

**Requirements Specifications**

**P04 : Dairy Farm Management System**

|  |  |
| --- | --- |
| **Name** | **Roll no** |
| Furqan Athar | 22100070 |
| Abdullah Saleem | 22100125 |
| Khawaja Junaid | 22100072 |
| Saad Qadeer | 22100209 |

**CS 491 (A) - S1**

**Submitted to**

**Sir Waqar Ahmad**

|  |  |  |
| --- | --- | --- |
| **Content** | **Totals** | **Obtained** |
| Introduction & system actors | 5 | 5 |
| Use case diagram | 10 | 10 |
| Use case descriptions | 20 | 12 |
| Class diagram | 20 | 10 |
| Sequence diagram | 20 | 10 |
| State diagram | 5 | 5 |
| Non-functional requirements | 5 | 5 |
| Who did what | 5 | 5 |
| Review checklist | 5 | 5 |
| Overall formatting/template | 5 | 5 |
| Late submission penalty | -20 |  |
| **Total** | **100** | **72** |
| Review | 20 | **20** |
| **Grand Total** |  |  |

**Table of Contents**

[1.](#_heading=h.gjdgxs) Introduction 3

[2.](#_heading=h.30j0zll) System Actors 4

[3.](#_heading=h.1fob9te) Use Cases 5

[3.1 Use Case Diagrams 5](#_heading=h.3znysh7)

[3.2 Description of Use Cases 5](#_heading=h.2et92p0)

[4.](#_heading=h.1t3h5sf) Class Diagram 7

[4.1 Diagram 7](#_heading=h.4d34og8)

[4.2 Description 7](#_heading=h.2s8eyo1)

[5.](#_heading=h.17dp8vu) Sequence Diagrams 8

[6.](#_heading=h.lnxbz9) State Diagrams 9

[6.1 Diagram details 9](#_heading=h.35nkun2)

[6.2 Diagram 9](#_heading=h.1ksv4uv)

[7.](#_heading=h.44sinio) Non-functional Requirements / Quality Attributes 10

[8.](#_heading=h.2jxsxqh) Who Did What? 11

[9.](#_heading=h.z337ya) Review checklist 11

# Introduction

In attempts to modernize the dairy industry in Pakistan, which is one of the greatest producers and consumers of milk and other dairy products in the region, our proposed and developed system will present a solution to better manage the dairy farms and optimize sales and help in better record keeping.

Furthermore, multiple dairy farm owners will have the opportunity to set up their accounts on our management system, as we aim to develop a multi-tenant software, which will aid them in getting more customers and for the better management and record keeping of their farms. The system will allow the customer to keep track of daily milk production, sales, expenses and maintenance. Moreover, the system will also provide an interface for the customers of the dairy farm to check their delivery and monthly invoice.

Potential users of our management system include dairy farm owners (who wish to digitize their records and keep an updated track of their farms) and the customers who wish to order dairy products, be it for domestic or commercial use.

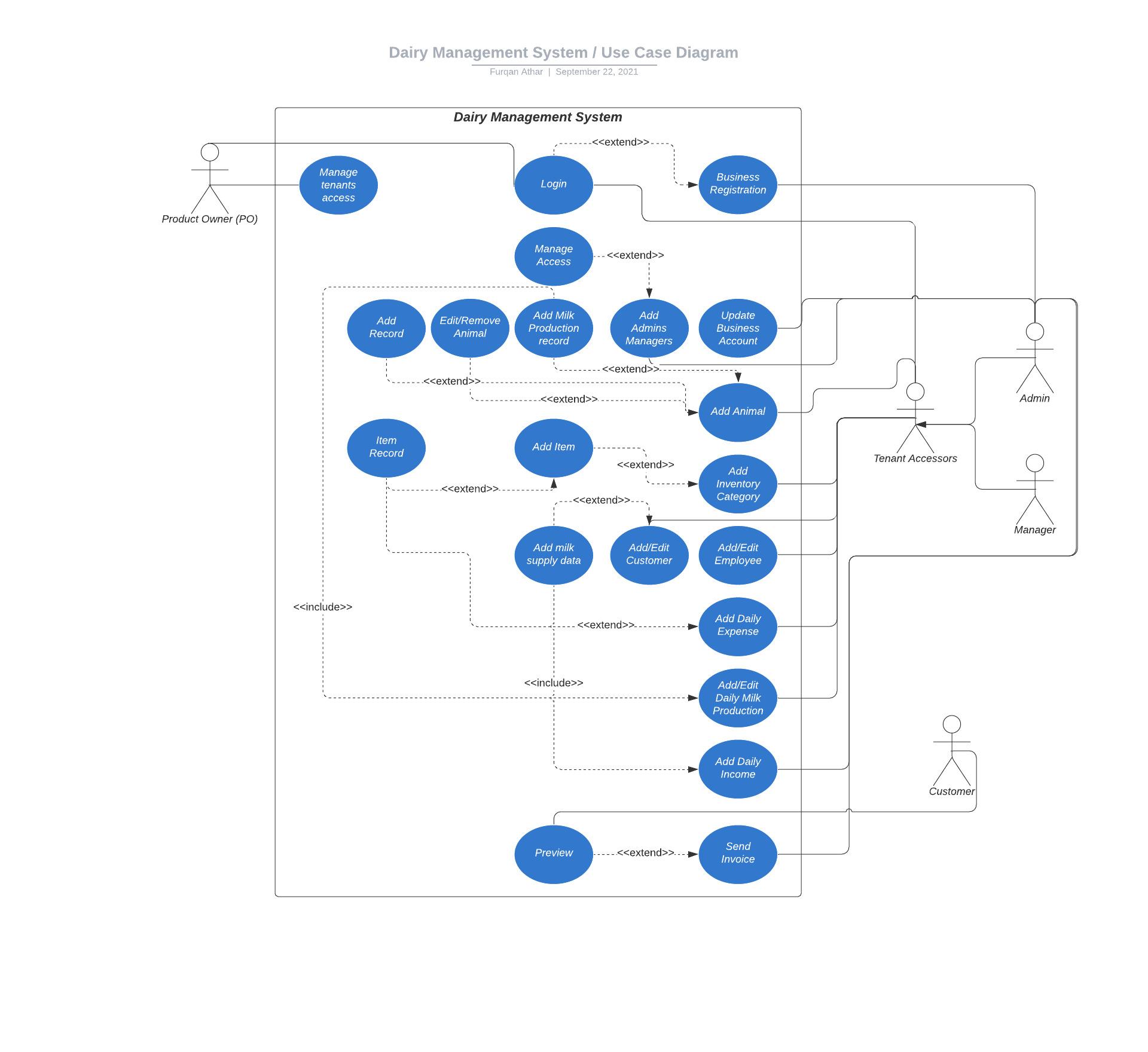
The main purpose of this product will be to provide multiple tenants a way to see various trends in their milk production, expenses and income streams and to make better decisions for the future.

# System Actors

|  |  |
| --- | --- |
| **Actor Name** | **Description** |
| Product Owner | The product owner will be present at the top of the hierarchical structure. The product Owner will ultimately decide which dairy farm owners can set up their accounts on their product. |
| Admin | Admins are the dairy farm owners/Companies themselves. A dairy farm owner will be able to set up their account/dashboard on the main product interface and is also known as the tenant admin. |
| Manager | The tenant manager will be the personnel that would be next in authority after the farm owner. The manager will be the one overlooking the logistics of the farm and is also known as the tenant manager. |
| Customer | The customer will be the actor that will be present at the lowest level of this hierarchical structure. Customer is the one who will get the milk supply and will be able to check his portal on daily basis |

# Use Cases

## Use Case Diagrams



## Description of Use Cases

<Write description of each use case separately using the template below.>

[Write use case description in a way that the roles of actor and the system become clear at each step. Refer to the sample provided in the template.

In general, there are issues with format of description and alternative paths. See my comments in some selected use cases below for specific examples.]

### Login for Super Admin (Product Owner)

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-001 | |
| **Purpose** | | The Product Owner wants to login to his website. | |
| **Pre-conditions** | |  | |
| **Post-conditions** | | Product Owner will be logged in to see the dashboard. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
|  | The PO visits the login page. | |
|  | The PO enters the login details (email, password) | |
|  | If the details are correct, he will be logged into the PO Dashboard. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
|  |  | |
| **Step #** | **Exception Paths** | |
|  | In step 3, if the details are incorrect, an error message will be displayed to the PO. | |

### Change access for tenants

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-002 | |
| **Purpose** | | The Product Owner wants to change access of multiple tenants from his dashboard | |
| **Pre-conditions** | | Complete UC-001 | |
| **Post-conditions** | | Access of a tenant will be changed. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The PO will select any of the tenants. | |
| **2.** | The PO can manage access by choosing from multiple options like Allow Access, Restrict Access and others. | |
| **3.** | The system will update the access. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. |  | |

### Register Account for Dairy Farm

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-003 | |
| **Purpose** | | The dairy farm owner wants to register his dairy farm to use our service as an admin. | |
| **Pre-conditions** | |  | |
| **Post-conditions** | | Dairy farm will be registered as a tenant to use service | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | On the registration page, the admin first provides details for his account credentials | |
| **2.** | After that,the admin will provide details regarding farm name and business domain. | |
| **3.** | The admin will click “Register”, the account details will be sent to the server for processing. | |
| **4.** | If registration is successful, the system will redirect the admin to the dashboard. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If any of the unique dairy information conflicts with the already present dairy, the system will display the error message. | |

### Addition of Admin/Manager to Management Team

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-004 | |
| **Purpose** | | The admin wants to add/edit admins and managers and want to manage their access | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | New admins and managers will be added into the management team. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin will go to the admin/manager addition page. | |
| **2.** | The admin will fill out the information like name, email and other details. | |
| **3.** | The admin will select the role for that person (Admin/Manager) | |
| **4.** | The admin will select the person status as well like allow access or restrict. | |
| **5.** | If successful, a success message will be displayed. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | If he wants to restrict someone's access immediately after adding, he can change his status to inactive/restrict. | |
| **Step #** | **Exception Paths** | |
| 1. | If any error occurs, an error message will be displayed. | |

### Update information of a Dairy Account

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-005 | |
| **Purpose** | | The admin wants to update information of the business account. | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | Dairy Account Information will get updated | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | On the business account information detail page, the admin will select “Edit”. | |
| **2.** | The admin will edit the details he wants to edit. | |
| **3.** | The admin will click “Update” | |
| **4.** | If successful, a success message will be displayed. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If the manager tries to update the account information, an error message will be displayed. | |
| 2. | If any other error occurs, an error message will be displayed. | |

### Add New Animal

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-006 | |
| **Purpose** | | The admin/manager wants to add details of new animals into their system. The page with animal details will be called an animal card | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | Animal detail will be added into the tenant system. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin/manager will click the ‘Add Animal’ button. | |
| **2.** | The admin/manager will add the animal details tag, name, image, date of birth etc. | |
| **3.** | The admin/manager will choose animal type like Milking, Child, Heifer etc | |
| **4.** | The admin/manager will select the animal status like Active, Inactive, Dead etc | |
| **5.** | The admin/manager will click “Save”. | |
| **6.** | If successful, the system will display a success message. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | In step 5, if any of the unique details conflicts with any of the already present animals, the system will display an error message. | |

### Adding animal record and editing any details

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-007 | |
| **Purpose** | | The admin/manager wants to add a record of an animal in the animal card. | |
| **Pre-conditions** | | Complete UC-003 and UC-006 | |
| **Post-conditions** | | Animal records will be added into the animal card. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | On the animal card page, The admin/manager will click “Edit” | |
| **2.** | The admin/manager will click the ‘Add Vaccination Record’ button. | |
| **3.** | The admin/Manager will select the date. | |
| **4.** | The admin/manager will add the note or description. | |
| **4.** | The admin/manager will click “Save”. | |
| **5.** | If successful, the system will display a success message. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 2, if the admin/manager clicks the 'Edit Vaccination Record’ button, he’ll change the date or description and move to step 4 | |
| **Step #** | **Exception Paths** | |
| 1. | In step 4, if any error occurs at the server side, the system will display an error message. [At which step, this error may occur?] | |

### Adding Categories to Inventory

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-008 | |
| **Purpose** | | The admin/manager wants to maintain/add inventory using categories. | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | Categories will be added into the inventory | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | On the inventory’s category addition page, the admin/manager will first select the option [which option?] from whether the quantities of items in that category will change on a regular basis or not? | |
| **2.** | The admin/manager selects the “regular basis” option. [Separate these steps into user actions and system responses, refer to the provided template.] | |
| **3.** | The system will ask whether the items to be included are Volume based or Weight based. | |
|  | If the admin/manager selects the “No on a regular basis” option, the system will not ask questions regarding volume/weight based items. [This should better be taken as alternative course of action.] Ignore this step, added into the alternate course of action | |
| **4.** | [this is clear from the sequence itself, you don’t need to mention such phrases.], The admin/manager will add the category name | |
| **5.** | The admin/manager will click on “Save” | |
| **6.** | The system will add a category into the inventory. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 2, the admin/manager can select the ‘Not on Regular basis’ option and proceed to step 4 directly. | |
| 2. | After completing step **6,** the admin/manager can click on ‘Delete’.[At which step in typical course of action will this path be taken?] | |
| 3. | The system will delete the category from the inventory. | |
| **Step #** | **Exception Paths** | |
| 1. | In step 5 , if any of the information of a new category conflicts with any of the already present categories, the system will display an error message. [At which step will this happen?] | |

### Add items in the inventory

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-009 | |
| **Purpose** | | The admin/manager wants to add items into the category with their stocks | |
| **Pre-conditions** | | Complete UC-003 and UC-008 | |
| **Post-conditions** | | Items will be added into the category | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
|  | If [Description should say, it will happen. What will happen otherwise?] the item is added in the volume category, the admin/manager can enter details of the item with their name and volume in ml. | |
| **1.** | The admin/manager will click on the category name and then will click on the ‘Add Item’ button. | |
| **2.** | If the ‘Weight’ category was clicked, the admin/manager will enter the name and weight (in kg). [Mention what will happen typically, take rest to the alternate courses of action.] | |
| **3.** | The admin/manager will click on “Save”. | |
| **4.** | The system will display a confirmation popup which will say “Are you sure? You’ll not be able to change this item's name again.” | |
| **5.** | If the admin/manager selects “Yes”, the request will be sent to the server and on successful completion, a success message will be displayed. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 2, if the ‘Volume’ category is selected, the admin/manager will add volume (in ml) instead of weight for the item. | |
| 2. | In step 2, if the ‘Miscellaneous’ category is selected, the admin/manager will add the discrete value of quantity for the item. | |
| 3. | In step 5, if the ‘No’ option is selected, the admin/manager will be redirected to the step 2. | |
| **Step #** | **Exception Paths** | |
| 1. | In step 3, if the admin/manager tries to add an item that is already present in that category, the system will display an error message and he’ll move back to step 2. | |

### Edit/Delete Inventory Item

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-010 | |
| **Purpose** | | The admin/manager wants to edit/delete the item in inventory having some exceptions. | |
| **Pre-conditions** | | Complete UC-003, UC-008 and UC-009 | |
| **Post-conditions** | | Items will be edited/removed from the inventory. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | On the inventory’s category page, the admin/manager will click “Edit” corresponding to an item. | |
| **2.** | Admin/Manager will update the stock manually. | |
| **3.** | The admin/manager will click on “save” | |
| **4.** | The system will update and add the record of inventory updates under that item’s details. | |
| **5.** | If successful, a system will display a success message. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | During step 1, 2 or 3, the admin/manager will ‘Delete’ corresponding to the item. | |
| 2. | The system will delete the item, if that item has no record. | |
| **Step #** | **Exception Paths** | |
| 1. | In alternate step 2, the system will not allow the delete event to happen if that item has some record and the system will display an error message. | |

### Add/edit/remove employees

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-011 | |
| **Purpose** | | The admin/manager wants to add/edit/remove employees data from the management system. | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | The employee will be added to the system’s database. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin/manager will click the ‘Add Employee’ button. | |
| **2.** | The admin/manager will add the employee’s salary. | |
| **3.** | The admin/manager will add the employee’s data like image, date of birth and other things. | |
| **4.** | The admin/manager will add the CNIC of that employee. | |
| **5.** | The admin/manager will click “Save”. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 1, the admin/manager will click the ‘Edit Employee’ button and move to step 2 for editing. | |
| 2. | In step 1, the admin/manager will click ‘Delete’. | |
| 3. | The system will display a confirmation message. | |
| 4. | If the admin/manager clicks ‘Yes’ and the system will verify. | |
| 5. | The system will display a success message if there are no records for that employee. | |
| **Step #** | **Exception Paths** | |
| 1. | In alternate step 4, if the employee has some record present, the system will display an error message because an employee having some history must not be deleted. | |
| 2. | In step 5, the system will display an error message if the cnic or any record conflicts with any already present employee. | |

### Add/Edit/Remove Customer Data

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-012 | |
| **Purpose** | | The admin/manager wants to add/edit/remove customers data from the management system. | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | The customer, to whom milk is being supplied, will be added to the system’s database. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | On the customer addition page, The admin/manager will add all the required fields. | |
| **2.** | The admin/manager will select the customer status like active/inactive | |
| **3.** | The admin/manager will click “Save”. | |
| **4.** | **OR** if edit is clicked for a customer, admin/manager can edit all the details they want. | |
| **5.** | **OR** if delete is clicked, the customer will be deleted. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If the customer has some record present like milk supplied history, and if the delete event occurs, an error will be displayed by the system because a customer having some history must not be deleted. | |

### Add daily expenses

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-013 | |
| **Purpose** | | The admin/manager wants to add the daily expenses. | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | The record of daily expenses will be added into the database with the specific date selected. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin/manager will first select the date. | |
| **2.** | The admin/manager will click the ‘Add from Inventory’ button. | |
| **3.** | The admin/manager will choose the option displayed from the list of inventory items | |
| **4.** | The admin/manager will enter the quantity consumed from the inventory. | |
| **5.** | The admin/manager will add the expense amount. | |
| **6.** | The admin/manager will keep going to step 2 if he wants to add more items from the inventory otherwise they will move to the below step. | |
| **7.** | The admin/manager will click “Save” | |
| **8.** | The system will display a confirmation popup which will say “Are you sure?You’ll not be able to edit this daily expense again”. | |
| **9.** | If the admin/manager selects “Yes”, the system will send a request to the server and the system will update the quantities of the items from the inventory if items are selected from the inventory. | |
| **10.** | If successful, the system will display a success message. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 2, the admin/manager clicks the 'Add Miscellaneous’ button. | |
| 2. | The admin/manager will add the name, note and expense amount for that item and follow the step 6. | |
| 3. | In step 9, the admin/manager selects ‘No’ and will move to step 6. | |
| **Step #** | **Exception Paths** | |
| 1. | In step 9, if the selected date is already in the database, the system will display an error message. | |
| 2. | In step 9, if the quantity of the medicine/food selected is greater than the quantity present in inventory, an error message will be displayed. | |

### Filter Daily Expense

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-014 | |
| **Purpose** | | The admin/manager wants to filter the daily expenses. | |
| **Pre-conditions** | | Complete UC-003 and UC-013 | |
| **Post-conditions** | | The filtered daily expenses list will be displayed. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | When the list of all daily expenses will be displayed, the admin/manager can select a specific date or a range of dates from the input form. | |
| **2.** | The system will display the results on runtime after filtration. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | On date selection, if the record is not present, a message will be displayed by the system that says “No Record Found”. | |

### Add/Edit Daily Milk Production

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-015 | |
| **Purpose** | | The admin/manager wants to add/edit daily milk production of each cow. | |
| **Pre-conditions** | | Complete UC-003 and UC-006 | |
| **Post-conditions** | | Daily milk production of each cow will be recorded into the database. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin/manager will click ‘Add Daily Milk Production’ button | |
| **2.** | The admin/manager will add milk records for both morning and evening for each active animal. | |
| **3.** | The admin/manager will click “Save” | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 1, the admin/manager will click on the ‘Edit Milk Record’ button and will follow step 2. | |
| 2. | In step 2, the admin/manager will add the total milk production of all animals in a single will. | |
| 3. | The system will automatically divide that total into each active animal. | |
| **Step #** | **Exception Paths** | |
| 1. | In step 3, the system will display an error message if the record on that date is already in the system. | |

### Filter Daily Milk Production and display Trends

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-016 | |
| **Purpose** | | The admin/manager wants to filter the daily milk production. | |
| **Pre-conditions** | | Complete UC-003 and UC-015 | |
| **Post-conditions** | | The filtered daily milk production list will be displayed. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | When the list of all daily milk production will be displayed, the admin/manager can select a specific date or a range of dates from the input form. | |
| **2.** | The system will display the graph that will show the trend of total milk production in the range of dates selected. And, | |
| **3.** | The system will display the results on runtime after filtration. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | On date selection, if the record is not present, a message will be displayed that says “No Record Found”. | |

### Add/Edit daily income

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-017 | |
| **Purpose** | | The admin wants to add/edit the daily income. | |
| **Pre-conditions** | | Complete UC-003 and UC-012 | |
| **Post-conditions** | | The daily income will be added to the database. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin will click on the ‘Add Daily Income’ button and select the date. | |
| **2.** | The admin will add the quantity of milk delivered to each customer with their corresponding rates. | |
| **3.** | The admin will select the “Paid/Unpaid” status for milk supply to each customer. | |
| **4.** | The admin will click on ‘Add Miscellaneous Source’ button | |
| **5.** | The admin will add the description and income amount corresponding to that. | |
| **6.** | The admin will follow step 3 again if he wants to add more income sources else will follow the next step. | |
| **7.** | The admin will click “Save”. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 1, the admin will click ‘Edit Daily Income’, select the date and will follow step 2 | |
| **Step #** | **Exception Paths** | |
| 1. | In step 7, the system will display an error message if the date selected is already in the database. | |

### Filter Daily Income and see Trends

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-018 | |
| **Purpose** | | The admin wants to filter the daily income. | |
| **Pre-conditions** | | Complete UC-003 and UC-013 | |
| **Post-conditions** | | The filtered daily income list will be displayed. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | When the list of all daily incomes will be displayed, the admin can select a specific date or a range of dates from the input form. | |
| **2.** | The graph, displayed by the system, will show the trend of total daily incomes in the range of dates selected. And, | |
| **3.** | The system will display results on runtime after filtration. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | On date selection, if the record is not present, the system will display the message that says “No Record Found”. | |

### Filters on customers based on area

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-019 | |
| **Purpose** | | The admin wants to filter the customers based on area | |
| **Pre-conditions** | | Complete UC-003 and UC-012 | |
| **Post-conditions** | | The filtered customers will be displayed | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | On the list of customers page, admin can search for the customer from the search box based on his name or address. | |
| **2.** | Also, he can select the area from the list of areas present in the dropdown. | |
| **3.** | The system will display the filtered results. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If no customer is present using the search field, “No Record Present” message will be displayed. | |

### Comparison of multiple cows’ milk production

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-020 | |
| **Purpose** | | The admin wants different options like the trend of milk production of an animal or want to compare milk production of each animal on the basis of selected date or the range of dates | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | The graph trends will be displayed. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin clicks on the ‘Compare Milk Productions’ button. | |
| **2.** | The admin checks the option: ‘All Cows’. | |
| **3.** | The admin selects a date. | |
| **4.** | The system will display a graph that shows the milk production of all cows on a selected date. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. | In step 3, the admin selects a range of dates. | |
| 2. | The system will display a graph of milk records for all cows over the selected range of dates | |
| 3. | In step 2, the admin unchecks the ‘All Cows’ option and selects some of the cows and follow step 3. | |
| **Step #** | **Exception Paths** | |
| 1. | The system will display a message saying ‘No record found’, if the admin selects a date that has no record. | |

### Customer login to his portal

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-021 | |
| **Purpose** | | The customer wants to login to his portal | |
| **Pre-conditions** | |  | |
| **Post-conditions** | | Customer will be logged in to his portal | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | Customer enters his login credentials | |
| **2.** | Customer clicks on login | |
| **3.** | If successful, will be redirected to his portal. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If wrong credentials are used, an error message will be displayed. | |

### Filtration on customer’s milk supplied data

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-022 | |
| **Purpose** | | The customer wants to filter the milk supplied data on his portal on the basis of date selected | |
| **Pre-conditions** | | Complete UC-021 | |
| **Post-conditions** | | Filtered data will be displayed. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The customer selects a date or range of dates. | |
| **2.** | The system will display the filtered results. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If not data on the date selected, it’ll show the “No Record Found” message. | |

### Invoice release to customers

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-023 | |
| **Purpose** | | The admin wants to release the invoice for the selected customers for all unpaid dues. | |
| **Pre-conditions** | | Complete UC-003 | |
| **Post-conditions** | | Invoice release will be sent to the customers on their portal. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The admin selects the customers from the list. | |
| **2.** | The admin selects the date range for which all dues are pending for each customer. | |
| **3.** | The admin clicks on “Send Invoice” | |
| **4.** | If successful, invoices will be visible on each customer’s portal. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If any error occurs, an error message will be displayed. | |

### Update status of Invoice from Customer’s portal

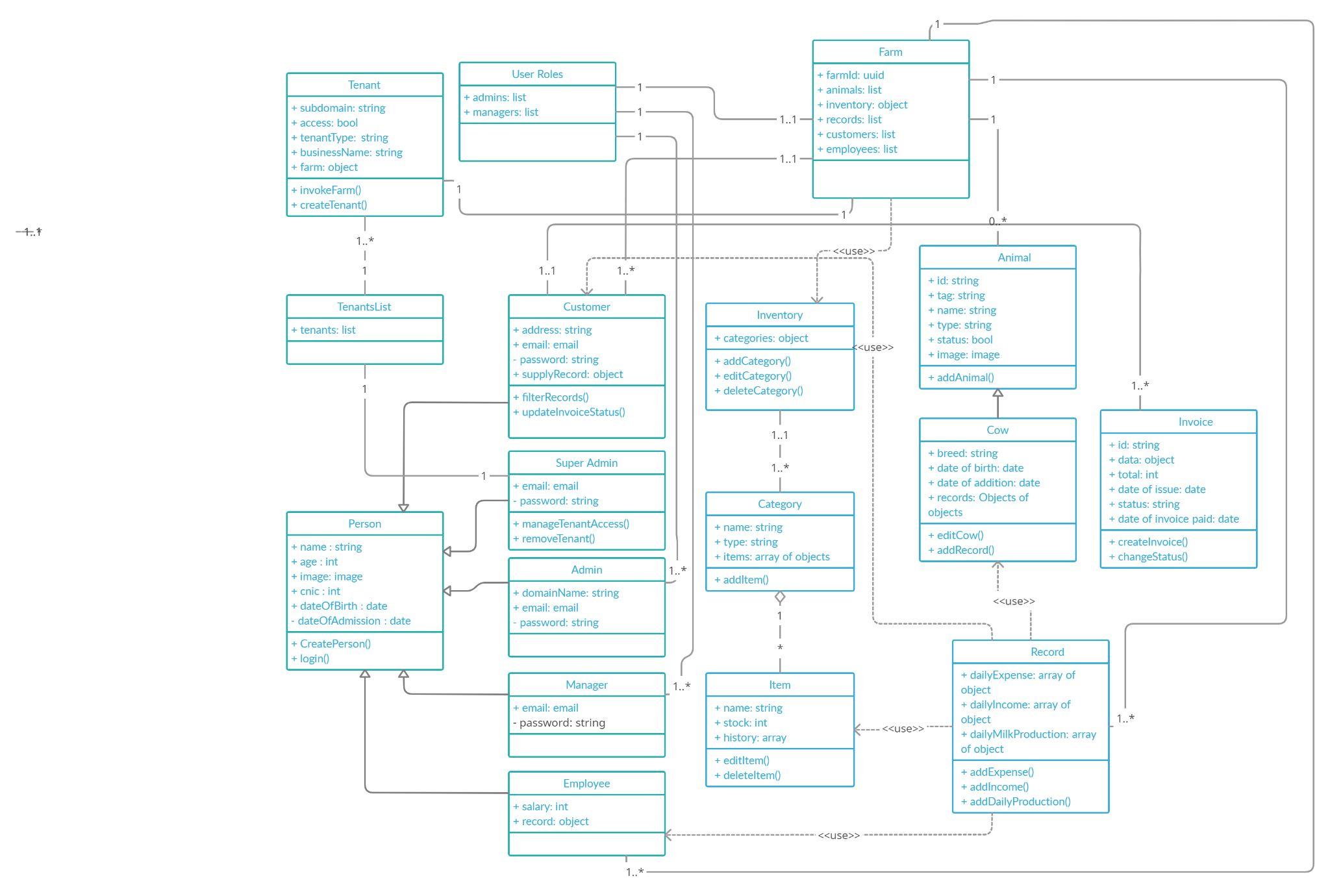
|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | | UC-024 | |
| **Purpose** | | The customer will review the invoice and change its status to paid if it is paid. | |
| **Pre-conditions** | | Complete UC-021/UC-003 | |
| **Post-conditions** | | Invoice status will be updated to paid and notification will be sent to the admins. | |
|  | | | |
| **Step #** | **Typical Course of Action** | |
| **1.** | The customer opens the invoice for review. | |
| **2.** | The customer changes its status to paid. | |
| **3.** | If successful, notification will be sent to admins. | |
|  | | | |
| **Step #** | **Alternate Courses of Action** | |
| 1. |  | |
| **Step #** | **Exception Paths** | |
| 1. | If any error occurs, an error message will be displayed. | |

# Class Diagram

* Missing classes for multi-tenant support
* Missing classes for farm, user roles,

## Diagram

* Tenant
* Tenant List
* User Roles
* Farm
* Super Admin



## Description

## Person It’ll be the person which contains basic attributes for a person to exist. Other persons will inherit this class for their common attributes.

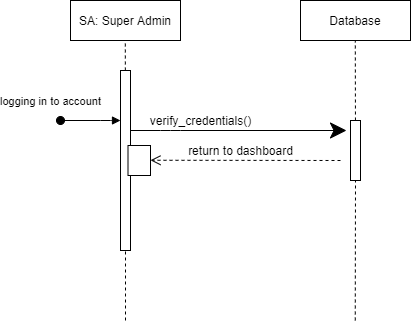
* + 1. **Super Admin**Super Admin is the owner of the product i.e. Dairy Management System. He’ll be able to manage access to multiple tenants from his dashboard and also, he can restrict them from using their business accounts.
    2. **Tenant**This class will have the information of a single tenant account and the associated single farm. Each tenant will have a single farm associated with it.
    3. **TenantList**This will contain a list of all tenants in our database and the super admin will have access to all of them and can restrict the access of any of them.
    4. **Farm**The farm will be associated with a tenant and it’ll consist of all other classes that are associated with it like animals, inventory and other things.
    5. **User Roles**This class will have a list of all admins and managers that are in the farm and can have multiple admins and managers in it.
    6. **Admin**He’ll be the tenant admin and will contain basic attributes inherited from the **Person Class**.
    7. **Manager**Manager class will inherit the person class for its existence.
    8. **Employee**  
       Employee class will inherit the person class and it also contains the salary attribute as well that is necessary for him to exist. It also has records associated with him that store how much money he has been paid and on which date.
    9. **Inventory**Inventory class will have multiple categories and it provides the function to add those new categories.
    10. **Category**  
        Category class will have the type which specifies that if it is a Volume based category or Weight based or the category whose items’ quantities cannot change with consumption.
    11. **Item**  
        This will store the details of items and provide methods like edit and delete and delete will only work if this item’s record is empty.
    12. **Animal**This class is the base class for animals and has basic attributes that are common among different animals.
    13. **Cow**  
        This class will have attributes that are for cows and inherit the Animal Class as well. This class will also have records objects as an attribute which stores different records like milk production record, medication record, vaccination record etc.
    14. **Record**  
        This is the most important class of all and it depends on various other classes for its existence. If we want to add daily expense, we have to refer to this class and that depends on the items class if we want to add the expense of the item present in the inventory i.e. Medicine.  
          
        Also, if we want to add daily income, it depends on the Customer Class because if we want to add the milk supplied to a customer then it depends on the existence of this Customer Class instance.  
          
        Likewise, for the milk production record of animals, we want the instance of Cow Class and hence dependency occurs.
    15. **Invoice**  
        Invoice class is for charging a customer on the unpaid supplies of milk and a single customer can have multiple invoices.

# Sequence Diagrams

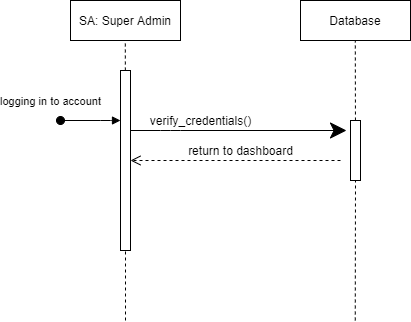
* See my comments that I have made for selected sequence diagrams. These comments are generally applicable to other diagrams as well.

## Login for Super Admin (Product Owner)

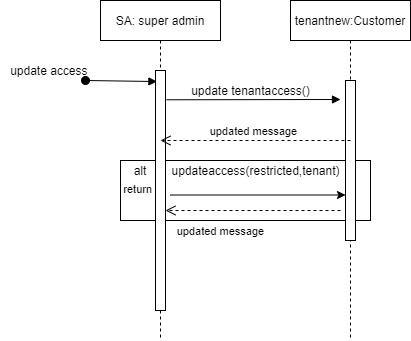
[What does this small box at the arrow head of “return to dashboard” show?]



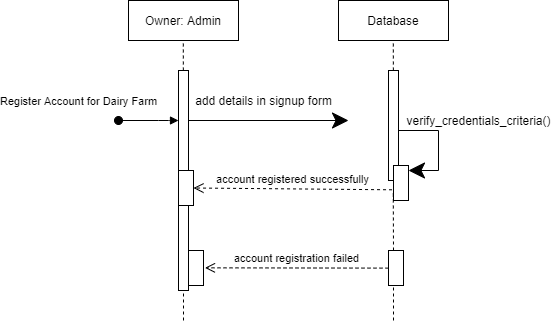
[updated diagram]



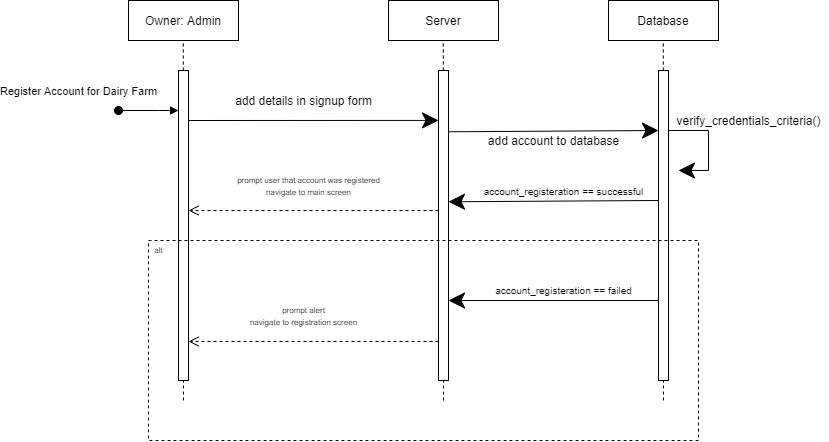
## Change access for tenants



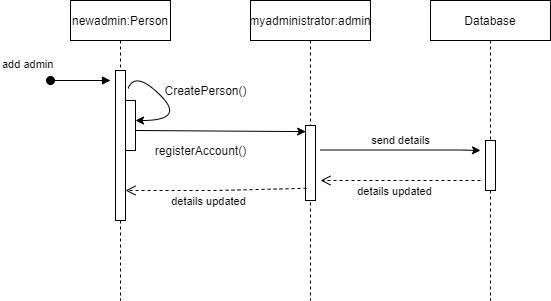
## Register Account for Dairy Farm



[updated diagram]



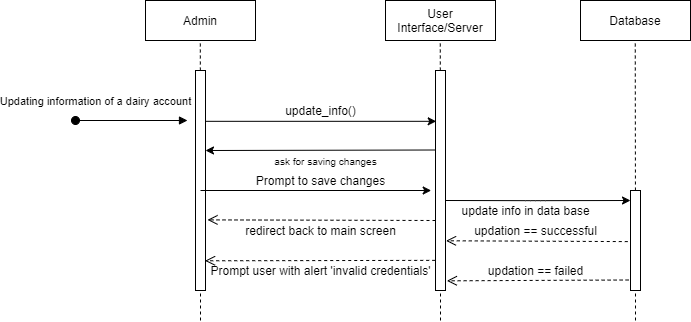
## Addition of Admin/Manager to Management Team



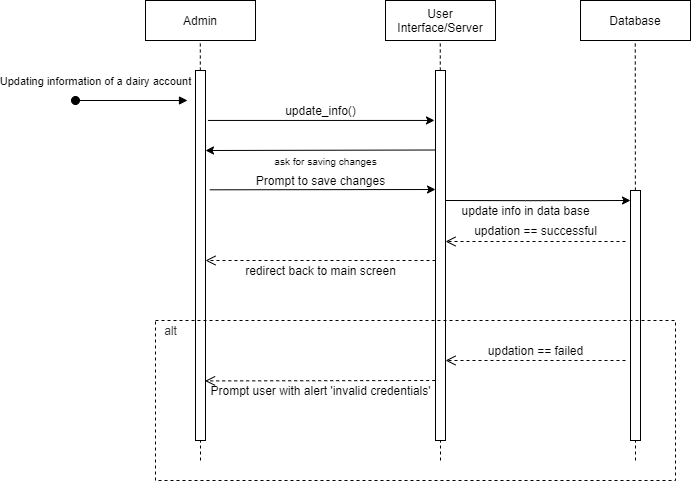
## Update Information of a Dairy Account

[There is a convention to show alternative paths in sequence diagrams, use that convention. Right now, it looks like both success and failure will happen in every execution.

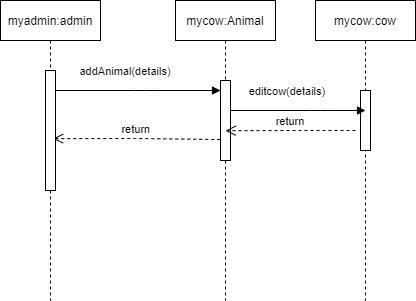
Moreover, user is prompted with “invalid credentials” first and then response of failure is received from the database. This is incorrect.]



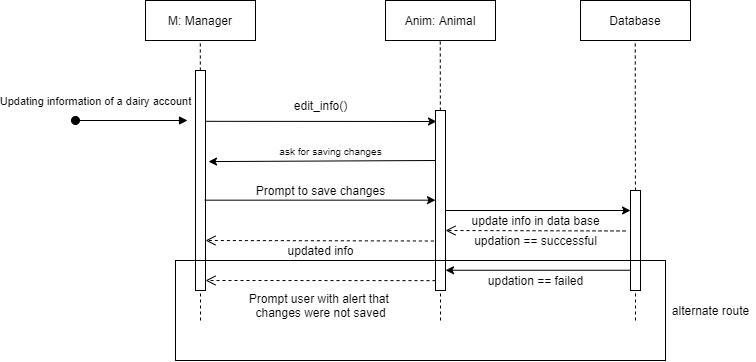
[updated diagram]



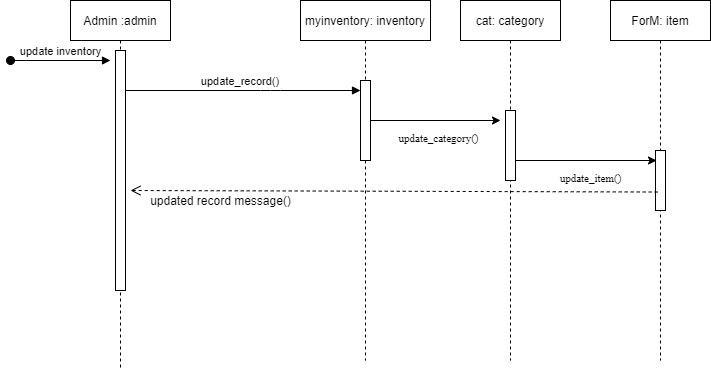
## Add new animal



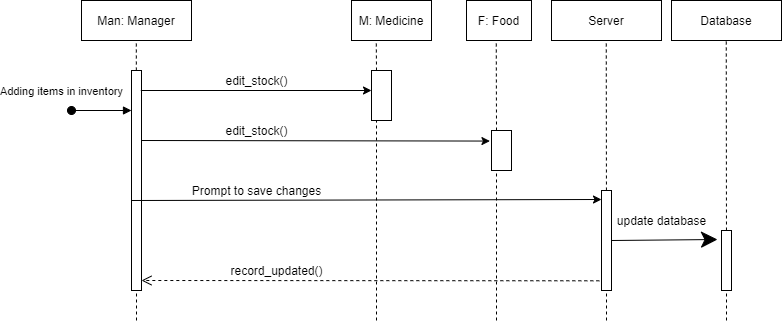
## Adding Animal Records



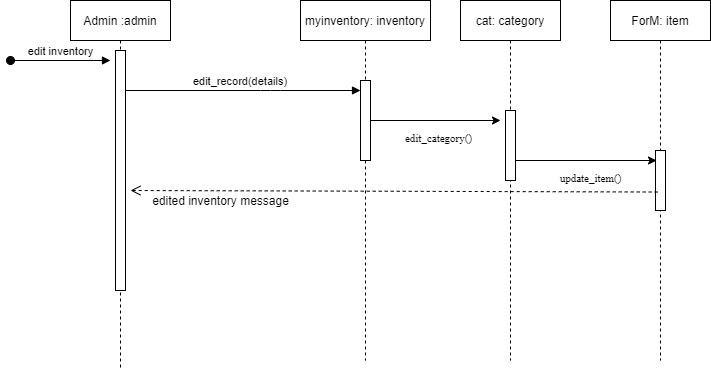
## Adding categories to the inventory



## Add items in the inventory

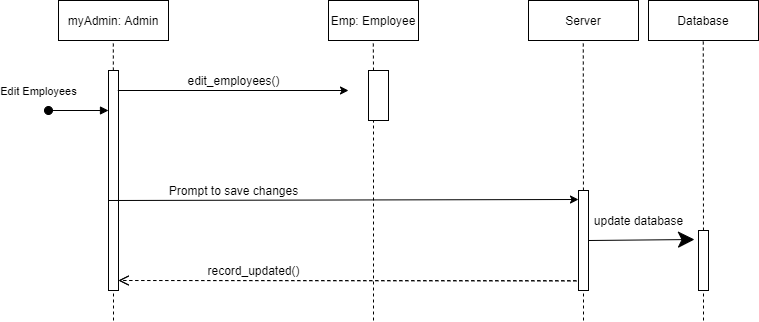


## Edit/Delete inventory item

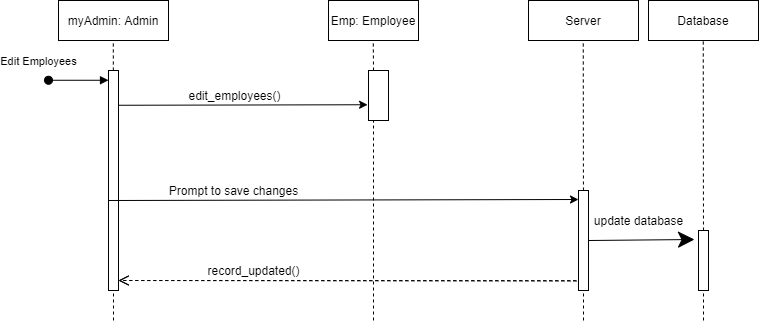


## Add/Edit Employees

[“Edit employees” call come later from the user but the call “edit\_employees()” has already taken place on the “emp” object. This is incorrect. ]



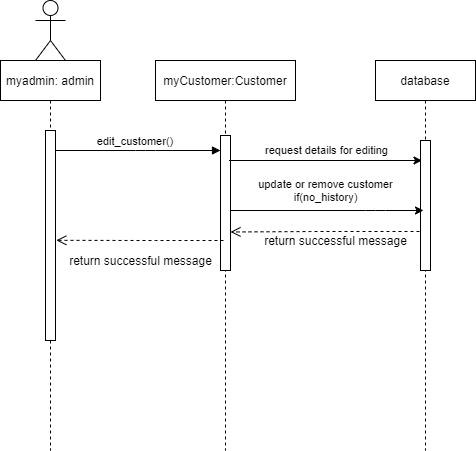
[updated diagram]



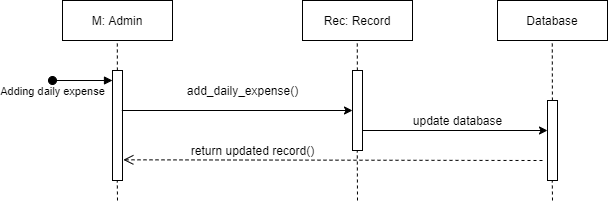
## Add/Edit/Remove Customer Data

[Is it “manager: admin” really an actor? It is not clear if it is representing an actor or a class in the following diagram. ]

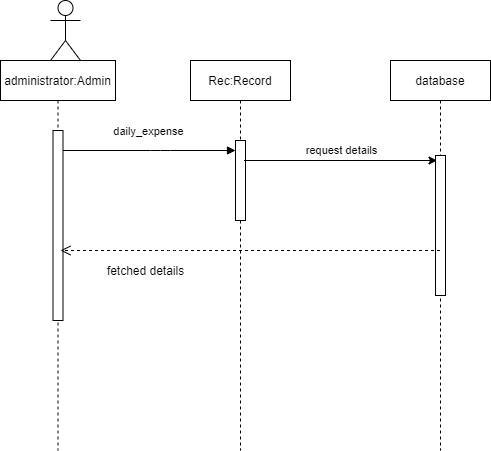
updated the object name of class admin for clarity



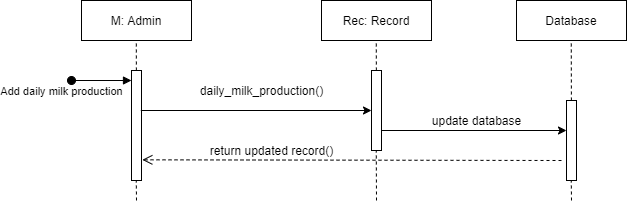
## Add Daily Expenses



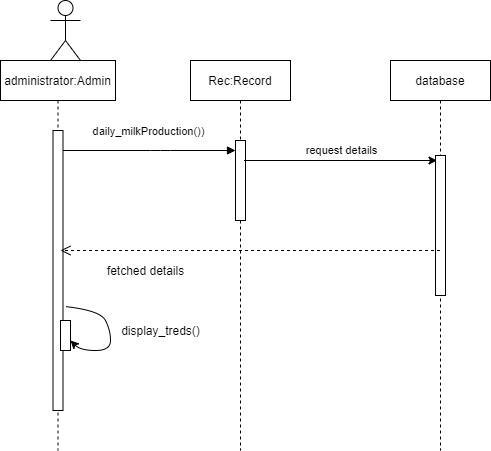
## Filter daily expense



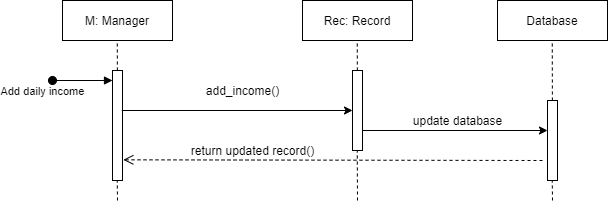
## Add/Edit Daily Milk Production



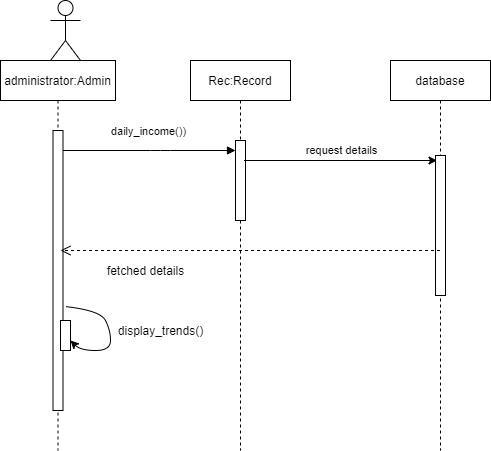
## Filter Daily milk production and display trends



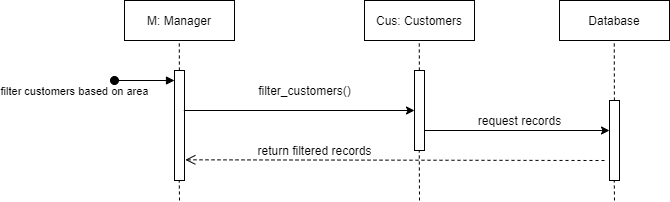
## Add/Edit Daily Income



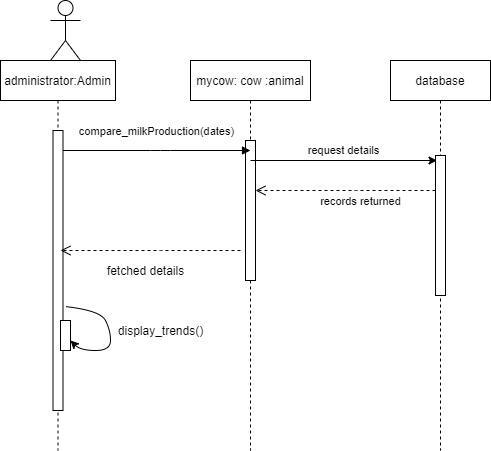
## Filter daily income and display trends



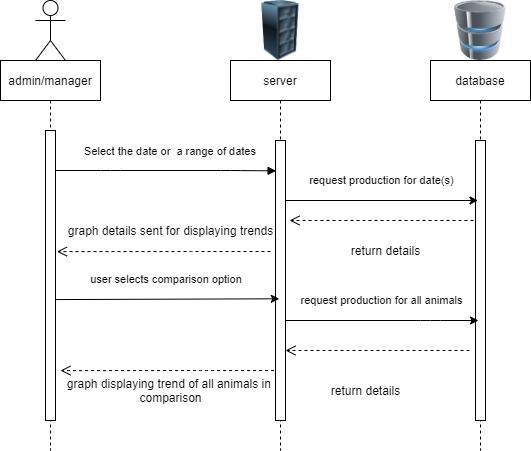
## Filters on Customer Data based on area



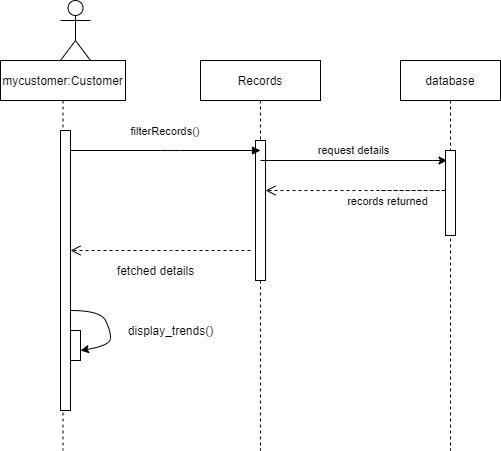
## Comparison of multiple cows’ milk production



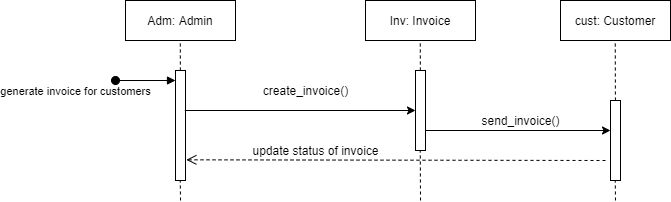
## Use case Name e.g., Transfer funds



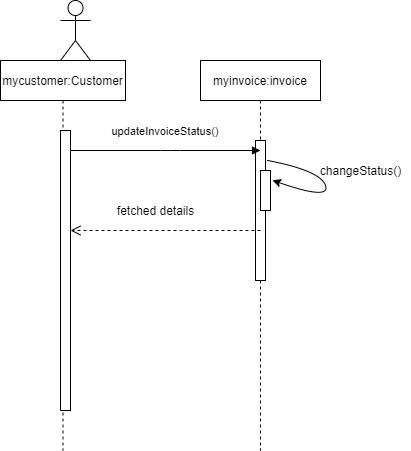
## Filtration on customer’s milk supplied data



## Invoice Release to Customers



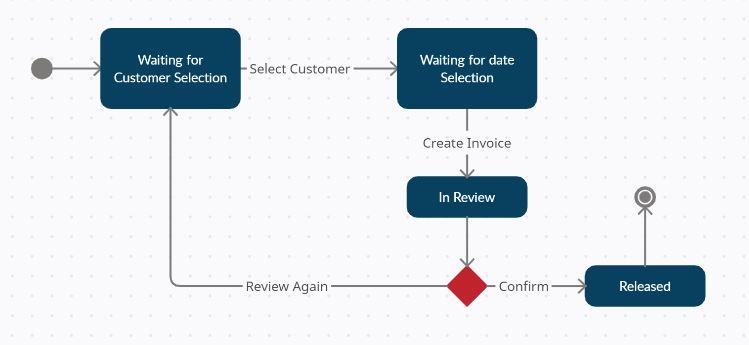
## Update status of Invoice from Customer’s portal



# State Diagrams

## State Diagram for Adding Daily Expenses

|  |  |
| --- | --- |
| State | Description |
| Waiting for customer selection | The system will be waiting for the customer selection to whom the invoice will need to be released. |
| Waiting for date selection | After selecting the customer, the system will be waiting for the date selection for which invoices should be released. |
| In Review | When the invoice is created, the invoice state will be set to “In Review”. |
| Released | When reviewed, the admin will confirm the invoice and state will be changed to “Released”. |

* + 1. State Diagram  
         
       

# Non-functional Requirements / Quality Attributes

|  |  |
| --- | --- |
| 1 | The system should not use extensive memory, not more than 1 GB memory at its time of execution. |
| 2 | The system should not fail more than 3 times for a given week and in case of failure must be available within 5 minutes. |
| 3 | Each page must load within 2 seconds after the user has made a request. |
| 4 | System should be secure against threats essentially the OWASP security standards which include input validation such as validation of data length or range, not disclose sensitive error information such as system details; Data protection by implementing least privilege which would restrict user to only functionality, data and information that they need to perform their tasks. |
| 5 | Privacy standards according to the GDPR be followed i.e. Not withholding personal data without the user’s consent, limitation of data storage that would mean no extra data is collected from the user. |
| 6 | The system should scale and be able to handle more traffic, 500 users at peak times that would be from 10am to 5pm. |
| 7 | For the Usability of the system the user must be able to navigate easily such as not going through a series of pages to make a payment slip request not more than 4 clicks for any request in the navigation. |
| 8 | The system should be expandable from the product owners’ side so that new features may be added such as a personalized data analysis graph. |
| 9 | The help documentation for the system should be released in the commonly understood language English and Urdu. |
| 10 | The system should cater for faulty inputs and must load or prompt the user for a correct input such as a price calculator accepting numbers and not letters. |
| 11 | The system should require authentications for all resources and information except that are intended to be public, least privilege principle will be followed and access granted would allow users to fulfill their purpose. As needed, access right can be given or withdrawn and in case of failure access controls should fail securely. |

# Who Did What?

|  |  |
| --- | --- |
| **Name of the Team Member** | **Tasks done** |
| Use Cases | Furqan Athar |
| State Diagrams | Furqan Athar |
| Class Diagram | Abdullah Saleem + Furqan Athar |
| Sequence Diagrams | Khawaja Junaid + Saad Qadeer |
| Non-Functional Requirements | Khawaja Junaid + Saad Qadeer + Abdullah Saleem |

# Review checklist

|  |  |
| --- | --- |
| **Section** **Title** | **Reviewer Name(s)** |
| Use Cases | Khawaja Junaid + Abdullah Saleem + Saad Qadeer |
| State Diagrams | Saad Qadeer |
| Sequence Diagrams | Furqan Athar |
| Class Diagrams | Khawaja Junaid |
| Non-Functional Requirements | Furqan Athar + Saad Qadeer + Khawaja Junaid + Abdullah Saleem |