**Final Project: Kitten Factory Ski Manufacturing**

Group 1: Luke Coffman, Jaxon Driver, Emma Lare

## Table of Contents

[Table of Contents](#_hp92tn10z00z)

[Use Case Model](#_d3h885bj5saj)

[Use Case Documents](#_b9o6tubw127v)

[Activity Diagrams](#_76hep4t3mzyg)

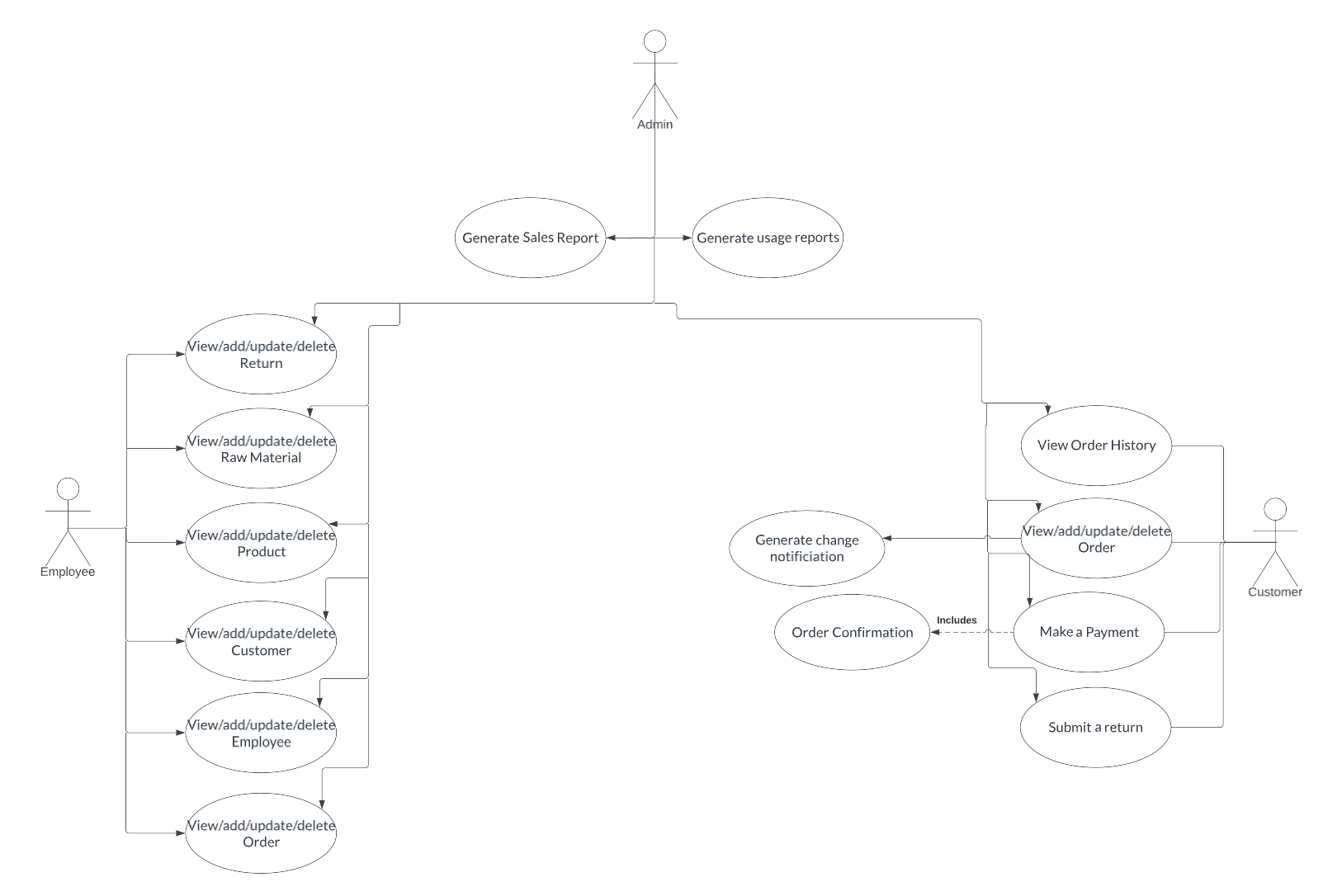
[Wire frame diagrams](#_ihv2dke546lh)

[ERD](#_txmmr611ji7m)

[Is the business objective(s) achieved?](#_3urfbitwczgn)

[Ethics and Critical Thinking](#_pvynsjofusrl)

## Use Case Model



## Use Case Documents

| Use Case #1 | View/add/update/delete a new employee | |
| --- | --- | --- |
| Objective | Employee Views/adds/updates/deletes employee(s) in the system | |
| Business Event | Views/adds/updates/deletes employee(s) in the system | |
| Primary Actor (s) | Employee | |
| Secondary Actor (s) | None. | |
| Pre-condition | Employee must be logged into the system.  Employee clicks on the ‘Edit Employees’ link in the home page. | |
| Post-condition | User is forwarded to the View Employee List Page. | |
| Failure outcomes | Employee is in the add new Employee page.  Employee is shown an error message ‘Error in adding a new Employee’. | |
| Flow of Events | Actor | System |
| 1 | Employee in the home page of the website(home.php).  User clicks on the edit employees page (edit -employees.php). | Edit Employees page is shown. |
| 2 | Employee clicks on either view/add/update/delete employee details. | Employee is shown perspective link |
| 3 | Employee submits the prompted form. | Connect to the database.  Insert employee details into the Employee table.  Forward employee to the view-employee-list.php page. |
| 4 | Employee sees the view-employee-list.php page. |  |

| Use Case #2 | Customer can view/add/update/delete an order | |
| --- | --- | --- |
| Objective | View/add/update/delete an order into the system | |
| Business Event | View/add/update/delete an order into the system | |
| Primary Actor (s) | Customer | |
| Secondary Actor (s) | None | |
| Pre-condition | Customer must be logged into the system, or consents to operating as a ‘guest’.  There must be at least one item in the cart before order can be submitted. | |
| Post-condition | User is forwarded to the payment information page. | |
| Failure outcomes | Item is out of stock. | |
| Flow of Events | Actor | System |
| 1 | Customer is on the homepage of the website.  User clicks on shopping cart. | Order detail page is shown. |
| 2 | Customer clicks on either view/add/update/delete order. | Customer is shown perspective links |
| 3 | Customer submits the corresponding order. | Connect to the database  Pull relevant information from customer and product tables.  Forward customer to the payment page. |
| 4 | Customer sees the payment page. |  |

| Use Case #3 | View/add/update/delete Raw Materials | |
| --- | --- | --- |
| Objective | View/add/update/delete Raw Material(s) into the system | |
| Business Event | View/add/update/delete Raw Material(s) into the system | |
| Primary Actor (s) | Employee | |
| Secondary Actor (s) | None | |
| Pre-condition | Employee must be logged into the system.  Employee clicks on the ‘Edit Raw Materials’ link in the home page | |
| Post-condition | User is forwarded to the View Raw Materials List Page. | |
| Failure outcomes | Employee is in the Raw Materials List Page  Employee is shown an error message “No Raw Materials in stock/available” | |
| Flow of Events | Actor | System |
| 1 | Employee is in the home page of the website (home.php)  User clicks on the Raw Materials List page (raw-materials-list.php) | Raw Materials List page is shown |
| 2 | Employee clicks on either view/add/update/delete Raw Materials | Employee is shown perspective link |
| 3 | Employee submits the prompted form | Employee is connected to the Raw Materials table in the database  Raw Materials are updated as used |

| Use Case #4 | View/add/update/delete Products | |
| --- | --- | --- |
| Objective | View/add/update/delete Product(s) into the system | |
| Business Event | View/add/update/delete Product(s) into the system | |
| Primary Actor (s) | Employee | |
| Secondary Actor (s) | None | |
| Pre-condition | Employee must be logged into the system.  Employee clicks on the ‘Edit Products’ link in the home page. | |
| Post-condition | User is forwarded to the Products List Page | |
| Failure outcomes | Employee is in the Products List page.  Employee is shown an error message ‘Unable to display products’ | |
| Flow of Events | Actor | System |
| 1 | Employee is in the home page of the website (home.php)  User clicks on the Products page (products.php) | Products List page is shown |
| 2 | Employee clicks on either view/add/update/delete Product details. | Employee is shown perspective link |
| 3 | Employee submits the prompted form | Employee is connected to the Products table in the database  Products are updated as sold |

| Use Case #5 | View/add/update/delete Customers | |
| --- | --- | --- |
| Objective | View/add/update/delete Customer(s) into the system | |
| Business Event | View/add/update/delete Customer(s) into the system | |
| Primary Actor (s) | Employee | |
| Secondary Actor (s) | None | |
| Pre-condition | Employee must be logged into the system.  Employee clicks on the ‘Edit Customers’ link in the home page | |
| Post-condition | User is forwarded to the View Customers List Page | |
| Failure outcomes | Employee is in the Customers List Page  Employee is shown an error message “Unable to display Customers” | |
| Flow of Events | Actor | System |
| 1 | Employee is in the home page of the website (home.php)  User clicks on the Customers List page (customers-list.php) | Customers List page is shown |
| 2 | Employee clicks on either view/add/update/delete Customers | Employee is shown perspective link |
| 3 | Employee submits the prompted form | Employee is connected to the Customers table in the database  Customers are updated as they purchase products |

| Use Case #6 | Generate Sales Report | |
| --- | --- | --- |
| Objective | Generate a Sales Report from sales recorded in the system | |
| Business Event | Generate a Sales Report from sales recorded in the system | |
| Primary Actor (s) | Admin | |
| Secondary Actor (s) | None | |
| Pre-condition | Admin must be logged into the system  Admin looks at the OrderLine to determine Quantity and Price | |
| Post-condition | Based off the Quantity and Price the Admin generates a sales report | |
| Failure outcomes | Admin is unable to retrieve accurate quantity and price so the sales report is inaccurate | |
| Flow of Events | Actor | System |
| 1 | Admin is in the home page of the website (home.php)  Admin clicks ‘Generate Report’ |  |
| 2 | Admin accesses the Quantity and Price within the Orders table and Products table | In the Database the data for Quantity and Price is displayed in the corresponding tables |
| 3 | Admin generates the Sales Report from multiplying the Quantity and Price | Quantity and Price are multiplied |
| 4 | Admin Generates Sales Report |  |

| Use Case #7 | Generate Popularity Reports | |
| --- | --- | --- |
| Objective | Generate purchasing reports based on type of product sold to display | |
| Business Event | Generate purchasing reports based on type of product sold to display | |
| Primary Actor (s) | Admin | |
| Secondary Actor (s) | None | |
| Pre-condition | There must be data of products purchased and quantity | |
| Post-condition | A display is shown of the most popular and least popular products | |
| Failure outcomes | There is no data for a selected category so an error will be thrown | |
| Flow of Events | Actor | System |
| 1 | Admin is in the Admin logged on page of the website(Admin-logged-on.php).  Admin clicks on the generate popularity report (popularity-report.php). | System has authenticated Admin |
| 2 | Admin selects desired products to display and time period | Selections are stored and sent to php command |
| 3 | An Admin selects display and a display is generated for inputted data | A select query is ran and displays data based on admins choices |

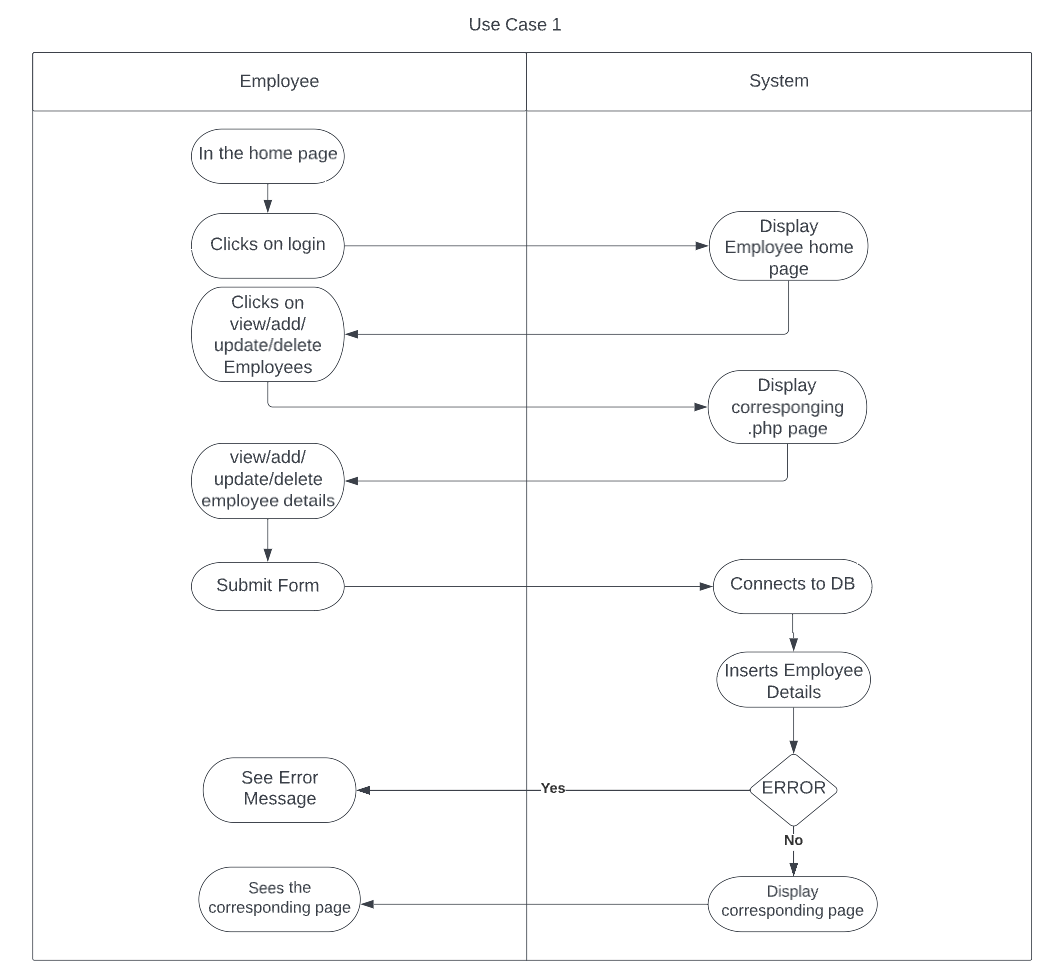
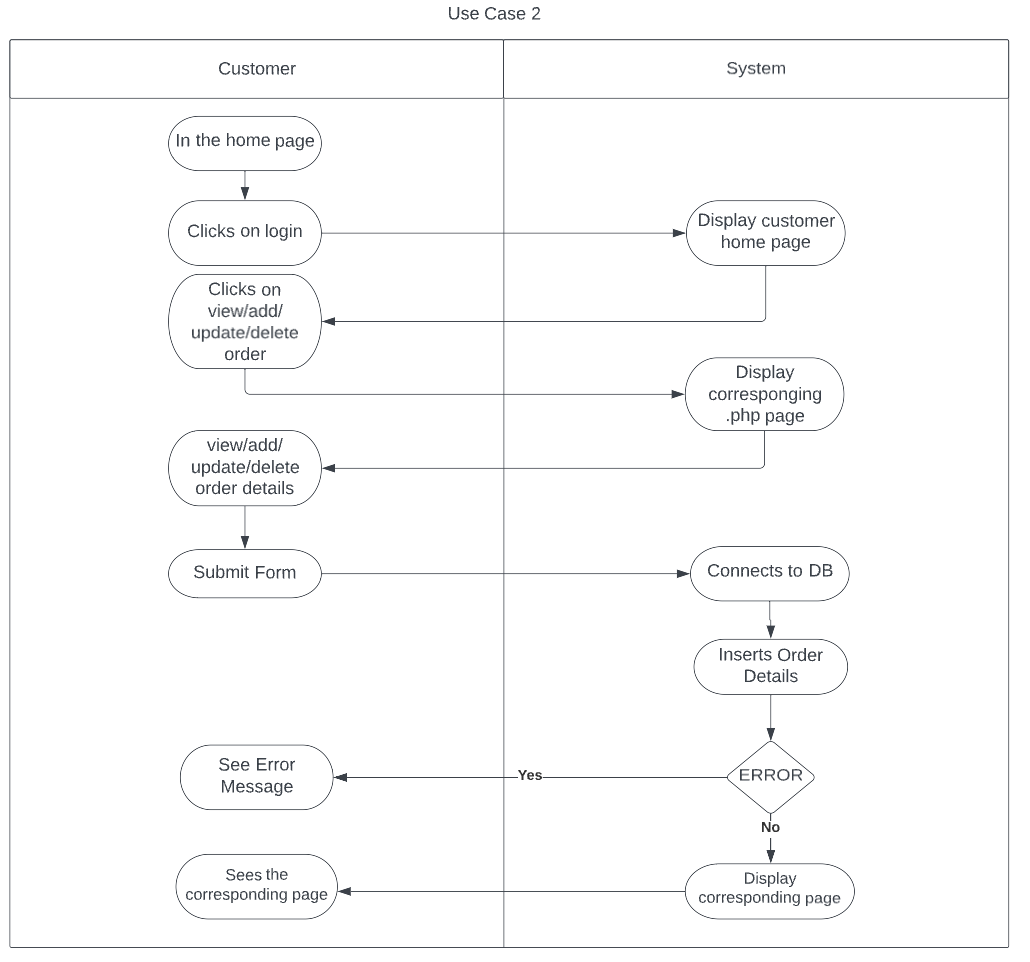
| Use Case #8 | View Order History | |
| --- | --- | --- |
| Objective | View order history | |
| Business Event | View order history | |
| Primary Actor (s) | Customer | |
| Secondary Actor (s) | None | |
| Pre-condition | Customer must be logged into the system  Customer clicks on “View Order History” | |
| Post-condition | Customers Order History is displayed | |
| Failure outcomes | Customer is in the View Order History and they either have not ordered anything so nothing is displayed, or the database for orders is unable to be accessed showing the error message “Unable to access Order History” | |
| Flow of Events | Actor | System |
| 1 | Customer logs in and is in the home page of the website (home.php)  User clicks on the View Order History page (view-order-history.php) | View Order History page is shown  Connection to Customer table and Orders table is established |
| 2 | Customer is able to see their order history if they made prior purchases | Customer is shown perspective link |

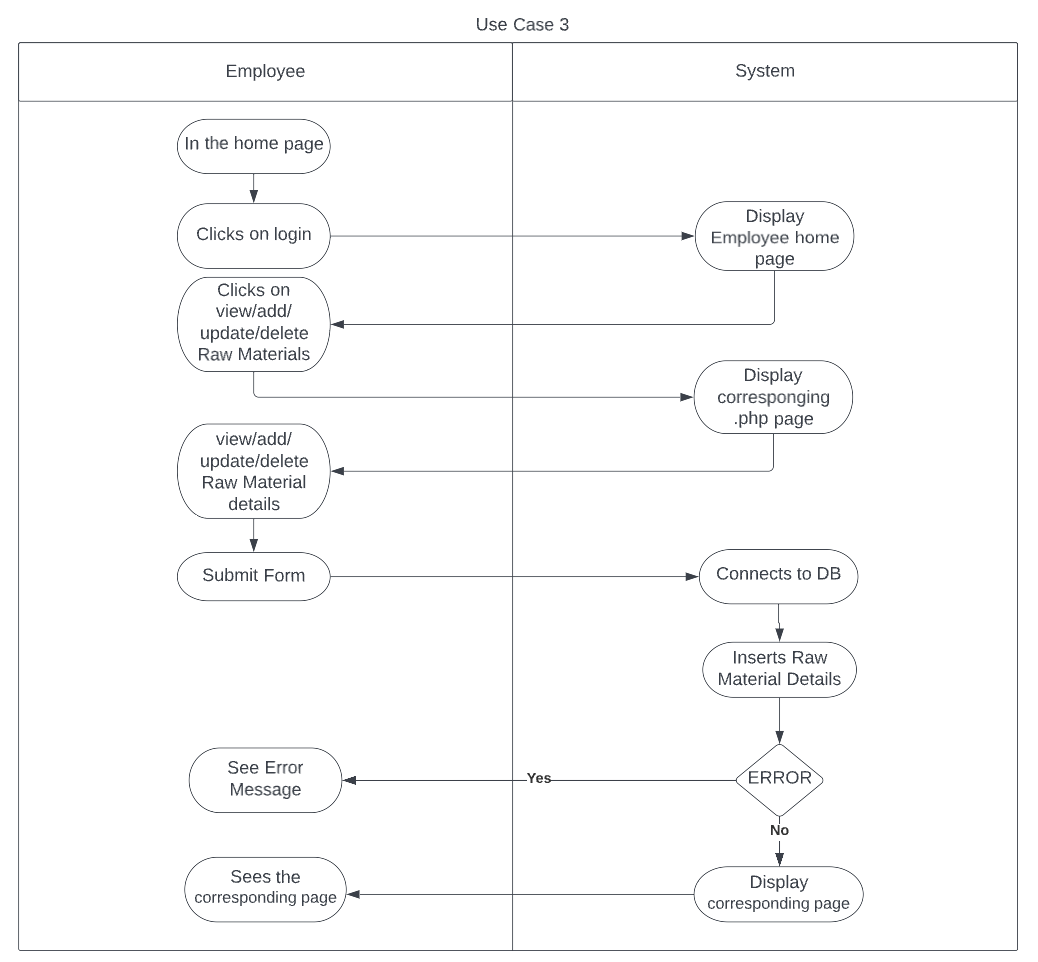
| Use Case #9 | View/add/update/delete Order | |
| --- | --- | --- |
| Objective | Employee Views/adds/updates/deletes order(s) in the system | |
| Business Event | Views/adds/updates/deletes order(s) in the system | |
| Primary Actor (s) | Employee | |
| Secondary Actor (s) | None | |
| Pre-condition | Employee must be logged into the system  Employee clicks on the “Edit Orders” link on the home page | |
| Post-condition | User is forwarded to the View Order List Page | |
| Failure outcomes | Employee is in the add new Order page.  Employee is shown an error message ‘Error in adding a new Order’. | |
| Flow of Events | Actor | System |
| 1 | Employee in the home page of the website(home.php).  User clicks on the edit orders page (edit-orders.php). | Edit Orders page is shown. |
| 2 | Employee clicks on either view/add/update/delete order details. | Employee is shown perspective link |
| 3 | Employee submits the prompted form. | Connect to the database.  Insert order details into the Orders table.  Forward employee to the view-order-list.php page. |

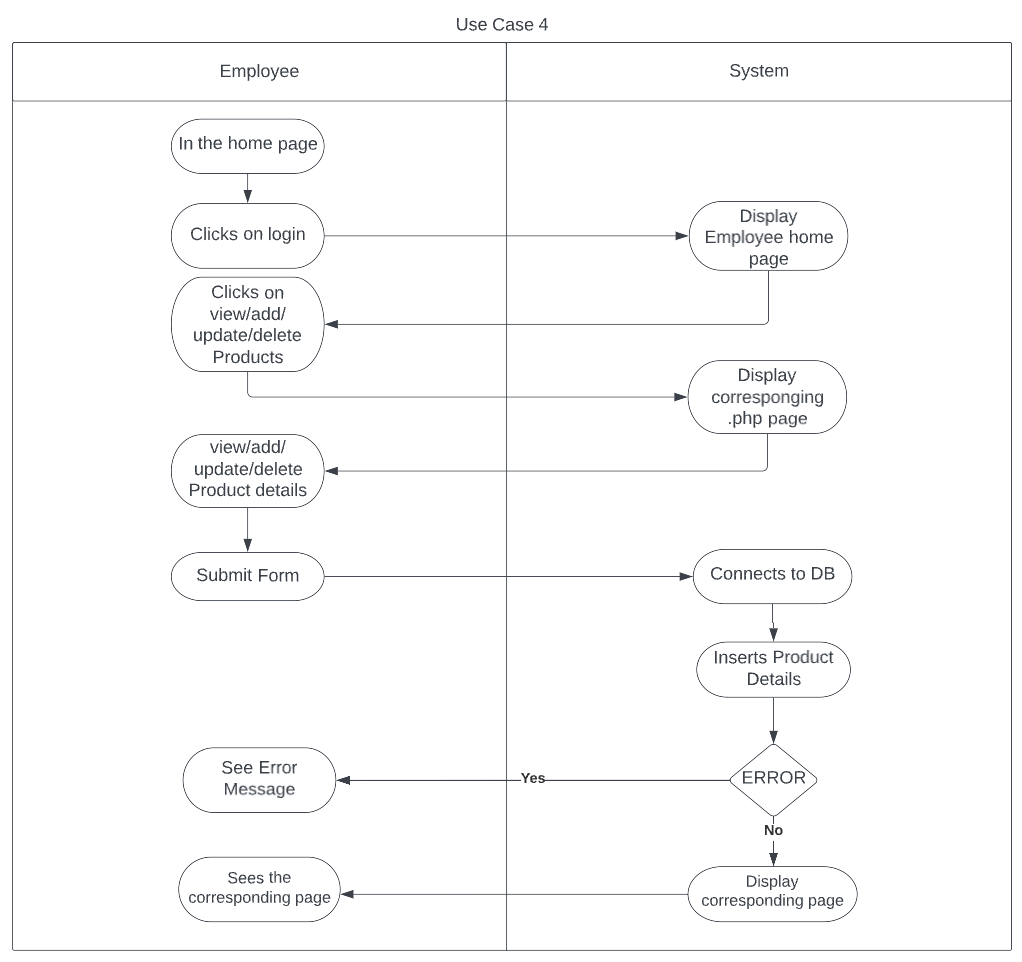
| Use Case #10 | Make a Payment | |
| --- | --- | --- |
| Objective | Customer makes payment for order to complete order. | |
| Business Event | Customer makes payment for order to complete order. | |
| Primary Actor (s) | Customer | |
| Secondary Actor (s) | None | |
| Pre-condition | Customer must be logged into the system  Customer must have an order placed in order to submit payment | |
| Post-condition | Payment is made confirming order | |
| Failure outcomes | Payment is declined spawning the message “Unable to process payment” | |
| Flow of Events | Actor | System |
| 1 | Customer is in the home page of the website (home.php)  User clicks on the Products page (products.php) | Products page is shown |
| 2 | Customer chooses which products they would like to purchase | Customer is shown perspective link |
| 3 | Customer confirms order and is sent to payment (payment.php) | Payment page is shown |
| 4 | Payment is received and order is confirmed to Customer | Payment is received and Order is processed |

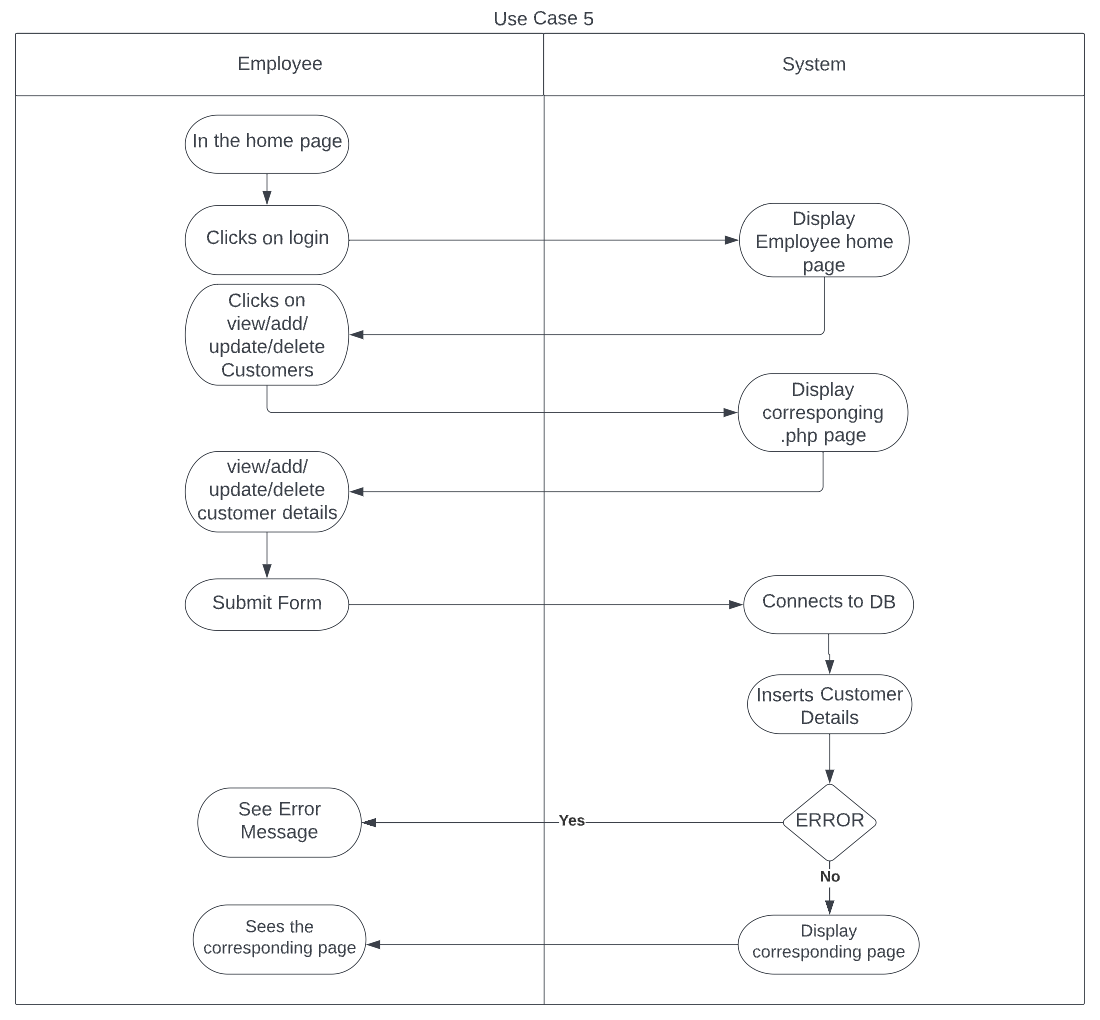
| Use Case #11 | Submit a return | |
| --- | --- | --- |
| Objective | Submit a return and receive a refund | |
| Business Event | Submit a return and receive a refund | |
| Primary Actor (s) | Customer | |
| Secondary Actor (s) | None | |
| Pre-condition | Customer must be logged into the system  Customer must have previously made an order | |
| Post-condition | Order is returned and a refund is issued | |
| Failure outcomes | If the order was never made initially then it can not be refunded | |
| Flow of Events | Actor | System |
| 1 | Customer is in the home page of the website (home.php)  User clicks on the Returns page (returns.php) | Returns page is shown |
| 2 | Customer puts in Order number for the order they would like to return | Customer is shown perspective link |
| 3 | Customer issues Order return with proper information for receiving refund  Customer receives refund | Order return and refund are processed  Refund is made |

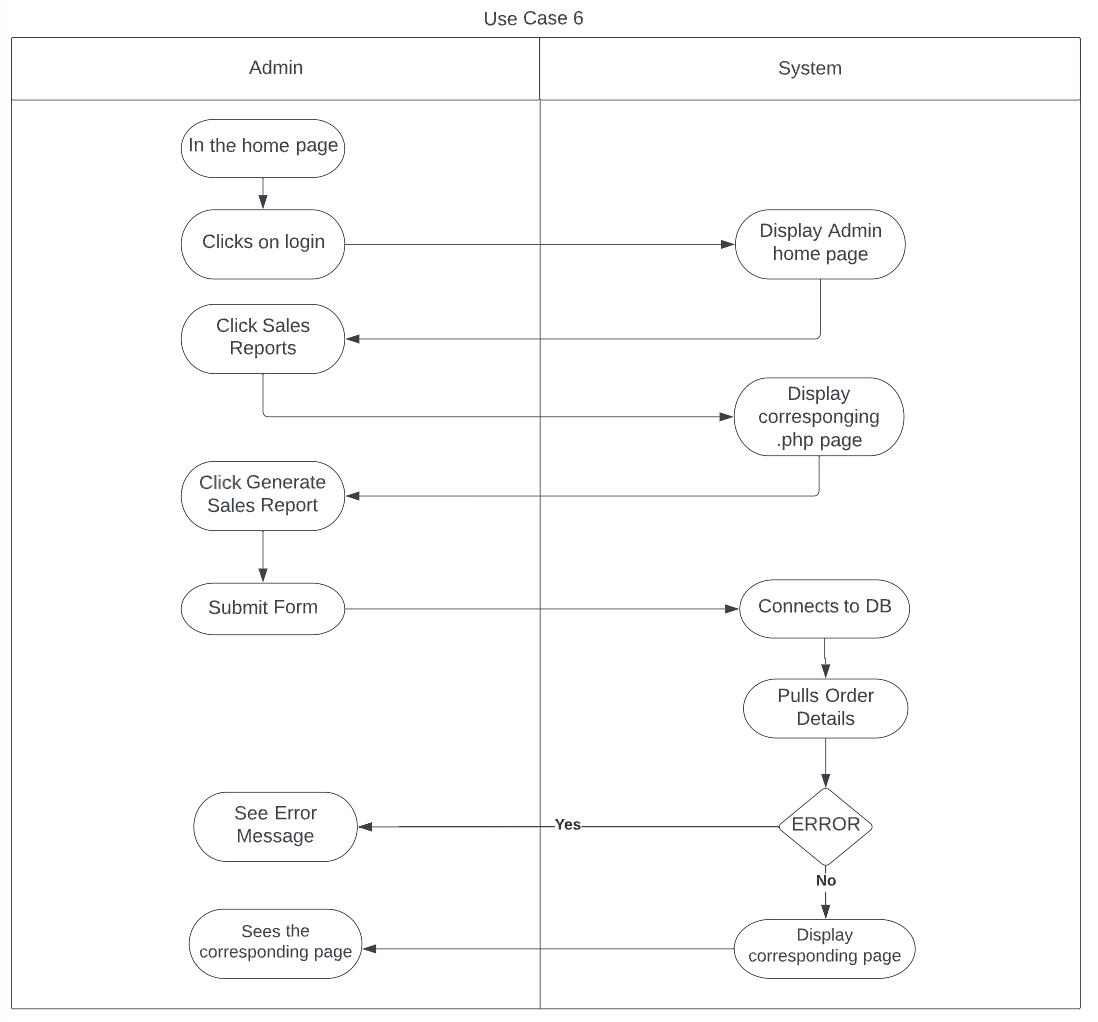
## Activity Diagrams

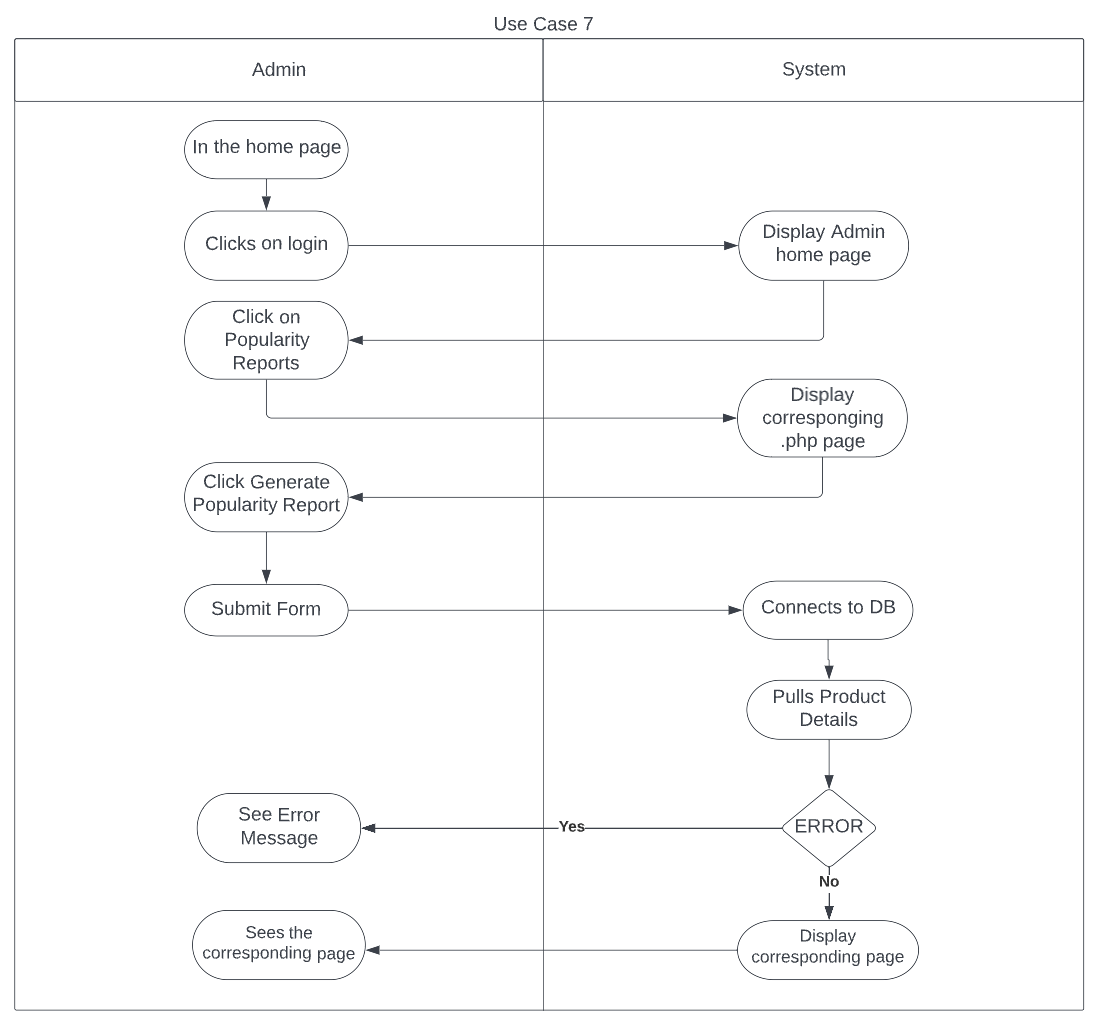
****

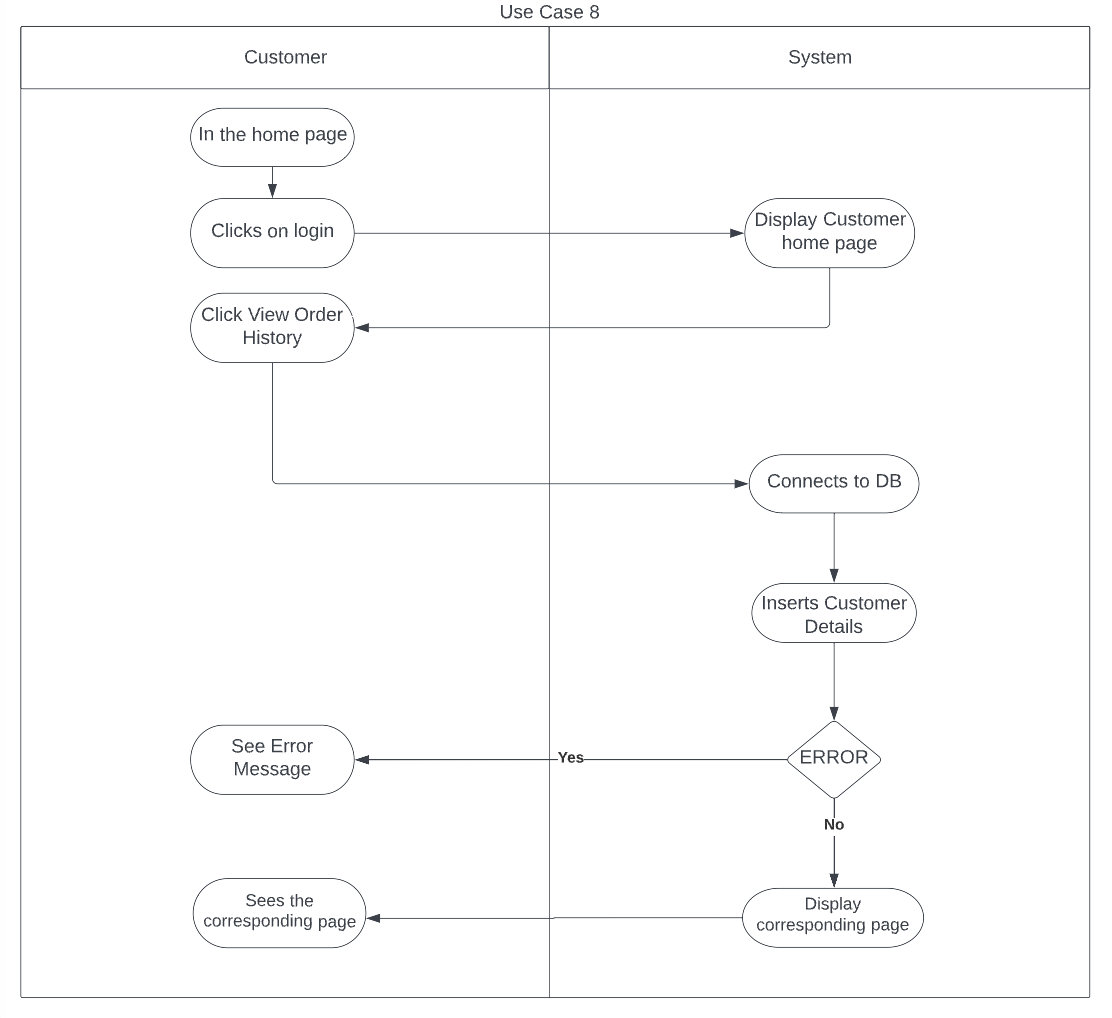


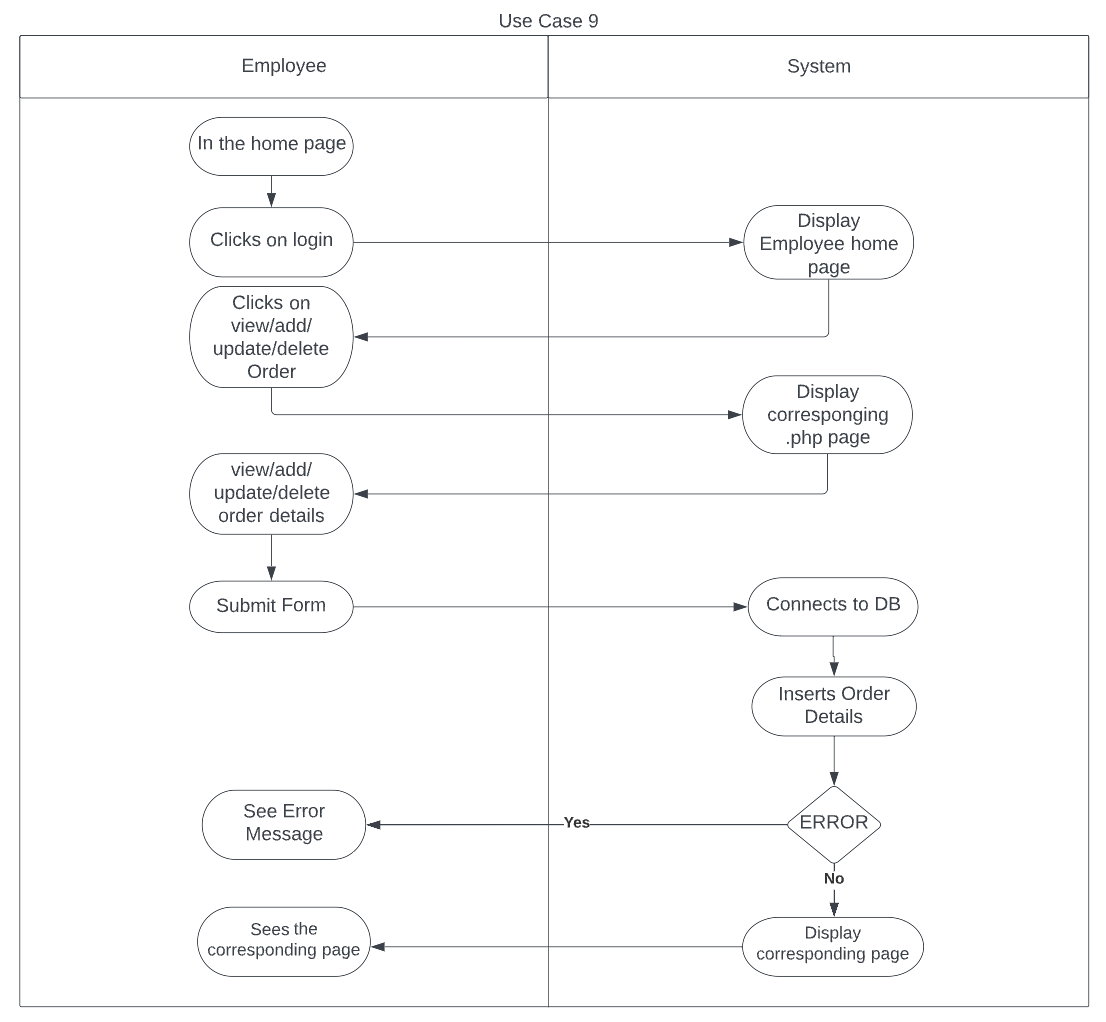


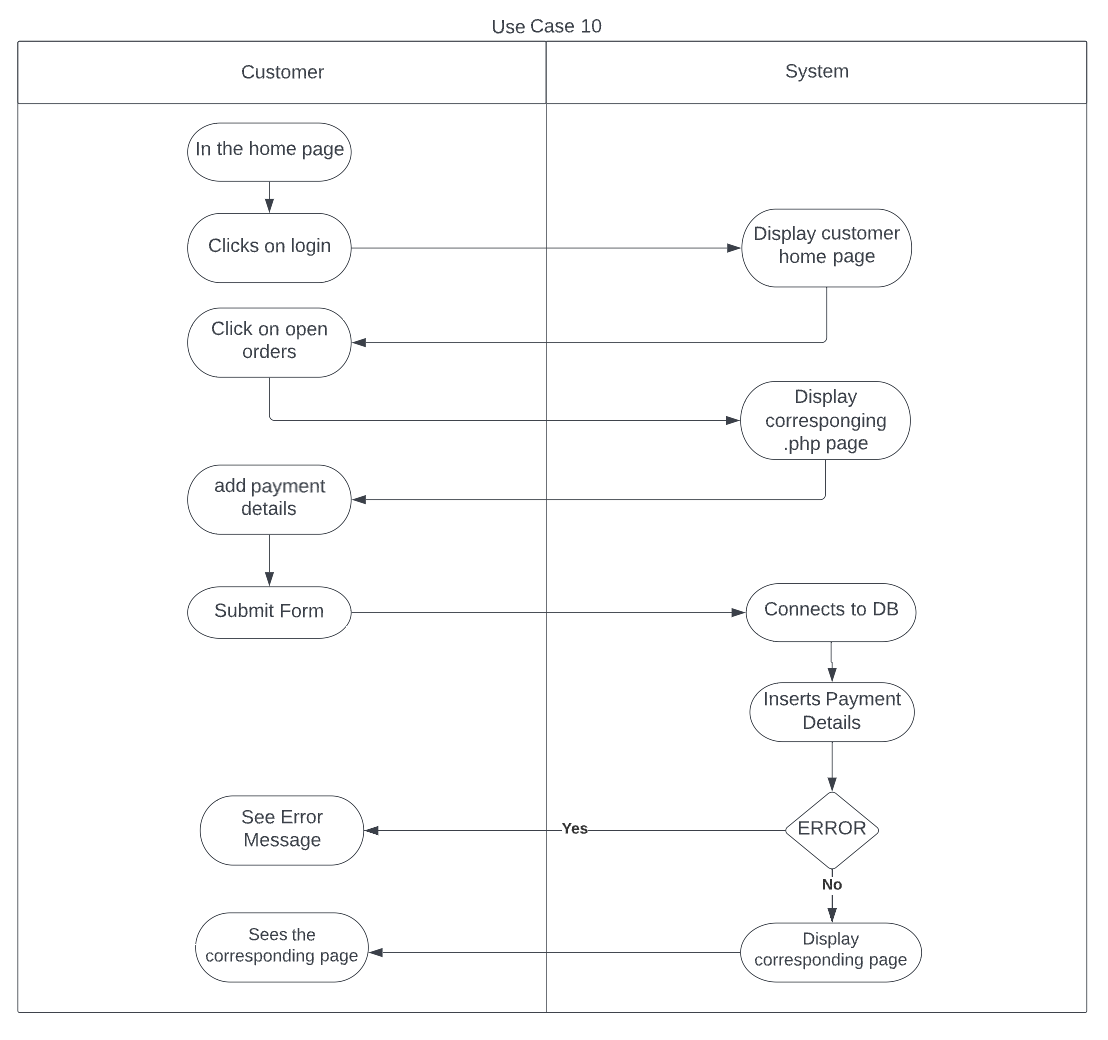


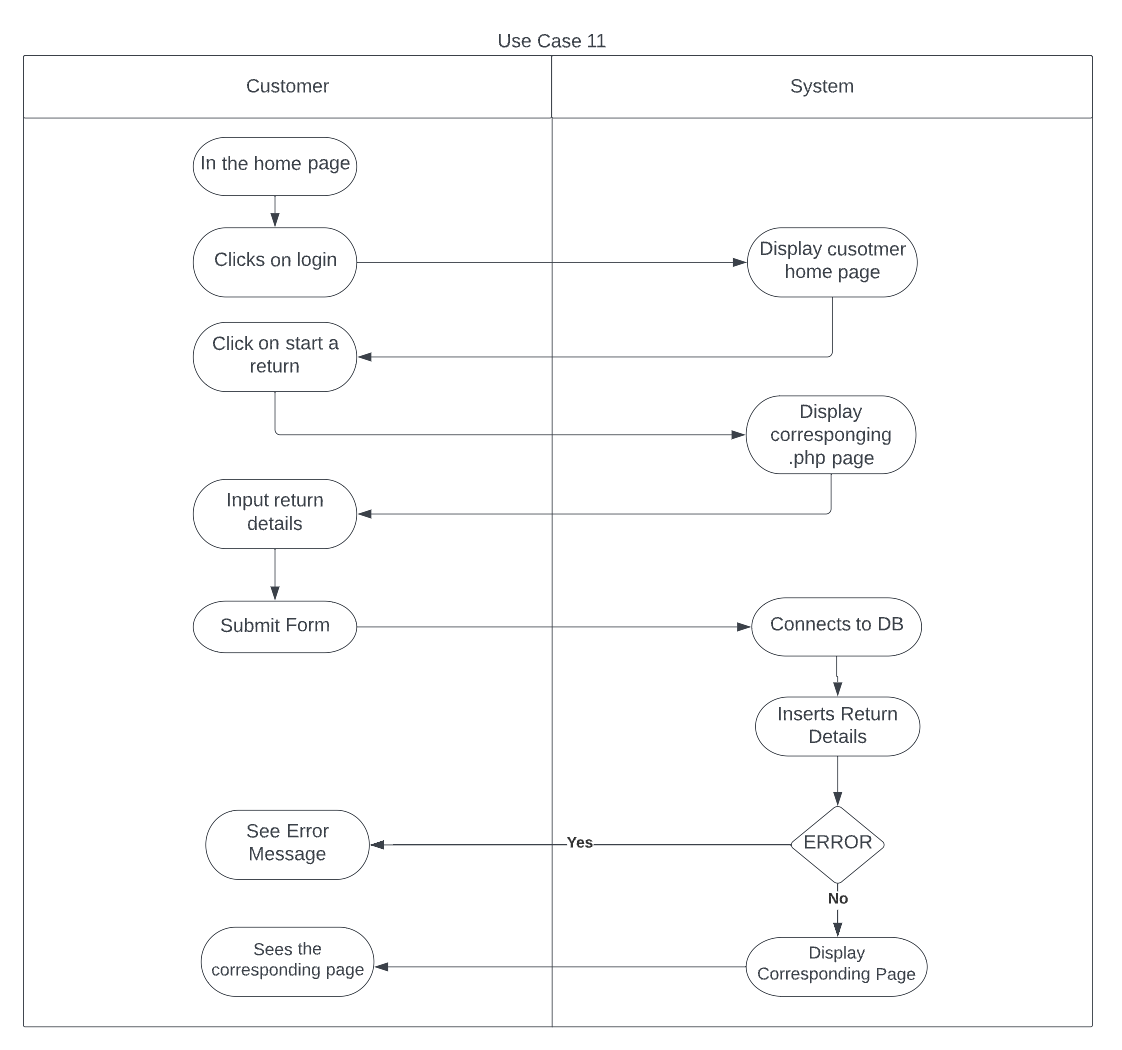






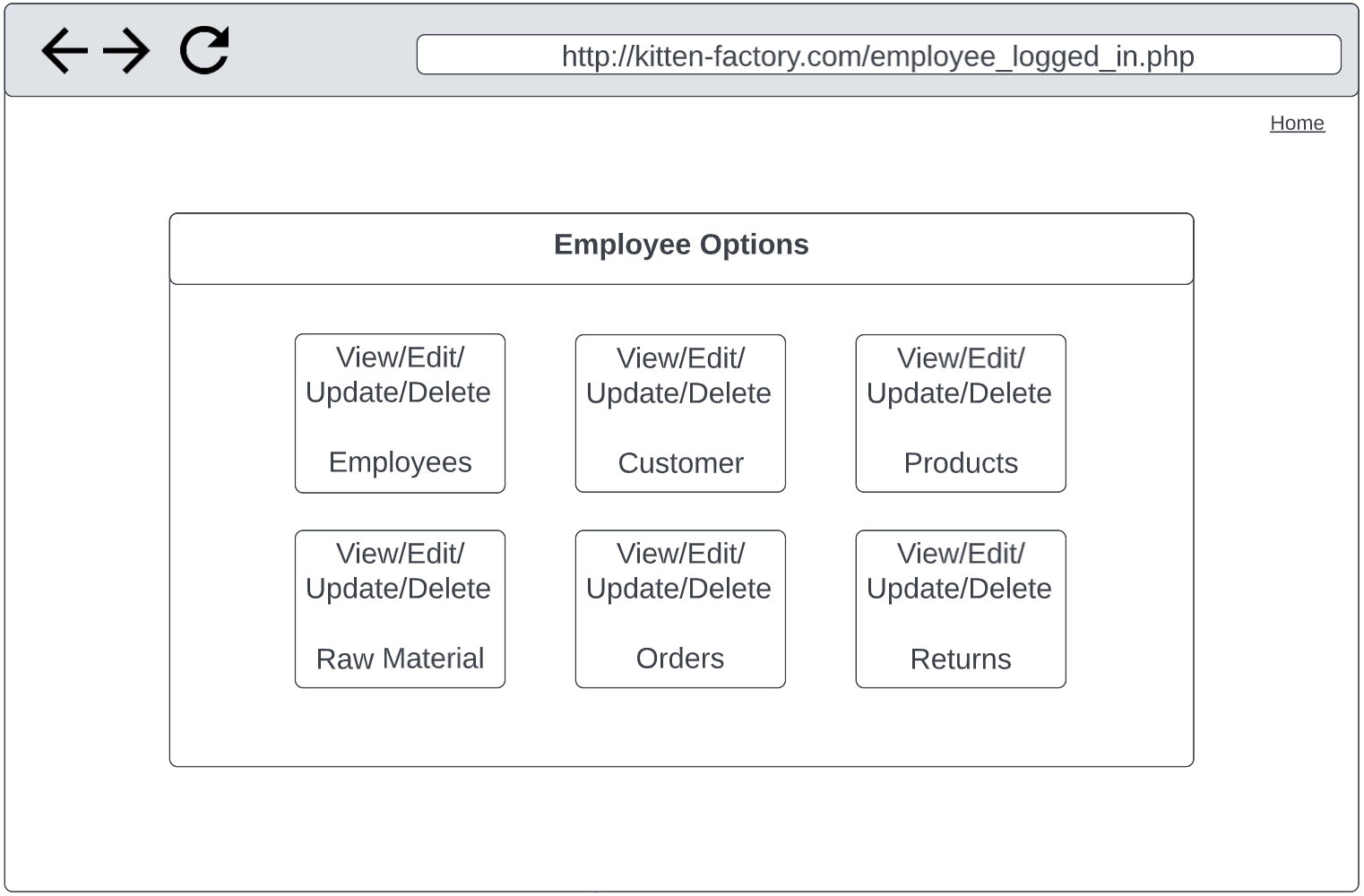


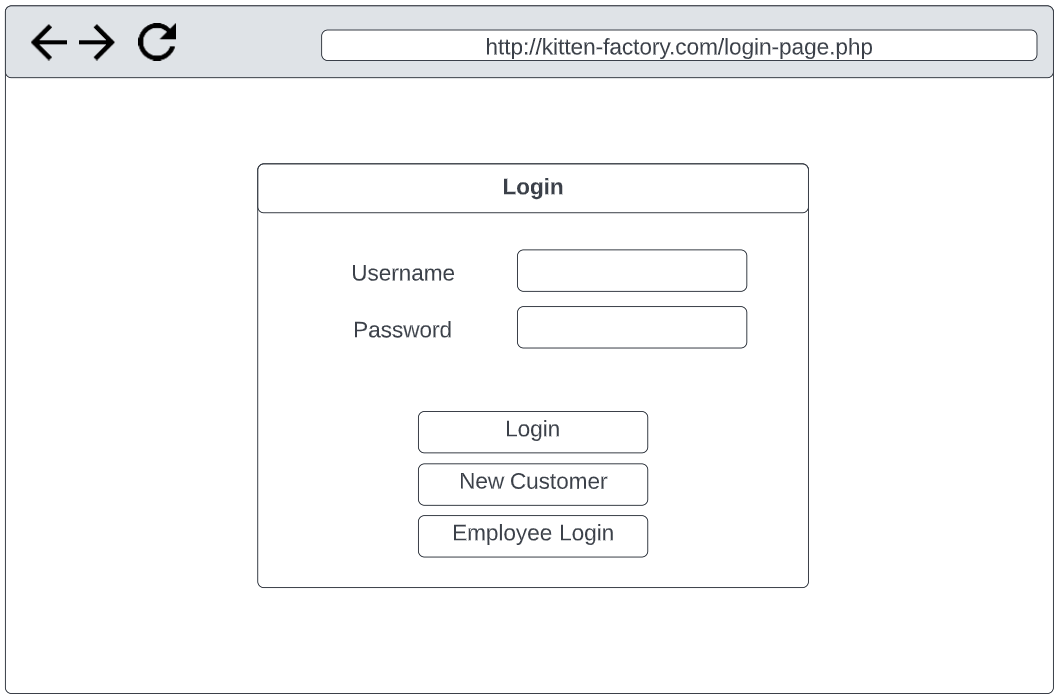


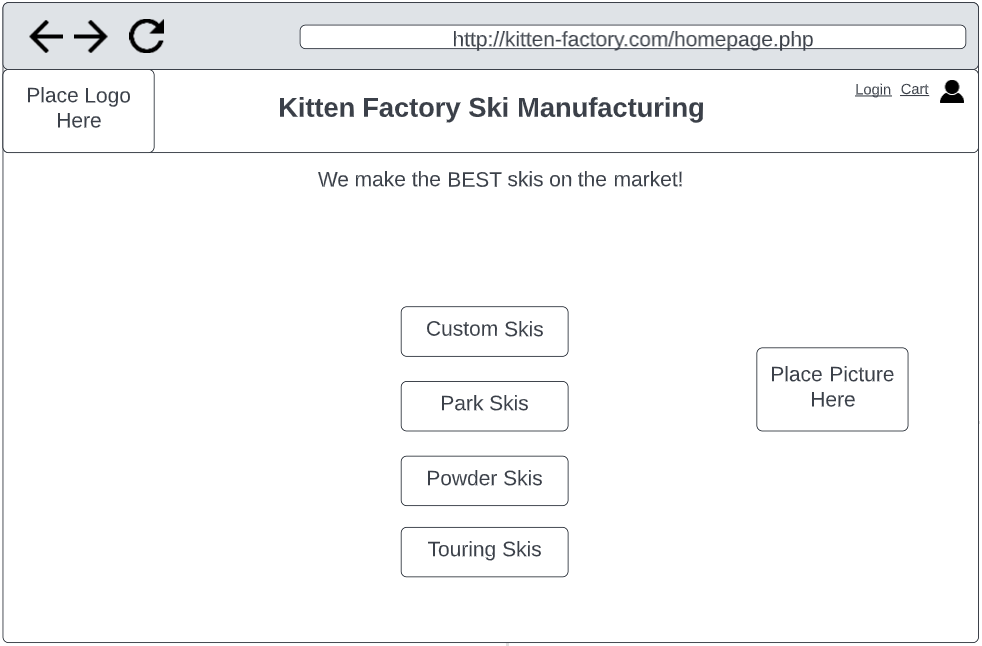
****

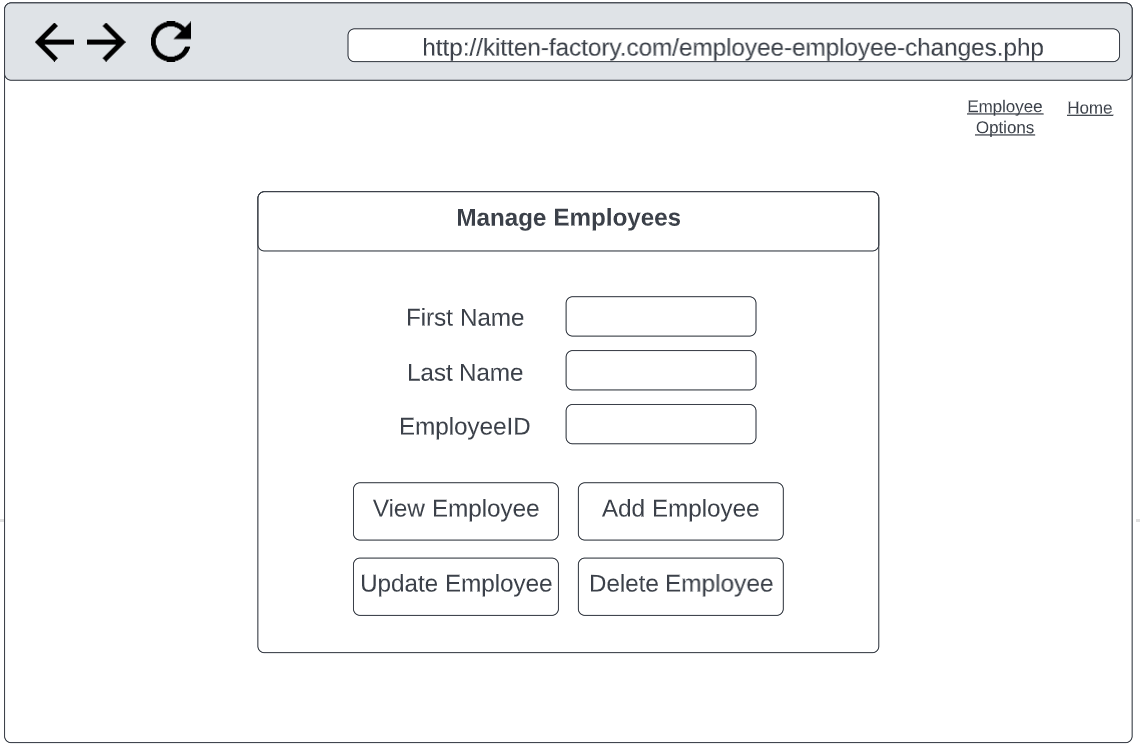
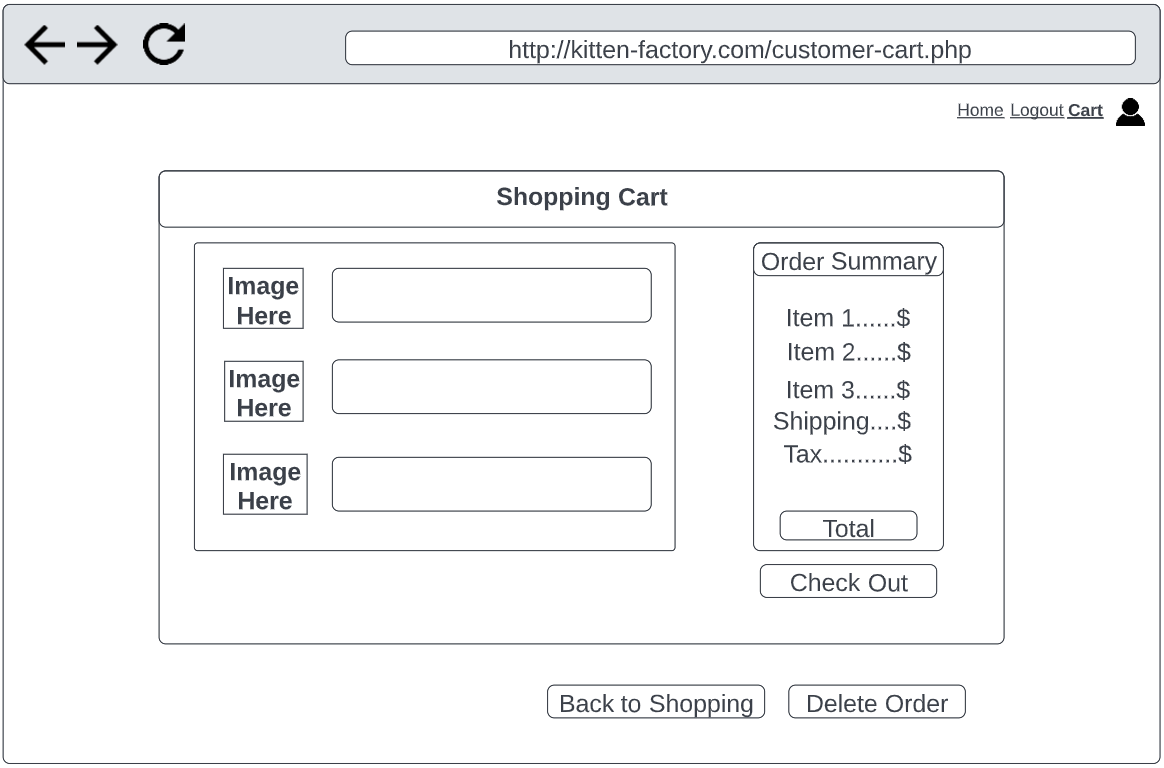
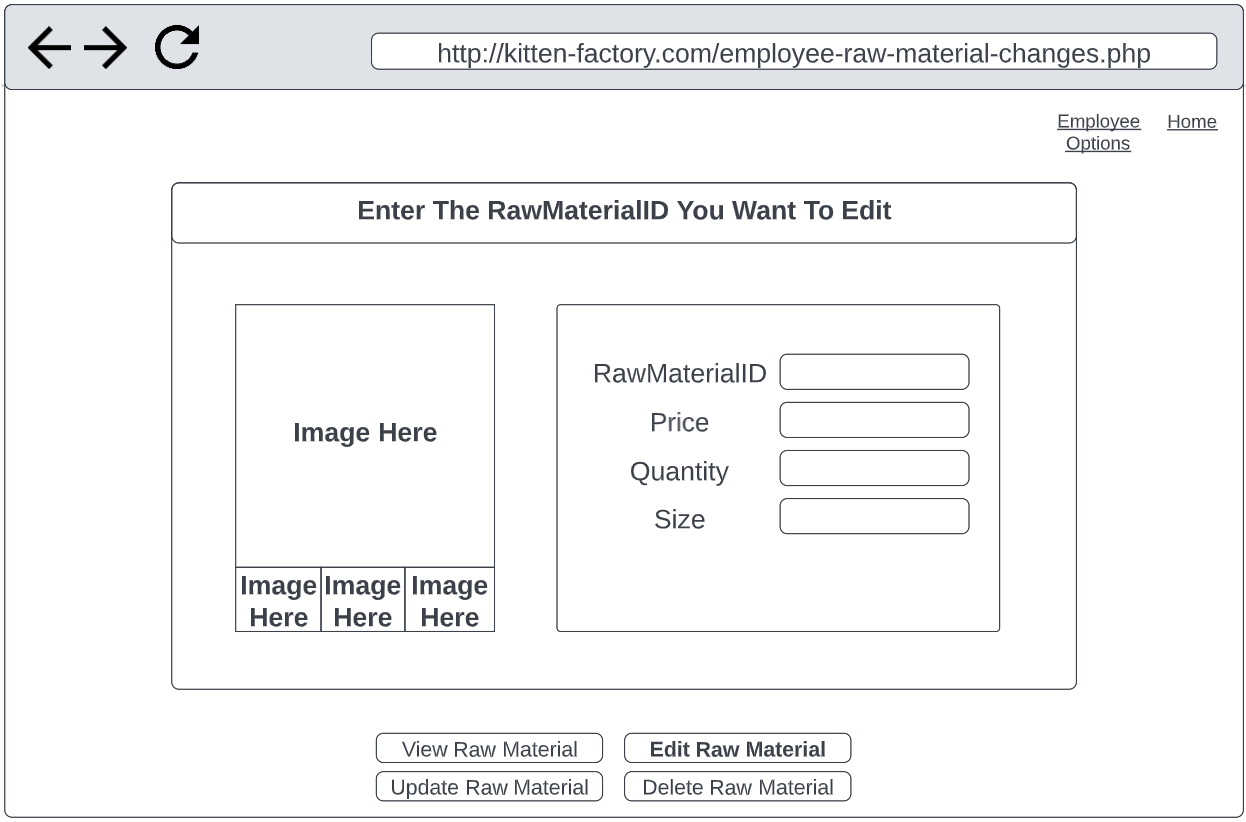
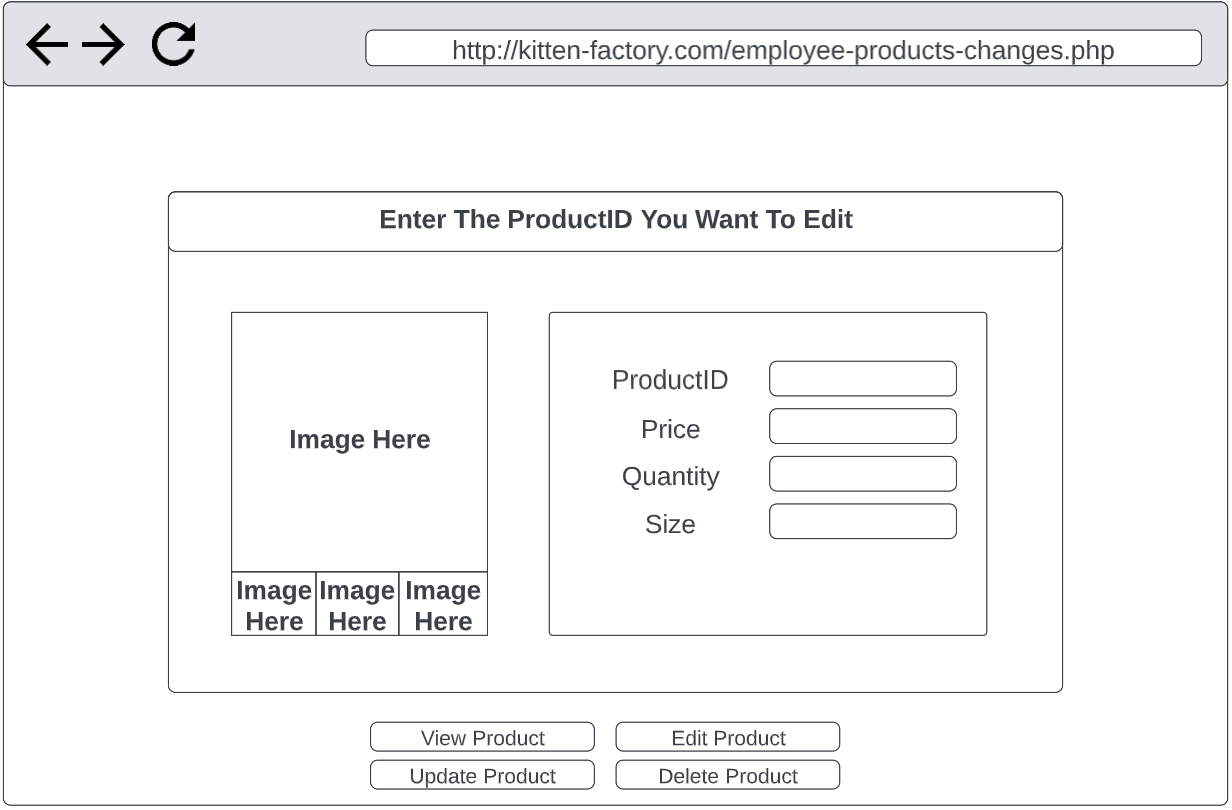
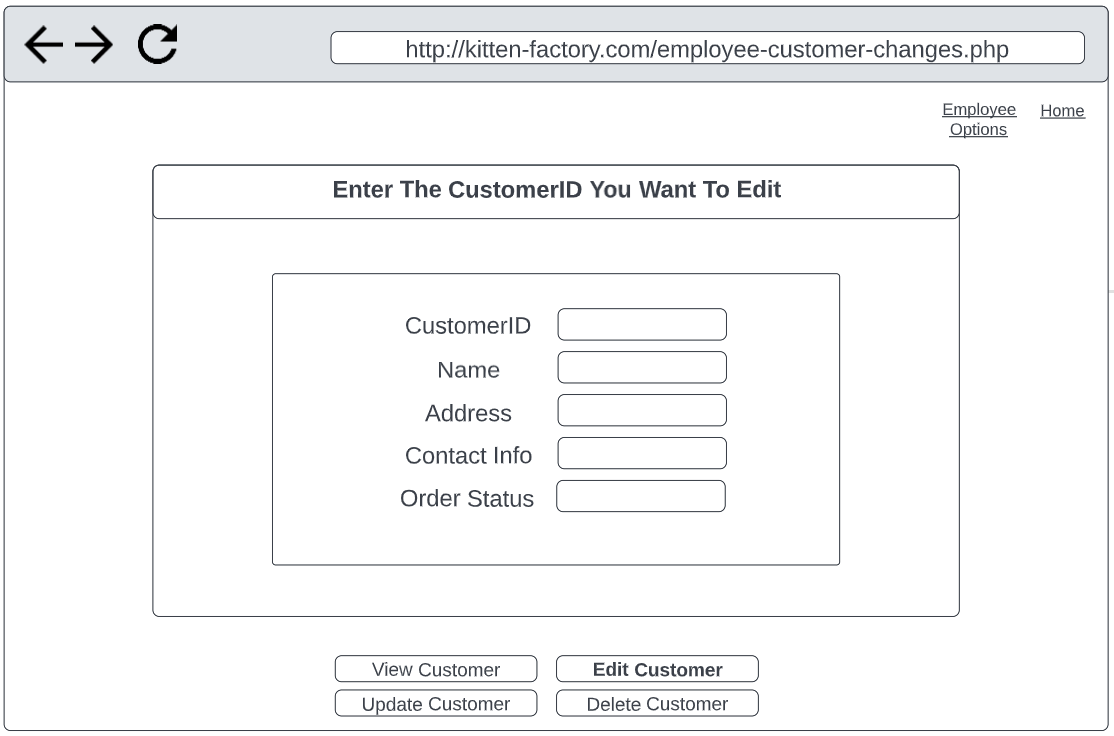
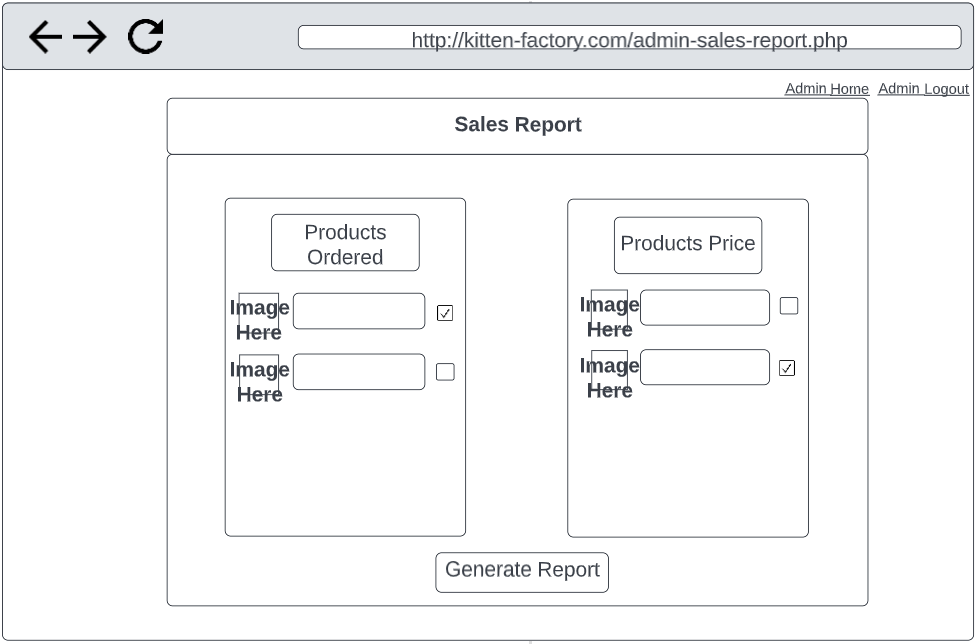
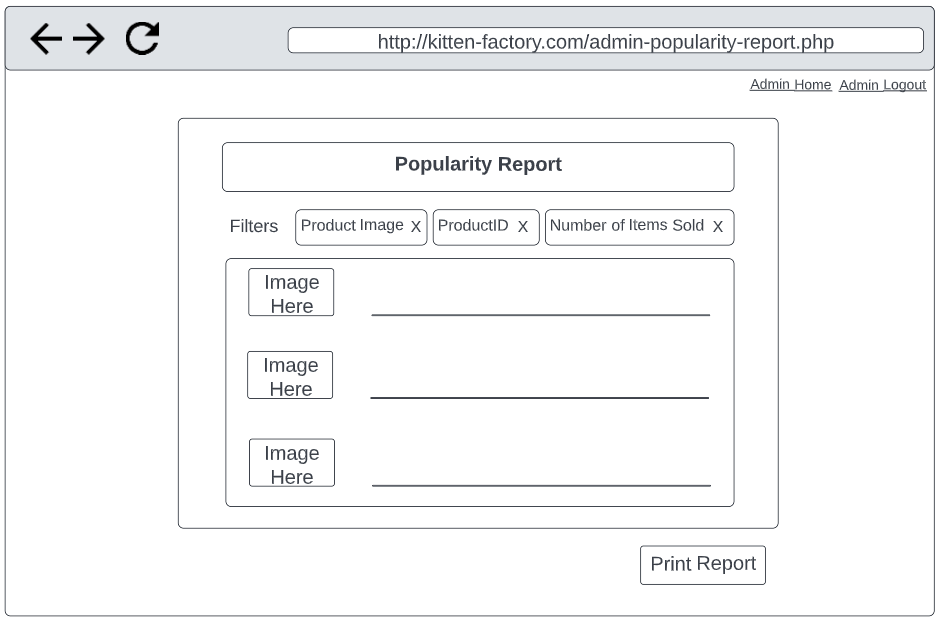
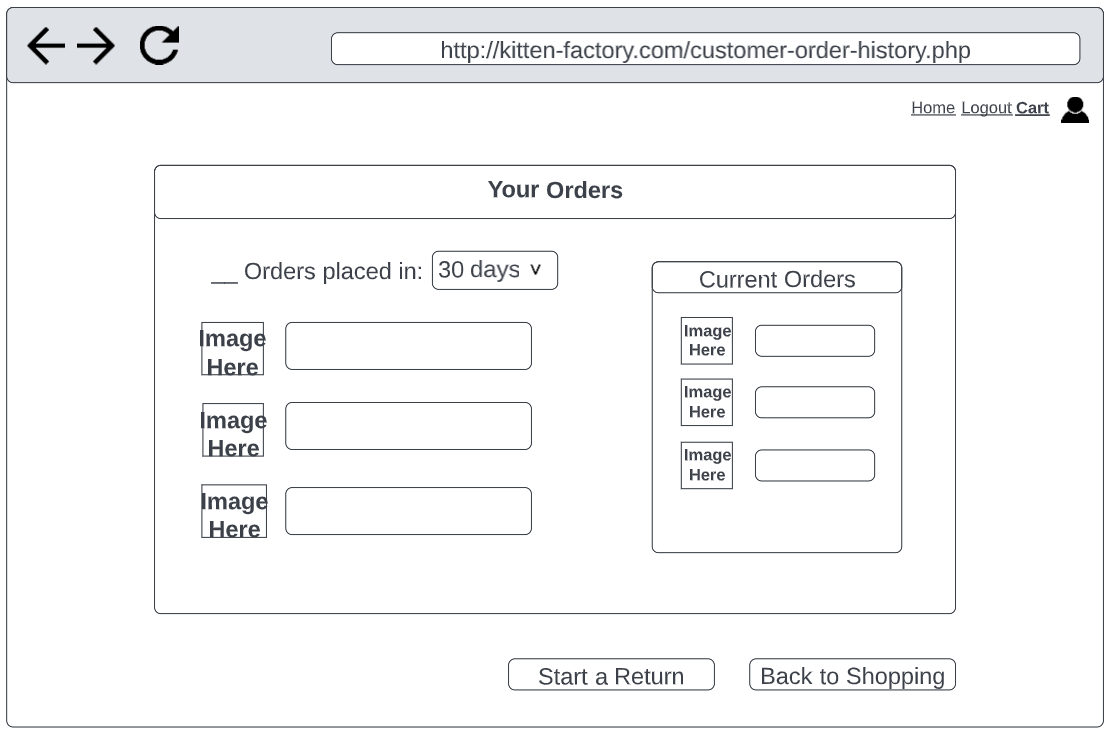
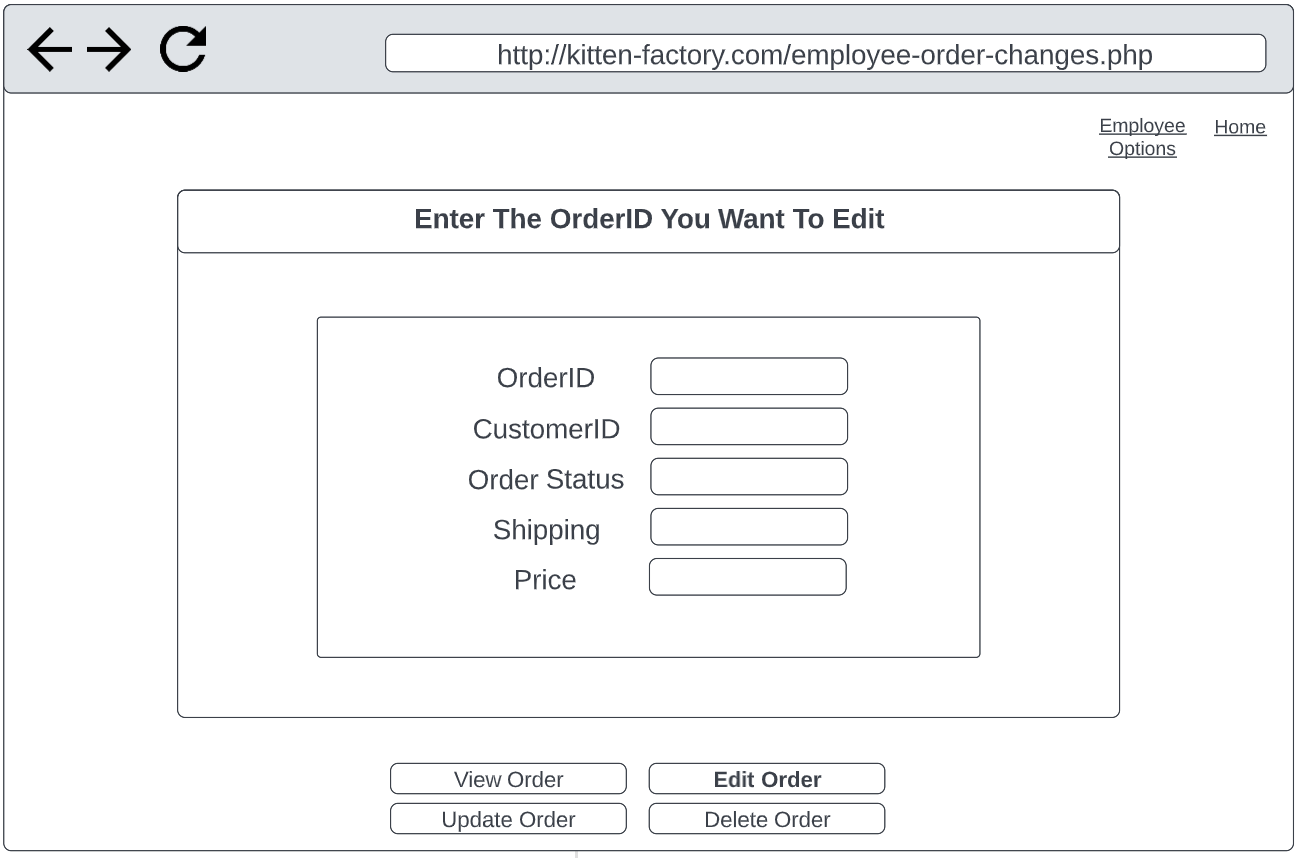
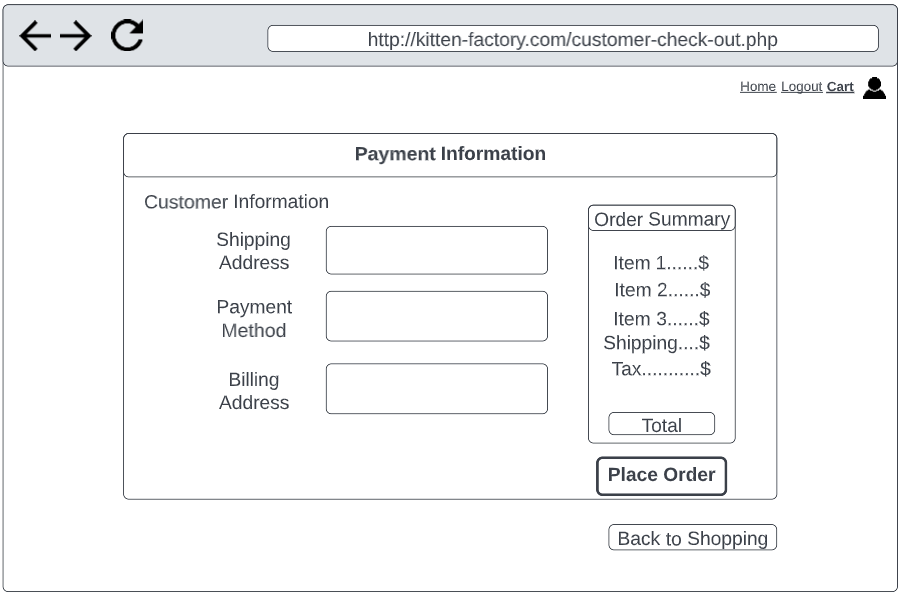
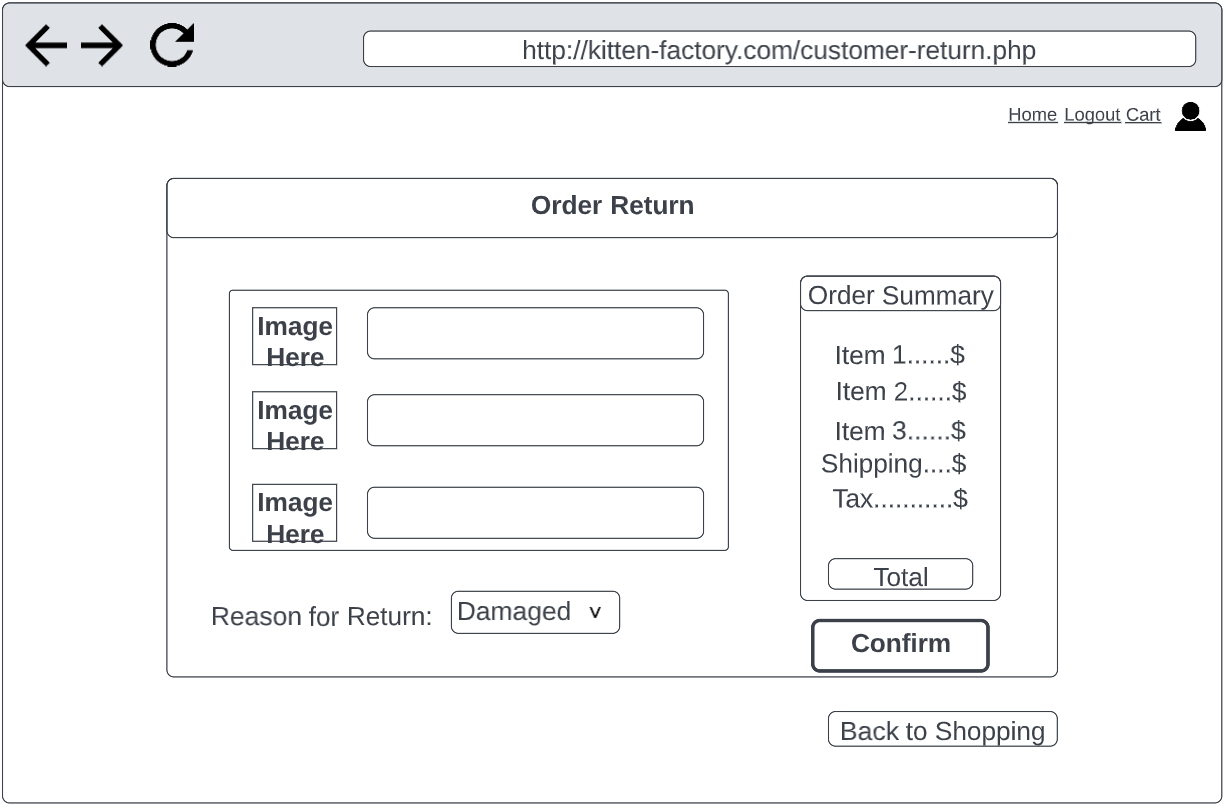
## Wire frame diagrams

Employee log on successful

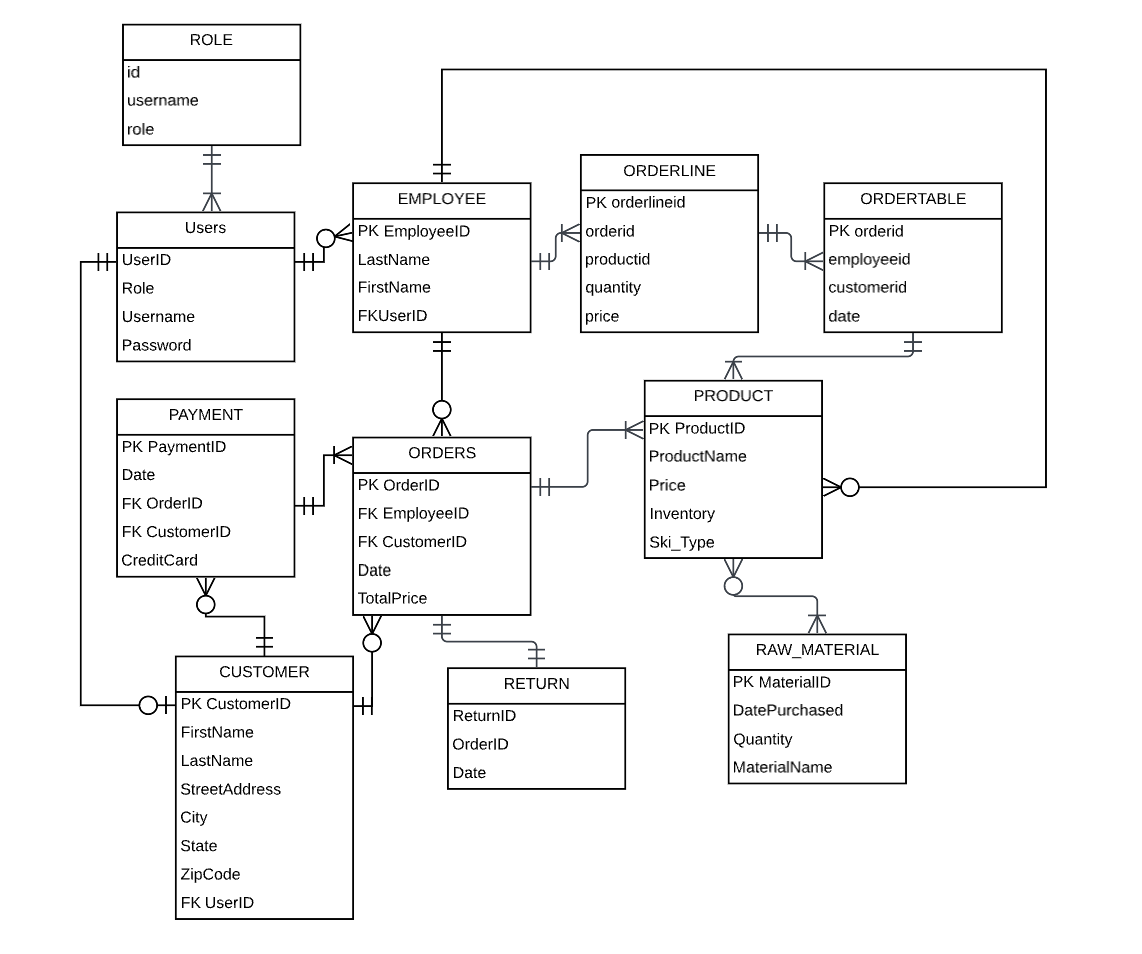


Login Page****

Homepage

1. View/add/update/delete a new employee
2. Customer View/Add/Update/Delete Order
3. View/Add/Update/Delete raw materials
4. View/Add/Update/Delete Products
5. Employee View/Add/Update/Delete Customers
6. Admin Generate Sales Report
7. Generate Popularity Reports
8. View Order History
9. Employee View/Add/Update/Delete Order
10. Make a payment
11. Submit a return

## ERD

****

## Is the business objective(s) achieved?

Kitten Factory’s objective is to manufacture and sell their custom made carbon fiber skis. The products produced by Kitten Factory are made to be lighter weight so that customers can go on a wide variety of terrain, including park, powder, touring, etc. Customers can go online and choose from our specialized offers or they can customize their very own pair of skis. Employees can monitor different user accounts as well as edit customer and product orders. And Admins can oversee the website and generate necessary reports that give information on how business is going. With these objectives in mind, we have created a web application with the corresponding databases to achieve the company’s goals.

## Ethics and Critical Thinking

Our group will keep into account how customers' personal credit card information will be used to purchase our products. This data is very important to keep secure as if this data is leaked or stolen, millions of dollars can be lost from customers and we will lose not only their trust but the trust of new customers. An example of this being very important is the Target case we talked about in class where ⅓ of customers' credit card data was leaked which lost them hundreds of millions of dollars.

Another factor we wanted to keep in mind was how important making our UI seamless and simple for a user. Poor UI even with really good architecture will hurt user experience and severely decrease the growth of our website and company. We also do not want to fall into the trap of customers accidentally selecting options they did not mean to which would lead to many more returns of our products which we want to limit as much as possible.

As we kept these problems in mind and worked to develop our web application and databases, we used critical thinking to make sure that we had a simple UI that our admins, customers, and employees can use. We wanted to make sure that our web application met all the necessary objectives in a manner that was easily executable, which pushed us to use critical thinking to solve our problems so we could complete our code. If an issue came up or a code didn’t work, we worked together to fix the issues so we could deliver our final product.