Food Box

Capstone Final Project by Rushikesh Nikam

Email id: rushikesh19nikam@gmail.com

# PROJECT INTRODUCTION:

Food-Box is an E-commerce website for managing products in a portal and selling different Food items to customers online.

In this document, you can see the prototype of product operations for both page Admin and Customer interaction.

The Project is divided into 2 Sprints each of one week.

# Product Back-Log:

* **Ticket Number(CAP001):-Admin Login Form**

**Description**:-Create a login form

**Acceptance Criteria**:-Should create an end-to-end full functional Login form

* **Ticket Number(CAP002):-User Login Form & Registration Form**

**Description**:-Create a login form

**Acceptance Criteria**:-Should create an end-to-end full functional Login form & registration form

* **Ticket Number(CAP003):-create a Home-Page**

**Description**:-Create Home Page

**Acceptance Criteria**:-Should create a fully functional Home page where all the products are listed

* **Ticket Number(CAP004):-Create manage products, manage purchases, manage customers**

**Description**:-Create manage products, manage purchases, manage customers pages

**Acceptance** **Criteria**:-Should create an end-to-end full functional pages

* **Ticket Number(CAP005):-Add Backend Functionality for view & update cart, make purchase**

**Description**:-Update Backend logic

**Acceptance Criteria**:-Should add backend logic

* **Ticket Number(CAP006):-Add Backend Functionality for viewing customers, updating products, managing purchase**

**Description**:-Update backend logic

**Acceptance Criteria**:-Should add backend logic

* **Ticket Number(CAP007):-Add search logic for required pages**

**Description**:-Update backend logic

**Acceptance Criteria**:-Should add backend logic

# Sprint Back-Log:

**1st Sprint:-**

* **Ticket Number(CAP001):-**

Created Admin Login Form with single [UserId=admin@gmail.com](mailto:UserId=admin@gmail.com) and Password=123 where Admin can manage products, customers, and purchases. Created backend with spring boot which handles the login Credentials

* **Ticket Number(CAP002):-**

Created User Login Form & Register Form where users can log in and browse products and can register. Created backend with spring boot which handles login credentials

* **Ticket Number(CAP003):-**

Created a Home page where users can see all the products. Created view cart, view active orders, and manage cart pages.

* **Ticket Number(CAP004):-**

Created managed products, manage customers, and manage purchases pages.

**2nd Sprint:-**

* **Ticket Number(CAP005):-**

Added User backend functionality for viewing the cart, updating the cart, make a purchase.

* **Ticket Number(CAP006):-**

Added Admin backend functionality for viewing customers, updating products, and managing purchases.

* **Ticket Number(CAP007):-**

Added backend functionality for all search results, update & delete functions

# GitHub Link:- https://github.com/RakeshGondala/SimplilearnCapstonProject\_FoodBox.git

# PRODUCT CAPABILITIES:

## Admin Operations:

* **Admin Login**: which is authorized according to data in the database.
* **Change Password**: Admin needs to enter the old password to authorize.
* **Manage Products**: Add, Delete, and Update Products.
* **Manage Customers**: View, Delete and Search Customers.
* **Manage Purchases/Orders**: View, Delete and Search Orders.

## Customer Operations:

* Register
* Login
* Search Products
* Choose quantity and category
* Add Cart
* View Cart
* Pay and Buy Products
* View previous active orders.

# TECHNOLOGIES USED:

* Eclipse IDE
* HTML
* MySQL
* Java Concepts
  + Spring Boot DevTools
  + Spring Web
  + Spring Data JPA
  + ThymeLeaf

## 

## Spring Concepts Used in Projects:

@SpringBootApplication: To initialize spring boot.

@Controller: for using class as a controller class

@Service: To indicate class as Service

@Repository: To indicate class/interface as Repository to contact with Database.

@Entity: To indicate class as the table in Database.

@Autowired: to auto-connect between Spring Beans, Services, and Repositories.

@PostMapping: to indicate URL links with Servlet post method

@GetMapping: to indicate URL links with the Get method in servlet.

@RequestParam and @RequestBody: Get values from the webpage.

javax.servlet.http.*HttpSession*: To manage Sessions with HTTP protocol.

org.springframework.ui.*Model*: to send data to view

java.sql.SQL*Exception:* To manage Database exceptions

Java.util.*regex*: to check string patterns like email.

JpaRepository: to get methods for CRUD operations.

jpa.repository.Query (@Query): To write native queries for custom methods for CRUD operations.

ThymeLeaf Template tags in HTML.

# FEATURES OF PROJECT:

Customers should log-in or Register to start shopping.

Customers can check current Items in the Cart and Previous orders.

Customers can Search for products by Product name, brand, or category.

Admin Login is verified by data from Database.

In the Manage Customers section admin can view the orders of a particular customer.

In the Manage Purchase(Purchase Report) section Admin can search according to the Date of purchase or Category of shoes.

# CONCLUSION:

As this is a prototype application appearance may not be much awestruck. Please try to evaluate the application’s operations and send us feedback.

**THANK YOU**

**Rushikesh Nikam.**