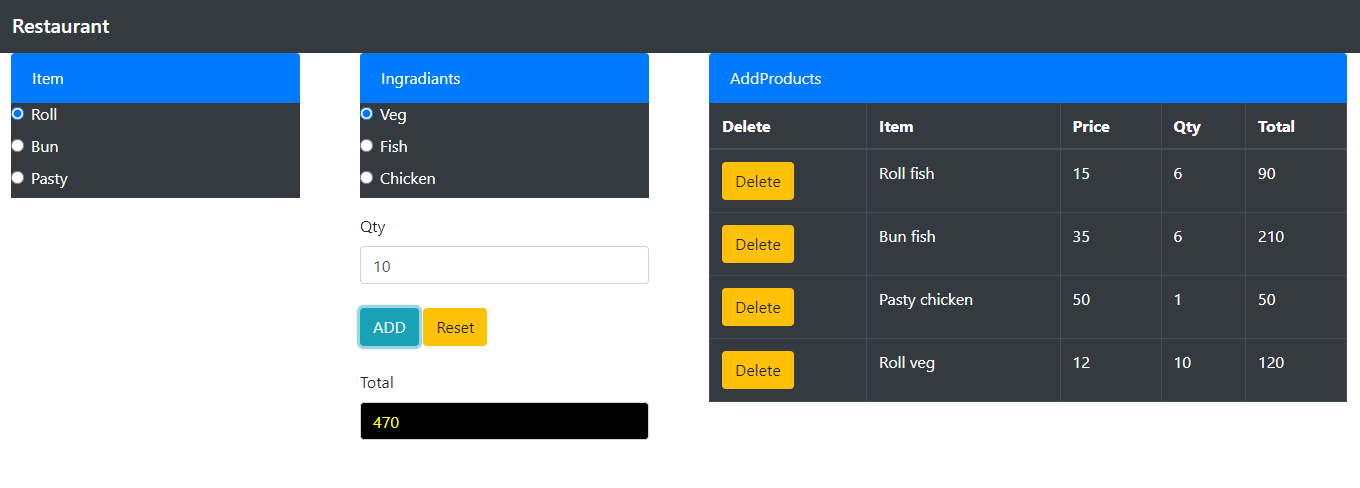
Login:



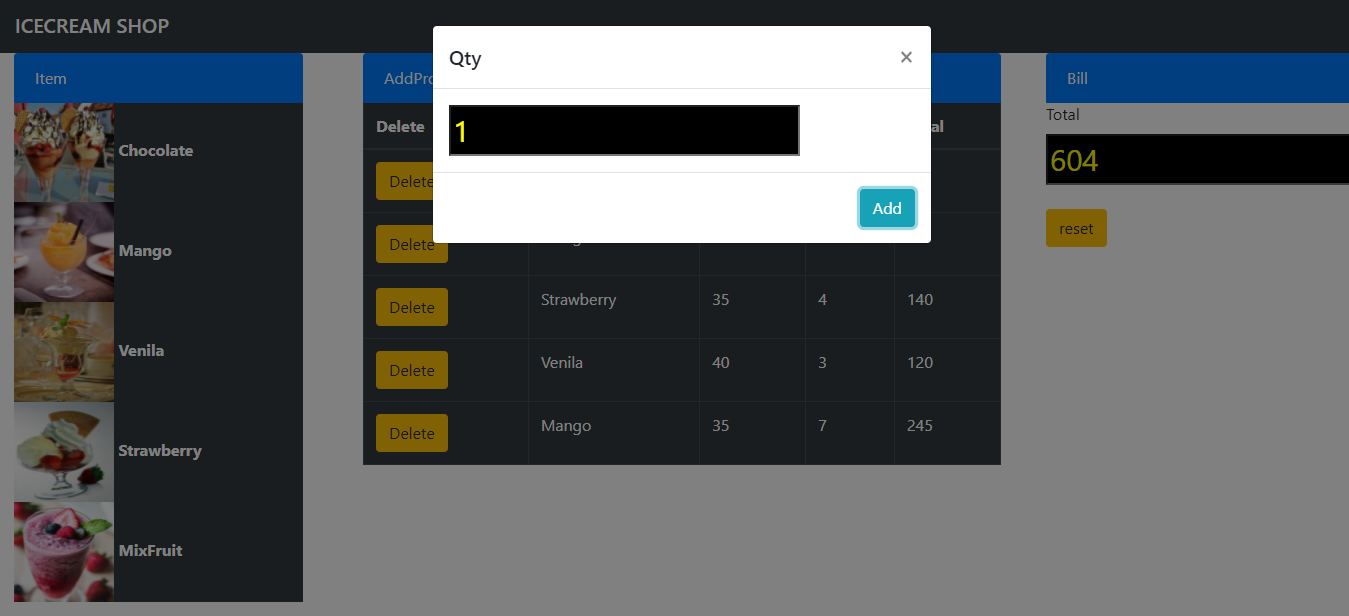
Register:



Restaurant Inventory:



IceCream Inventory:



**Advantages and Disadvantages:**

**Advantages include:**

The advantages of advanced inventory management software, which is utilised by many types of retail, wholesale, and manufacturing businesses, cannot be overstated. Inventory management software helps organisations eliminate the clog by facilitating appropriate administration, real-time inventory monitoring, financial integration, extensive analytics, and timely provision of services. Inventory management software enables proper management, real-time inventory monitoring, financial integration, extensive analytics, and timely service supply, allowing firms to reduce the complexity of their operations and advance strategically.

* Integration with all the necessary back-end systems with the accounting - Advanced inventory management software allows for synchronisation between any back-end systems, such as inventory software, e-commerce platforms, and accounting systems. As a result, data input needs are substantially reduced, order processing is made even easier, and accurate and comprehensive financial reports are available on demand.
* Provide details for proper planning and forecasting – An Inventory management system can do more than just storing inventory data and prices. With some analysis of continuously acquired data it can also predict future inventory and their prices, hence also helping with proper planning and forecasting.
* Utilizes cloud-based inventory monitoring throughout various locations - The use of cloud-based inventory management makes centralised inventory management from numerous channels easier. Store owners and employees may use a single dashboard to track orders and complete a variety of business operations, regardless of where they are in the world.
* Eliminate the risk of human errors - Most procedures, such as inventory management, ordering, and so on, rely heavily on people, especially when a business is just getting started. Human error, on the other hand, is to blame for stock-level inaccuracies. And the cost of such mistakes will swiftly mount up. Hence, by digitalizing the process of Inventory Management the risk of human error is eliminated.

**Disadvantages include:**

* Prone to system crash - Every computerised system must be concerned about the danger of a system crash. A corrupted hard disc, power outages, and other technological issues may result in the loss of critical data. At the very least, corporate activities are hampered as a result of a lack of data availability. To protect against system failure, business owners must back up their data on a regular basis.
* At high risk for malicious hacking - Hackers and cybercriminals are always seeking for new ways to gain information about a company or a consumer. In the search for substantial financial data or personal information of suppliers or consumers, an inventory platform linked to point-of-sale systems and accounting is a valuable resource to be hacked into.

**Applications:**

The Inventory Management system is a Java Spring Software targeted to small and medium organizations. Here, the Customer is the Vendor and has his power of authority. Some of the Scope are –

* Only one person responsible for modifying Inventory.
* It is security driven.
* Can be used to track items in a warehouse.
* Can also be used to keep track of items release with the help of previous data.

**C****onclusion:**

To conclude, Inventory Management System is a very simple Java Spring Boot Software basically suitable for small organization. It contains all of the essential items for a small business. Our team has succeeded in developing an application that allows us to edit, insert, and delete items as needed.

Though it has its limitations and not fully developed, our team strongly believes that the implementation of this system will surely benefit the said organizations.

**Future Scope:**

The final integration of the project can be achieved and data being collected can be further analysed in order to predict and forecast the inventory and its prices. So as to further aid in planning of inventory while managing it.

**Bibliography:**

1 - Ziukov, Serhii. "A literature review on models of inventory management under uncertainty." (2016).

2 - Hassan, Ummi Kalsom, Shahreen Kasim, Rohayanti Hassan, Hairulnizam Mahdin, Azizul Azhar Ramli, Mohd Farhan Md Fudzee, and Mohamad Aizi Salamat. "Most Stationery Inventory Management System." Acta Electronica Malaysia 2, no. 2 (2018): 10-13.

**Appendix:**

**Code:**

**Controller: (IceController.java)**

package com.example.IceCreamInventory;

import java.time.LocalDateTime;

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

import org.springframework.stereotype.Controller;

import org.springframework.ui.Model;

import org.springframework.validation.BindingResult;

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

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

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

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

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

import com.example.IceCreamInventory.User;

import com.example.IceCreamInventory.SecurityService;

import com.example.IceCreamInventory.UserService;

import com.example.IceCreamInventory.UserValidator;

@Controller

public class IceController {

@RequestMapping(value = "/index", method = RequestMethod.GET)

public String index()

{

return "index";

}

@PostMapping("/first")

public String ice(User user)

{

System.out.println(user.getUsername());

System.out.println(user.getEmail());

return "ice";

}

@PostMapping("/first")

public String rest(User user)

{

System.out.println(user.getUsername());

System.out.println(user.getEmail());

return "rest";

}

@Autowired

private UserService userService;

@Autowired

private SecurityService securityService;

@Autowired

private UserValidator userValidator;

@GetMapping(value = "/registration")

public String registration(Model model) {

model.addAttribute("userForm", new User());

model.addAttribute("waktu", LocalDateTime.now());

return "registration";

}

@PostMapping(value = "/registration")

public String registration(@ModelAttribute("userForm") User userForm, BindingResult bindingResult, Model model) {

userValidator.validate(userForm, bindingResult);

if (bindingResult.hasErrors()) {

return "registration";

}

userService.save(userForm);

securityService.autoLogin(userForm.getUsername(), userForm.getPasswordConfirm());

return "login";

}

@GetMapping(value = "/login")

public String login(Model model, String error, String logout) {

if (error != null)

model.addAttribute("error", "Your username and password is invalid.");

if (logout != null)

model.addAttribute("message", "You have been logged out successfully.");

return "login";

}

@GetMapping(value = {"/", "/login"})

public String welcome(Model model) {

return "first";

}

}

**ICECREAM.html**

<html>

<head>

<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>

</head>

<body>

<nav class="navbar navbar-dark bg-dark">

<span class="navbar-brand mb-0 h1">ICECREAM SHOP POS</span>

</nav>

<div class="row">

<div class="col-sm-3">

<div class="container">

<div class="list-group-item list-group-item-action active">Item</div>

<div class="panel-body bg-dark" style="color: white">

<form id="frm-project">

<div class="form-group">

<div>

<img src="images/choc.jpg" id="Chocolate" class="photo" width="100" height="100" data-toggle="modal" data-target="#exampleModal">

<b>Chocolate</b>

</div> <div>

<img src="images/mango.jpg" id="Mango" class="photo" width="100" height="100" data-toggle="modal" data-target="#exampleModal">

<b>Mango</b>

</div> <div>

<img src="images/venila.jpg" id="Venila" class="photo" width="100" height="100" data-toggle="modal" data-target="#exampleModal">

<b>Venila</b>

</div> <div>

<img src="images/sww.jpg" id="Strawberry" class="photo" width="100" height="100" data-toggle="modal" data-target="#exampleModal">

<b>Strawberry</b>

</div> <div>

<img src="images/mixfruit.jpg" id="MixFruit" class="photo" width="100" height="100" data-toggle="modal" data-target="#exampleModal">

<b>MixFruit</b>

</div> </div> </form> </div> </div> </div>

<div class="col-sm-6">

<div class="container">

<div class="list-group-item list-group-item-action active">AddProducts</div>

<table id="tbl-item" class="table table-dark table-bordered" cellpadding="0" cellspacing="0" width="100%" align="center">

<thead>

<tr>

<th>Delete</th>

<th>Item</th>

<th>Price</th>

<th>Qty</th>

<th>Total</th>

</tr>

<tbody> </tbody> </table> </div> </div >

<div class="col-sm-3">

<div class="list-group-item list-group-item-action active">Bill</div>

<div>

<label>Total</label>

<input type="text" style="color: yellow; background: black; font-size: 30px;" id="total" name="total" placeholder="Total" required>

</div> </br> <div>

<input type="button" class="btn btn-warning" value="reset" name="reset" id="reset">

</div> </div> </div>

<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel" aria-hidden="true">

<div class="modal-dialog" role="document">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="exampleModalLabel">Qty</h5>

<button type="button" class="close" data-dismiss="modal" aria-label="Close">

<span aria-hidden="true">&times;</span>

</button>

</div>

<div class="modal-body">

<div>

<input type="number" style="color: yellow; background: black; font-size: 30px;" id="qty" name="qty" placeholder="Qty" required>

</div> </div>

<div class="modal-footer">

<input type="button" class="btn btn-info" value="Add" name="add" id="add" onclick="add()">

</div> </div> </div></div>

<script src="component/jquery/jquery.js"></script>

<script src="component/jquery/jquery.min.js"></script>

<script type="text/javascript">

var total = 0;

var tot = 0;

var item = null;

var price = 0;

var a = "";

$("img").on("click",function()

{

a = $(this).prop('id')

});

function add()

{

if(a == "Chocolate")

{

item = "Chocolate";

price = 32; }

else if(a == "Mango")

{

item = "Mango";

price = 35;

}

else if(a == "Venila")

{

item = "Venila";

price = 40;

}

else if(a == "Strawberry")

{

item = "Strawberry";

price = 45;

}

else if(a == "MixFruit")

{

item = "Strawberry";

price = 35;

}

var qty = $('#qty').val();

tot = qty \* price;

var table1 =

"<tr>" +

"<td><button type='button' name='record' class='btn btn-warning' onclick='deleterow(this)'>Delete</td>" +

"<td>" + item + "</td>" +

"<td>" + price + "</td>" +

"<td>" + qty + "</td>" +

"<td>" + tot + "</td>" +

"</tr>" ;

total += Number(tot);

$('#total').val(total);

$("table tbody").append(table1);

$("exampleModal").modal('toggle');

$('#qty').val("1");

}

function deleterow(e)

{

total\_cost = parseInt($(e).parent().parent().find('td:last').text(),10);

total -= total\_cost;

$('#total').val(total);

$(e).parent().parent().remove();

}

$('#reset').click(function()

{

location.reload();

});

</script>

</body>

</html>

**Restaurant.html**

**<html>**

**<head>**

**<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">**

**<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>**

**<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>**

**<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>**

**</head>**

**<body>**

**<nav class="navbar navbar-dark bg-dark">**

**<span class="navbar-brand mb-0 h1">Restaurant</span>**

**</nav>**

**<div class="row">**

**<div class="col-sm-3">**

**<div class="container">**

**<div class="list-group-item list-group-tem-action active">Item</div>**

**<div class="panel-body bg-dark" style="color: white">**

**<form id="frm-project">**

**<div class="form-group">**

**<div class="form-check">**

**<input class="form-check-input" type="radio" name="pos" id="pos" value="Roll" checked>**

**<label>Roll</label>**

**</div>**

**<div class="form-check">**

**<input class="form-check-input" type="radio" name="pos" id="pos" value="Bun" checked>**

**<label>Bun</label>**

**</div>**

**<div class="form-check">**

**<input class="form-check-input" type="radio" name="pos" id="pos" value="Pasty" checked>**

**<label>Pasty</label>**

**</div> </div> </form> </div> </div>**

**</div>**

**<div class="col-sm-3">**

**<div class="container">**

**<div class="list-group-item list-group-tem-action active">Ingradiants</div>**

**<div class="panel-body bg-dark" style="color: white">**

**<form id="frm-project">**

**<div class="form-group">**

**<div class="form-check">**

**<input class="form-check-input" type="radio" name="ind" id="ind" value="veg" checked>**

**<label>Veg</label>**

**</div>**

**<div class="form-check">**

**<input class="form-check-input" type="radio" name="ind" id="ind" value="fish" checked>**

**<label>Fish</label>**

**</div>**

**<div class="form-check">**

**<input class="form-check-input" type="radio" name="ind" id="ind" value="chicken" checked>**

**<label>Chicken</label>**

**</div> </div> </form> </div> <div>**

**<label>Qty</label>**

**<input type="number" id="qty" name="qty" placeholder="Qty" class="form-control" required>**

**</div> </br> <div> <input type="button" class="btn btn-info" value="ADD">**

**<input type="button" class="btn btn-warning" value="Reset" name="reset" id="reset"**

**</div> </br> <div>**

**<label>Total</label>**

**<input type="text" style="color: yellow; background: black" id="total" name="total" placeholder="Total" class="form-control" required>**

**</div> </div> </div>**

**<div class="col-sm-6">**

**<div class="container">**

**<div class="list-group-item list-group-tem-action active">AddProducts</div>**

**<table id="tbl-item" class="table table-dark table-bordered" cellspacing="0" width="100%" align="center">**

**<thead>**

**<tr>**

**<th>Delete</th>**

**<th>Item</th>**

**<th>Price</th>**

**<th>Qty</th>**

**<th>Total</th>**

**</tr>**

**<tbody>**

**</tbody>**

**</table>**

**</div> </div></div>**

**<script type="text/css" src="component/jquery/jquery.js"></script>**

**<script type="text/css" src="component/jquery/jquery.min.js"></script>**

**<script type="text/javascript">**

**var total = 0;**

**var tot = 0;**

**$(document).ready(function() {**

**$("input[type='button']").click(function(){**

**var radio1 = $("input[name='pos']:checked").val();**

**var radio2 = $("input[name='ind']:checked").val();**

**var price = null;**

**if(radio1 == "Roll")**

**{**

**if(radio2 == "veg")**

**{**

**price = 12;**

**}**

**else if(radio2 == "fish")**

**{**

**price = 15;**

**}**

**else if(radio2 == "chicken")**

**{**

**price = 30;**

**}**

**}**

**else if(radio1 == "Bun")**

**{**

**if(radio2 == "veg")**

**{**

**price = 30;**

**}**

**else if(radio2 == "fish")**

**{**

**price = 35;**

**}**

**else if(radio2 == "chicken")**

**{**

**price = 40;**

**} }**

**else if(radio1 == "Pasty")**

**{**

**if(radio2 == "veg")**

**{**

**price = 40;**

**}**

**else if(radio2 == "fish")**

**{**

**price = 45;**

**}**

**else if(radio2 == "chicken")**

**{ price = 50; } }**

**var qty = $('#qty').val();**

**tot = qty \* price;**

**var table1 =**

**"<tr>" +**

**"<td><button type='button' name='record' class='btn btn-warning' onclick='deleterow(this)'>Delete </td>" +**

**"<td>" + radio1 + " " + radio2 + "</td>" +**

**"<td>" + price + "</td>" +**

**"<td>" + qty + "</td>" +**

**"<td>" + tot + "</td>" +**

**"</tr>" ;**

**total += Number(tot);**

**$('#total').val(total);**

**$("table tbody").append(table1);**

**});**

**});**

**function deleterow(e)**

**{**

**total\_cost = parseInt($(e).parent().parent().find('td:last').text(),10);**

**total -= total\_cost;**

**$('#total').val(total);**

**$(e).parent().parent().remove();**

**}**

**$('#reset').click(function()**

**{**

**location.reload();**

**}**

**);**

**</script>**

**</body>**

**</html>**