**System Requirement Development**

With the aim of code generation, we've used a free and open-source UML (Unified Modelling Language) Diagramming Application named ArgoUML, written in Java and released under the open source Eclipse Public License. (Anon., n.d.) ArgoUML is used for designing class diagram and such. Also we have used Diagrams.net, a free and open source cross-platform graph drawing software developed in HTML5 and JavaScript. (Anon., n.d.) Diagrams.net is used for designing sequence diagrams and collaboration diagrams and such.

**Usecase Delete Item**

|  |  |  |
| --- | --- | --- |
| Use Case No. | Uc-04 | |
| Use Case Name | Delete Item | |
| Priority | High | |
| Actor | Manager | |
| Description | This use case allows the manager to delete items | |
| Pre-condition | Uc-06 | |
| Post condition | Manager has successfully deleted the item | |
| The fundamental course of action | User action | System responses |
| 1. User is on updated items where the option for delete items are given  2. User clicks delete item button  5. User clicks confirm delete button & affirms request | 3. System responds to delete item request  4. System generates message to confirm deletion  6. System delete items & displays deletion message  7. Use case exit |
| An alternative course of action | If in 4. user fails to confirm deletion the system will re-direct to stage 1 | |

**Usecase Login**

|  |  |  |
| --- | --- | --- |
| Use Case No. | Uc-01 | |
| Use Case Name | Login | |
| Priority | High | |
| Actor | Manager, customer | |
| Description | If this use case is successful the manager | |
| Pre-condition | None | |
| Post condition | Login success | |
| The fundamental course of action | User action | System responses |
| 1. Customer is on the Home page for login  3. Customer/Manager  enters user credentials and  clicks the login button | 2. System requests for login  credentials from Customer/  manager  4. System runs verification  on credentials  5. Manager/Customer  successfully logs in  6. Use case exit |
| An alternative course of action | 3.1. If user credentials are incorrect or not valid, the program will pop a message for validation and would allow re-filling of credentials, resuming back to stage 3 | |

**Usecase view order**

|  |  |  |
| --- | --- | --- |
| Use Case No. | Uc-0 | |
| Use Case Name | View ordering | |
| Priority | High | |
| Actor | Customer, manager | |
| Description | The actor can choose to view ordering made and all details in  relevance to the orderings | |
| Pre-condition | Uc-01, | |
| Post condition | A list of all orderings made is displayed in format | |
| The fundamental course of action | User action | System responses |
| 1. Customer/manager  is on view ordering page  [Based on actors, customer would  see his/her orderings, while  administration would be able  to see all orderings]  2.2. In case of administration, manager fills in ordering ID or  just views all orderings | 2. System displays relevant  orderings according to actors  2.1. In case of  administration, he/she can  view orderings according to  ordering D automatically  generated or all orderings  3. Orderings are displayed  4. System displays  confirmation of retrieval  5. Use case exit |
| An alternative course of action | 2.2.1. In case of manager filling ordering ID, if it is invalid the  system would validate and pop an error while re-directing to  stage 2.2 | |

**Usecase Logout**

|  |  |  |
| --- | --- | --- |
| Use Case No. | Uc-0 | |
| Use Case Name | Logout | |
| Priority | High | |
| Actor | Customer, manager | |
| Description | This use case allows both actors to logout of the system | |
| Pre-condition | Uc-01 | |
| Post condition | Customers have successfully logged out of the system | |
| The fundamental course of action | User action | System responses |
| 1. Actors wish to logout  2. Actors click logout button | 3. System directs to logout  request  4. System logs out  5. System displays logged out  message  6. Use case exit |
| An alternative course of action | None | |

**Implementation Requirements**

This section focuses on the development of the planned system and the way it's to be implemented.

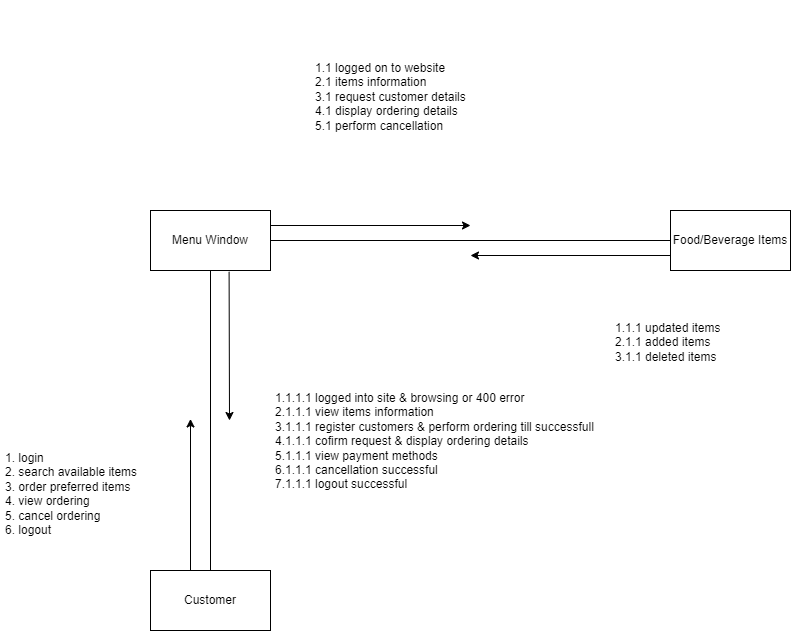
1. We want to concentrate on coding and selecting a desirable get advantages programming language.
2. We want to contemplate verification and validation.

When considering point 1, we want to decide on a language carefully. It’s the language that may define the constraints and quality of the system. PHP, MySQL, HTML, CSS are a number of the best options you may take.

Advantages that follow when using PHP

1. It is a scripting language which will run on any device in spite of operating systems.
2. It has less learning curve, because it is simple and straightforward to use. (pulkitagarwal03pulkit, n.d.)
3. It has powerful library support to use varied function modules for data representation.

**Collaboration Diagram – Customer**



# References

Anon., n.d. *ArgoUML.* [Online]   
Available at: https://en.wikipedia.org/wiki/ArgoUML  
[Accessed 03 03 2022].

Anon., n.d. *Diagrams.net.* [Online]   
Available at: https://en.wikipedia.org/wiki/Diagrams.net  
[Accessed 03 03 2022].

pulkitagarwal03pulkit, n.d. [Online]   
Available at: https://www.geeksforgeeks.org/advantages-and-disadvantages-of-php/