Ateneo de Manila University

CS 123 -- Introduction to Software Engineering

SY 2015-2016

**Final Report**

SUBMITTED BY:

Team Name:

Group C - Cenation

Team Members:

Cordero, Adrian Paolo

Dela Cruz, Viktor Mikhael

Guevara, April Gale

Torrijos, Jose Emmanuel

SUBMITTED TO:

Mrs. Marlene M. De Leon

CS 123, Lecturer, Introduction to Software Engineering

Ateneo de Manila University

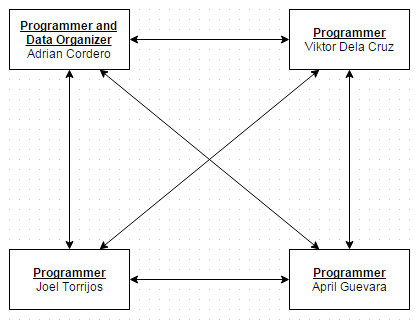
**TABLE OF CONTENTS**

1. [CONTEXT OF THE PROJECT](#h.wqdp61jdnm2w) – Page 1
2. TEAM ORGANIZATION – Page2
   1. Organization, Rationalization and Roles – Page 3
3. [SPRINT PLANS](#h.jnsoqduykw3t) – Page 3
4. [ACTIVITY NETWORK DIAGRAM](#h.r1ivxf7amu07) – Page 5
   1. Activities, Task Distribution and Duration – Page 5
   2. Dependencies – Page 7
   3. Activity Network Diagram & Staff Allocation – Page 8
5. [EERD AND DATABASE DICTIONARY](#h.lnr7cowvjyh9)
   1. EERD – Page 9
   2. Database Dictionary – Page 9
6. [USE CASE DIAGRAM](#h.77k283ty2c9d) – Page 13
7. [USE CASE DESCRIPTION](#h.9pcokz9jcj3b) – Page 14
8. [LESSONS LEARNED](#h.hj3epk9m3klf) – Page 39

# **CONTEXT OF THE PROJECT**

# Food Buddy is a system designed by Joel Torrijos, April Guevara, Viktor Dela Cruz and Adrian Cordero. It is an ordering website that caters to cinemas. The project was started on the idea of giving moviegoers a way of ordering snacks from inside the cinema, to make the most of their viewing experience. The system works for the entire mall the cinema is located in, meaning the moviegoers may order from any restaurant in the mall. The ones in charge of the system for a specific mall are one or more boys, and an admin for each restaurant in the mall. The boy is in charge of receiving, assembling and delivering orders. The admin takes care of updating the restaurant’s menu in the system. Ideally, malls can be registered into the system by contacting the system developers, who take care of adding, deleting and editing malls and restaurants in the system.

# **TEAM ORGANIZATION**



**Organization, Rational, and Roles**

For our organization, we chose to go with the Democratic Team approach. This is because we are egoless programmers who recognize that each of us, individually, is equal to one another in a sense that we are jacks of all trades but masters of none. Rather than relying on one person to “carry” us, we found it more appropriate and rewarding to work together in an attempt to combine our skills to achieve an effect resembling the “mastering” of a skill. We understand that we are only as good as the other member and that our success will be measured by the equal contributions of each.

Given our organization, each member has an established line of communication with all the others. Work distribution will be discussed and agreed upon by the entire team. However, we will have one member, Adrian Cordero, with an informal and additional role of the **data organizer**. This person will have an added task of collecting and organizing the team’s files so that the team will have record of the project progress. Aside from that, everyone, namely Adrian Cordero, Viktor Dela Cruz, April Guevara, and Joel Torrijos, will be playing the role of a **programmer**. Programmers are tasked with the designing, modelling, coding, implementing, and modifying of the system being worked on.

# **SPRINT PLANS**

1. Tasks to Accomplish by October 15
   1. Set up a mall database and allow the adding of a mall on the admin system.
   2. Set up an account database and allow the creation and viewing of accounts
   3. Client Story – I can create an account and see the account details.
   4. Admin Story – I can view and add to a list of malls.
   5. Boy Story – I can do nothing right now.
2. Tasks to Accomplish by October 29
   1. Set up restaurant and food menu database on the admin system.
   2. Allow orders to be sent using the client system, with real food items from the admin system.
   3. Allow clients to edit account data.
   4. Receives the orders on the order system, with details of the order minus the timestamp.
   5. Client Story – I can see the menus offered by restaurants in malls, and can send my order. I can also change the detail my account holds.
   6. Admin Story – I can view and add to lists of malls, restaurants and menus.
   7. Boy Story – I can see the orders, but do not know when they come in yet.
3. Tasks to Accomplish by November 12
   1. Implement loading function on admin system.
   2. Allow deletion of restaurants and editing of menu on the admin system.
   3. Implement all buttons on order system, along with the timestamp.
   4. Split Admin into two: Admin and Dev. Dev handles the malls and accounts while Admin refers to the restaurant accounts which handles the menu.
   5. Client Story – I can see the menus offered by restaurants in malls, and can send my order if I have sufficient load amount.
   6. Dev Story – I can add, edit and remove malls. I can demote and promote accounts.
   7. Admin Story – I can edit an item’s availability. I can add, edit and remove food items from restaurants. I can add load as soon as the client gives the corresponding amount in cash.
   8. Boy Story – I can see the orders, queue them up and delete them, and check when I am able to perform each task.
4. Tasks to Accomplish by November 26
   1. Finish Minimum Viable Product.
   2. Fix layout for each web page.
   3. Client Story – I can see the menus offered by restaurants in malls, and can send my order if I have sufficient load amount.
   4. Dev Story – I can add, edit and remove malls. I can demote and promote accounts.
   5. Admin Story – I can edit an item’s availability. I can add, edit and remove food items from restaurants. I can add load as soon as the client gives the corresponding amount in cash.
   6. Boy Story – I can see the orders, queue them up and delete them, and check when I am able to perform each task. I can send notifications to the client to inform them of the status of their orders.

# **ACTIVITY NETWORK DIAGRAM**

**Activities**

1. Set up Databases
2. Set up User Pages
3. Improve layout for User Pages

**Tasks, Distribution and Duration per Activity**

|  |  |  |
| --- | --- | --- |
| **Set-up Databases** | | |
| **Person** | **Tasks** | **Duration** |
| **April** | [T1] Set-up Mall Database | 1 day |
| [T2] Set-up Restaurant Database | 1 day |
| [T3] Set-up Food Database | 1 day |
| **Joel** | [T4] Set-up Accounts Database | 1 day |
| **Viktor** | [T5] Set-up Orders Database | 1 day |
| **Adi** | [T6] Finish EERD | 1 day |

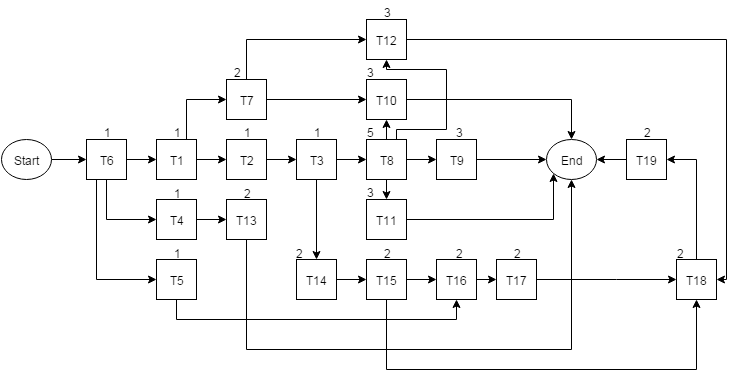
|  |  |  |
| --- | --- | --- |
| **Set-up User Pages** | | |
| **Person** | **Tasks** | **Duration** |
| **April** | [T7] Allow adding into Mall Database via admin and set-up admin mall page | 2 days |
| [T8] Allow adding into Restaurant and Food Database via admin and set-up admin restaurant and food page | 5 days |
| [T9] Allow removal (deleting) of entries from Food Database via admin | 3 days |
| [T10] Allow removal (deleting) of entries from Mall and Restaurant Database via admin | 3 days |
| [T11] Allow editing of Food Database via admin | 3 days |
| [T12] Allow editing of Mall and Restaurant Database via admin | 3 days |
| **Joel** | [T13] Allow Clients to create an account and record it into Accounts | 2 days |
| [T14] Read Food from Database based on settings of Admin | 2 days |
| [T15] Send orders to Order Database | 2 days |
| **Viktor** | [T16] Set-up Boy page and display orders from Order Database into Boy page | 2 days |
| [T17] Allow queueing and deletion of orders on Boy page | 2 days |
| **Adi** | [T18] Create GUI for each page | 2 days |

|  |  |  |
| --- | --- | --- |
| **Improve Layout for User Pages** | | |
| **Person** | **Tasks** | **Duration** |
| **Adi** | [T19] Improve page graphics | 2 days |

**Dependencies**

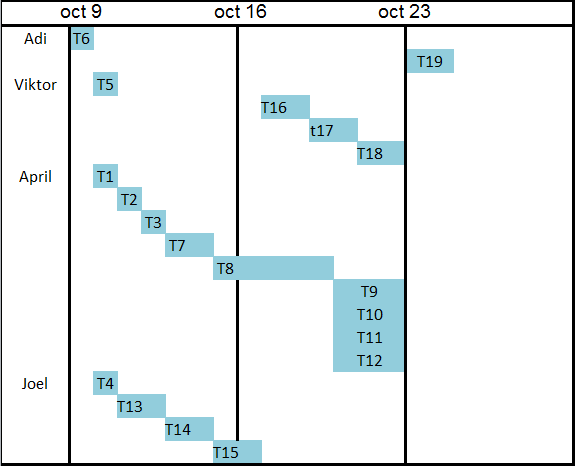
|  |  |  |
| --- | --- | --- |
| **Task** | **Duration (days)** | **Dependencies** |
| **T1** | 1 | T6 |
| **T2** | 1 | T1, T6 |
| **T3** | 1 | T1, T2, T6 |
| **T4** | 1 | T6 |
| **T5** | 1 | T6 |
| **T6** | 1 |  |
| **T7** | 2 | T1 |
| **T8** | 5 | T2, T3 |
| **T9** | 3 | T8 |
| **T10** | 3 | T7, T8 |
| **T11** | 3 | T8 |
| **T12** | 3 | T7, T8 |
| **T13** | 2 | T4 |
| **T14** | 2 | T1,T2,T3 |
| **T15** | 2 | T14 |
| **T16** | 2 | T5, T15 |
| **T17** | 2 | T16 |
| **T18** | 2 | T12, T15, T17 |
| **T19** | 2 | T18 |

**Activity Network**



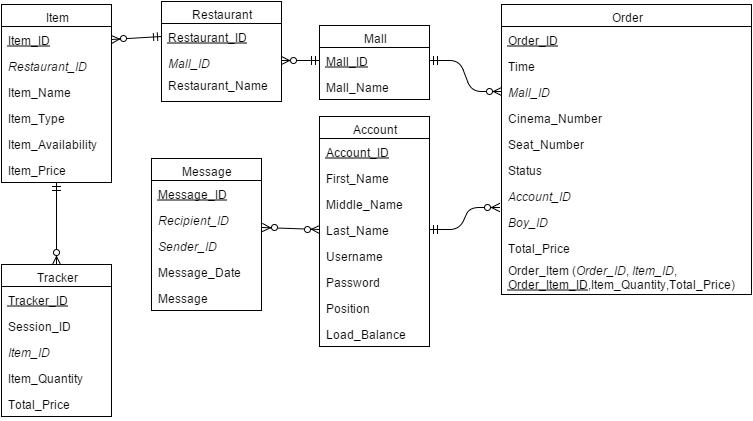
The critical path is T6-T1-T2-T3-T14-T15-T16-T17-T18-T19, and the project lasts for 16 days.

**Staff Allocation**



# **EERD AND DATABASE DICTIONARY**

**EERD**



**Data Dictionary:**

|  |  |
| --- | --- |
| **TABLE NAME:** | ACCOUNT |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| Account ID | Integer | 11 Digits | Yes | No |
| First Name | Text | 15 Characters | No | No |
| Middle Name | Text | 15 Characters | No | No |
| Last Name | Text | 15 Characters | No | No |
| Username | VarChar | 15 Characters | No | No |
| Password | VarChar | 15 Characters | No | No |
| Position | Text | 10 Characters | No | No |
| Load Balance | Decimal | 10 Digits | No | No |
| Mall ID | Integer | 11 Digits | No | Yes |
| Restaurant ID | Integer | 11 Digits | No | Yes |

# 

|  |  |
| --- | --- |
| **TABLE NAME:** | ORDER |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| **Order ID** | Integer | 11 Digits | Yes | No |
| **Time** | DATETIME | 6 | No | No |
| **Mall ID** | Integer | 11 Digits | No | Yes |
| **Cinema Number** | VarChar | 2 Characters | No | No |
| **Seat Number** | VarChar | 3 Characters | No | No |
| **Status** | VarChar | 10 Characters | No | No |
| **Account ID** | Integer | 11 Digits | No | Yes |
| **Boy ID** | Integer | 11 Digits | No | Yes |
| **Total Price** | Decimal | 10 Digits | No | No |

# 

|  |  |
| --- | --- |
| **TABLE NAME:** | MALL |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| **Mall ID** | Integer | 7 Digits | Yes | No |
| **Mall Name** | VarChar | 32 Characters | No | No |

# 

|  |  |
| --- | --- |
| **TABLE NAME:** | RESTAURANT |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| **Restaurant ID** | Integer | 11 Digits | Yes | No |
| **Mall ID** | Integer | 11 Digits | No | Yes |
| **Restaurant Name** | VarChar | 255 Characters | No | No |

# 

|  |  |
| --- | --- |
| **TABLE NAME:** | ITEM |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| **Item ID** | Integer | 11 Digits | Yes | No |
| **Restaurant ID** | Integer | 11 Digits | No | Yes |
| **Item Name** | VarChar | 255 Characters | No | No |
| **Item Type** | VarChar | 15 Characters | No | No |
| **Item Availability** | Integer | 1 Digit | No | No |
| **Item Price** | Decimal | 5 Digits, 0 Decimal Places | No | No |

# 

|  |  |
| --- | --- |
| **TABLE NAME:** | ORDER ITEM |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| **Order ID** | Integer | 11 Digits | Yes | Yes |
| **Item ID** | Integer | 11 Digits | Yes | Yes |
| **Order Item ID** | Integer | 11 Digits | Yes | No |
| **Item Quantity** | Integer | 11 Digits | No | No |
| **Total Price** | Integer | 11 Digits | No | No |

# 

|  |  |
| --- | --- |
| **TABLE NAME:** | TRACKER |

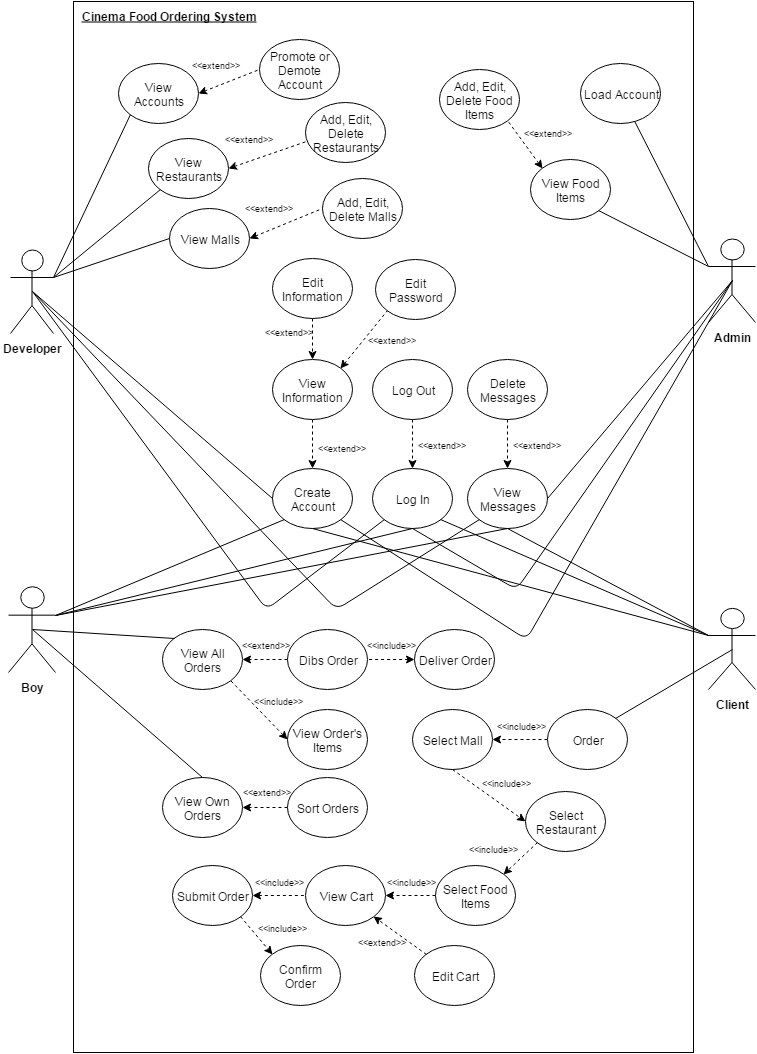
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| **Tracker ID** | Integer | 11 Digits | Yes | No |
| **Session ID** | VarChar | 32 Characters | No | No |
| **Item ID** | Integer | 11 Digits | No | Yes |
| **Item Quantity** | Integer | 11 Digits | No | No |
| **Total Price** | Integer | 11 Digits | No | No |

# 

|  |  |
| --- | --- |
| **TABLE NAME:** | MESSAGE |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **FIELD NAME** | **DATA TYPE** | **LENGTH** | **PK?** | **FK** |
| **Message ID** | Integer | 11 Digits | Yes | No |
| **Recipient ID** | Integer | 11 Digits | Yes | Yes |
| **Sender ID** | Integer | 11 Digits | Yes | Yes |
| **Message Date** | DATETIME |  | No | No |
| **Message** | Text | 255 Characters | No | No |

# **USE CASE DIAGRAM**



# **USE CASE DESCRIPTIONS**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Create Account | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | Creating an account for user identification | | |
| **OVERVIEW:** | The user will be asked for personal information to create a unique account with. The account will be the channel which will allow the user to interact with the system. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records. | | |
| **PRE-CONDITION/S:** | None | | |
| **POST CONDITION/S:** | The user will have an account by the end of the process. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user opens up the Log In page. | |  | Alternative for Step 4: The username put in is already in the system. The system displays an error message and cancels the process. |
| 1. The user clicks the Create Account button. | | 1. The system opens up the Create Account page. | Alternative for Step 4: The two copies of the password are not the same. The system displays an error message and cancels the process. |
| 1. The user inputs a first name, middle name and last name, along with a username and a password put in two times, then presses the Submit button. | | 1. The system takes the data and creates an account record, displaying a message to show the result of the creation. | Alternative for Step 2: Any or all of the information asked for is left blank and the Submit button is pressed. The system displays an error message. |
|  | |  | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Log In | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | Logging in to an account | | |
| **OVERVIEW:** | The user must log in to an account using the correct keys to access the information the account holds and do the tasks it can perform. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records. | | |
| **PRE-CONDITION/S:** | The user must have an account, and must know the log-in details. | | |
| **POST CONDITION/S:** | The user can access the account. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user opens up the Log In page. | |  | Alternative for Step 2: The username does not match any record. The system displays an error message and cancels the process. |
| 1. The user puts in a username and a password then presses the Log In button. | | 1. The system retrieves the account record and logs the user in. | Alternative for Step 2: The username-password combination is incorrect. The system displays an error message and cancels the process. |
|  | |  | Alternative for Step 2: Any or all of the information asked for is left blank and the Log In button is pressed. The system displays an error message and cancels the process. |
|  | |  | Exception for Step 2: The user puts in information but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Log Out | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | Logging out of an account | | |
| **OVERVIEW:** | The user logs out of an account because there is no more need to use it. The system saves all data and stores it, closing access to the account so no one else can use it until it is needed again. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records. | | |
| **PRE-CONDITION/S:** | The user must have an account, and must be logged in. | | |
| **POST CONDITION/S:** | The user account is logged out. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user is using any page in the system. | |  | Exception for Step 3: The user leaves the window. The system does not save the account data logged in. |
| 1. The user presses the log out button. | | 1. The system automatically logs out, and stops accessing the account record. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Information | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | Seeing the information an account holds | | |
| **OVERVIEW:** | The user can view the information held by his or her account. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records. | | |
| **PRE-CONDITION/S:** | The user must have an account, and must be logged in. | | |
| **POST CONDITION/S:** | The user’s account information is displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user is using any page in the system. | |  |  |
| 1. The user presses the Account Information button. | | 1. The system takes the user’s record from the Account table and organizes the information in a table on the Account page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Edit Information | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | Editing the information an account holds | | |
| **OVERVIEW:** | The user can change the information held by his or her account, and save the changes. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records. | | |
| **PRE-CONDITION/S:** | The user must have an account, and must be logged in to the account. | | |
| **POST CONDITION/S:** | Any changes to the account’s data that the user makes will be saved. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user is using the Account page. | |  | Alternative for Step 4: Any or all of the information asked for is left blank and the Submit button is pressed. The system displays an error message and cancels the process. |
| 1. The user presses the Edit button. | | 1. The system opens the Edit Account page. | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |
| 1. The user is allowed to edit his or her first name, middle name and last name then presses the Submit button. | | 1. The system saves the changes and displays a message showing the successful edit. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Edit Password | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | Editing the password an account holds | | |
| **OVERVIEW:** | The user can change the password needed to access his or her account, and save the changes. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records. | | |
| **PRE-CONDITION/S:** | The user must have an account, and must be logged in. | | |
| **POST CONDITION/S:** | Any changes to the account’s password that the user makes will be saved. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user is using any page in the system. | |  | Alternative for Step 4: The old password is incorrect. The system displays an error message and cancels the process. |
| 1. The user presses the Edit Password button. | | 1. The system opens the Edit Password page. | Alternative for Step 4: The two copies of the new password are not the same. The system displays an error message and cancels the process. |
| 1. The user puts in the old password and the new password twice then presses the Submit button. | | 1. The system saves the changes and displays a message showing the successful edit. | Alternative for Step 4: Any or all of the information asked for is left blank and the Log In button is pressed. The system displays an error message and cancels the process. |
|  | |  | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Messages | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | View the messages sent to an account | | |
| **OVERVIEW:** | The user can view messages from other accounts sent to his or her account, about notifications regarding the system. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records, and Message table in database for checking messages. | | |
| **PRE-CONDITION/S:** | The user must have an account, and must be logged in. | | |
| **POST CONDITION/S:** | The messages sent to the user’s account are displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user is using any page in the system. | |  | Alternative for Step 4: No checkboxes are pressed before the View Message button. The system does nothing. |
| 1. The user presses the Messages button. | | 1. The system takes records from the Message table linked to the account and displays them in a table on the Messages page. | Alternative for Step 4: More than one or all checkboxes are pressed then the View Message button is pressed. The system only displays the details of the first message selected. |
| 1. The user presses a checkbox next to a message then presses the View Message button. | | 1. The system shows the details of the selected message on the Messages page. | Exception for Step 4: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Delete Messages | | |
| **ACTOR/S:** | Client, Boy, Admin, Developer | | |
| **PURPOSE:** | Delete the messages sent to an account | | |
| **OVERVIEW:** | The user can delete messages from other accounts. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking user records, and Message table in database for checking messages. | | |
| **PRE-CONDITION/S:** | The user must be logged in to an account and must have gone through the View Messages use case. | | |
| **POST CONDITION/S:** | Any deleted messages will be removed from the system. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The user is using the Messages page. | |  | Alternative for Step 4: No checkboxes are pressed before the Delete button is pressed. The system does nothing. |
| 1. The user presses a checkbox next to a message then presses the Delete button. | | 1. The system removes the selected record from the Message table and reflects the change on the page. | Alternative for Step 4: More than one or all checkboxes are pressed then the View Message button is pressed. The system deletes all messages selected. |
|  | |  | Exception for Step 4: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Malls | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | View the malls in the system | | |
| **OVERVIEW:** | The developer can view the malls in the system. | | |
| **CROSS-REFERENCE/S:** | Mall table in database for checking mall records, and connection to Account database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account. | | |
| **POST CONDITION/S:** | All malls in the system will have their information displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Developer is using any page in the system. | |  |  |
| 1. The Developer presses the Malls button. | | 1. The system takes all records from the Mall table and organizes the information in a table on the All Malls page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Add Malls | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Adding malls into the system | | |
| **OVERVIEW:** | The developer can add mall records into the system. | | |
| **CROSS-REFERENCE/S:** | Mall table in database for checking mall records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account, and must have gone through the View Malls use case. | | |
| **POST CONDITION/S:** | New mall records created will be stored into the system. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Developer is using the All Malls page. | |  | Alternative for Step 4: The information asked for is left blank and the Add Mall button is pressed. The system displays an error message. |
| 1. The Developer presses the Add button. | | 1. The system displays a form on the page asking for a mall name. | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |
| 1. The Developer puts in a name then presses the Add Mall button. | | 1. The system creates a new mall record and stores it in the Mall table. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Edit Malls | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Editing malls in the system | | |
| **OVERVIEW:** | The developer can edit mall records in the system. | | |
| **CROSS-REFERENCE/S:** | Mall table in database for checking mall records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account, and must have gone through the View Malls use case. | | |
| **POST CONDITION/S:** | Any mall records edited will be updated in the system. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Developer is using the All Malls page. | |  | Alternative for Step 2: No checkboxes are pressed before the Edit button is pressed. The system displays a notification asking the user to select a mall. |
| 1. The Developer presses a checkbox next to a mall then presses the Edit button. | | 1. The system displays a form on the page asking for a mall name. | Alternative for Step 2: More than one or all checkboxes are pressed then the Edit button is pressed. The system only allows editing of the first mall selected. |
| 1. The Developer puts in a name then presses the Save Changes button. | | 1. The system edits the selected mall record in the Mall table and notifies the user of the result of the operation. | Alternative for Step 4: The information asked for is left blank and the Save Changes button is pressed. The system still displays a notification but does nothing to the selected mall record. |
|  | |  | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |
|  | |  | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Delete Malls | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Deleting malls in the system | | |
| **OVERVIEW:** | The developer can delete mall records from the system. | | |
| **CROSS-REFERENCE/S:** | Mall table in database for checking mall records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account, and must have gone through the View Malls use case. | | |
| **POST CONDITION/S:** | Any mall records deleted will be removed from the system. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Developer is using the All Malls page. | |  | Alternative for Step 2: More than one or all checkboxes are pressed then the Delete button is pressed. The system deletes the records of all selected malls. |
| 1. The Developer presses a checkbox next to a mall then presses the Delete button. | | 1. The system removes the selected mall record from the Mall table and reflects the changes on the page. | Alternative for Step 3: There is a boy or there are restaurants linked to the selected mall then the Delete button is pressed. The system displays a notification telling the user to remove any records linked to the mall first before it can be deleted. |
|  | |  | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Restaurants | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | View the restaurants in the system | | |
| **OVERVIEW:** | The developer can view the restaurants in the system. | | |
| **CROSS-REFERENCE/S:** | Restaurant table in database for checking restaurant records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account. | | |
| **POST CONDITION/S:** | All restaurants in the system will have their information displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Developer is using any page in the system. | |  |  |
| 1. The Developer presses the Restaurants button. | | 1. The system takes all records from the Restaurant table and organizes the information in a table on the All Restaurants page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Add Restaurants | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Adding restaurants into the system | | |
| **OVERVIEW:** | The developer can add restaurant records into the system. | | |
| **CROSS-REFERENCE/S:** | Restaurant table in database for checking restaurant records, and connection to Account and Mall tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account, and must have gone through the View Restaurant use case. | | |
| **POST CONDITION/S:** | New restaurant records created will be stored into the system. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Developer is using the All Restaurants page. | |  | Alternative for Step 4: The information asked for is left blank and the Add Restaurant button is pressed. The system still displays the notification saying the user. |
| 1. The Developer presses the Add button. | | 1. The system displays a form on the page asking for a restaurant name and a mall to link it to. | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |
| 1. The Developer puts in a name, selects a mall then presses the Add Restaurant button. | | 1. The system creates a new restaurant record linked to the selected mall record and stores it in the Restaurant table, displaying a notification on the result of the operation. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Edit Restaurants | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Editing restaurants in the system | | |
| **OVERVIEW:** | The developer can edit mall records in the system. | | |
| **CROSS-REFERENCE/S:** | Restaurant table in database for checking restaurant records, and connection to Account and Mall tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account, and must have gone through the View Restaurant use case. | | |
| **POST CONDITION/S:** | Any restaurant records edited will be updated in the system. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Developer is using the All Restaurants page. | |  | Alternative for Step 2: No checkboxes are pressed before the Edit button is pressed. The system displays a notification asking the user to select a restaurant. |
| 1. The Developer presses a checkbox next to a restaurant then presses the Edit button. | | 1. The system displays a form on the page asking for a restaurant name and a mall to link it to. | Alternative for Step 2: More than one or all checkboxes are pressed then the Edit button is pressed. The system only allows editing of the first mall selected. |
| 1. The Developer puts in a name, selects a mall then presses the Save Changes button. | | 1. The system edits the selected restaurant record in the Mall table and notifies the user of the result of the operation. | Alternative for Step 4: The information asked for is left blank and the Save Changes button is pressed. The system still displays a notification but does nothing to the record. |
|  | |  | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |
|  | |  | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Delete Restaurants | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Deleting restaurants in the system | | |
| **OVERVIEW:** | The developer can delete restaurant records from the system. | | |
| **CROSS-REFERENCE/S:** | Restaurant table in database for checking restaurant records, and connection to Account and Mall tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account, and must have gone through the View Restaurant use case. | | |
| **POST CONDITION/S:** | Any restaurant records deleted will be removed from the system. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Developer is using the All Restaurants page. | |  | Alternative for Step 2: More than one or all checkboxes are pressed then the Delete button is pressed. The system deletes the records of all selected malls. |
| 1. The Developer presses a checkbox next to a restaurant then presses the Delete button. | | 1. The system removes the selected restaurant record from the Restaurant table and reflects the changes on the page. | Alternative for Step 3: There is an admin or there are food items linked to the selected restaurant then the Delete button is pressed. The system displays a notification telling the user to remove any records linked to the restaurant first before it can be deleted. |
|  | |  | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Promote Account | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Change the position of client accounts into either boy or admin | | |
| **OVERVIEW:** | The developer can view the non-developer accounts in the system and promote clients into admins or boys. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking account records. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account. | | |
| **POST CONDITION/S:** | Any changes to any other account’s position that the user makes will be saved. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Developer is using any page in the system. | |  | Alternative for Step 4: No checkboxes are pressed before any Promote button is pressed. The system displays a notification asking the user to select an account. |
| 1. The Developer presses the Promote/Demote Account button. | | 1. The system loads the All Accounts page. | Alternative for Step 4: More than one or all checkboxes are pressed then any Promote button is pressed. The system only allows promoting of the first account selected. |
| 1. The Developer presses a checkbox next to an account then presses the Promote to Boy/Promote to Admin button. | | 1. The system displays a form on the page asking for a mall selection/a restaurant-mall selection. | Alternative for Step 6: The information asked for is left blank and the Promote button is pressed. The system still displays a message but does not update any account. |
| 1. The Developer puts in the requested information then presses the Promote to Admin/Promote to Boy button. | | 1. The system updates the account record’s status and links it to the selected mall/restaurant-mall. | Exception for Step 6: The user puts in information but leaves the page. The system cancels the process. |
|  | |  | Exception for Step 4: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Demote Account | | |
| **ACTOR/S:** | Developer | | |
| **PURPOSE:** | Change the position of boy or admin accounts into client | | |
| **OVERVIEW:** | The developer can view the non-developer accounts in the system and demote admins and boys into clients. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking account records. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Developer account. | | |
| **POST CONDITION/S:** | Any changes to any other account’s position that the user makes will be saved. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Developer is using any page in the system. | |  | Alternative for Step 4: No checkboxes are pressed before the Demote button is pressed. The system still displays the message but actually does nothing. |
| 1. The Developer presses the Promote/Demote Account button. | | 1. The system loads the All Accounts page. | Alternative for Step 4: More than one or all checkboxes are pressed then the Demote button is pressed. The system changes the position of all the accounts selected. |
| 1. The Developer presses a checkbox next to an account then presses the Demote button. | | 1. The system changes the position of the selected client, displaying a message reflecting the changes on the page. | Exception for Step 4: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Food Items | | |
| **ACTOR/S:** | Admin | | |
| **PURPOSE:** | View the food items of the restaurant the admin account is linked to | | |
| **OVERVIEW:** | The admin can view the food items of the account’s restaurant. | | |
| **CROSS-REFERENCE/S:** | Item table in database for checking item records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to Admin account. | | |
| **POST CONDITION/S:** | All food items linked to the account’s restaurant will be displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Admin is using any page in the system. | |  |  |
| 1. The Admin presses the Food Items button. | | 1. The system takes all records from the Item table and organizes the information in a table on the Restaurant Menu page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Add Food Items | | |
| **ACTOR/S:** | Admin | | |
| **PURPOSE:** | Adding food items into the system | | |
| **OVERVIEW:** | The admin can add food item records into the system. | | |
| **CROSS-REFERENCE/S:** | Item table in database for checking food item records, and connection to Account and Restaurant tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to an Admin account, and must have gone through the View Food Items use case. | | |
| **POST CONDITION/S:** | New food item records created will be stored into the system. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Admin is using the Restaurant Menu page. | |  | Alternative for Step 4: Any or all of the information asked for is left blank and the Add Menu Item button is pressed. The system removes the form and displays an error message |
| 1. The Admin presses the Add button. | | 1. The system displays a form on the page asking for an item name, price, type and its availability. | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |
| 1. The Admin puts in the requested information then presses the Add Menu Item button. | | 1. The system creates a new food item record and stores it in the Item table. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Edit Food Items | | |
| **ACTOR/S:** | Admin | | |
| **PURPOSE:** | Editing food items in the system | | |
| **OVERVIEW:** | The admin can edit food item records in the system. | | |
| **CROSS-REFERENCE/S:** | Item table in database for checking food item records, and connection to Account and Restaurant tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to an Admin account, and must have gone through the View Food Items use case. | | |
| **POST CONDITION/S:** | Any food item records edited will be updated in the system. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Admin is using the Restaurant Menu page. | |  | Alternative for Step 2: No checkboxes are pressed before the Edit button is pressed. The system displays a notification asking the user to select a food item. |
| 1. The Admin presses a checkbox next to a food item then presses the Edit button. | | 1. The system displays a form on the page asking for an item name, price, type and its availability. | Alternative for Step 2: More than one or all checkboxes are pressed then the Edit button is pressed. The system only allows editing of the first food item selected. |
| 1. The Admin puts in the requested information then presses the Save Changes button. | | 1. The system edits the selected food item record in the Item table and notifies the user of the result of the operation. | Alternative for Step 4: Any or all of the information asked for is left blank and the Save Changes button is pressed. The system removes the form and displays an error message. |
|  | |  | Exception for Step 4: The user puts in information but leaves the page. The system cancels the process. |
|  | |  | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Delete Food Items | | |
| **ACTOR/S:** | Admin | | |
| **PURPOSE:** | Deleting food items in the system | | |
| **OVERVIEW:** | The admin can delete food item records from the system. | | |
| **CROSS-REFERENCE/S:** | Item table in database for checking food item records, and connection to Account and Restaurant tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to an Admin account, and must have gone through the View Food Items use case. | | |
| **POST CONDITION/S:** | Any food item records deleted will be removed from the system. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Admin is using the Restaurant Menu page. | |  | Alternative for Step 2: More than one or all checkboxes are pressed then the Delete button is pressed. The system deletes the records of all selected food items. |
| 1. The Admin presses a checkbox next to a food item then presses the Delete button. | | 1. The system removes the selected food item record from the Item table and reflects the changes on the page. | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Load Account | | |
| **ACTOR/S:** | Admin | | |
| **PURPOSE:** | Add value to a client account’s load | | |
| **OVERVIEW:** | The admin can add load to a client account and notify the client. | | |
| **CROSS-REFERENCE/S:** | Account table in database for checking mall records, and connection to Message table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to an Admin account. | | |
| **POST CONDITION/S:** | Loading will update the client account and send a message to the account. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Admin is using any page in the system. | |  | Alternative for Step 4: No checkboxes are pressed before the Add Load button is pressed. The system displays a notification asking the Admin to select an account. |
| 1. The Admin presses the Load Accounts button. | | 1. The system takes all Client records from the Account table and organizes the information in a table on the Clients page. | Alternative for Step 4: More than one or all checkboxes are pressed then the Add Load button is pressed. The system only displays a form for the first client selected. |
| 1. The Admin presses a checkbox next to an account then presses the Load Account button. | | 1. The system shows a form to fill up on the Clients page. | Alternative for Step 6: The Add Load button is pressed but no amount is specified. The system will add zero load and still go through Steps 7 and 8. |
| 1. The Admin specifies an amount in the form then presses the Add Load to Account button. | | 1. The system updates the client record and shows a notification indicating the result | Exception for Step 4: The user presses one or more checkboxes but leaves the page. The system cancels the process. |
|  | | 1. The system creates a message record to show who sent the load, to whom, how much was sent and when. | Exception for Step 6: The user fills up the form but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Orders | | |
| **ACTOR/S:** | Boy | | |
| **PURPOSE:** | View all orders the system | | |
| **OVERVIEW:** | The boy can view all the orders recorded in the system. | | |
| **CROSS-REFERENCE/S:** | Order table in database for checking orders, and connection to Account. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Boy account. | | |
| **POST CONDITION/S:** | All orders in the system will be displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Boy is using any page in the system. | |  |  |
| 1. The Boy presses the All Orders button. | | 1. The system takes all records from the Order table, with links to the Orderitems table, and shows the information in a table on the Mall Orders page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Order Items | | |
| **ACTOR/S:** | Boy | | |
| **PURPOSE:** | View the items linked to a specific order | | |
| **OVERVIEW:** | The boy can view the details of the food items linked to an order. | | |
| **CROSS-REFERENCE/S:** | Order and Orderitems tables in database for checking order records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Boy account, and must have gone through the View Orders use case. | | |
| **POST CONDITION/S:** | All food items linked to a chosen order will be displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Boy is using the Mall Orders page. | |  | Alternative for Step 2: No checkboxes are pressed before the View Order button is pressed. The system the order items table but it is empty. |
| 1. The Boy presses a checkbox next to an order then presses the View Order button. | | 1. The system takes all records from the Orderitems table linked to the selected order record, and shows the information in a table on the Mall Orders page. | Alternative for Step 2: More than one or all checkboxes are pressed then the View Order button is pressed. The system displays the order items found in all orders. |
|  | |  | Exception for Step 2: The user presses on or more checkbox but leaves the page. The system cancels the process. |
|  | |  | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Dibs Order | | |
| **ACTOR/S:** | Boy | | |
| **PURPOSE:** | Link an order record and a boy account | | |
| **OVERVIEW:** | The boy account can be linked to to order records. | | |
| **CROSS-REFERENCE/S:** | Order and Account tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Boy account, and must have gone through the View Orders use case. | | |
| **POST CONDITION/S:** | Any orders called dibs upon will be linked to the boy account. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Boy is using the Mall Orders page. | |  | Exception for Step 2: The user presses on or more checkbox but leaves the page. The system cancels the process. |
| 1. The Boy presses a checkbox next to an order then presses the Dibs button. | | 1. The system links the selected order to the boy account and changes its status from Waiting to Assembling, updating the table on the Mall Orders page. | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Deliver Order | | |
| **ACTOR/S:** | Boy | | |
| **PURPOSE:** | Delete an order and remove its links to a boy account | | |
| **OVERVIEW:** | The boy removes an order from the system. | | |
| **CROSS-REFERENCE/S:** | Order and Account tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Boy account, and must have gone through the View Orders use case. | | |
| **POST CONDITION/S:** | Any orders delivered will be deleted from the system. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Boy is using the Mall Orders page. | |  | Exception for Step 2: The user presses on or more checkbox but leaves the page. The system cancels the process. |
| 1. The Boy presses a checkbox next to an order then presses the Dibs button. | | 1. The system deletes the selected order and destroys its link to the boy account updating the table on the Mall Orders page. | Exception for Step 2: The user presses one or more checkboxes but leaves the page. The system cancels the process. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Own Orders | | |
| **ACTOR/S:** | Boy | | |
| **PURPOSE:** | View the orders linked to the boy account | | |
| **OVERVIEW:** | The boy can view the food items of the orders claimed dibs on. | | |
| **CROSS-REFERENCE/S:** | Order and Orderitems tables in database for checking order item records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Boy account. | | |
| **POST CONDITION/S:** | All orders linked to the boy account will be displayed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Boy is using any page in the system. | |  |  |
| 1. The Boy presses the Orders to Prepare button. | | 1. The system takes all food item records from the Items table linked to orders from the Order table linked to the boy account and organizes the information in a table on the Claimed Orders page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Sort Orders | | |
| **ACTOR/S:** | Boy | | |
| **PURPOSE:** | Sort the orders linked to the boy account | | |
| **OVERVIEW:** | The boy can sort the items of the orders claimed dibs on for convenience, sorted either by the restaurant the items can be found in or by the clients who ordered the items. | | |
| **CROSS-REFERENCE/S:** | Order and Orderitems tables in database for checking order item records, and connection to Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Boy account, and must have gone through the View Own Orders use case. | | |
| **POST CONDITION/S:** | The orders claimed dibs on will be sorted depending on which sorting display is chosen. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Boy is using the Claimed Orders page. | |  |  |
| 1. The boy presses one of two buttons, the Sort by Restaurant button or the Sort by Account button. | | 1. The system sorts the food items accordingly, depending on the button pressed. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Order | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | Place an order | | |
| **OVERVIEW:** | The client can place an order in the system. | | |
| **CROSS-REFERENCE/S:** | Account table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account. | | |
| **POST CONDITION/S:** | The client’s order will start being processed. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Client is using any page in the system. | |  |  |
| 1. The Client presses the Order Food button. | | 1. The system opens the Choose a Mall page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Select Mall | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | Select a mall | | |
| **OVERVIEW:** | The client can select a mall to order from. | | |
| **CROSS-REFERENCE/S:** | Account table in database, with connections to Mall table in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account, and must have started the Order use case. | | |
| **POST CONDITION/S:** | The mall selected will be sent to the Select Restaurant use case. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Client is using the Choose a Mall page. | |  | Alternative for Step 2: The user presses the I’m Here at This Mall button without selecting a mall. The system will still go to the Available Menus page but the next use cases will not perform the normal flow of events. |
| 1. The client selects a mall from a drop-down list, then presses the I’m Here at This Mall button. | | 1. The system opens up the Available Menus page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Select Restaurant | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | Select a restaurant from the mall selected in the previous use case | | |
| **OVERVIEW:** | The client can select any restaurant in the mall chosen in the last use case to order from. | | |
| **CROSS-REFERENCE/S:** | Account table in database, with connections to Mall and Restaurant tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account, and must have gone through the Select Mall use case. | | |
| **POST CONDITION/S:** | The restaurant selected will be sent to the Select Food Items use case, along with the mall selection from the Select Mall use case. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Client is using the Available Menus page. | |  | Alternative for Step 3: The user goes back to Step 2 even after Step 3 is finished. The system continues the process from there. |
| 1. The client clicks a drop-down list. | | 1. The system loads the drop down list with the restaurants linked to the mall selected from the Select Mall use case. | Exception for Step 2: The user did not select a mall in the Select Mall use case. No restaurant will be shown in the drop down list. |
| 1. The user selects a restaurant then presses the Choose a Restaurant button. | | 1. The system displays the food items linked to the selected restaurant on a table in the page. | Exception for Step 1: The user goes back to the Select Mall use case. The system goes back to the Choose a Mall page. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Select Food Items | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | Select food items from the restaurant and mall selected in the previous use cases | | |
| **OVERVIEW:** | The client can select any food item in the restaurant and mall chosen in the last use cases to add to a temporary cart. | | |
| **CROSS-REFERENCE/S:** | Account table in database, with connections to Mall, Restaurant, Item and Tracker tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account, and must have gone through the Select Restaurant use case. | | |
| **POST CONDITION/S:** | Any food items selected will be sent to the View Cart use case, along with the mall and restaurant selections from the Select Mall and Select Restaurant use cases. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Client is using the Available Menus page. | |  | Alternative for Step 3: The user decides not to add the item to the cart and exits the pop-up. Return to Step 1. |
| 1. The Client presses an Add to Cart button next to an item. | | 1. The system displays a pop-up asking the user for the quantity of the item selected. | Alternative for Step 4: The user sets no quantity then presses the Okay button. The system will not put the item in the Tracker table. |
| 1. The Client sets a quantity of the product to be ordered then presses the Okay button. | | 1. The system puts the food item and its quantity to the Tracker table in the database | Exception for Step 2: The user did not select a mall in the Select Mall use case, and thus no restaurant in the Select Restaurant use case. No food items will appear. |
|  | |  | Exception for Step 4: The user sets a quantity but leaves the page. The system cancels the process. |
|  | |  | Exception for Step 1: The user goes back to the Select Restaurant use case. The system saves the data manipulated in this use case. |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | View Cart | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | View the current cart | | |
| **OVERVIEW:** | The client can view and remove the items in placed in the temporary cart. | | |
| **CROSS-REFERENCE/S:** | Account table in database, with connections to Mall, Restaurant, Item and Tracker tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account, and must have gone through the Select Restaurant use case. | | |
| **POST CONDITION/S:** | All food items from the Select Food Item use case may be sent to the Edit Cart or Submit Order use cases, along with the mall and restaurant selections from the Select Mall and Select Restaurant use cases. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Client is using the Available Menus page. | |  | Exception for Step 1: The user goes back to the Select Food Item use case. The system saves the data manipulated in this use case. The system goes back to the Available Menus page. |
| 1. The Client presses the View Cart button. | | 1. The system takes all records from the Tracker table and organizes the information in a table on the Current Cart page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Edit Cart | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | Edit the current cart | | |
| **OVERVIEW:** | The client can remove the items placed in the temporary cart. | | |
| **CROSS-REFERENCE/S:** | Account table in database, with connections to Mall, Restaurant, Item and Tracker tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account, and must have gone through the View Cart use case. | | |
| **POST CONDITION/S:** | Any food items removed from the cart will be removed, and those that remain will be sent to the Select Food Items or Submit Order use cases, along with the mall and restaurant selections from the Select Mall and Select Restaurant use cases. | | |
| **Actor Actions** | | **System Actions** | **Alternatives and Exceptions** |
| 1. The Client is using the Current Cart page. | |  | Exception for Step 1: The user goes back to the Select Food Item use case. The system saves the data changes in this use case. The system returns to the Available Menus page. |
| 1. The Client presses a Remove from Cart button next to an item. | | 1. The system removes the food item from the Tracker table and reflects the change on the page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Submit Order | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | Compiles the order | | |
| **OVERVIEW:** | The client views the details of the cart before finally creating an order record. | | |
| **CROSS-REFERENCE/S:** | Account table in database, with connections to Mall, Restaurant, Item and Tracker tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account, and must have gone through the View Cart use case. | | |
| **POST CONDITION/S:** | The mall, restaurant and food items selected from the Select Mall, Select Restaurant and Select Food Item use cases, respectively, will be Sent to the Confirm Order use case before making the order. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Client is using the Current Cart page. | |  |  |
| 1. The Client presses the Submit button. | | 1. The system loads the Order Form page. |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE NAME:** | Confirm Order | | |
| **ACTOR/S:** | Client | | |
| **PURPOSE:** | Creates the order | | |
| **OVERVIEW:** | The client creates an order based on the temporary data stored from the Select Mall, Select Restaurant and Select Food Item use cases. | | |
| **CROSS-REFERENCE/S:** | Account table in database, with connections to Mall, Restaurant, Item and Tracker and Order tables in database. | | |
| **PRE-CONDITION/S:** | The user must be logged in to a Client account, and must have gone through the Submit Order use case. | | |
| **POST CONDITION/S:** | The mall, restaurant and food items selected from the Select Mall, Select Restaurant and Select Food Item use cases, respectively, will be stored in a new order record, along with the cinema number and seat number. | | |
| **Normal Flow** | | **Exceptions** | **Alternative** |
| 1. The Client is using the Order Form page. | | 1. The system has loaded a form asking the client for a cinema number and a seat number. | Alternative for Step 3: Any or all of the information asked for is left blank and the Confirm Order button is pressed. The system displays an error message and cancels the process. |
| 1. The client puts in a cinema number and a seat number then presses the Confirm Order button. | | 1. The system creates a new order record and stores it in the Order table. | Exception for Step 1: The user goes back to the View Cart use case. The system does not save the data given in this use case. The system goes back to the Current Cart page. |
|  | |  | Exception for Step 3: The user puts in information but leaves the page. The system cancels the process. |

# **LESSONS LEARNED**

There are a number of things we learned about software development in the course of this project. The two key ideas we learned about developing software are good communication and organized planning and execution. These ideas sum up the experience we have had in developing Food Buddy for CS 123.

.We learned about good communication being key to software development because the process of creating the software is a team effort. We learned to work together, seeing that each one had the success of the project in mind. Criticism had to be accepted as we went along with our project because it was important to hear everyone’s input to improve the project - this also meant sharing of ideas was encouraged. The project taught us to look after each other if we wanted a successful project, with some members not getting too far ahead of the others while the others were doing their best to catch up to the work. The process of developing Food Buddy meant working together to create our product.

Proper planning and organization were also essential to the project. This was our first time actually developing software. At times the tasks seemed easy, but we learned the hard way that we had to stay on our toes when it happens. Things actually happen very fast, and if you do not catch up to them you could do badly. We learned to do plan ahead to make sure that we knew how things should be going, and we did our best to effectively do things one at a time to make the most of each level of progress. After an initial failure, we made sure not to take the work too lightly, and to anticipate future failures and incorporate them in our plans. We learned that software development is a field that always keeps you on your toes and gets your mind critically thinking.

These two main lessons we learned help us finish the minimum viable product for Food Buddy. Granted, there’s still a lot to be done before the system can be perfect, but with these lessons in mind our group can tackled the project more effectively, and these lessons will help us in projects to come.