Belize Cattle Tracker

Report 3: System Specifications & System Design

Group 2 Members:
Joanne Y.
Chimezirim A.
Rene S.
Miguel A.
Juan S.
Cameron T.

Submitted To:
CMPS4131
Mr. Manuel Medina
May 16, 2022

Contribution Breakdown

SIGNATURE BLOCK			
Statement	I did my share of the work, and I have a general understanding of the contents of the assignment.		
Team Member	**Contribution**	**Signature**	**Date**
Joanne Yong	Project Management. Did Problem Statement, Glossary, Preliminary Designs, Plan of Work, Hardware Requirements. Assisted with Use Cases, Traceability Matrix, Fully Dressed Description. Did Data Model and Persistent Data Storage, Class Diagrams, Data Types and Operation Signatures, User Interface Design and Implementation, and Project Management. Assisted with Interaction Diagrams, conceptual model, Algorithms, Concurrency, and Data Structures. Assisted with summary of changes. Assisted with OCL. Did History of Work.		05/16/2022
Miguel Avila	Assisted with Problem Statement. Did System Specifications. User Effort Estimation Identifying Subsystems		05/16/2022

	Did Data Structures and Design of Test.		
Rene Sanchez	Did Problem Statement. Did Casual Description Assisted with Use Case Diagrams Assisted with System Sequence Diagrams Did Subsystems Did Architectural Styles Did Conceptual Model, Interaction Diagrams, User Interface Design and Implementation, Assisted with updating glossary of terms. Did Design Patterns.		05/16/2022
Juan Carlos Sarabia	Did traceability matrix Fully Dressed Description Assisted with Identifying Sub System Did Connectors and Network Protocols. Assisted with Conceptual Model (Domain Models).		05/16/2022
Chimezirim Amagwula	Did System Specifications/Requirements. Assisted with User Effort Estimation. Assisted with Traceability Matrix. Assisted with Problem Statement. Did Global Control Flow. Assisted with Project Management. Did Conceptual Model, System Operation Contracts, Traceability Matrices, Interaction Diagrams, Concurrency.		05/16/2022

	Assisted with summary of changes. Assisted with OCL.		
Cameron Tillett	Did the Use Case Diagrams Did the System Sequence Diagrams Did the Architectural Styles Did the Mapping Subsystems to Hardware Algorithms		05/16/2022

Responsibility Matrix

Note: See point allocations in the table below.		Team Member Name					
		Joanne	Rene	Miguel	Chimeziri m	Juan	Cameron
Responsibility Level	Project management (*10 points*)	95%			5%		
	Sec.1: Customer Statement of Requirements (*9 points*)	80%	15%		5%		
	Sec.2: System Requirements (*6 points*)		20%	40%	40%		
	Sec.3: Functional Requirements Specification (*30 points*)	6%	29%		15%	25%	25%
	Sec.4: User Interface Specs (*15 points*)	50%		20%	30%		
	Sec.5: System Architecture (15 *points*)	15%	25%	15%	15%	15%	15%
	Sec.6: Domain Analysis *(25 points)*	27.5%	27.5%		27.5%	17.5%	

	Sec.7: Interaction Diagrams *(30 points)*	30%	35%		35%		
	Sec.8: Class Diagrams & Interface Specification *(15 points)*	80%			20%		
	Sec.9: Algorithms & Data Structures *(12 points)*	15%		32.5%	32.5%		20%
	Sec.10: User Interface Design & Implementation *(4 points)*	50%	50%				
	Sec.11: Test Case Design*(15 points)*	20%	30%	50%			
	Sec.12: History of Work (*19 points*)	100%					
	Sec.13: Summary of Changes	33.3%	33.3%		33.3%		

Responsibility Allocation Chart

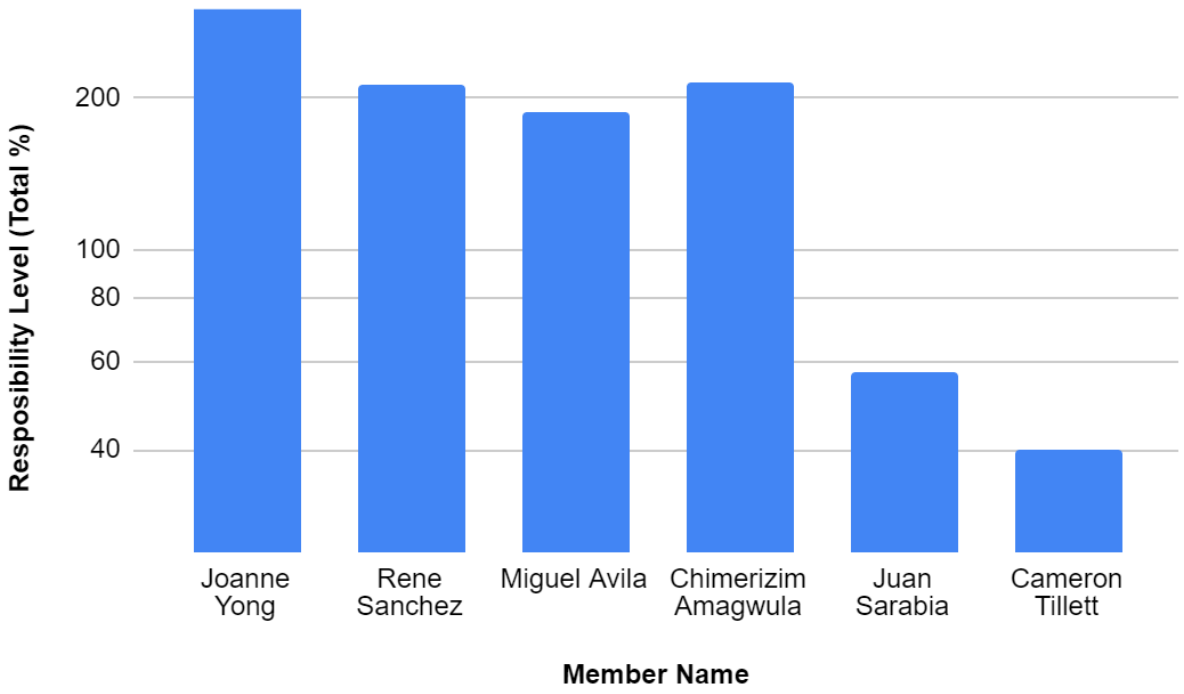


Table of Content

Customer Statement Requirements

Problem Statement

Today's corporate sector has undergone various transformations as a result of the ease of access to technology. In reality, most, if not all, businesses nowadays utilize technology in some form or another to manage and market their goods. As a result, we intend to obtain a new system that will allow our cattle information to be traced and stored, at each production stage, into one system where stakeholders such as BAHA, licensed/authorized farmer and slaughterhouses, and packaging, production and sales managers can easily enter, access, manage and use cattle information. Apart from that, we also intend on providing our customers with a platform that will allow them to view information about the specific meat produce they plan to purchase at a grocery store using modern technologies. This would serve as a quality control for meat products, and also as a rationale for why certain produce cost more than others, as creating high-quality produce is our main goal.

Meat processing companies frequently place a greater emphasis on the manufacturing and distributing aspects of their businesses, failing to recognize that certain customers want to know about the quality and grading of the goods they are purchasing. When purchasing meat produce at a grocery store, buyers frequently inquire how fresh the produce is or even why some meat products cost more than others. As a result, we wish to distinguish ourselves apart from the rest of the meat processing enterprises in the country (Belize) by giving our customers an insight of the grade and quality of the product they're buying, ensuring convenience and consistency.

To begin, most grocery stores rely on "word of mouth" to respond to a concerned customer who wants to know how long the product has been in the store before purchasing the certain meat product. However, this is deemed very inconvenient and inefficient for us meat producers since it impacts how our customers perceive the quality of our goods and because there is no information provided to the customer based on the produce's date of slaughter/processing, they wouldn't know much about the freshness of the produce if the store manager is not truthful about it.

Secondly, another matter is that some clients are concerned about the health of the cattle from whom the meat was produced. Consumers are typically concerned about the cattle's health throughout its life, worrying that they may eat meat from a cattle that was not in good health and had to be treated with heavy doses of antibiotics, which could have long-term negative side effects on the consumer's health. Unfortunately, no information is offered to the customer depending on the specific meat product that they are purchasing to help them decide whether or not to buy the product based on the information provided on the packaging. As a result, some customers are likely to be cautious and skeptical of the meat products produced by our company.

Following that, an additional matter is cattle information management. When purchasing cattle from licensed and recognized farms in Belize, there is an ear tagging and registration system in place for the cattles that have recently been born on the farm. These ear tags are unique and individualized to a certain calf, and they feature a number that denotes their unique cattle number provided to them by BAHA when the cattle is registered by the farmer. It can be difficult for BAHA to keep track of the number of cattles registered from different registered farms as well as

their biographic information because there is a lot of information to go through manually by papers and as a result, we'd want to provide a more accessible and structured way for all of our stakeholders to collect and maintain biographical/processing data for the numerous cattle they've acquired from the various farms.

To address the problem of obtaining and keeping biographical data on cattle purchased from farms, the system should allow farmers to enter the animal's data starting at birth. The system must allow the farmer to disclose critical information for each animal sold to our organization, such as the animal's date of birth, breed, ear tag number and other important information. As previously stated, the cattle obtained from various farms will be identified by ear tags with unique codes. The farmers and our managers will be able to identify a specific cattle on our system using this code, which will lead to all the biographical data from the animal's birth stage that the farmers would have entered. After the farmers' biographical data has been collected, each animal's information will be forwarded to the slaughtering division, where the animal's slaughter details will be added to its current record.

The slaughterhouse managers are responsible for killing and sanitizing the meat that they get from the farmers. This process should be carefully documented by the employees as they will be responsible for keeping track of each animal that is slaughtered. This is critical to the operation of this system because customers might want their meat to be slaughtered a certain way. Managers are able to keep track of each cattle brought into the warehouse and when slaughtered, there should be an indication that the animal has been slaughtered and is now ready to be transferred to the packaging/processing managers. Having a label to keep track of each animal

that has been slaughtered makes the process much easier and helps maintain clarity within the managers of the different stages of production.

Another problem is linkage between processing and sales stage information. When sending meat products to various grocery stores, it can be quite hectic to gather information about the business, such as its location, name, inventory demands, and other trade specifics. When data has to be saved and written in manually by hand each time there is a sale, it becomes extremely difficult to manage and keep track of preexisting purchasers and new customers; it also causes unnecessary redundancy. Because of this, we'd want to have a digitalized way for our sales managers to keep track of trade specifics of different purchasers in order to capture and save the information in case they're returning customers.

From the issues and scenarios stated, we are now requesting for your company to create and execute a system that will handle the problems that have been addressed and presented. We would like your company to begin the process of changing our manual cattle management system to an automated cattle traceability system. This system should have several user-friendly interfaces: one for our farmers or BAHA, another for our slaughterhouse managers, and a third for our packaging/processing managers, and a fourth for our sales managers, all of which should be accessible by BAHA.

This central system should enable BAHA or a registered/authorized farmer for a meat production firm (Running W) to construct a cattle profile for each individual cattle that the farmer has registered with BAHA. The farmer can either input the information themself or leave the

registration documents with BAHA and have BAHA do it. Based on the hair sample submitted by the farmer, BAHA will send breed information to the farmer. Aside from DNA information, the farmer/BAHA would also add birth information to the system such as the cattle ID (given by BAHA when the cattle is registered), birth date, gender, health concerns, weight, reproductive status, owner (farmer), antibiotic types, rearing type, and so on. The farmer/BAHA should also be able to search for a specific cattle profile by its cattle ID, and the farmer/BAHA can make modifications to a specific cattle depending on their weight, feed, health changes, and so on. These changes can be updated throughout the cow's development. When a cattle has been sold to the slaughterhouse or has passed away, these statuses should be present on the profile, and there should also be a cease (for the farmer) for that cattle profile so that no misleading information is added by the farmer after those events. Also, when the status of the cattle is marked as deceased, the cattle profile should not be deleted.

When the farmer sells the cattle to the slaughterhouse, they provide the slaughterhouse managers the cattle's ID so that they may add slaughter information to the already produced birth information (from the farmer). This information should contain the date of slaughter, the procedures used, and the location of the farmer, among other things. The information from the previous step should not be editable by the slaughterhouse management. Employees at the slaughterhouse will be able to categorize if the meat is of a higher quality than the others and why that is so at this point, which is where we decide the quality and grade of the meat (lean percentage and fat percentage). The slaughterhouse will divide the cattle into chunks after it is slain, and attach a traceability number (same cattle ID) on the different parts that came from a specified cow. When it is transported to the packaging people/meat manufacturing firm, they will

chop it into its respective portions for the various meat products. When the slaughterhouse has shipped the meat chunks to the meat manufacturing firm, the cattle profile's status must include a slaughtered status, as well as a cease (for the slaughterhouse manager) for that cattle profile, so that the slaughter manager does not add any misleading information after those events.

Additionally, after the meat chunks are transported from the slaughterhouse to the meat manufacturing company, our production managers will supervise the procurement and separation of the various meat parts so that the packaging managers can pack the various meat produced into their designated packaging type. Packaging managers will ensure that the meat products are appropriately wrapped and that all packaged meat information is captured and preserved on the system after receiving the various meat pieces from the production stage. This meat information will be added to the pre-existing slaughter and birth information, but the production or packaging manager should not be allowed to change the preexisting information. The traceability number (cattle ID) issued by the slaughterhouse, the kind of meat product, the weight of the product, the packaging and expiration date, the batch number, and the store location/name should all be included into the system at this stage. After this stage is completed, the packaging manager should collaborate with the sales managers to ensure that the meat produce inventory is delivered properly. Once again, when our meat processing company has distributed the products to the various stores, the cattle profile's status must include a packaged and delivered status, as well as a cease (for the packaging and production managers) for that cattle profile, so that these managers do not add any misleading information after the meat produce has been sold.

Our sales managers maintain and keep track of all trade information for the company's buyers in terms of sales. The use of this system should alleviate this challenge by allowing them to record

information on purchasers and their trade information, as well as save it so that it may be utilized again if they are returning buyers.

Apart from handling the data from the various phases, we'd want to include a package printing system that allows us to create product labels that are all unique to the cattle they originated from since they carry the traceability number (cattle ID). This would make it easier for BAHA to keep track of the registered cattle that have been tagged, purchased, slaughtered, and processed into meat products. This system should also develop new packaging labels for our various beef products, each of which would have a generated QR code that purchasers could scan to learn more about the cow from which the product was derived.

Our customers should be able to learn more about the specific meat product and its quality with this system in place. From the packaging of the specified meat product, customers should be able to obtain information such as the cattle's date of birth, breed, weight, feed type, health history, origin (farm), grade, and price of the meat. The system should provide data on the whole meat manufacturing process. The type of treatment, the type of cut, the processing date, the expiry date, the packing type, shipping and delivery data, and so on must all be included in the details. Based on the information supplied by the system, our consumers should be able to make an informed decision about which meat product to purchase. For this, we would like to include scanning features in which customers can use to access the biographical information of the specified meat product.

Lastly, because meat products raise health concerns and necessitate some form of monitoring and quality control in order for our company to follow guidelines set forth by institutions such as the

BAHA to produce goods that meet industry standards, this system should aid BAHA in maintaining cattle trace information (from birth to slaughter) in order to enforce the maintenance and abidance of those guidelines.

Glossary of Terms

Terms	Definition
Corporate sector	The portion of a country's economy that involves private businesses.
Transformations	A thorough or dramatic change in form or appearance.
Technology	The application of scientific knowledge for practical purposes, especially in industry.
Business	The practice of making one's living by engaging in commerce/trade.
Manage	Maintain control or influence over.
Utilize	Make practical and effective use of.
Market	Advertise or promote.

Production Manager	Coordinate and control manufacturing processes so that products are delivered on time and within budget.
Packaging Manager	Manages and oversees the packaging staff on designated shifts to achieve operational objectives and implements quality control checks to ensure the packaging meets environment, health, and safety regulations.
Cattle	Large ruminant animals with horns and cloven hoofs, domesticated for meat or milk, or as beasts of burden; cows.
Information	Facts provided or learned about something or someone.
Platform	A standard for the hardware of a computer system, determining what kinds of software it can run.

Meat Produce	Meat products from an animal such as pig, cow, etc.
Quality Control	A system of maintaining standards in manufactured products by testing a sample of the output against the specification.
Rationale	A set of reasons or a logical basis for a course of action or a particular belief.
Manufacturing	The making of articles on a large scale using machinery; industrial production.
Purchasing	Acquire (something) by paying for it; buy.
Convenience	The state of being able to proceed with something with little effort or difficulty.
Enterprises	A project or undertaking, typically one that is difficult or requires effort.
Word of Mouth	Oral communication; talking.
Consistency	Conformity in the application of something, typically that which is necessary for the sake of logic, accuracy, or fairness.

Customer	A person or organization that buys goods or services from a store or business.
Inconvenient	Causing trouble, difficulties, or discomfort.
Inefficient	Not achieving maximum productivity; wasting or failing to make the best use of time or resources.
Quality of goods	Decides the quality of people required for its sales, marketing and brand promotion. Informed buyers can correlate and compute the reliable equation between the product's characteristics & property and its company's person's character & personality.
Grade of goods	Grading is the procedure of categorization of products into dissimilar groups, on the source of some of its significant characteristics such as quality, size, etc. Products of diverse qualities should be divided into groups or lots and related quality products are put into a grade.
Store Manager	The most senior member of the staff in a

	store, who is responsible for the overall running of the store.
Dose	A quantity of a medicine or drug taken or recommended to be taken at a particular time.
Antibiotics	A medicine (such as penicillin or its derivatives) that inhibits the growth of or destroys microorganisms.
Packaging	Materials used to wrap or protect goods.
Cautious	Careful to avoid potential problems or dangers.
Skeptical	Not easily convinced; having doubts or reservations.
Licensed	Having an official license.
Recognized	Acknowledge the existence, validity, or legality of.
Ear tagging	Ear tags are needed for animal identification. They make it possible for us to identify and keep accurate records about each calf, heifer, steer, cow and bull.

Registration	The action or process of registering or of being registered.
Individualize	Tailor (something) to suit a particular individual.
Maintain	Cause or enable (a condition or state of affairs) to continue.
Manual	Relating to or done with the hands.
Automated system	System of operating or controlling a process by highly automatic means, as by electronic devices, reducing human intervention to a minimum. a mechanical device, operated electronically, that functions automatically, without continuous input from an operator.
Traceability	The quality of having an origin or course of development that may be found or followed.
Interface	A device or program enabling a user to communicate with a computer.

User friendly	(Of a machine or system) easy to use or understand.
Sequential	Forming or following in a logical order or sequence.
Biographic information	Biographical information is that which pertains to a person's life. For example: date of birth. date of death.
Exporting	Send (goods or services) to another country for sale.
Processing date	Date/day in which the meat product is taken into the factory and made into different meat products.
QR code	A machine-readable code consisting of an array of black and white squares, typically used for storing URLs or other information for reading by the camera on a smartphone.
Bar code	A machine-readable code in the form of numbers and a pattern of parallel lines of varying widths, printed on and identifying a

	product.
Industry Standards	Voluntary agreements that establish requirements for products, practices, or operations in a given field.
Breed	A stock of animals or plants within a species having a distinctive appearance and typically having been developed by deliberate selection.
Necessitate	Make (something) necessary as a result or consequence.
Abide	Accept or act in accordance with (a rule, decision, or recommendation).
Guidelines	A general rule, principle, or piece of advice.
BAHA	The Belize Agricultural Health Authority is the Competent Authority for agricultural health and food safety in Belize.
Alleviate	To relieve or lessen the load.
Inventory	A complete list of items such as property,

	goods in stock, or the contents of a building.
Enquiry	Act of asking for information.
Database	A database is a collection of data that is organized, which is also called structured data. It can be accessed or stored in a computer system.
System	A set of things working together as parts of a mechanism or an interconnecting network; a complex whole.
Database Query	A request to access data from a database to manipulate it or retrieve it.
Render	The process of generating an image from a model by means of computer software
Requirement	A thing that is compulsory; a necessary condition.
Verify	Make sure or demonstrate that (something) is true, accurate, or justified.
Confirmation	The action of confirming something or the state of being confirmed.

Database Connection	A database connection is a facility in computer science that allows client software to talk to database server software, whether on the same machine or not. A connection is required to send commands and receive answers, usually in the form of a result set.
Generate	To produce or create.
QR Code	A machine-readable code consisting of an array of black and white squares, typically used for storing URLs or other information for reading by the camera on a smartphone.
Domain Model	Is a visual representation of conceptual classes or real-situation objects in a domain.
Boundary	A limit of something abstract, especially a subject or sphere of activity.
Entity	A thing with distinct and independent existence.
Use Case	Term that describes how a user uses a system to accomplish a particular goal.

Priority	A thing that is regarded as more important than others
Filter	A filter in a database hides (filters out) unwanted records displaying only the records you want to see.
Cross Reference	A reference to another text or part of a text, typically given in order to elaborate on a point.
Exception	A person or thing that is excluded from a general statement or does not follow a rule.
Pre-Condition	A condition that must be fulfilled before other things can happen or be done.
Post-Condition	Post Condition is a statement or set of statements describing the outcome of an action if true when the operation has completed its task.
Data Model	A data model organizes data elements and standardizes how the data elements relate to one another.

Schema	Is a collection of metadata that describes the relationships between objects and information in a database.
Sequence Diagram	Represents the flow of messages in the system and is also termed as an event diagram.
Interaction Diagram	The purpose of interaction diagrams is to visualize the interactive behavior of the system.
Class Diagram	A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.
Interface Specification	Interface specifications provide the standardized mechanism in which subsystems can effectively communicate with each other and enable them to operate as independent modules that, when collectively implemented, support the entire CM technical reference

	model.
Data Types	A particular kind of data item, as defined by the values it can take, the programming language used, or the operations that can be performed on it.
Validate	Check or prove the validity or accuracy of.
Controller	A controller, in a computing context, is a hardware device or a software program that manages or directs the flow of data between two entities.
Algorithm	A process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.
Data Structures	Data structure is a storage that is used to store and organize data. It is a way of arranging data on a computer so that it can be accessed and updated efficiently.
Array	An array is a collection of items stored at contiguous memory locations. The idea is to

	store multiple items of the same type together.
Concurrency	The fact of two or more events or circumstances happening or existing at the same time.
Multi-threading	A technique by which a single set of code can be used by several processors at different stages of execution.
Synchronize	Cause to occur or operate at the same time or rate.
Alphanumeric	Consisting of or using both letters and numerals.
Mock-up	A model or replica of a machine or structure, used for instructional or experimental purposes.

System Requirements

Enumerated Functional Requirements

ID	PW	REQ-x
Req1	3	The system will contain a list of all the approved farms and slaughterhouses in Belize.
Req2	3	The system will contain all relevant data of the animal (Sickness, weight) when registered
Req3	1	The system allows for basic user setting manipulations such as changing password, username etc.
Req4	2	The system allows for Slaughterhouse and Production Managers to access and update information about specific meat products.
Req5	3	The system records all stages of an animal and steps within the entire meat production process.
Req6	4	The system allows the user/customer to view a report of all information regarding the grade, process date, expiring date, etc about the meat by scanning the QR Code (quality of the meat).

Req7	3	The system allows packaging managers to insert product information into the system based on the cut of meat.
Req8	2	System provides BAHA the ability to generate an array of reports as desired.
Req9	2	The system provides search features to find desired information on available meat products.
Req10	4	Depending on the role allotted, the system allows for various role specific operations.
Req 11	4	The system allows for the creation of new cattle profiles once they have been registered. (Farmer)
Req 12	4	The system allows each role to search for the cattle through the ID number and view the information about it.
Req 13	2	The system allows packaging managers to print the QR code.
Req14	2	The system allows farmers to update livestock information (live status, fertility)
Req 15	3	The system allows BAHA to filter information based on what type of reports that they would like to generate

Most Important = 4 | Least Important = 1

Enumerated Nonfunctional Requirements

ID	PW	REQ-x
NONREQ1	1	The system must have a friendly user interface for users (Usability)
NONREQ2	3	The system must be efficient and provide instantaneous results (Usability)
NONREQ3	1	The system will only allow the customers to view the desired information (security)
NONREQ4	3	The system must produce accurate information for the cattle/meat that is being purchased (Reliability)
NONREQ5	2	The system must log out inactive users (packaging managers) after a 10 minutes of being inactive.
NONREQ6	2	There must be sufficient documentation on how the device should be used. (Usability)
NONREQ7	3	Role specific operations must be constantly updated and available for use. (Reliability)
NONREQ8	2	Reports must be generated in less than 30 seconds after QR code has been scanned. (Performance)
NONREQ9	1	Websites must be updated continually to avoid any crashes or

		inconsistency. (Reliability)
NONREQ10	2	The system must always be live and the different features must be available for each role to be able to perform their jobs (Availability)
NONREQ11	3	The system allows for different roles in the login screen. (Usability)
NONREQ12	2	The system allows for (BAHA) to be created. This admin role should be able to view all the information being added by the other roles and have full access to all of the system's features. (Reliability)
NONREQ13	1	The system shows each stage that the cattle went through up to when it was slaughtered. (Reliability)
NONREQ14	3	The system terminates role access for a livestock when the livestock has been moved to the next stage. (Reliability)

Most Important = 3 | Least Important = 1

On Screen Appearance Requirements

ID	PW	REQ-x
AR1	1	The system must not be complicated for users to learn, there must be a simple yet effective user interface.

AR2	1	The system must be enhanced for touchscreen usage.
AR3	2	The system must remain on a description of the meats being selected. Screen. Must return to the home screen afterwards.
AR4	2	The system will be available on all major web browsers.
AR5	1	Each stage must be clearly labeled when displayed on the screen.

Most Important = 3 | Least Important = 1

Functional Requirements Specification

Stakeholders

- BAHA
- Farmer
- Slaughterhouse - (Running W)
- Packaging/Processing - (Running W)
- Customer

Actors and Goals

Actor	Roles	Types	Goals
System	Responsible for: 1.) Allowing for profile creation for Farmers and the various Management staffs (the slaughterhouse employees, Packaging managers, Business Managers) 2.) Provide a user interface that allows	Initiating	1.) Provide customers with a way to accurately verify the quality of meat before purchasing 2.) Provide a user interface that is easy to use 3.) Provide communication between the managers

	customers to easily navigate and access features as they please. 3.) Generate a comprehensive description of any sickness or illnesses that the animal had before being slaughtered. 4.) Produce accurate information for the cattle/meat that is being purchased. 5.) Allow for role specific operations such as the updating information and deleting information.		and Farmers. 4.) Allow for the easy update of cattle information by the different roles 5.) Provides an opportunity for Spanish speakers to partake in its features. 6.) Provides a frictionless experience to users.

	6.) Provide Warning if the cattle was compromised during packaging or slaughtering. 7.) Updates in real time as information is being changed.		
Customer	Responsible for: 1.) Viewing generated reports about meat that is being purchased 2.) Navigating Interface to access available features. 3.) Initiate features by taking actions as indicated.	(participating) Initiating	1.) Make educated decisions based on generated reports 2.) Easily utilize the user interface to access different features 3.) Filter the information that is being displayed.

Farmers	Responsible for: 1.) Creating their profile for their role. 2.) Creating a profile for registered animals. 2.) Ensuring that each cattle is being properly tagged and that each animal has a unique ID number 3.) Inserting accurate information about each cattle that is tagged into the system 4.) Stay in contact with the managers through a feature that allows communication between managers.	Initiating (participating)	1.) Update information effortlessly 2.) Stay in touch with managers for more consistent and accurate updates 3.) Manage cattle tag numbers more efficiently

	5.) Get notified when there is an urgent need for cattle meat.		
BAHA	Responsible for: 1.) Can be allowed to insert information about meat products alongside farmers. 2.) Generating a variety of reports based on different specifications on the meat product. For example, they are allowed to generate a report to see all the cows that are of a certain breed. 3.) View all the information that is	Initiating	1.) Stay updated about all the cattles that are being slaughtered and sold 2.) Make decisions about cattle distribution

	being added by the farmers, slaughterhouses and the packagers. They should be allowed to view it, however they should not be allowed to change or modify it.	(participating)	
Managers (Slaughterhouse) (Packaging)	Responsible for: 1.) Creating profile 2.) Accessing information about cattle before and after slaughter 3.) Update information as the cattle gets passed down through the managers.	Initiating	1.) Update information effortlessly 2.) Stay in touch with farmers for more consistent and accurate updates 3.) Manage cattle information more efficiently

	4.) Get notified when there is an urgent need for cattle meat 5.) Communicate with other managers through the system.	(participating) (participating)	

Casual Description

Name	Description	Requirements Covered
UC-1 Create Cattle Profile	Farmers can make cattle profiles and add the given cattle ID numbers containing its specific cattle birth information such as the breed, gender, etc.	Req 11, Req 10

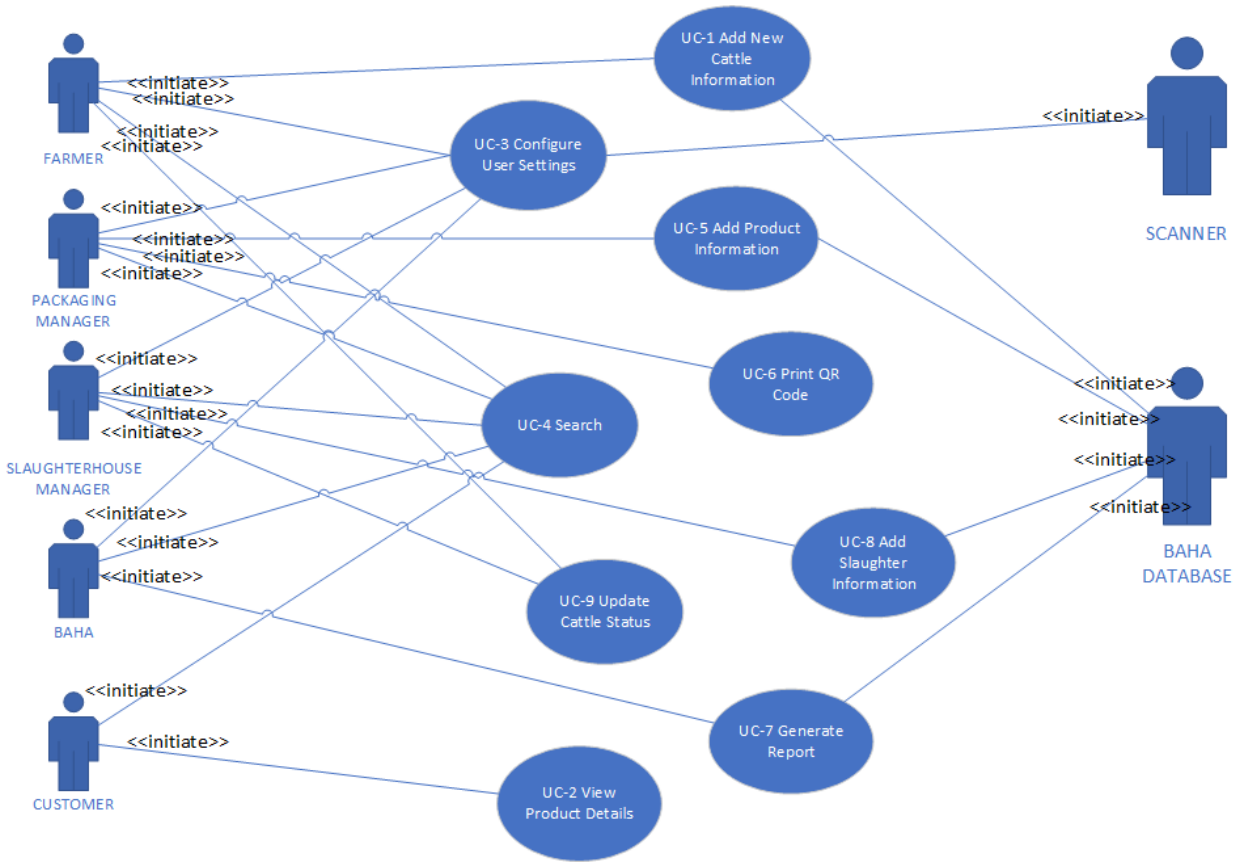
Name	Description	Requirements Covered
UC-2 View Product Details	Customers will scan QR code on specific meat products. After the QR code is scanned, the customer gets redirected to a webpage displaying the cattle information. The customer can then view all available details of a specific meat product being purchased. The information displayed will consist of breed, health history, feed type, meat quality, processing details, farms, etc.	Req6, Req2, Req 10
UC-3 Configure User Settings	Users except for customers must have access to basic user settings and the ability to manipulate these settings by changing password, etc.	Req3
UC-4 Search	When the ID number for a specific cattle is entered into the system, all the appropriate	Req9, Req 12

Name	Description	Requirements Covered
	information for that animal should be available for viewing.	
UC-5 Add Product Information	Packaging managers can log in to their profile. Which provides them with features to access pre existing meat information and to insert additional information to the specific cattle profiles. Packaging managers can use the cattle ID to keep track and add to the right cattle profile. They can also generate the QR code containing the cattle information.	Req4, Req 10
UC-6 Print QR Code	Packaging managers should have the ability to generate and print QR codes containing the cattle information for that specific product, which will be available on the product packaging for customers to scan and view.	Req13

Name	Description	Requirements Covered
UC-7 Generate Report	BAHA can log in to the system and have access to all the features and data on the system since it is designed for them to track and trace cattle from birth to death. The system should also allow for BAHA to freely generate reports based on specified filters such as by breed, date of birth, gender, etc.	Req8, Req 10,Req 15
UC-8 Add Slaughter Information	The slaughterhouse manager should have access to the preexisting cattle information and should only be able to add on new information to the cattle profile such as the slaughter date, location and method. They should also be able to update the cattle life status (eg: death).	Req 4, Req 10

Name	Description	Requirements Covered
UC-9 Update Cattle Status	The farmers should be to update the weight changes, feeding type, health issues contracted, etc. while the cattle is still on their farm. If the cattle passes away under the watch of the farmer, the farmer should also be able to update the live status of the cattle to (eg: dead).When the cattle is slaughtered at the slaughterhouse, the slaughterhouse manager should also be able to update the live status of the cattle to (eg: dead).	Req10, Req14

Use Case Diagram



Traceability Matrix

The traceability matrix below maps out the system requirements to use cases. The purpose of the matrix is to ensure the system requirements are met and all use cases have a purpose.

Key Words: UC - Use Cases, PW - Priority, REQ - Functional Requirements, NONREQ - Non-functional requirements, and AR - On-Screen Requirements.

REQ	PW	UC1	UC2	UC3	UC4	UC5	UC6	UC7	UC8	UC9
REQ1	**3**									
REQ2	**3**		**x**							
REQ3	**1**			**x**						
REQ4	**2**					**x**			**x**	
REQ5	**3**									
REQ6	**4**		**x**							
REQ7	**3**									
REQ8	**3**							**x**		
REQ9	**2**				**x**					
REQ10	**4**	**x**	**x**			**x**		**x**	**x**	**x**
REQ11	**4**	**x**								
REQ12	**3**				**x**					
REQ13	**2**						**x**			
REQ14	**3**	**3**								**x**
REQ15	**3**								**x**	

Max PW		**4**	**4**	**1**	**3**	**4**	**2**	**4**	**4**	**3**
Total PW		**8**	**10**	**1**	**5**	**6**	**2**	**7**	**9**	**7**

Fully Dressed Description

Fully Dressed Description of Add New Cattle Information

Use Case UC-1	**Create Cattle Profile**
Related Requirement	Functional Requirement(s): REQ10, REQ11
Initiating Actor(s)	Farmers
Actor's Goal	• Login using profiles • Create animal profiles • Enter in cattle biographical information
Participating Actors	System
Preconditions	Farmers want to create an animal profile.
Post Conditions	Farmers create profiles for cattles.
Flow of Events for Main Success Scenarios	→ Logins into the system using profile → Clicks on create profile → Enter new cattle information. → Submits information. ← Shows newly created cattle profile with biographical

	information.
Flow of Events for Extensions(Alternate Scenarios)	No alternate scenarios

Fully Dressed Description of View Product Details

Use Case UC-2	**View Product Details**
Related Requirement	Functional Requirement(s): REQ2, REQ6, REQ10
Initiating Actor(s)	Customer
Actor's Goal	Use system to view product information
Participating Actors	System
Preconditions	Customer wants to view a report about meat being purchased.
Post Conditions	Customers view reports about meat products that they would like to purchase.
Flow of Events for Main Success Scenarios	→ Scans QR code on meat product packaging ← System shows the meat information
Flow of Events for Extensions(Alternate Scenarios)	No alternate scenarios

Fully Dressed Description of Add Product Information

Use Case UC-5	**Add Product Information**
Related Requirement	Functional Requirement(s): REQ4, REQ10
Initiating Actor(s)	Packaging Managers
Actor's Goal	● Login using profiles ● Insert additional information of products ● Generate QR Code
Participating Actors	System
Preconditions	Managers want to insert additional information on certain meat produce.
Post Conditions	Managers inserted additional information on the product.
Flow of Events for Main Success Scenarios	→ Logins into the system using profile → Search for cattle profile with cattle ID → Clicks add information ← System shows the cattle information →Inserts additional information
Flow of Events for Extensions(Alternate Scenarios)	Managers enters invalid credentials → Logins into the system using profile

	← System recognizes wrong password ← System notifies user on error

Fully Dressed Description of Generate Report

Use Case UC-7	**Generate Report**
Related Requirement	Functional Requirement(s): REQ8, REQ10
Initiating Actor(s)	BAHA
Actor's Goal	• Login in using profile • View all data in the system on cattle
Participating Actors	System
Preconditions	BAHA wants to track and trace cattle from birth to death
Post Conditions	None
Flow of Events for Main Success Scenarios	→ Logins into the system using profile → Selects a specific cattle → Clicks on viewProductDetails ← System display cattle trace
Flow of Events for Extensions(Alternate Scenarios)	No alternate scenarios

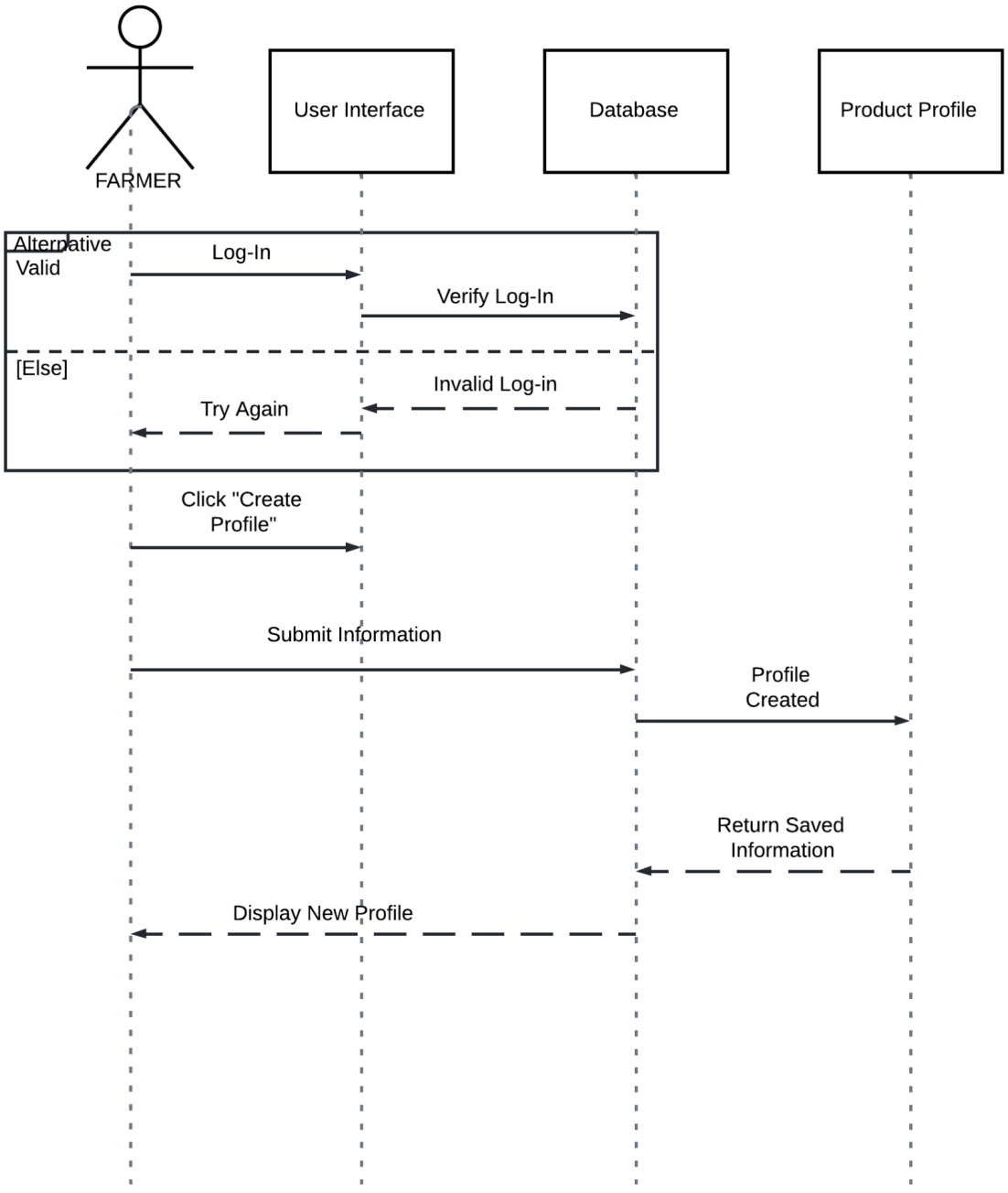
Fully Dressed Description of Add Slaughter Information

Use Case UC-8	**Add Slaughter Information**
Related Requirement	Functional Requirement(s): REQ4, REQ10
Initiating Actor(s)	Slaughterhouse manager
Actor's Goal	● Login in using profile ● View all data in the system on cattle ● Add cattle slaughter information ● Update cattle life status
Participating Actors	System
Preconditions	Slaughterhouse manager wants to add slaughter information of cattle
Post Conditions	Slaughterhouse manager added slaughter information of cattle
Flow of Events for Main Success Scenarios	→ Logins into the system using profile → Search for cattle profile with cattle ID → Clicks add information ← System shows the cattle information →Inserts additional information
Flow of Events for Extensions(Alternate	No alternate scenarios

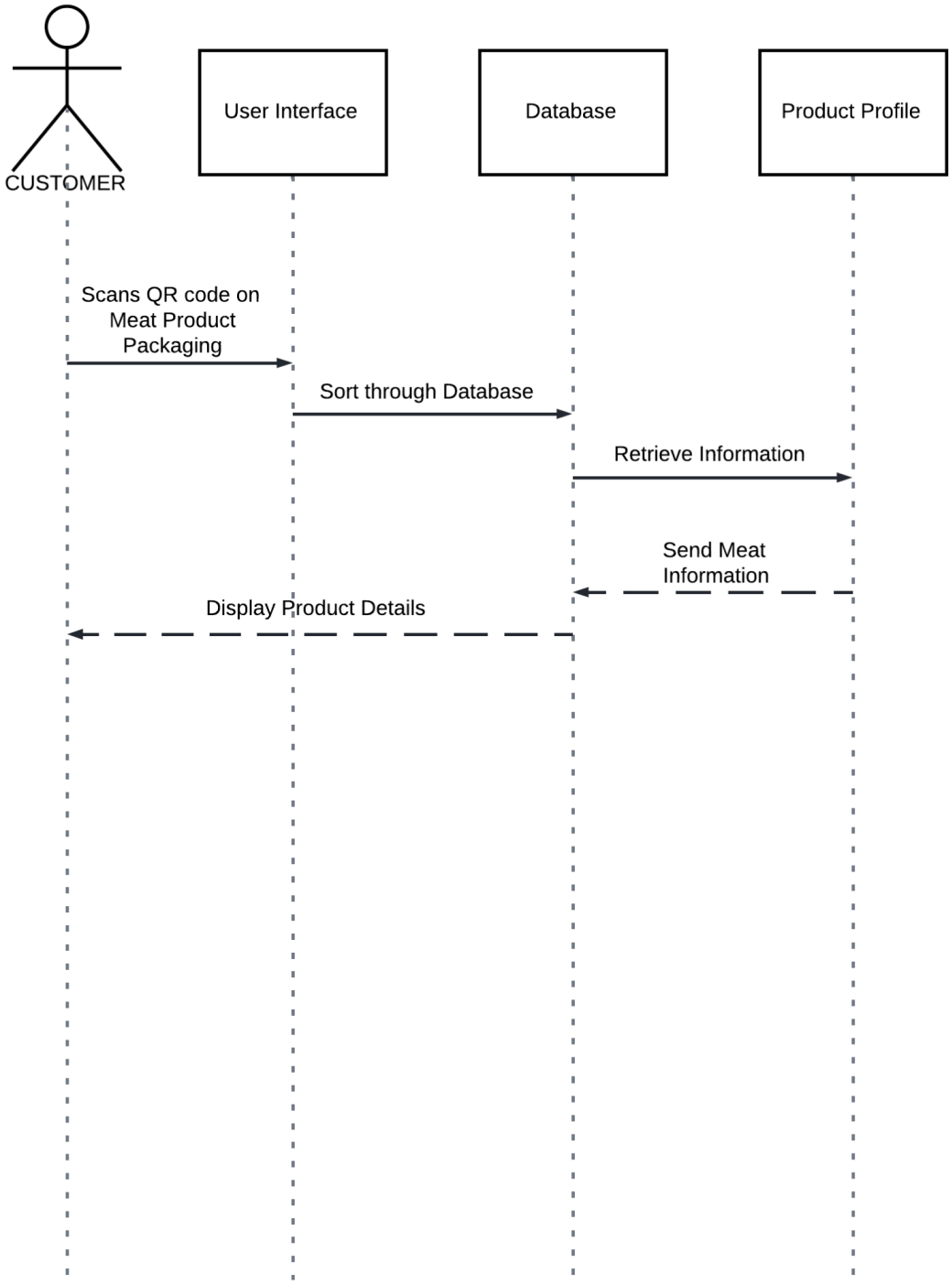
Scenarios)	

System Sequence Diagram

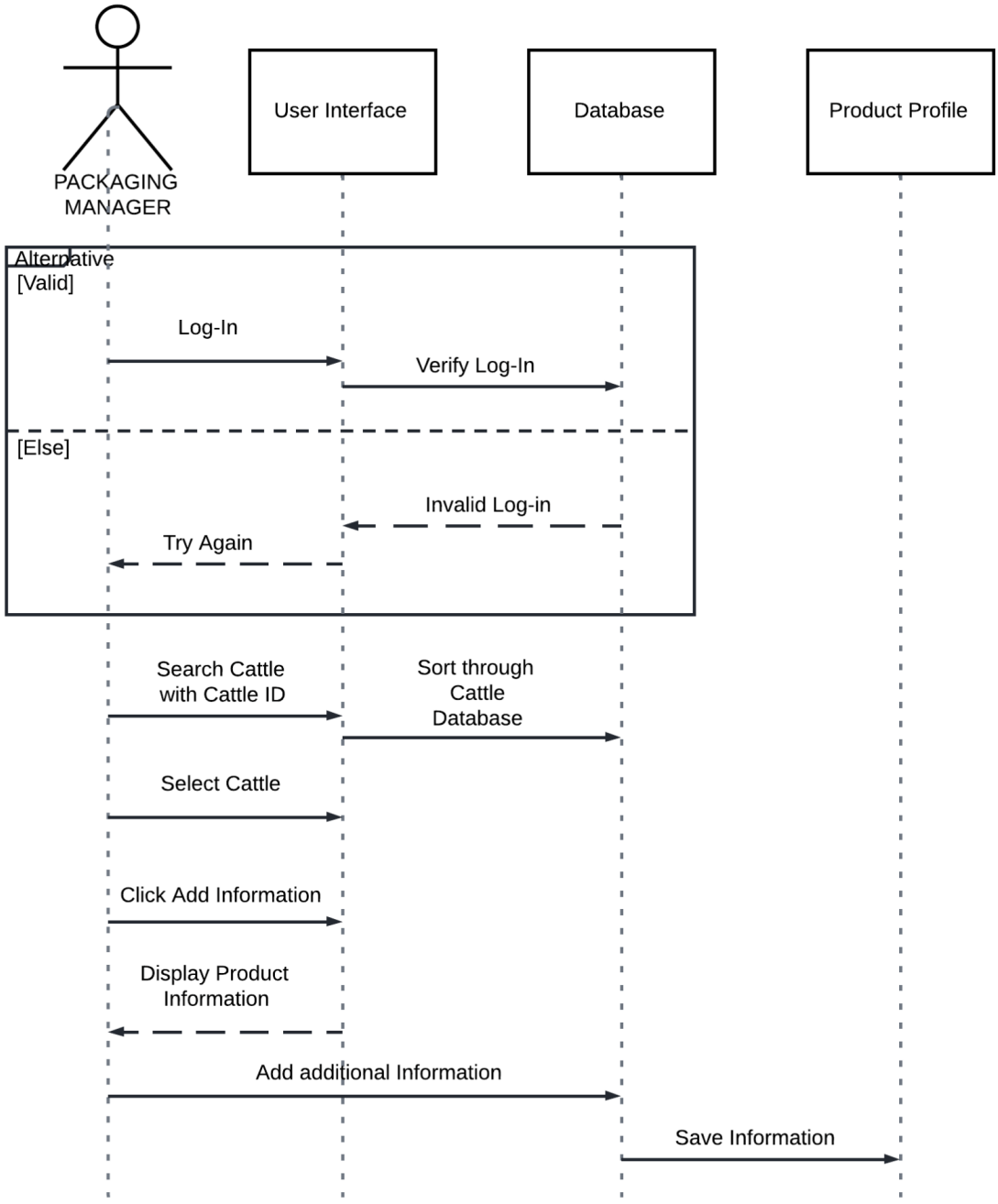
Use Case 1: Create Cattle Profile



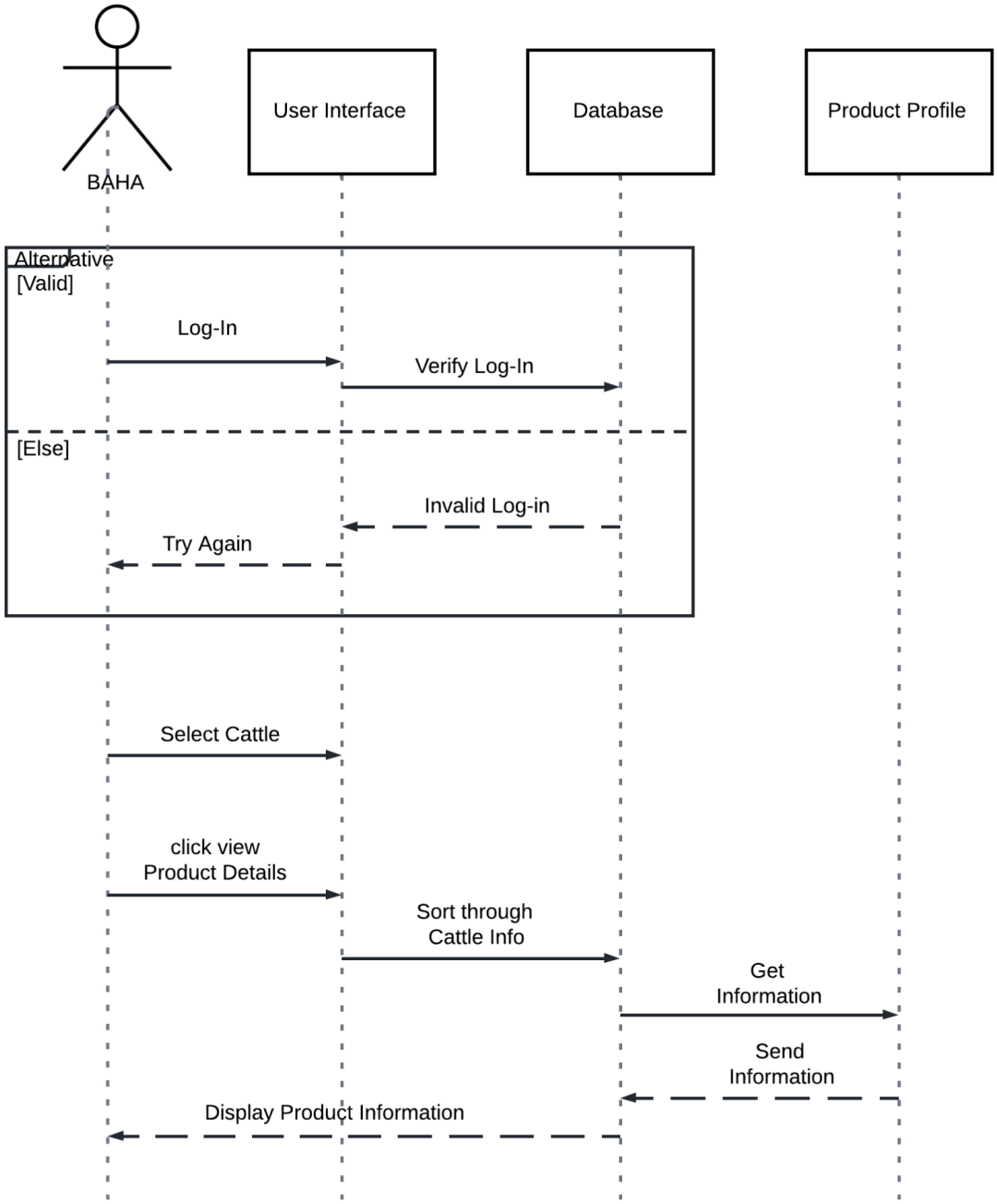
Use Case 2: viewProductDetails


CUSTOMER
User Interface
Database
Product Profile
Scans QR code on Meat Product Packaging
Sort through Database
Retrieve Information
Send Meat Information
Display Product Details

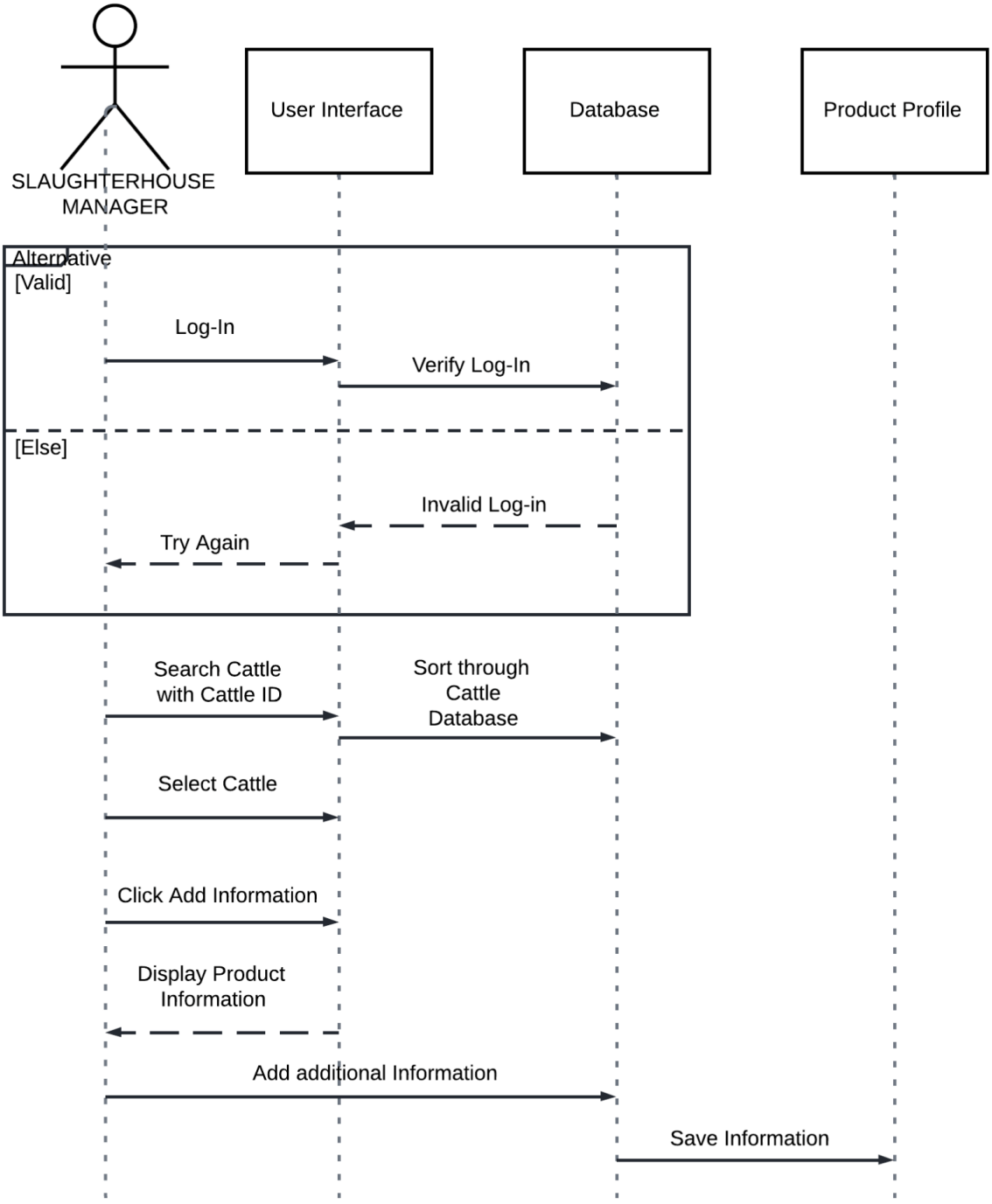
Use Case 5: addProductInformation



Use Case 7: generateReport



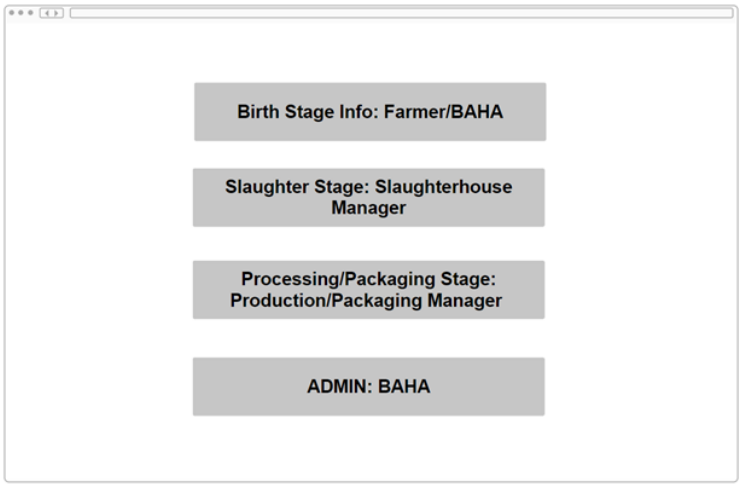
Use Case 8: addSlaughterInformation



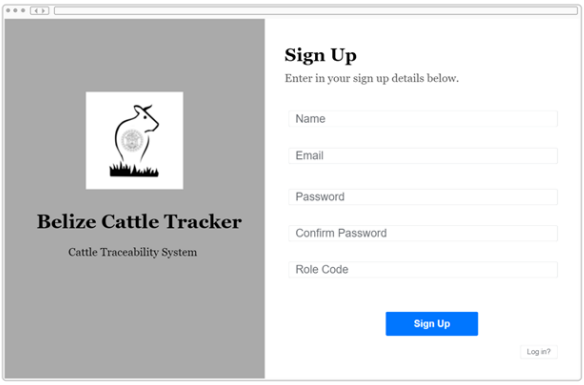
User Interface Specification

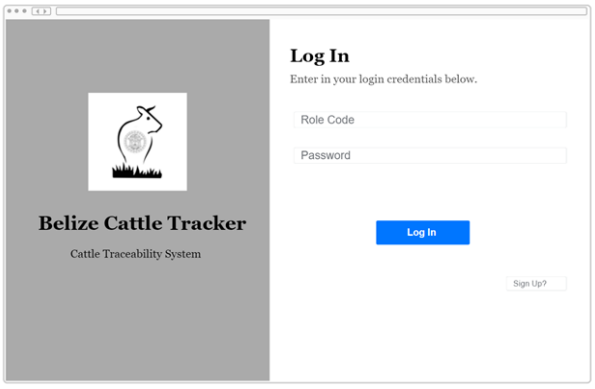
Preliminary Designs (MOCKUPS)

Role Portal

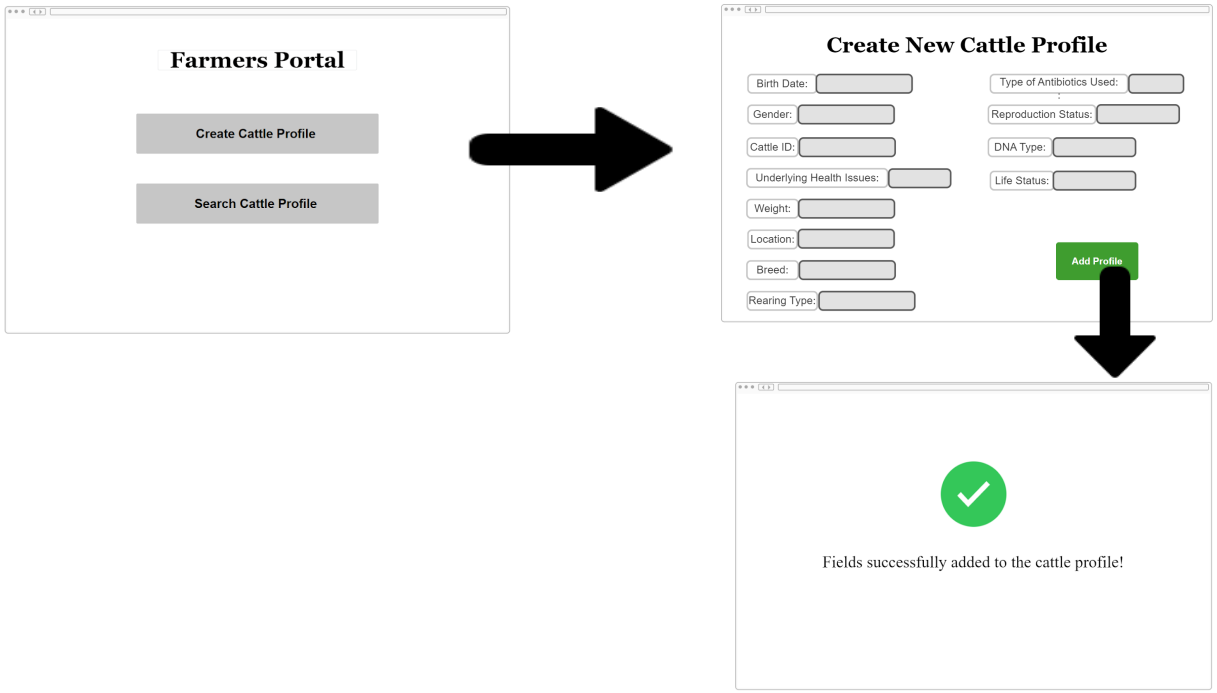


Log In and Sign Up

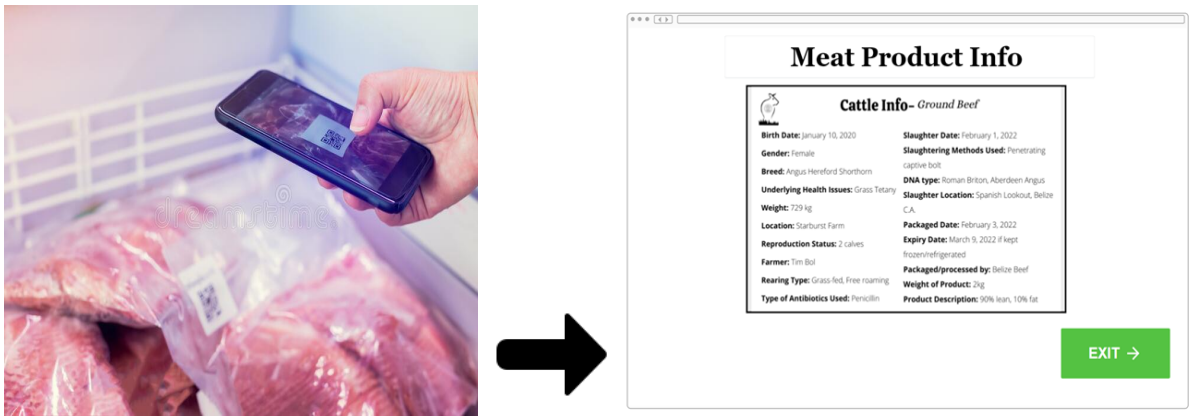




UC-1 Create Cattle Profile

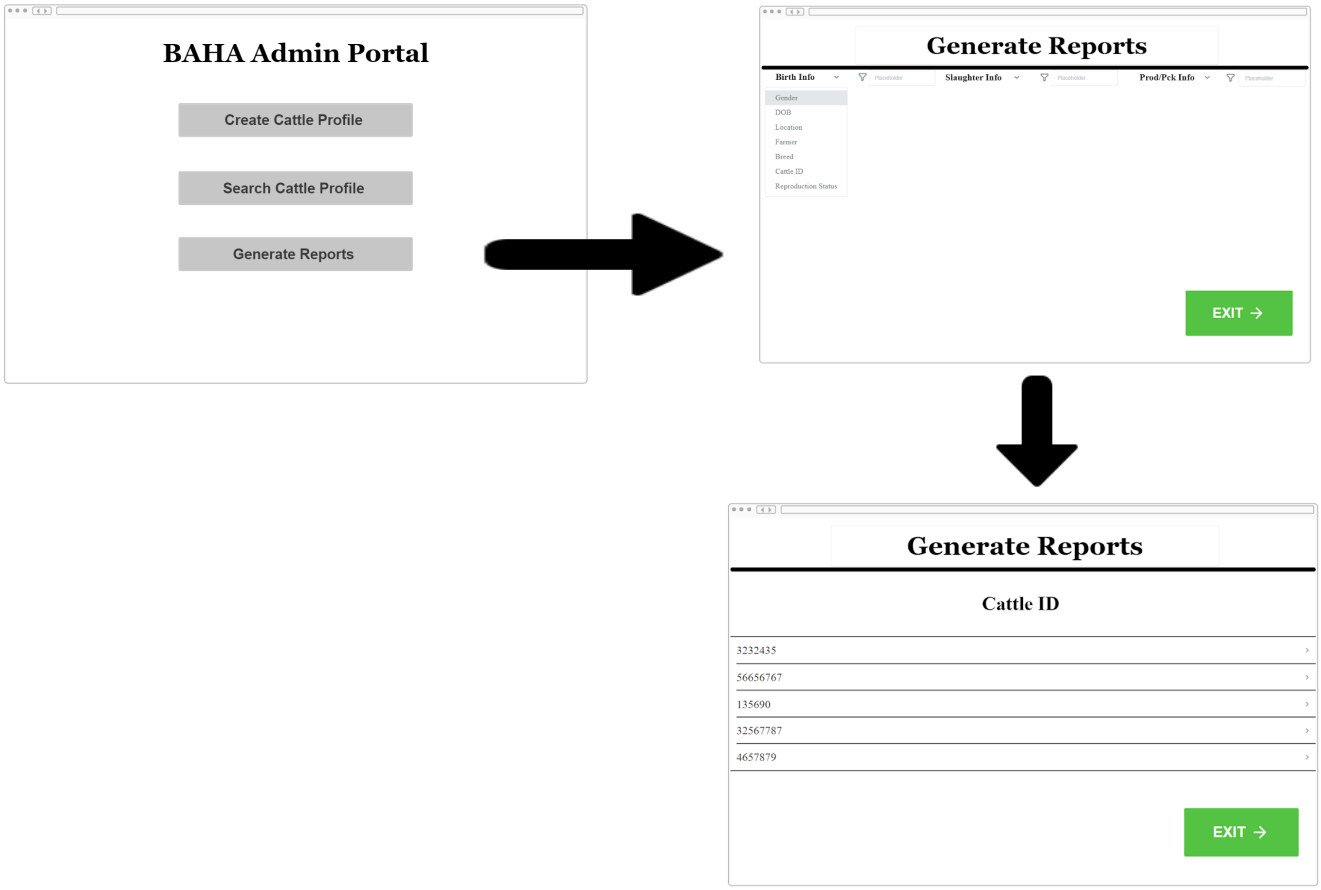


UC-2 View Product Details

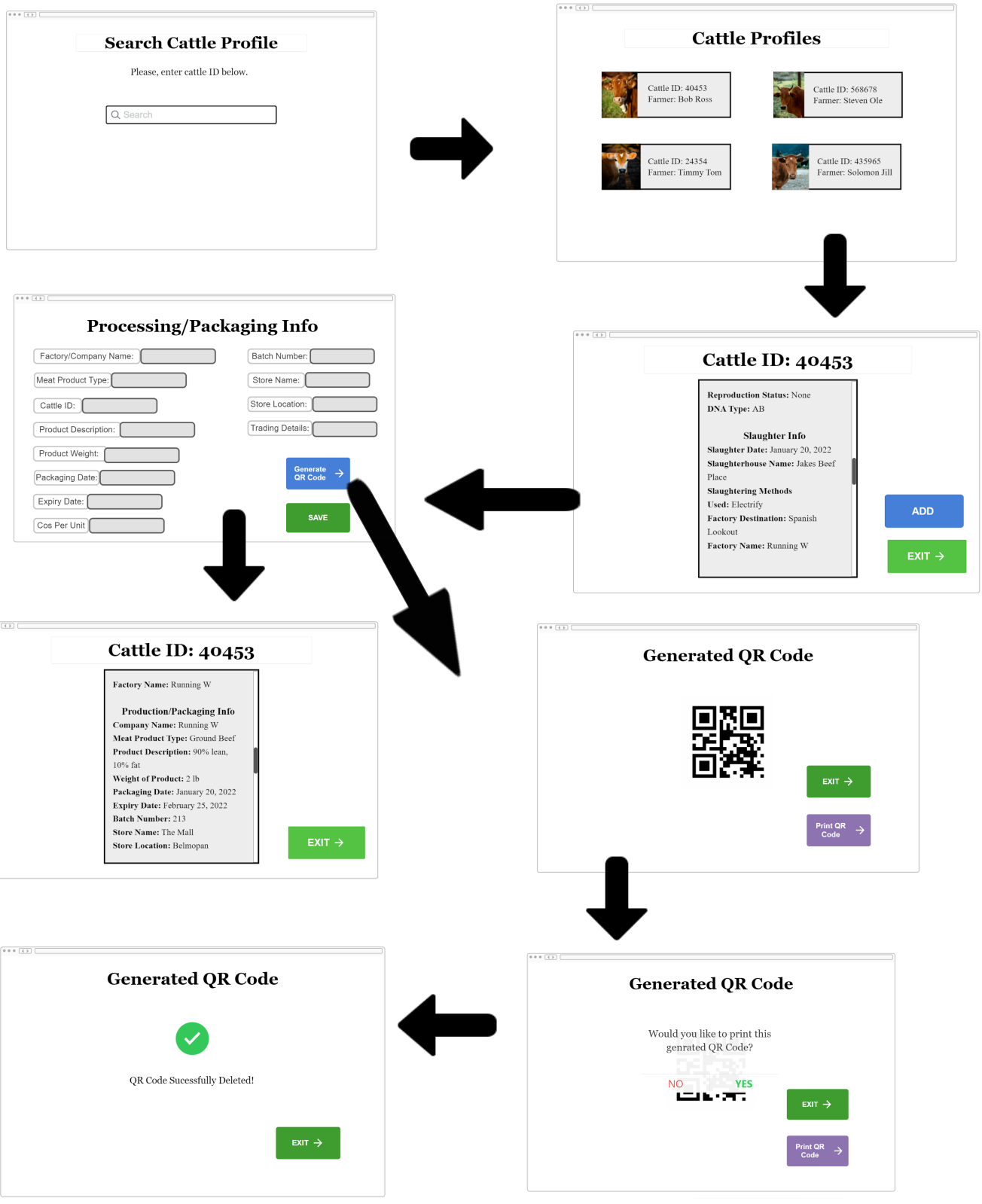


UC-8 Add Slaughter Information

UC-7 Generate Report



UC-5 Add Product Information



User Effort Estimation

UC 1:

Create Cattle Profile: (16 clicks and various keystrokes)

1. Click Create Profile(1 click)
2. Enter cattle id and other information about the animal (xy keystrokes and 12 clicks to switch to the different fields)
3. Click Add Profile to create a new cattle profile. (1 click)
4. Click “yes” on Confirmation Dialogue (1 click).
5. Click “exit” to close the success message (1 click).

UC 2:

View Product Details: (2 clicks and scrolling)

1. Click on camera and the Scan QR code (if code is valid, then the system will prompt user with the product detail)
2. Scroll down to view more information of the specified meat produce.
3. Click exit to leave the info view.

UC 5:

Add Product Information: (19 clicks and various keystrokes)

1. Click on search bar (1 click)
2. Click on cattle profile (1 click)

3. Click on Add (1 click)
4. Enters processing/packaging information about the meat product (xy keystrokes, where xy is the number of keystrokes entered in the 14 fields and 14 clicks to switch to the different fields)
5. Click on “Save” (1 click)
6. Click on “Generate QR Code” (1 click)
7. Click on “Print QR Code” (1 click)
8. Clicks on “Exit” to leave (1 click)

UC 7:

Generate Report: (6 clicks)

1. Click on “Generate Reports” (1 click)
2. Click on “Generate Birth Info Reports” button (1 click)
3. Click on “Generate Slaughter Info Reports” button (1 click)
4. Click on “Generate Product/Packaging Info Reports” button (1 click)
5. Click on desired filter (1 click)
6. Enter filter criteria (xy keystrokes, where xy is the number of keystrokes entered in the filter field, eg: filterings by “cattleID”)
7. Click on print button to print the report (1 click)
8. Clicks the “←” to leave “Generate Reports” (1 click)

UC 8:

Add Slaughter Information: (10 clicks and various keystrokes)

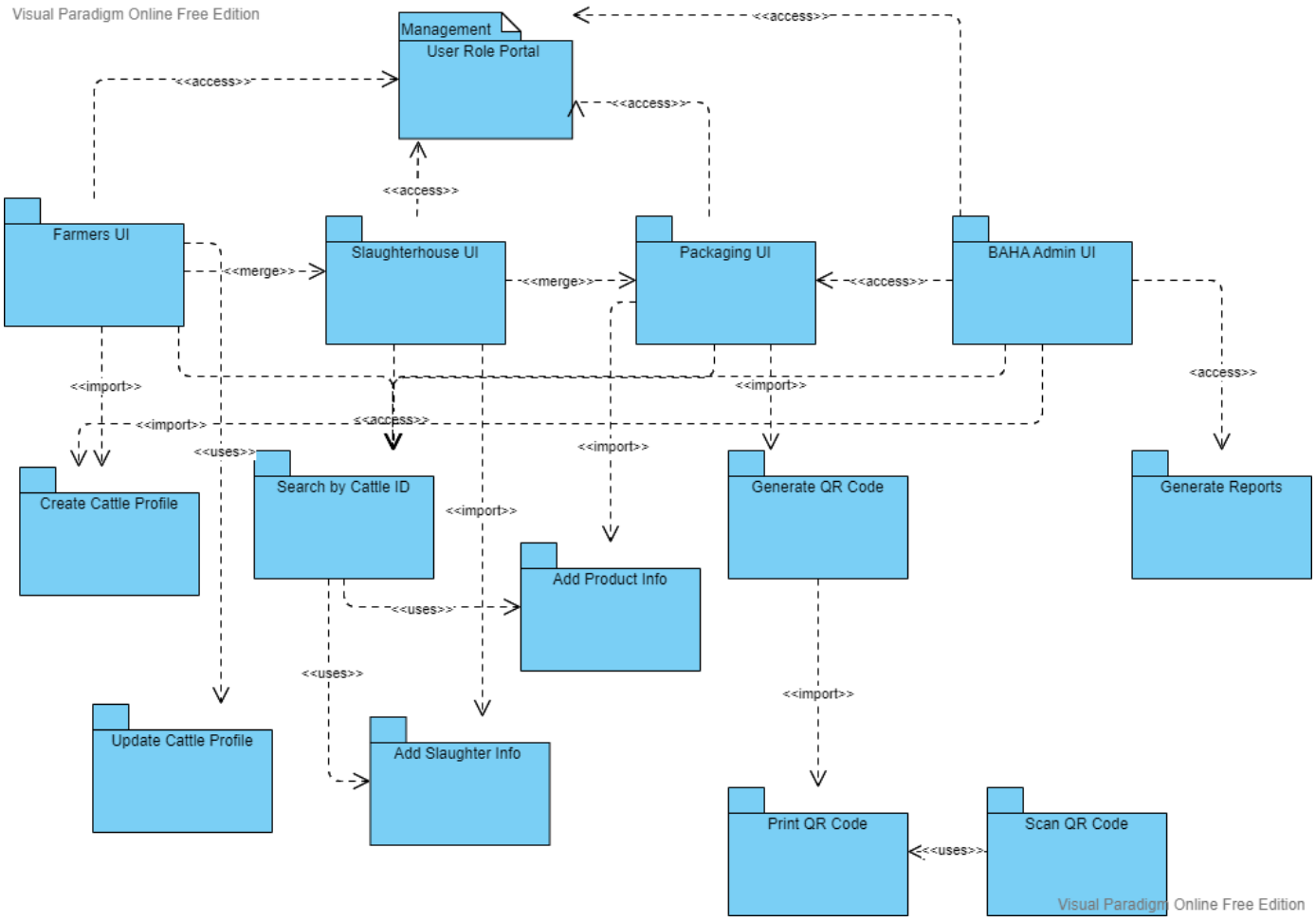
1. Click on search bar (1 click)
2. Click on cattle profile (1 click)
3. Click on Add (1 click)
4. Enters slaughter information about the animal to the cattle's information (xy keystrokes, where xy is the number of keystrokes entered in the 6 fields and 6 clicks to switch to the different fields)
5. Clicks on "Add" to Update the Cattle's Information (1 click)
6. Click "yes" on Confirmation Dialogue (1 click).
7. Click "exit" to close the success message (1 click).

System Architecture

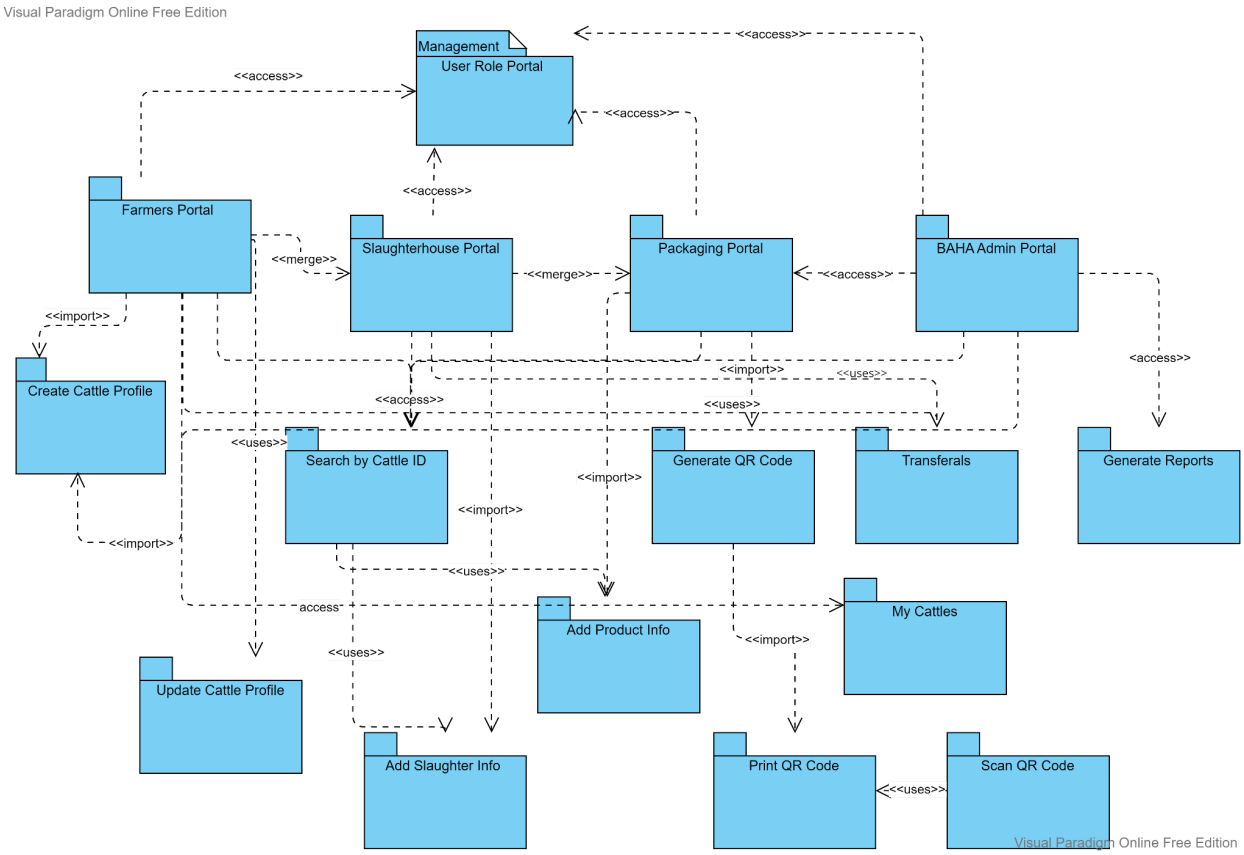
Identifying Subsystems

The front end is the first impression the customer gets from a system. To develop a user-friendly graphical user interface we will employ css library bootstrap and html. The back end is the server-side code that will be running on the hosting server. The code will have complex functionality. It is what will get the data to be displayed in the front end of the system. The system will contain the important part which is the database. This is where all the data for the system will be stored permanently. This system must be planned and designed properly to avoid any issues at the development stage. It is also important for the system to have a reliable and safe database management system.

Before Changes



After Changes



Explanation of Changes: There were minimum changes made to the overall flow and connections of the UML package diagrams. The only changes that were made was that the Tranferrals and My Cattles Packages were added. The reason for adding the transferrals is because the UI component will be needed to transfer off the cattle information to the next stage, it is a non requirement function (NONREQ14) but it is crucial for the birth and slaughter stage as it aids with the efficiency and reliability of transferring over the information to the next stage. Additionally, the My Cattles Page was also added to the updated UML package diagram as it is a main UI that will allow easier access for the farmer to view/update the cattle profiles they have made (this was not a part of the priority use cases, but it was just added so that the farmer would

not have to go through a long search process just to access the specific cattle profile that they would like to view).

Architecture Styles

To put into simple terms, software architecture refers to the organization of a system. In essence, it is the blueprint which includes all components and the way these parts communicate, the operation environment, and the principles of design used in the creation of the software. In most cases, the upgrading and the evolution of the software for future use will also be considered in the architectural style.

In this system, the architectural style is Layered. Layered architecture patterns are n-tiered patterns where the components are organized into horizontal layers. This method is traditionally used for designing most software and is made to be self-independent. This means that all the components are interconnected but are not totally dependent on each other.

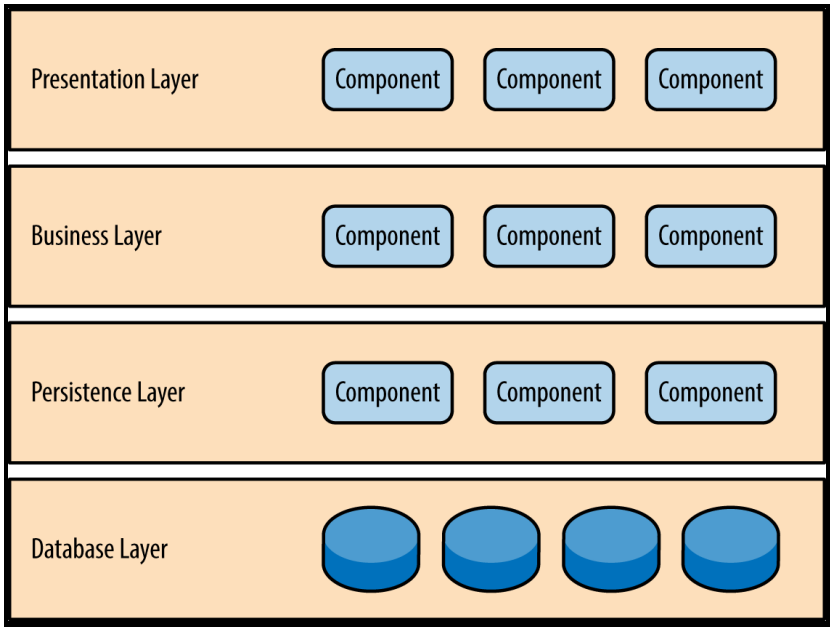
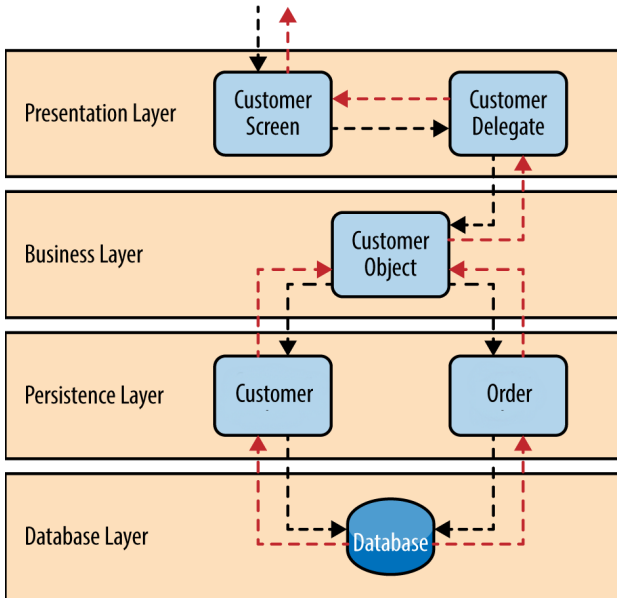


Diagram of Layered Architecture

Layered architecture has 4 layers:

- Presentation layer, which contains the user interface layer where we see and enter data into an application.
- Business layer, the layer responsible for executing business logic as per the request.
- Application layer, which acts as a medium for communication between the ‘presentation layer’ and ‘data layer’.
- Data layer, which contains a database for managing data.

Below is a diagram representing just how these layers communicate.



In the case of the Belize Cattle Tracker, the user interface is displayed to the client, this is the presentation layer. When the client logs in the information transfers to the business layer, which acts as an authenticator and security. If the login was invalid, then the business layer would send a message back to the presentation layer displayed as “Invalid”. If the login was successful, we can move into the persistent layer, which has all the persistent data stored in the ram. Finally, we have the database layer, this layer holds all the data that BAHA, Farmers and Managers will insert.

Mapping Subsystems to Hardware

In software engineering, subsystems refer to hardware. The Belize Cattle Tracker is a system that would need to run on multiple computers considering BAHA would have a database that would serve as a main server. Farmers are to register their cattle with BAHA whereby they will create an animal profile. This profile will be held in BAHA’s system. With that, the clients (managers, farmers and customers) will run the system through their own devices and keep these profiles updated.

Connectors and Network Protocols

The Belize Cattle Tracker system will be hosted on a web server, this will make it able for customers and administrators to interact with the system from anywhere with an internet connection. Hypertext Transfer Protocol Secure(HTTPS) will be utilized as the communication protocol across the network to ensure security. HTTPS encrypts the data while transferred on the internet and therefore secures all communications that occur between the clients’ and the server.

Moreover HTTPS will allow data to be uploaded on the website, system, and database.

Global Control Flow

The system will be an event driven system. This means that it will be a multiuser system and that it will allow for the use of multiple roles using the system simultaneously, with each role being able to perform a different task. This means that program execution will be largely determined by the system. When an event is triggered by one of the roles, an action will then be executed in response to it. By doing this, it makes the system a lot more flexible and open to changes and this also allows the system to be responsible for controlling the flow and execution of the program. An example of this would be when a customer is scanning an item's qr code in order to generate the appropriate data for it. A trigger will be set up to respond to the event which would be the scanning being done. When the user scans the item, the action will then be executed and the appropriate information will be extracted from the database and displayed on the customer's screen.

The system will be a real time system, which means that it is subjected to certain time constraints. Since the system will be periodic, dynamic tasks will be incorporated. These tasks will then be invoked at the occurrence of an event. So for example when a QR code is generated, these tasks will occur at arbitrary time intervals and the execution time should be fast so that it meets the deadline that is set relative to the event happening. However since most of the tasks will be aperiodic and have soft deadlines, the time constraints allow for more flexibility and deadlines should be met, but not meeting them will not cause catastrophic results.

Hardware Requirements

For these requirements, a PC and a local server will be required for the system to work properly.

Screen Display:

- Has to be adaptable to both mobile and desktop.
- Minimum resolution of 640 × 480 pixels on laptops and desktops.
- Minimum resolution of 240 x 320 pixels on smartphones.

Disk Storage

- Since this will be a website and not an application, disk storage will not be required, any phone that has access to the internet and a camera that can scan a QR code is eligible to use this system. However to ensure smooth operation, it is recommended that your phone has at least 2 gigabytes of space on your hard disk.

Special Instrument:

- Barcodes for product packaging so that store cashiers can scan to access the cost per unit for the particular product.
- QR codes for customers to scan and access biographical info of the cattle the produce came from.

Network Bandwidth

- Minimum network bandwidth of 1 Mbps
- This will be needed because information will be transmitted over the internet when the

QR code is scanned. If the network bandwidth is too low, then the information will be slow and the user will not be able to view the information in a quick manner.

Operating System

- Windows 7 and above will be needed for the website to work properly.

Section 2

Analysis and Domain Modeling

Conceptual Model

(I) Concept Definitions

Concept Definition for CreateCattleProfile(UC1)

Responsibility Description	Type	Concept Name
RS.1 Coordinate actions of concept associated with the use case delegate the work to other concepts.	D	Controller
RS.2 Verifies if the user is a farmer.	D	Verifier
RS.3 System displays farmer portal.	K	PortalDisplay
RS.4 Select option to create cattle profile.	D	ProfileCreator
RS.5 Enter cattle information.	D	EnterInfo
RS.6 Selects the save option.	D	SaveInfo
RS.7 Database stores cattle information.	D	Database Connection
RS.8 Notifies farmers that information was successfully added.	D	Notifier

Concept Definition for ViewProductDetails(UC2)

Responsibility Description	Type	Concept Name
RS.1 Coordinate actions of concept associated with the use case delegate the work to other concepts.	D	Controller
RS.2 User enters roleID into the form.	K	Interface Page
RS.3 Medium for customer to scan QR Code	K	Scanner
RS.4 QR Code is scanned using the Medium	K	QR Code
RS.5 Prepare a database query that best matches the QR Code and retrieve the records from the database.	D	DatabaseRetriever
RS.6 Database provides product details.	D	Database Connection
RS.7 Render the retrieved records for sending to the actor's web browser for display.	K	DisplayReport

Concept Definition for AddProductInformation(UC5)

Responsibility Description	Type	Concept Name
RS.1 Coordinate actions of concept associated with the use case delegate the work to other concepts.	D	Controller
RS.2 User enters roleID into the form.	K	Interface Page
RS.3 Verifies if the user is a packaging manager.	D	Verifier
RS.4 System displays packaging portal.	K	PortalDisplay
RS.5 System prompts for the search filter criteria	K	SearchRequest
RS.6 Prepare a database query that best matches the actor's search criteria and retrieve the records from the database.	D	DatabaseRetrieve
RS.7 Render the retrieved records for sending to the actor's web browser for display.	K	DisplayReport
RS.8 Select add option for displayed document.	D	AddInfo
RS.9 Enter packaging information.	D	EnterInfo
RS.10 Selects the save option.	D	SaveInfo
RS.11 Store saved info into database.	D	DatabaseStore
RS.12 Notifies Actor that information was successfully added	D	Notifier

RS.13 Selects the generate QR code option.	D	GenerateQR
RS.15 System prompts the React library to generate a new unique QR code.	K	QRRequest
RS.16 Prepare a database query that best matches the QR's data requirements and retrieve the records from the database.	D	QRDataRetrieve
RS.17 Render the retrieved QR for sending to the actor's web browser for display.	D	DisplayQR

Concept Definition for GenerateReport(UC7)

Responsibility Description	Type	Concept Name
RS.1 Coordinate actions of concept associated with the use case delegate the work to other concepts.	D	Controller
RS.2 User enters roleID into the form.	K	Interface Page
RS.3 Verifies if the user is a BAHA admin.	D	Verifier
RS.4 System displays the BAHA admin portal.	K	PortalDisplay
RS.5 Select option to Generate Reports.	D	ReportGenerator
RS.6 System displays search form .	K	SearchFormDisplay

RS.7 Select search filter/cattle category.	D	FilterSearch
RS.8 System prompts for the search filter criteria.	K	SearchRequest
RS.9 Prepare a database query that best matches the actor's search criteria and retrieve the records from the database.	D	DatabaseRetrieve
RS.10 Render the retrieved records for sending to the actor's web browser for display.	K	DisplayRecords
RS.11 Select cattle id.	D	SelectID
RS.12 Render the retrieved profile/report for sending to the actor's web browser for display.	K	DisplayReport

Concept Definition for AddSlaughterInformation(UC8)

Responsibility Description	Type	Concept Name
RS.1 Coordinate actions of concept associated with the use case delegate the work to other concepts.	D	Controller
RS.2 User enters roleID into the form.	K	Interface Page
RS.3 Verifies if the user is a slaughterhouse manager.	D	Verifier
RS.4 System displays the slaughterhouse portal.	K	PortalDisplay

RS.5 System prompts for the search filter criteria	K	SearchRequest
RS.6 Prepare a database query that best matches the actor's search criteria and retrieve the records from the database.	D	DatabaseRetrieve
RS.7 Render the retrieved records for sending to the actor's web browser for display.	D	DisplayReport
RS.8 Select add option for displayed document.	D	AddInfo
RS.9 Enter slaughter information.	D	EnterInfo
RS.10 Selects the save option.	D	SaveInfo
RS.11 Notifies Actor that information was successfully added	D	Notifier
RS.12 Store saved info into database.	D	DatabaseStore

(II) Association Definitions

Association Definition for CreateCattleProfile(UC1)

Concept Pair	Association Description	Association Name
Verifier ↔ PortalDisplay	System verifies the role ID of an actor and displays their portal.	Verifies

PortalDisplay↔ PortalCreator	When the system displays the actor's portal the actor can create a cattle profile.	Presents Portal
PortalDisplay↔ InterfacePage	The portal renders and displays on the Interface Page.	Shows Interface Page
ProfileCreator↔ EnterInfo	Before a cattle profile is created, the user has to enter in the cattle info for that account.	Creates Profile
EnterInfo↔Data base Connection	When the actor presses the save button after the cattle info is entered, the information gets stored to a table in the database.	Initiates Database
SaveInfo↔Notif ier	The actor is prompted with a success message when the save button is selected.	Notifies Success
Database Connection↔N otifier	The system saves the entered cattle information into the database. After that, it notifies the actor with a confirmation message.	Notifies Success

Association Definition for ViewProductDetails(UC2)

Concept Pair	Association Description	Association Name
Controller ↔ Database Connection	Connects to the database	Connects Database

Scanner ↔ Display Report	Once the QR code is scanned, the record is retrieved from the database	Retrieves Record
Database Connection ↔ Display Report	Provides data that matches the actor's QR code	Provides Data
Display Report↔InterfacePage	The report info renders and displays on the Interface Page.	Shows Interface Page

Association Definition for AddProductInformation(UC5)

Concept Pair	Association Description	Association Name
Controller ↔ DatabaseRetrieve	Controller passes search requests to Database Connection.	Conveys Request
PortalDisplay↔ SearchRequest	When the portal is displayed, the actor can click the search request button to search for a criteria (cattleID).	Search Data
DatabaseRetrieve ↔	Provides data that matches the Actor's search criteria.	Provides Data

DisplayReport		
Display Report↔InterfacePage	The report info renders and displays on the Interface Page.	Shows Interface Page
AddInfo↔ EnterInfo	When the add button is selected, the actor can enter in packaging information.	Enters Info
AddInfo↔ SaveInfo	When the save button is selected, the system will save the added information and push the data to the database.	Saves Data
SaveInfo ↔ DatabaseStore	Once the information has been saved, the information is placed into a table in the database.	Stores Data
GenerateQR↔QRRequest	There should be a prompt for a QR code to be generated once the user selects the option for it.	Selects Info
QRRequest↔QRRetrieve	Once the QR code has been generated, the accurate information should be retrieved from the database.	Retrieves Data
QRRetrieve↔DisplayQR	The Information that has been retrieved from the Database is displayed on the device.	Displays DataVeri

Association Definition for GenerateReport(UC7)

Concept Pair	Association Description	Association Name
Controller ↔ DatabaseRetrieve	Controller passes search requests to Database Connection.	Conveys Request
Portal Display ↔ Search Request	When the portal is displayed, the actor can click the generate reports button to search for cattle and retrieve reports about them.	Search Data
DatabaseRetrieve ↔ DisplayRecords	Provides data that matches the Actor's search criteria.	Provides Data
Display Records↔InterfacePage	The report info renders and displays on the Interface Page.	Shows Interface Page
Display Records ↔ Select ID	When the records are displayed, the actor can click a cattle id to access the animal's profile/report.	Search Report
DatabaseRetrieve ↔ DisplayRecords	Provides data that matches the Actor's search criteria.	Provides Data

Association Definition for AddSlaughterInformation(UC8)

Concept Pair	Association Description	Association Name
Controller ↔ DatabaseRetrieve	Controller passes search requests to Database Connection	Conveys Request
Verifier ↔ PortalDisplay	System verifies the role ID of an actor and displays their portal.	Verifies
PortalDisplay↔ InterfacePage	The portal renders and displays on the Interface Page.	Shows Interface Page
PortalDisplay↔ SearchRequest	When the portal is displayed, the actor can click the search request button to search for a criteria (cattleID).	Search Data
SearchRequest ↔DataabaseRetreive	Once a search request is entered, the system will be prompted to retrieve the records from the database.	Retrieves Data
DatabaseRetrieve ↔ DisplayReport	Provides data that was retrieved from the database and displays the report to the actor on a newly rendered page.	Provides Data
Display	The report info renders and displays on the	Shows Interface Page

Report↔InterfacePage	Interface Page.	
AddInfo↔ EnterInfo	When the add button is selected, the actor can enter in slaughter information.	Enters Info
AddInfo↔ SaveInfo	When the save button is selected, the system will save the added information and push the data to the database.	Saves Data
SaveInfo ↔ DatabaseStore	Once the information has been saved, the information is placed into a table in the database.	Stores Data

(III) Attribute Definitions

Attribute Definition for CreateCattleProfile(UC1)

Concept	Attributes	Attribute Description
Verifier	User ID	Used to identify the Actor's identity, which will determine what operations that can be performed in turn
	UserRole	Determines the Actor's role

Database Connection	User's Identity	Prepare a Query that retrieves based on the class that is selected
Notifier	Data Status	Used to show a message that the Actor's Data has been successfully added to the Database.
ProfileCreator	Cattle ID	Used to Create the Cattle ID, that will be added to the Database.

Attribute Definition for ViewProductDetails(UC2)

Concept	Attributes	Attribute Description
QR Code	QR ID	Determines the ID of the QR Code that was scanned
	Class Information	Identifies the class of the QR code that was scanned
Scanner	Medium Type	Used to Determine what kind of Device is being used to scan QR Code

Attribute Definition for AddProductInformation(UC5)

Concept	Attributes	Attribute Description
Verifier	User ID	Used to identify the Actor's identity, which will determine what operations that can be performed in turn
	UserRole	Determines the Actor's role
Search Request	Search Criteria	Needed to Prepare a Database Query to return a result that matches the Actor's search.
AddInfo	CattleInfo	Used to Add information about a specific Cattle into the Database. Can be broken down into more Attributes such as "cut of meat", "breed" and etc.
SaveInfo	ClassID	Used to update the cattle Data in the Database after new information has been added.
Notifier	Data Status	Used to show a message that the Actor's Data has been successfully added to the Database.
QRDataRetrieve	QR ID	Used to Identify the proper QR information so that the accurate information can be displayed

		on the screen.

Attribute Definition for GenerateReport(UC7)

Concept	Attributes	Attribute Description
Verifier	User ID UserRole	Used to identify the Actor's identity, which will determine what operations that can be performed in turn Determines the Actor's role
Filter Search	Filter Request	Used to Determine what information to filter out
Search Request	Search Criteria	Needed to Prepare a Database Query to return a result that matches the Actor's based on the filter provided.
DatabaseRetrieve	ClassID	Used to Identify the cattle information for the specific class
SelectID	Cattle ID	Needed to extract information from the Database in order to match the Cattle ID that

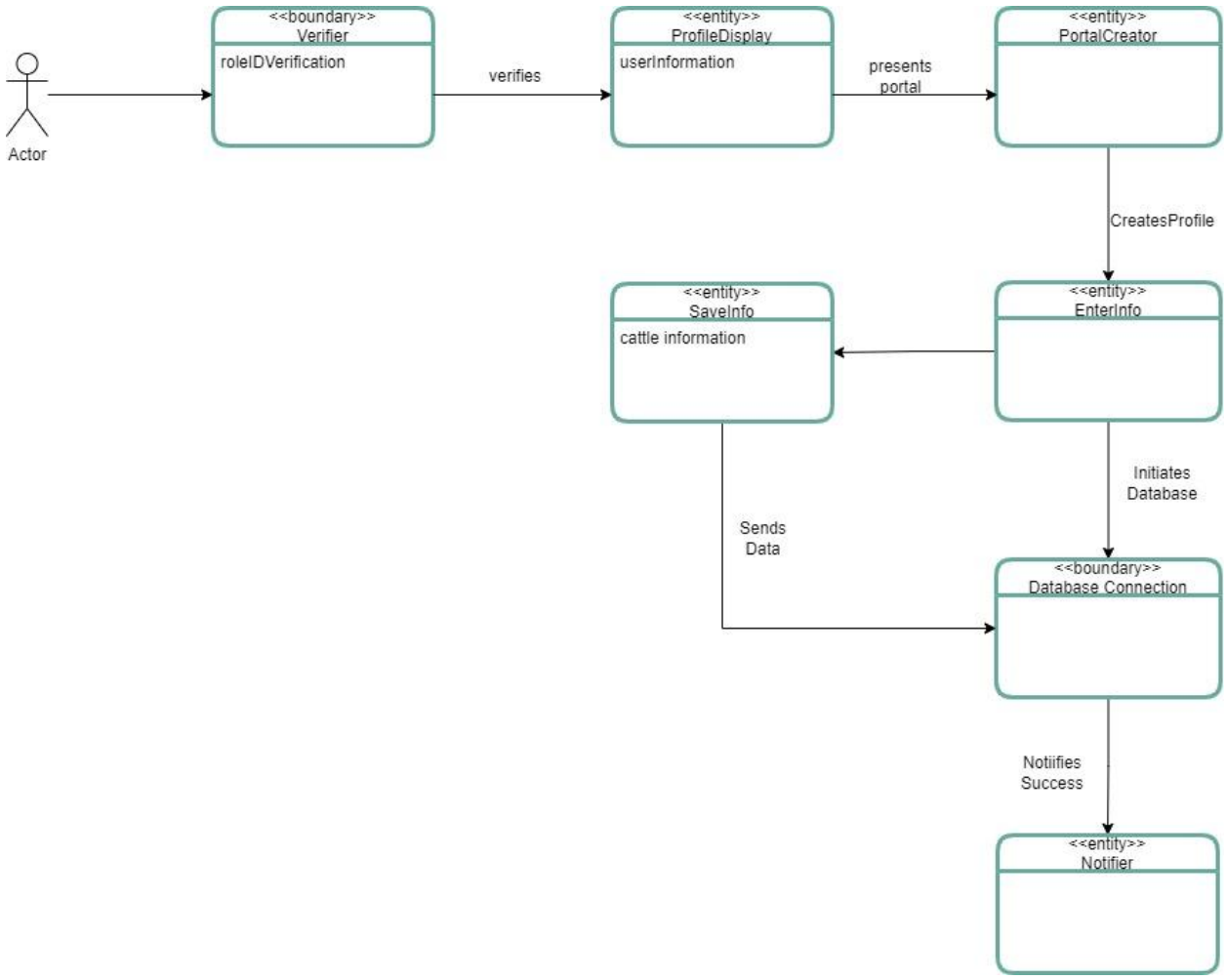
		was entered, which is then displayed on the screen.

Attribute Definition for AddSlaughterInformation(UC8)

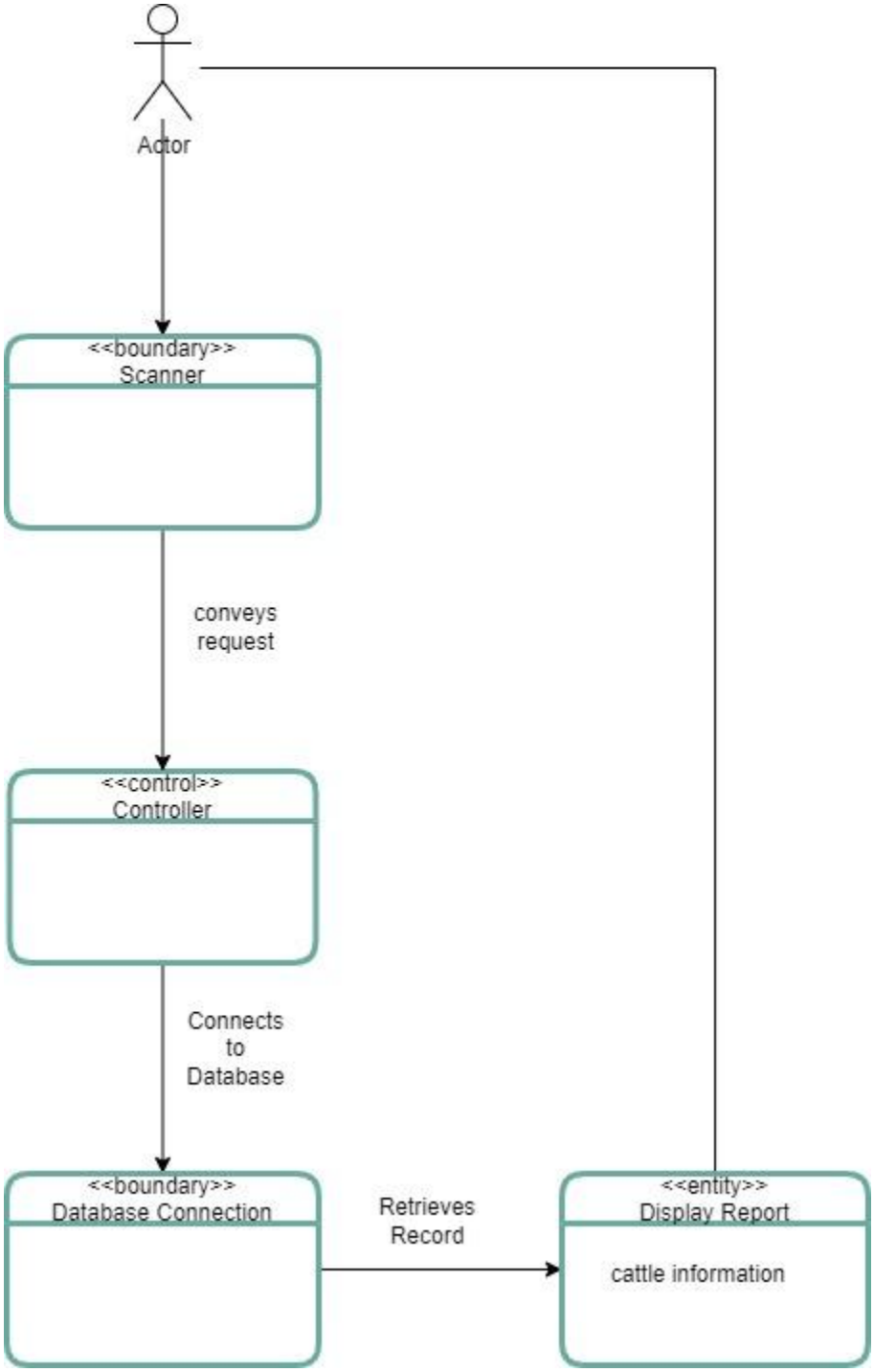
Concept	Attributes	Attribute Description
Verifier	User ID UserRole	Used to identify the Actor's identity, which will determine what operations that can be performed in turn Determines the Actor's role
Search Request	Search Criteria	Needed to Prepare a Database Query to return a result that matches the Actor's search.
AddInfo	CattleInfo	Used to Add information about a specific Cattle into the Database. Can be broken down into more Attributes such as "cut of meat", "breed" and etc.
SaveInfo	ClassID	Used to update the cattle Data in the Database after new information has been added.

Domain Models

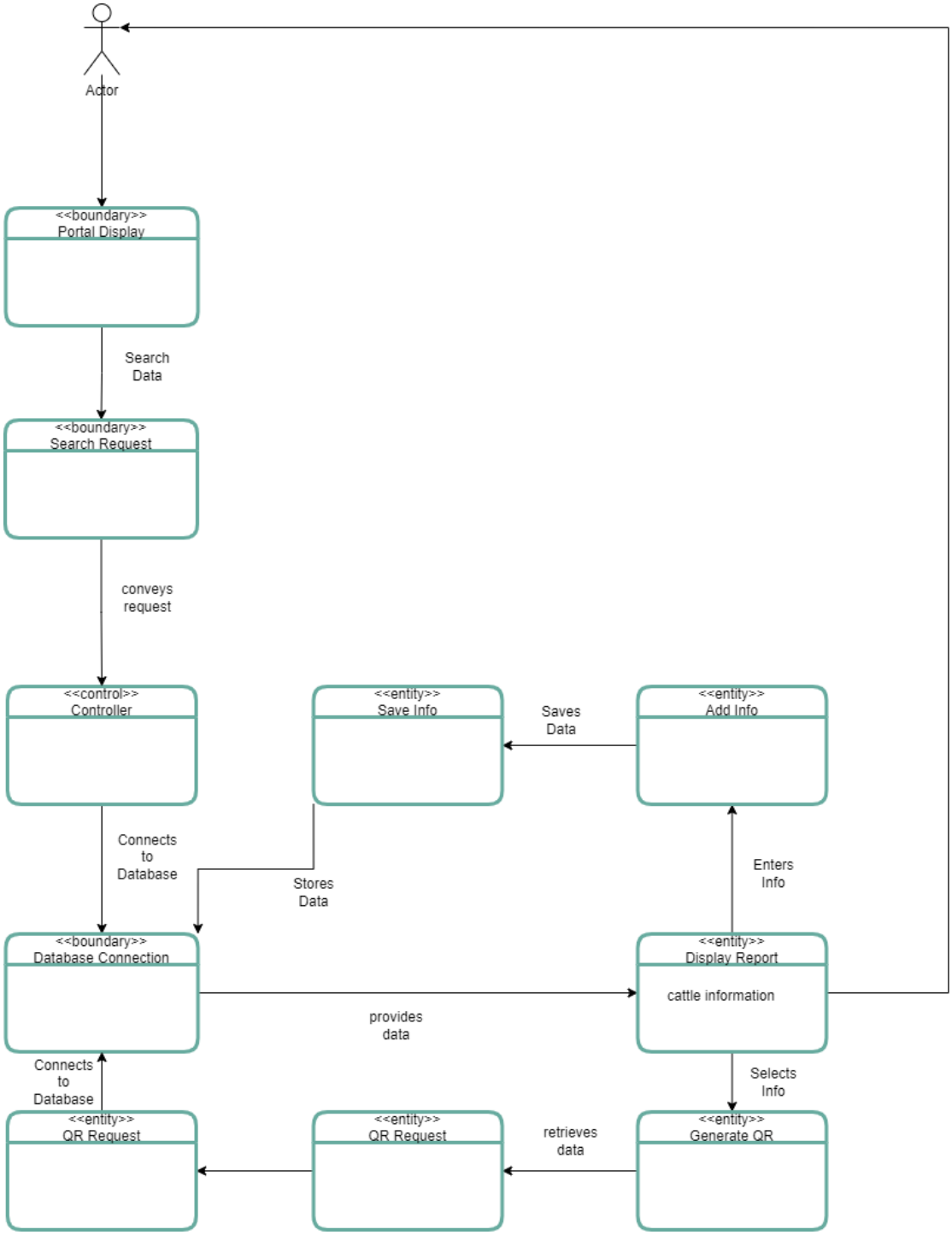
Domain Model of UC-1 CreateCattleProfile



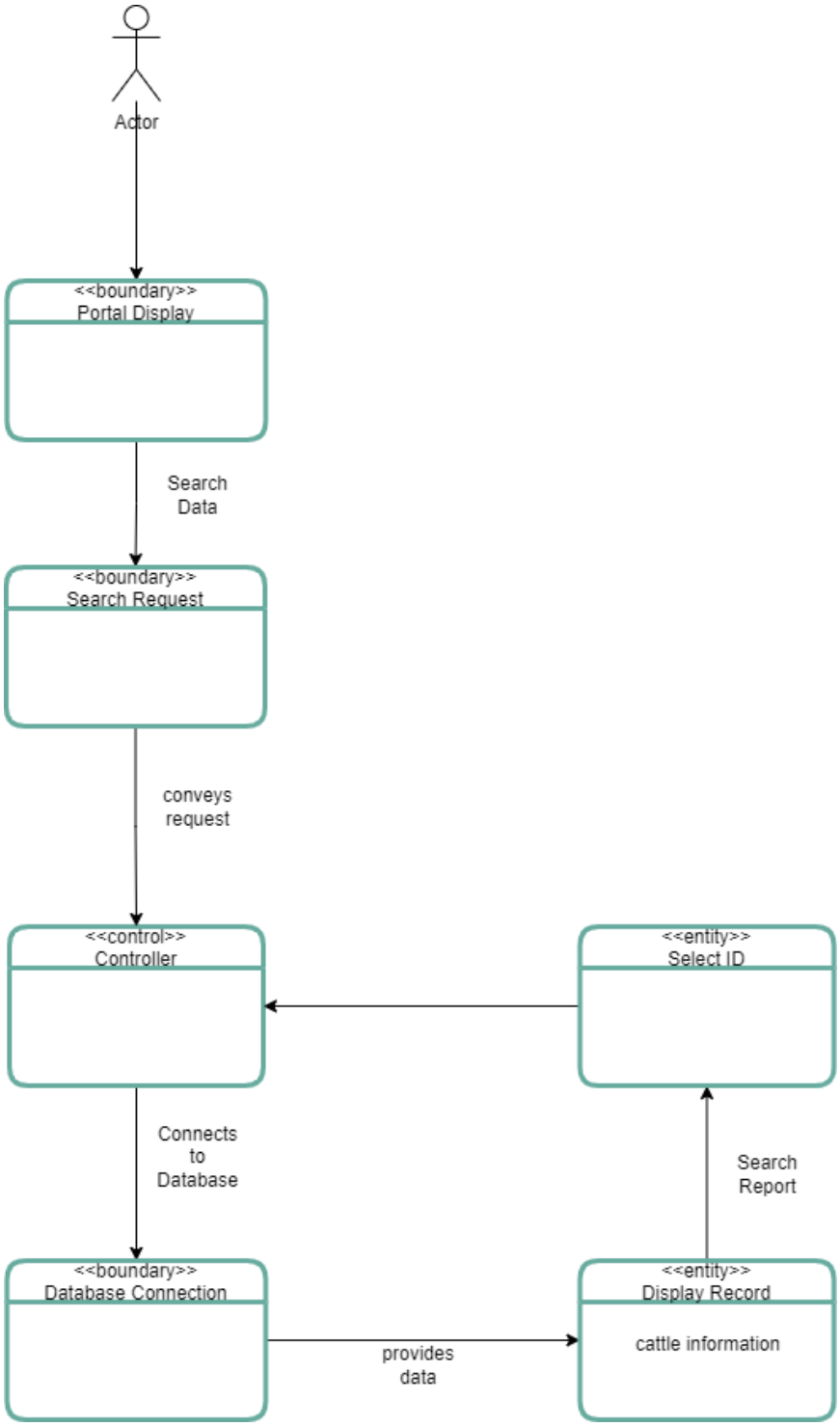
Domain Model of UC-2 viewProductDetails



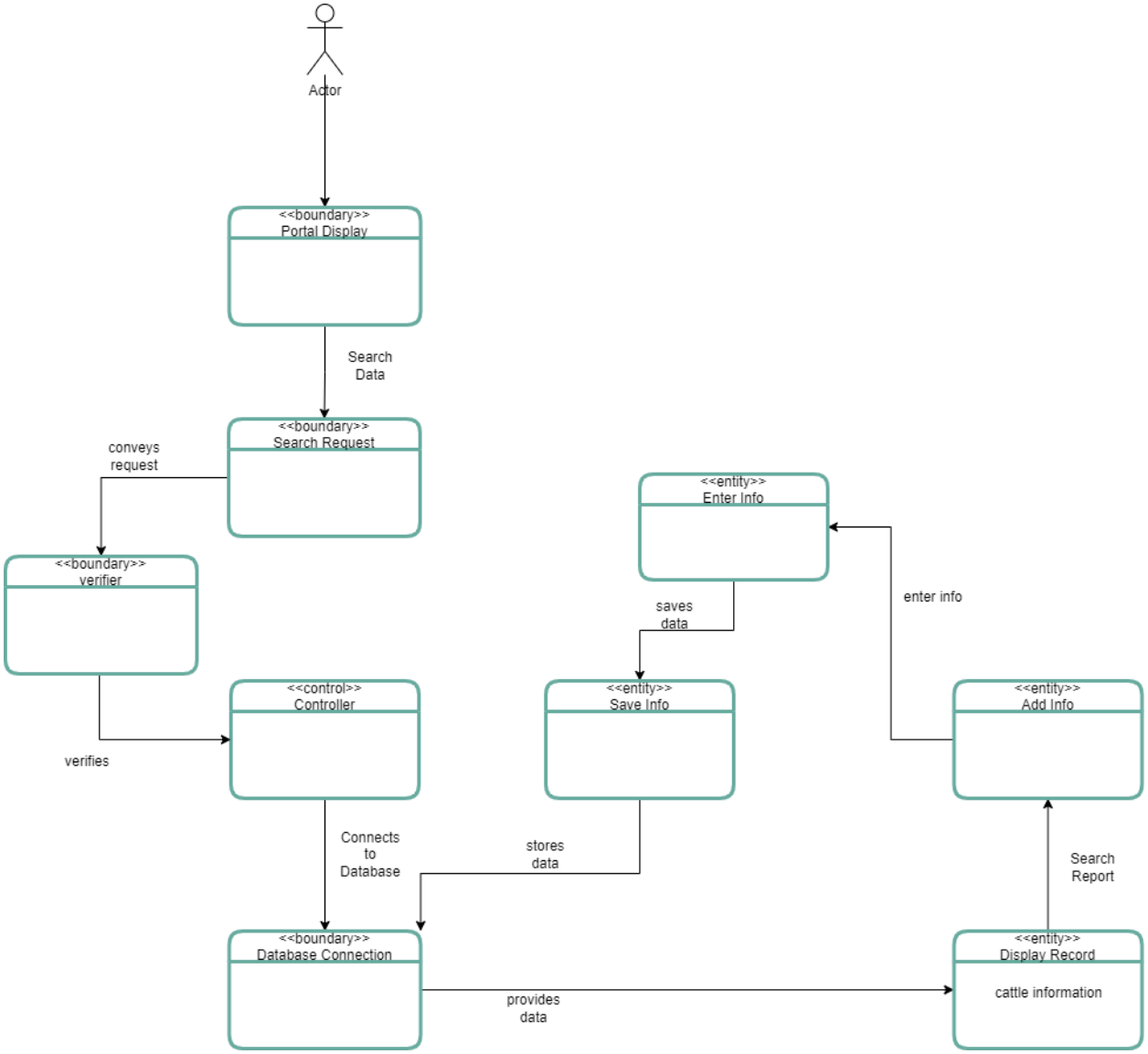
Domain Model of UC-5 addProductInformation



Domain Model of UC-7 generateReport



Domain Model of UC-8 addSlaughterInformation



There were no changes/evolutions of the domain models as there were no extreme system changes made for the set up and flow of the system. Therefore, flow of the system (for the priority use cases) kept the same implementations as the above listed domain models.

(IV) Traceability Matrix

Domain Concepts	PW	Controller	Verifier	Portal Display	Profile Creator	EnterInfo	SaveInfo	DatabaseConnection	Notifier	Scanner	QR Code	DisplayReport	SearchRequest	DatabaseRetreive	AddInfo	DatabaseStore	GenerateQR	QRRequest	QR Data Retrieve	DisplayQR	ReportGenerator	SearchFormDisplay	Filter Search	SearchRequest	CriteriaSearch	Select ID	DisplayRecords
UC1	9	x	x	x	x	x	x	x	x						x	x												
UC2	11	x						x		x	x	x		x			x	x	x	x								x
UC3	3	x												x	x	x												
UC4	6	x											x	x								x		x		x	x	
UC5	8	x	x	x		x	x	x	x				x	x	x	x	x	x	x			x		x		x		
UC6	2	x								x		x					x	x	x	x								x
UC7	9	x	x	x		x	x	x			x	x	x	x	x		x	x	x		x	x	x	x	x	x	x	x
UC8	11	x	x	x		x	x	x	x	x	x	x	x	x	x	x						x		x		x		
UC9	7	x				x	x	x					x	x		x			x			x		x		x		

Each priority use case corresponds to ideas that are used to ensure that the use case performs what it is supposed to do. The Database Connection is used in almost every use case since it is critical to the overall application's operation. The reason for this is that most of our tasks include the system retrieving and storing data in a database. Because the search concept allows a user to access a specific cow profile through its cattleID, it is critical for those use cases that need information to be updated to a cattle profile. In certain circumstances, the notifier is also used to provide success or completion notifications to the user (for example, when adding information to a livestock profile).

System Operation Contracts

System Contracts for CreateCattleProfile

Contract Name	verifyInfo
Operation	**VerifyInfo(string info)**
Cross Reference	**Use Cases: Create Cattle Profile**
Responsibilities	**Verify that user credentials is valid**
Type	**System**
Exceptions	**Credentials were entered incorrectly**
Pre-Conditions	**Information needs to be entered.**
Post-Conditions	**Information is either valid or Invalid, If it is Valid, then user gains access to the system, else the user is prompted to re-enter credentials**

Contract Name	createInfo
Operation	**createInfo(string info)**
Cross Reference	**Use Cases: Create Cattle Profile**
Responsibilities	**Creates a New Cattle profile which is then added**

	to the database
Type	**System**
Exceptions	**No information is entered or The wrong type of information is entered in the wrong field.**
Pre-Conditions	**Information needs to be entered to be saved**
Post-Conditions	**Information has been received and successfully added to the Database and the Actor is notified**

System Contracts for ViewProductDetails

Contract Name	recordScan
Operation	**recordScan(scan: QRCode)**
Cross Reference	**Use Cases: ViewProductDetails**
Responsibilities	**Parses the QRCode so that the information can be retrieved from the database and displayed**
Type	**System**
Exceptions	**None**

Pre-Conditions	**Scanner is readily accessible because the user has the camera app open.**
Post-Conditions	**QRCode information is successfully parsed and displayed to the user.**

Contract Name	receiveQRInfo()
Operation	**receiveQRInfo()**
Cross Reference	**Use Cases: ViewProductDetails**
Responsibilities	**Receives QRCode information from the scanner**
Type	**System**
Exceptions	**Unsuccessful Scan (User scans QRCode Improperly, eg: Only captures half of the QR Code)**
Pre-Conditions	**User has already scanned QRCode**
Post-Conditions	**Information has been received and verified**

System Contracts for AddProductInformation

Contract Name	displayRecords
Operation	**displayRecords()**
Cross Reference	**Use Cases: Add Product Information**
Responsibilities	**Display Information that matches the Actor's Search Criteria**
Type	**System**
Exceptions	**No information has been Entered/ Enters Criteria that doesn't exist in the Database**
Pre-Conditions	**Information has been entered.**
Post-Conditions	**Information has been validated and displayed for the Actor to view.**

Contract Name	notifyUser
Operation	**NotifyUser()**

Cross Reference	**Use Cases: Add Product Information**
Responsibilities	**Notifies the User that the information has been successfully added**
Type	**System**
Exceptions	**Incorrect Information entered (Enters a numerical value in a field that requires text values)**
Pre-Conditions	**User has clicked the Add button**
Post-Conditions	**Information has been validated and successfully inserted**

System Contracts for GenerateReport

Contract Name	filterResult
Operation	**filterResult()**
Cross Reference	**Use Case: GenerateReport**
Responsibilities	**Filters the information stored in the database based on the category provided by the Actor.**

Type	**System**
Exceptions	**Invalid Filter Category Provided**
Pre-Conditions	**Filter category has been selected**
Post-Conditions	**Information has been extracted from the database and is displayed to the screen**

Contract Name	displayReport
Operation	**displayReport()**
Cross Reference	**Use Case: GenerateReport**
Responsibilities	**Displays the retrieved profile/report to the actor's web browser for display**
Type	**System**
Exceptions	**N/A**
Pre-Conditions	**Cattle ID has been selected**
Post-Conditions	**Cattle ID was validated and the accurate information is displayed for the user to view**

System Contracts for AddSlaughterInformation

Contract Name	retrieveInfo
Operation	**retrieveInfo()**
Cross Reference	**Use Cases: AddSlaughterInformation**
Responsibilities	**Retrieves the accurate information that matches the user's input from the database.**
Type	**System**
Exceptions	**Invalid search criteria provided**
Pre-Conditions	**Search criteria needs to be inputted to the form**
Post-Conditions	**Once the criteria has been validated and the appropriate data has been extracted, it is displayed on screen.**

Contract Name	updateInfo
Operation	**updateInfo()**
Cross Reference	**Use Cases: AddSlaughterInformation**
Responsibilities	**Updates the database based on the information**

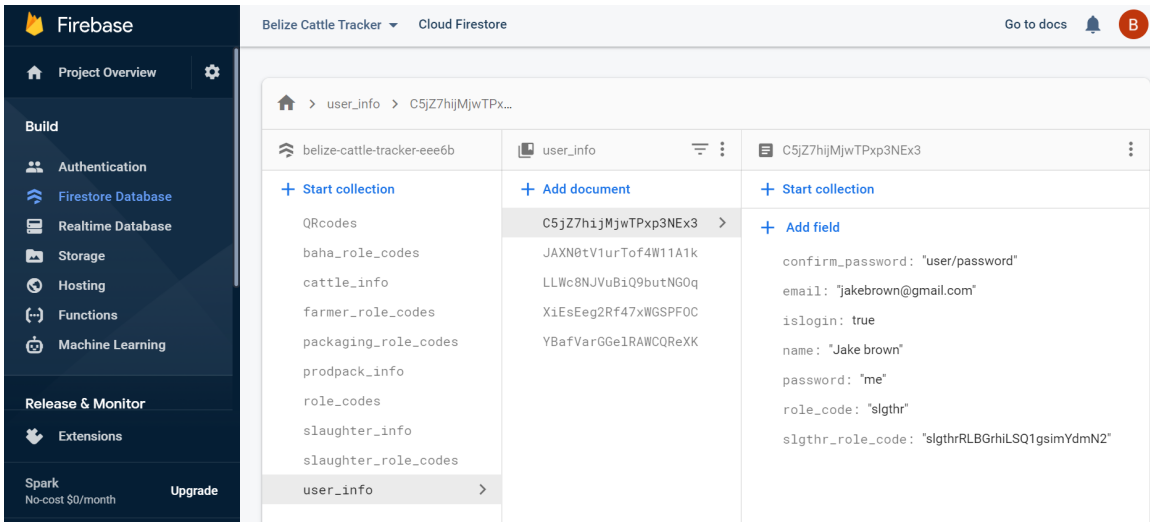
	provided by the Actor
Type	**System**
Exceptions	**No information is given**
Pre-Conditions	**createInfo from Use Case CreateCattleProfile needs to have been executed**
Post-Conditions	**The database is updated and the information is successfully added. The Actor is notified that the information has been added.**

Data Model and Persistent Data Storage

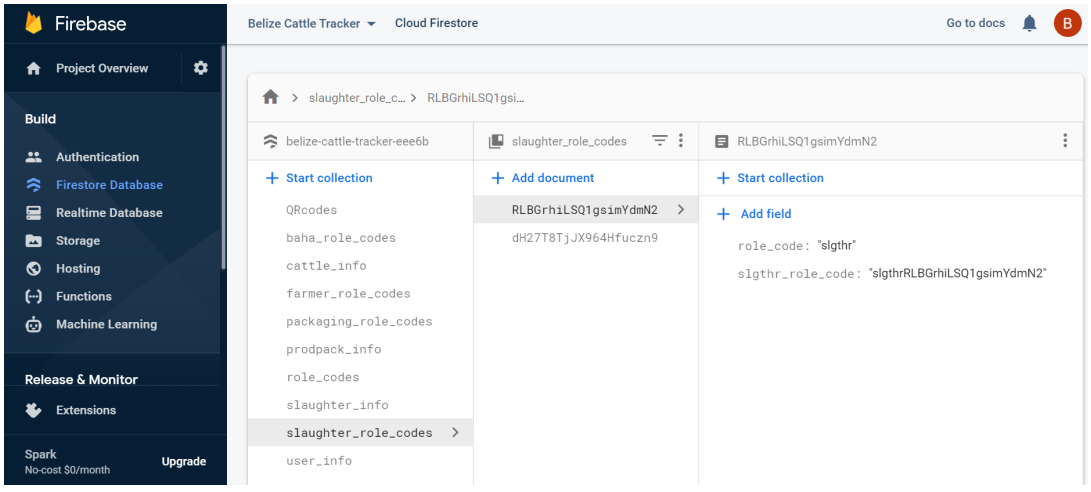
The Belize Cattle Tracker consists of various scenarios where data has to be retrieved, passed, and displayed. These retrieved and stored data must outlive a single execution of the system as new data will be added and updated to previously available data, because of this, the data will be stored in a cloud-hosted database (Firebase). Since it is a noSQL database, it allows for data to be stored and synced between different fields in real time. Being a non relational database, SQL queries such as JOINS cannot be used to link tables together so document references and api links have to be included in the code of the function; though it can sometimes be linked through similar fields in the different tables (Delaney, 2018). Moreover, in regards to Firebase, a collection would be considered a table in a relational database, a document would be considered

a table attribute and a field would be considered a table row/tuple. (Additionally, Firebase is also a schema-less database. Below is our Firebase Setup. Collections are linked through the "cattleid" field.

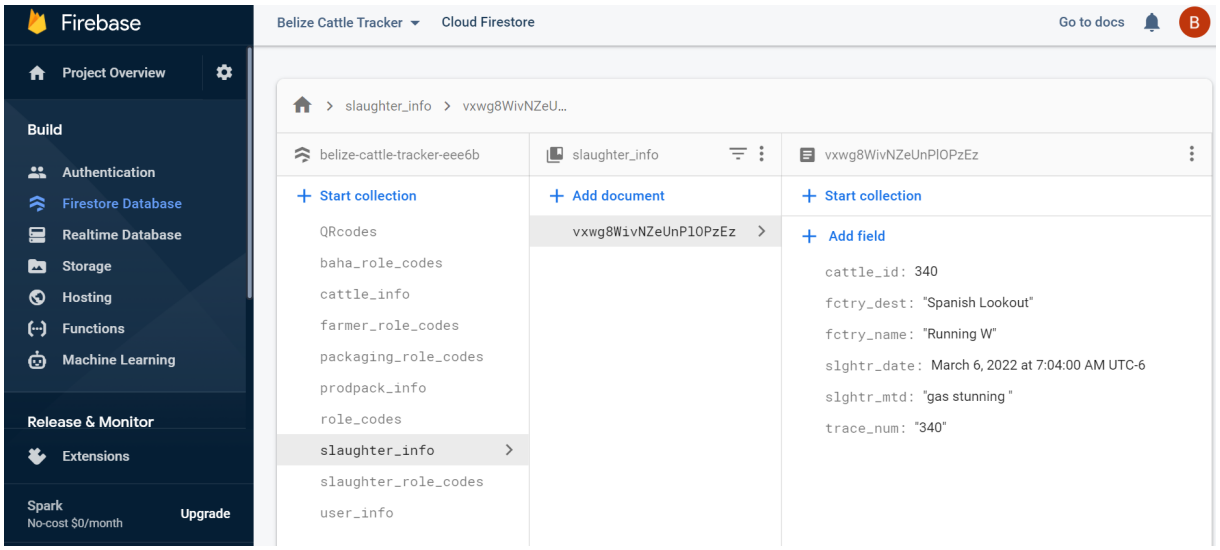
"Schema" for user_info document to store login information.



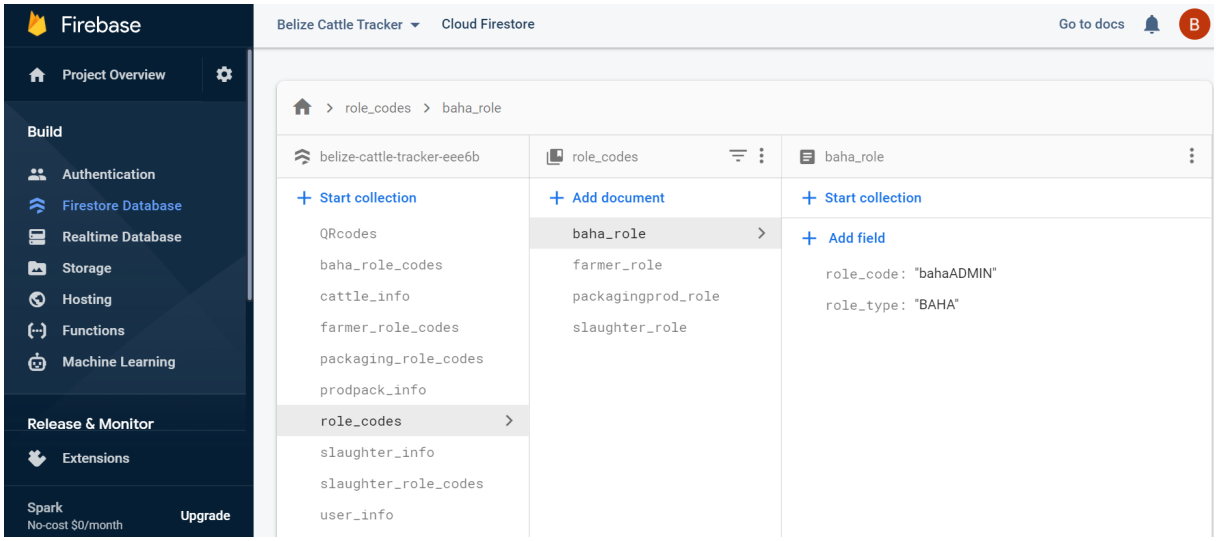
"Schema" for slaughter_role_codes document to store unique role codes for different slaughterhouse managers who have signed up.



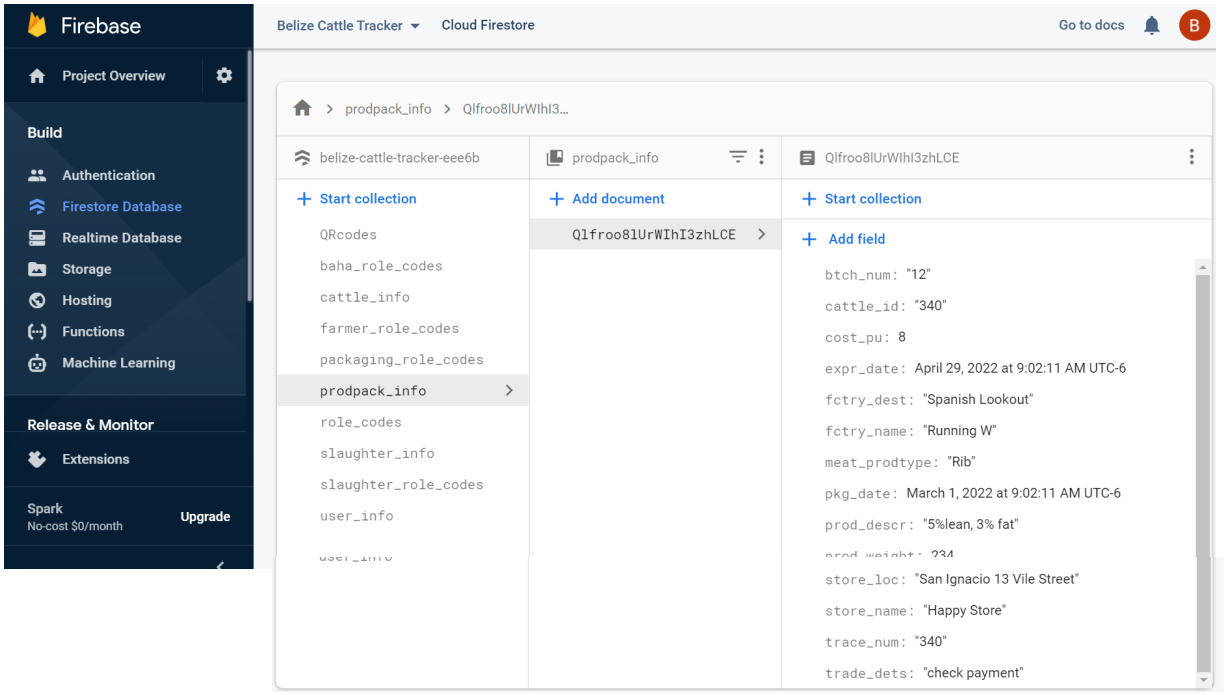
“Schema” for slaughter_info document to store slaughter info that was added on to the cattle info.



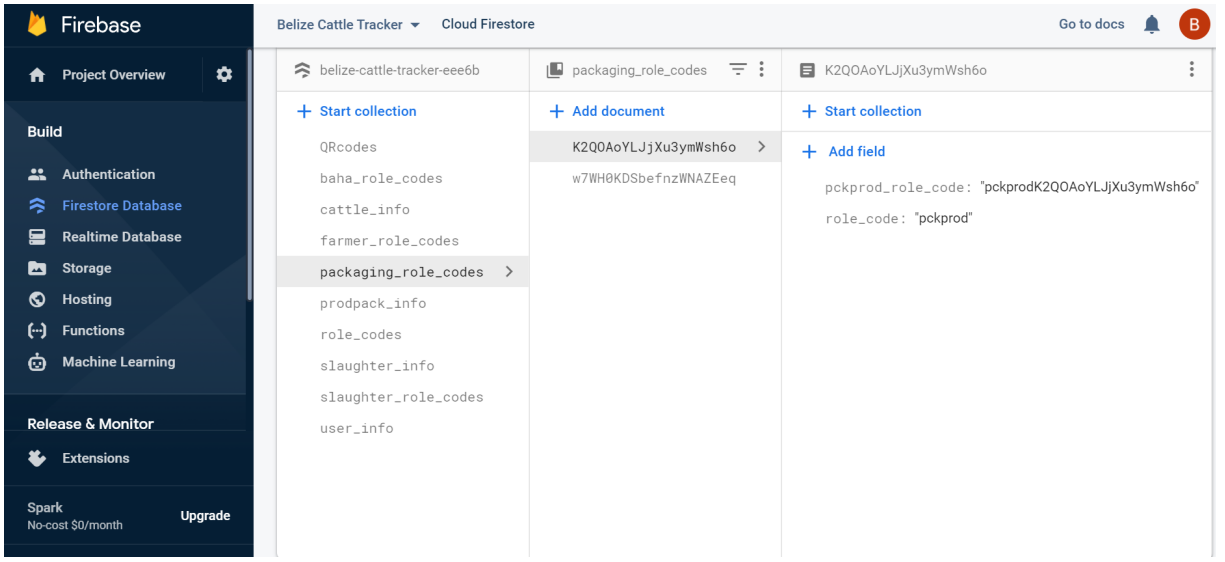
“Schema” for role_codes document to store role types and their code.



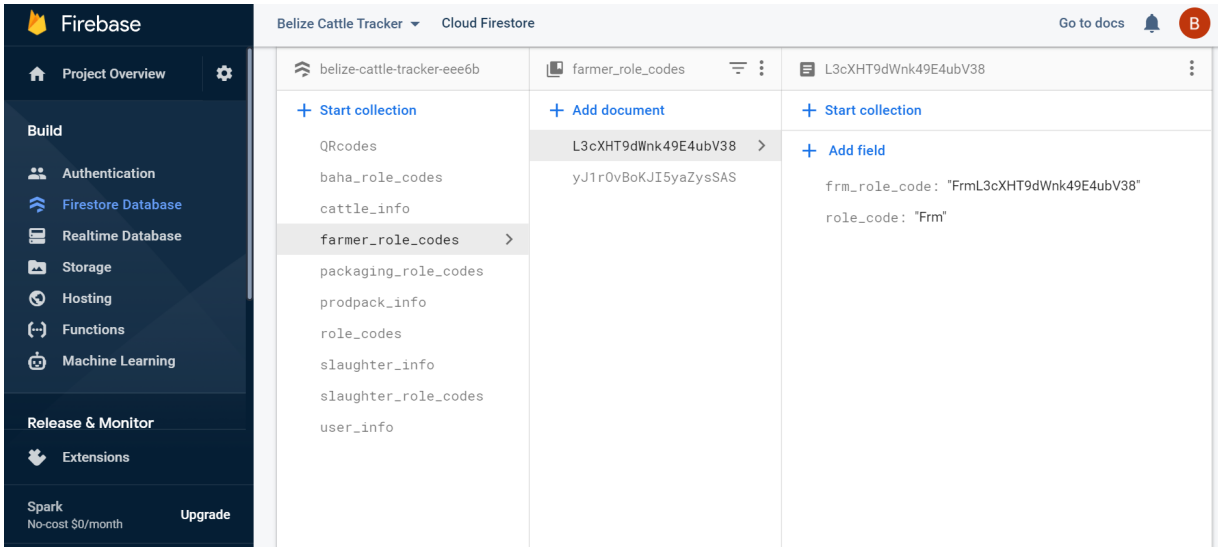
"Schema" for prodpack_info document to store the product info that was added on to the cattle and slaughter info.



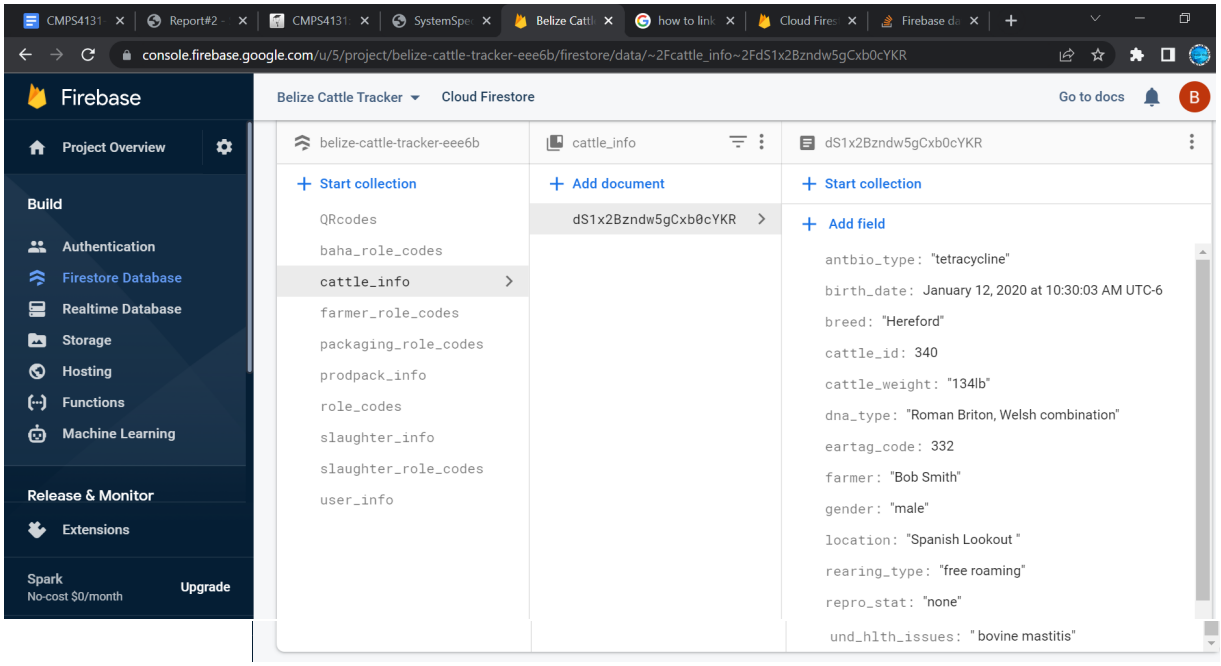
"Schema" for packaging_role_codes document to store the unique role codes for different packaging managers who have signed up.



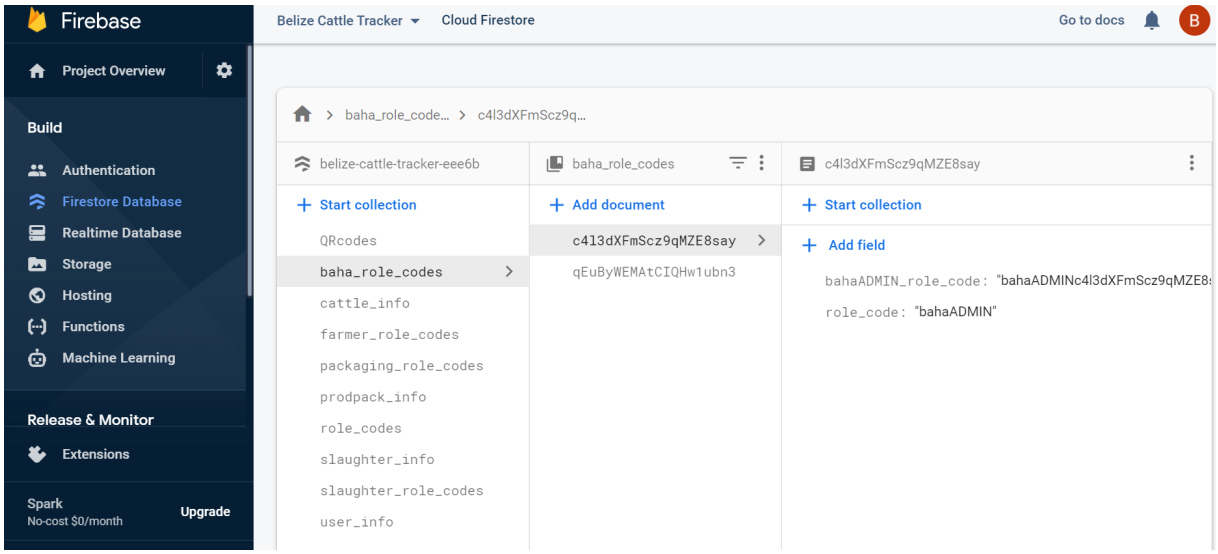
"Schema" for farmer_role_codes document to store the unique role codes for different farmers who have signed up.



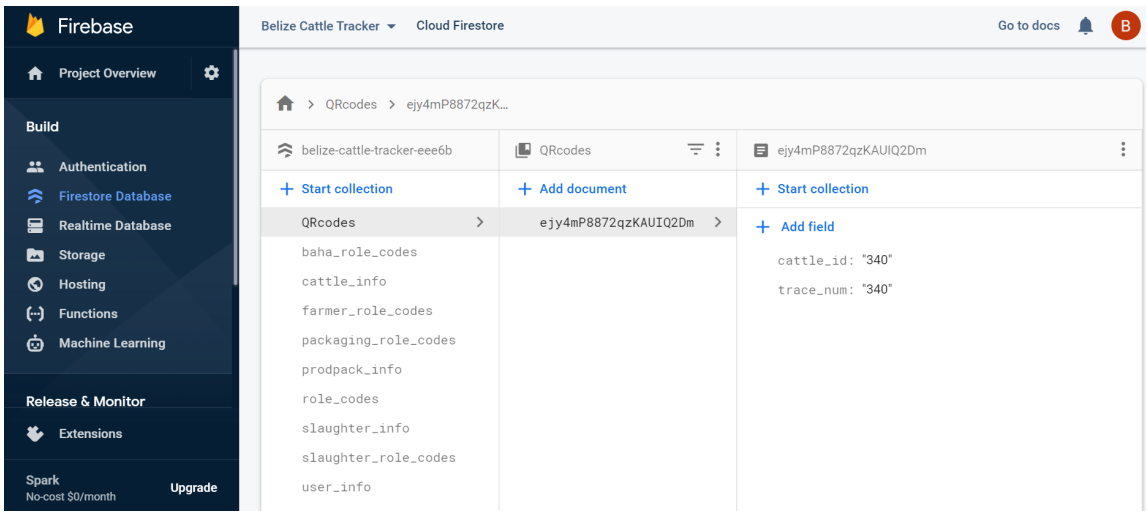
"Schema" for cattle_info document to store the information that was added by the farmer in order to create or update a cattle profile.



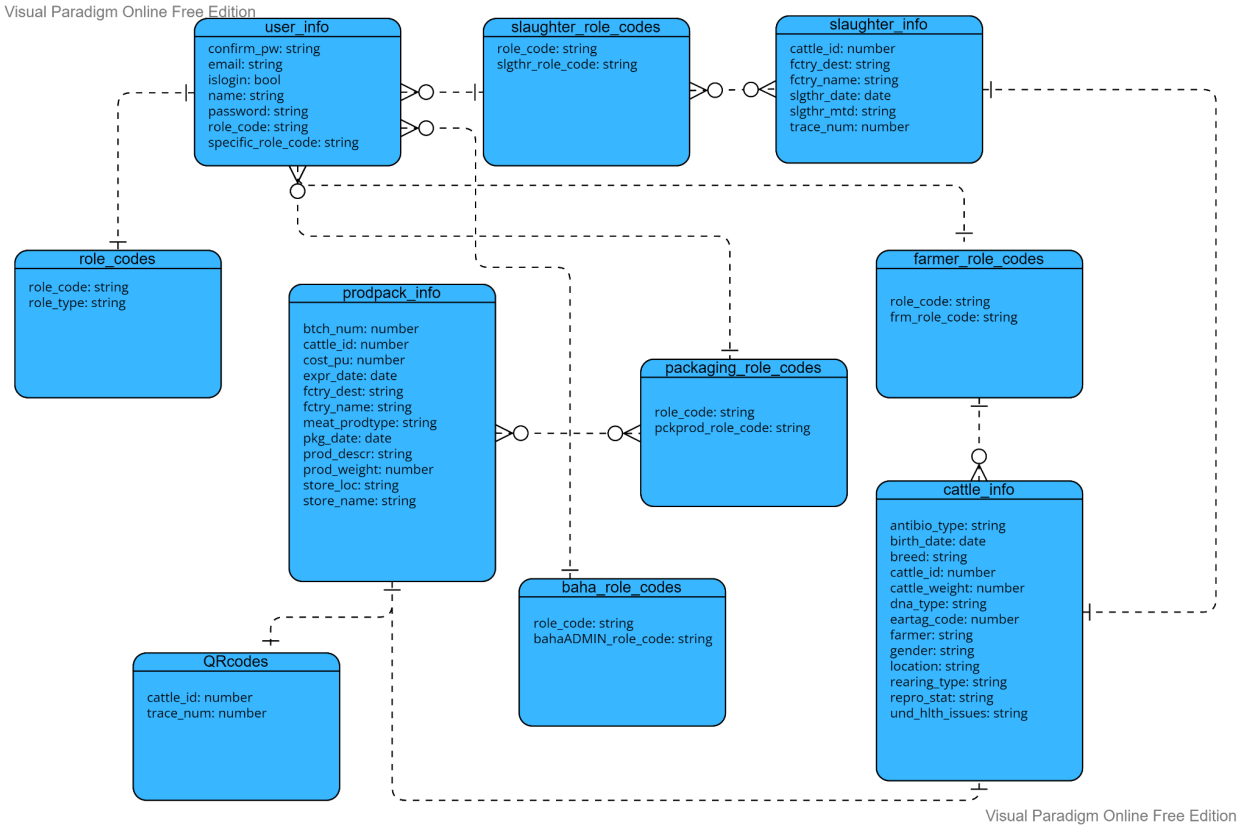
“Schema” for baha_role_codes document to store the unique role codes for the BAHA admins.



“Schema” for QRcodes document to store the uniquecattleid and tracenum for a specific cattle. This table will be used to make the collectionGroup function (in React code) easier to pull information for a specified trace number. Eg: packaging. cattle, and slaughter info that has the same tracenum value will be pulled and linked to a generated QR code.



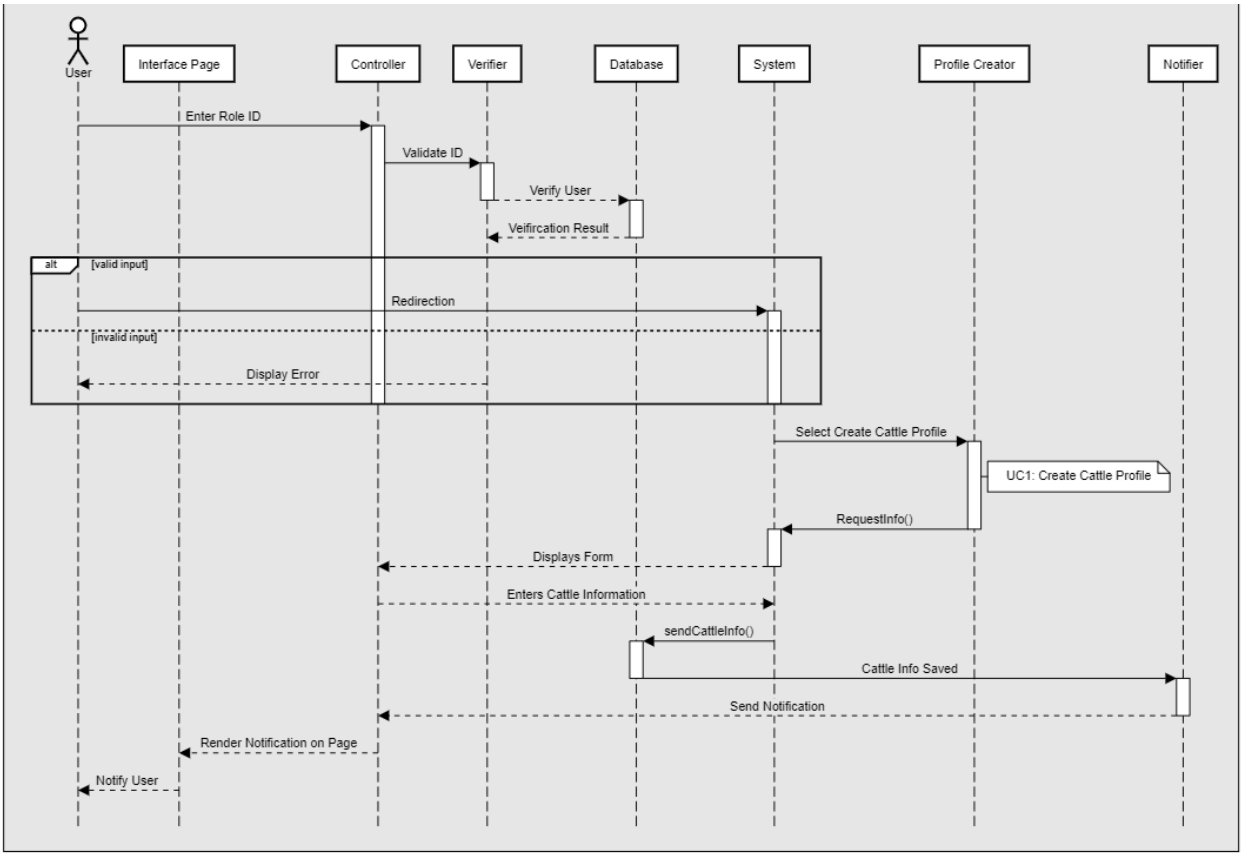
Below is a clear representation of what the **database schema** works/looks like with the assumption that the Firestore Database is in terms of a relational SQL based database.



Interaction Diagrams

Sequence Diagram for UC 1 - Create Cattle Profile

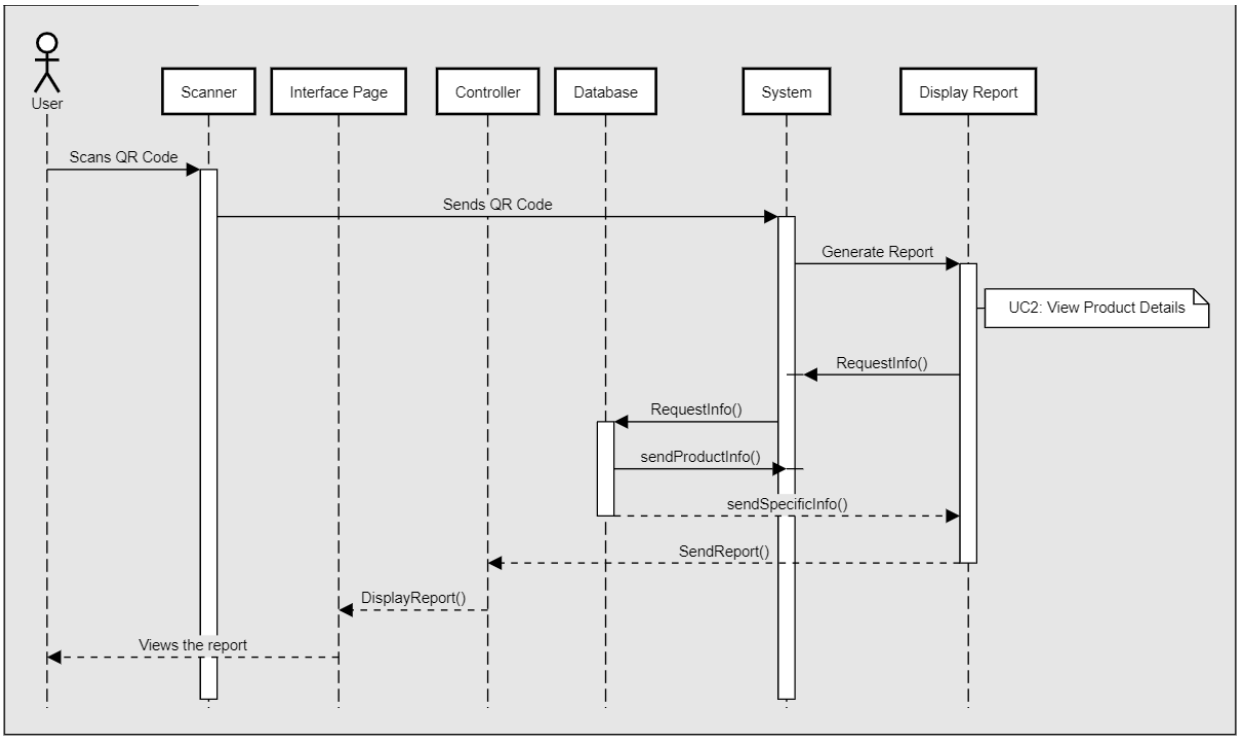
Create Cattle Profile is responsible for creating and notifying a farmer of a successful creation of a cattle profile. A farmer can enter their ID to verify their role and specify their portal. The CONTROLLER then sends the farmer role ID to the VERIFIER to confirm the farmer ID from the database. When the farmer ID is retrieved from the database, the system will be notified and the FARMERPORTAL will be displayed. From there, the farmer can select CREATECATTLEPROFILE which would initiate the system's PROFILECREATOR to display the CATTLEBIRTHINFOFORM. The farmer can then enter and save the CATTLEBIRTHINFO which would push the data to the DATABASE. After the information is pushed to the database, the farmer receives a successful confirmation message from the NOTIFIER, letting them know that the profile was successfully created. The concept(s) (User Interface, Database, Product Profile) evolved into (Interface Page, Controller, Verifier, Database, System, Profile Creator and Notfier) respectively.



Sequence Diagram for UC 2 - View Product Details

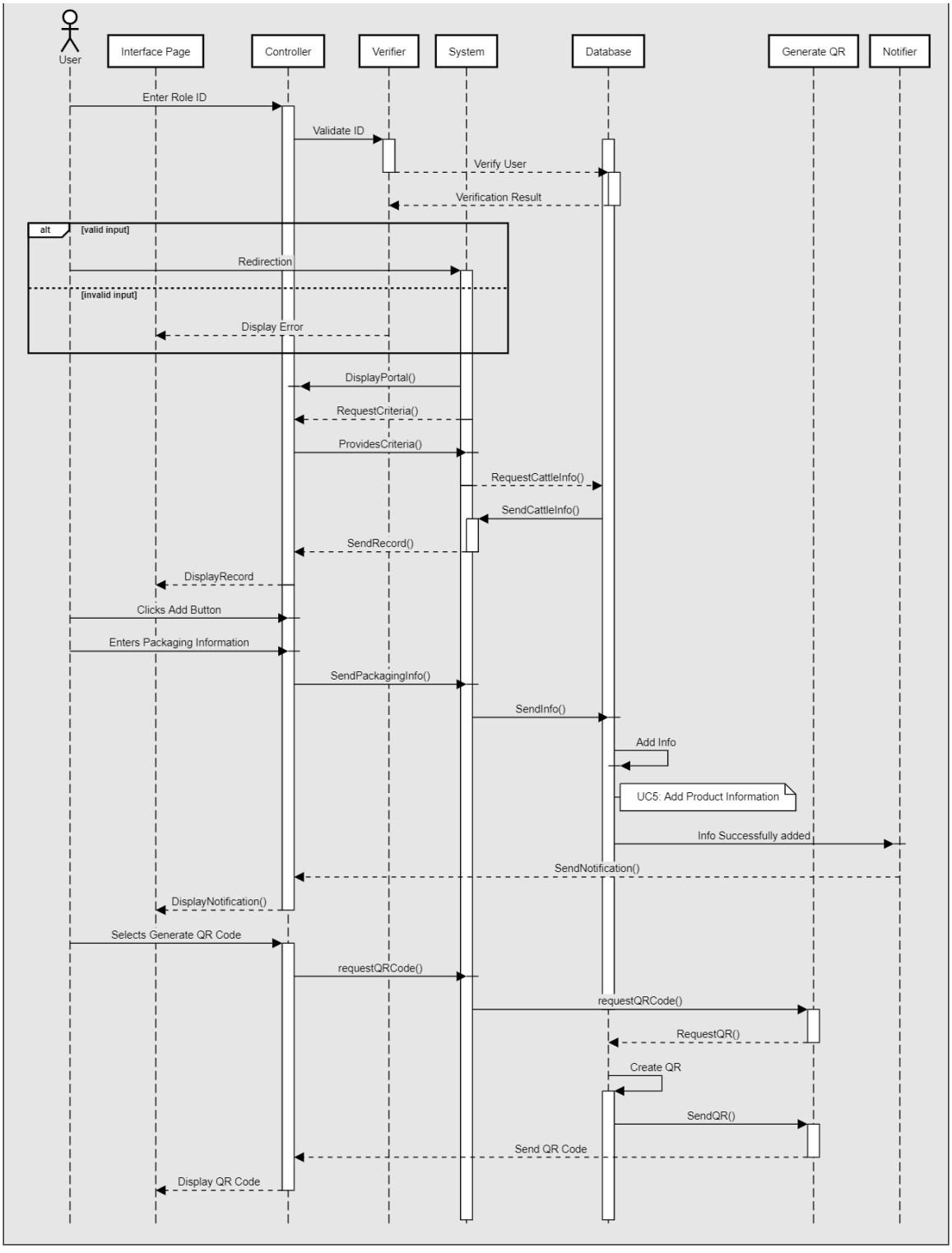
View Product Details is responsible for displaying the product details file to the customer when the QR code is scanned. A customer can use any form of smart device medium that has a camera with the ability to scan QR codes. When the customer scans the QR code, the system gets prompted to GENRATEREPORT. From there, the system will invoke DISPLAYREPORT, which will query the DATABASE for the required information. After retrieving the data from the DATABASE, the DISPLAYREPORT delivers the file containing the necessary records to the CONTROLLER, allowing the INTERFACEPAGE to DISPLAYREPORT. Finally, the consumer may access the PRODUCTDETAILSREPORT via the INTERFACEPAGE. The concept(s) (User

Interface, Database, Product Profile) evolved into (Scanner, Interface Page, Controller, Database, System, and Displasy Report) respectively.



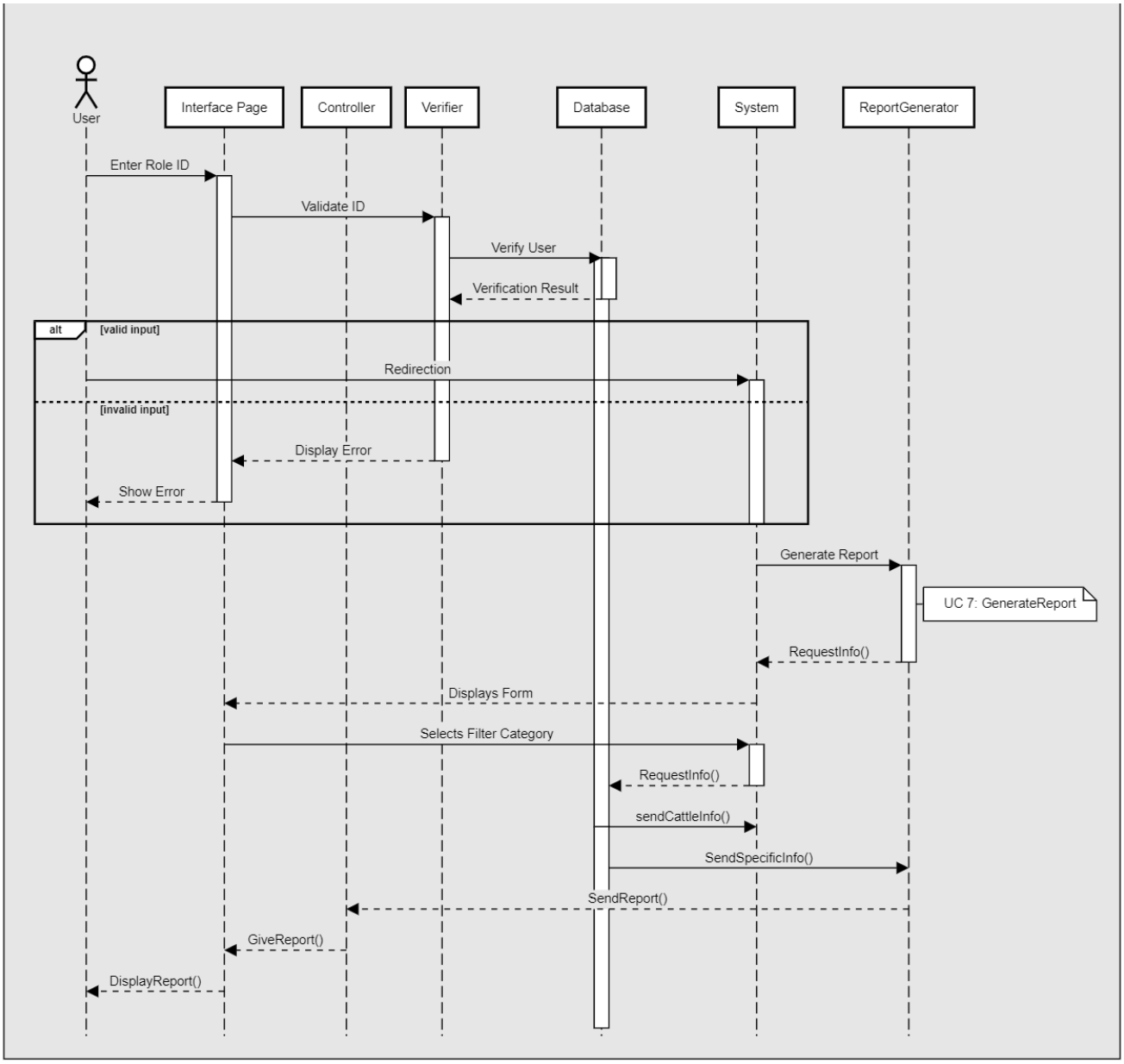
Sequence Diagram for UC 5 - Add Product Information

Add Product Information is responsible for adding and notifying a package manager of a successful addition of packaging information to a cattle profile. A packaging manager can enter their ID to verify their role and specify their portal. The CONTROLLER then sends the packaging manager role ID to the VERIFIER to confirm the packaging manager's ID from the database. When the packaging manager ID is retrieved from the database, the system will be notified and the PACKAGING PORTAL will be displayed. From there, the packaging manager can search for a specific cattle ID which will enable the SYSTEM to request for the cattle information from the DATABASE. The DATABASE will send the requested information to the SYSTEM, which will send the information to the INTERFACEPAGE where it will be displayed to the USER. The packaging manager can then add packaging information to that cattle profile and save the PACKAGINGINFO which would push the data to the DATABASE. After the information is pushed to the database, the packaging manager receives a successful confirmation message from the NOTIFIER, letting them know that the information was successfully added. The packaging manager can then select to generate a QR code, this would prompt the SYSTEM to REQUESTQRCODE. The SYSTEM will invoke GENERATEQR to request a QR code from the DATABASE. The DATABASE will create the QR code and send it to GENERATE QR which will send the QR code to the CONTROLLER. The CONTROLLER will send it to the INTERFACEPAGE where it will be displayed to the packaging manager/USER. The concept(s) (User Interface, Database, Product Profile) evolved into (Interface Page, Controller, Verifier, System, Database, Generate QR and Notfier) respectively.


User
Interface Page
Controller
Verifier
System
Database
Generate QR
Notifier
Enter Role ID
Validate ID
Verify User
Verification Result
alt
[valid input]
Redirection
[invalid input]
Display Error
DisplayPortal()
RequestCriteria()
ProvidesCriteria()
RequestCattleInfo()
SendCattleInfo()
SendRecord()
DisplayRecord
Clicks Add Button
Enters Packaging Information
SendPackagingInfo()
SendInfo()
Add Info
UC5: Add Product Information
Info Successfully added
SendNotification()
DisplayNotification()
Selects Generate QR Code
requestQRCode()
requestQRCode()
RequestQR()
Create QR
SendQR()
Send QR Code
Display QR Code

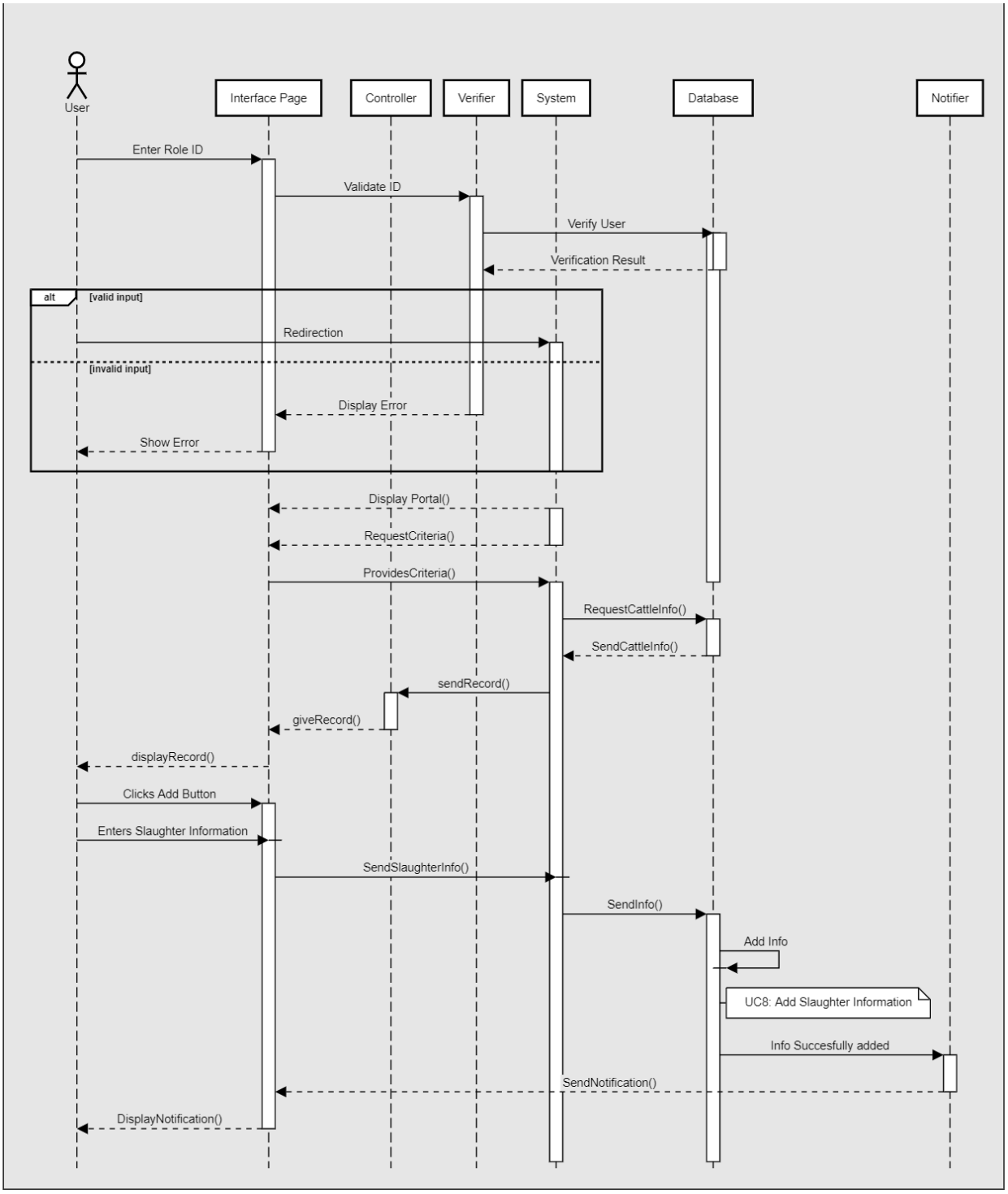
Sequence Diagram for UC 7 - Generate Report

Generate Report is responsible for generating a report of an existing cattle profile. A BAHA Admin can enter their ID to verify their role and specify their portal. The CONTROLLER then sends the BAHA admin role ID to the VERIFIER to confirm the admin ID from the database. When the BAHA admin ID is retrieved from the database, the system will be notified and the BAHAADMIN PORTAL will be displayed. From there, the admin can select GENERATEREPORTS which would initiate the system's REPORTGENERATOR to display the GENERATEREPORTFORM. The USER will select the search criteria to find a cattle profile. This will prompt the system to request the information from the DATABASE. The DATABASE sends the specific cattle information to the REPORTGENERATOR, which will send the report to the CONTROLLER. The CONTROLLER will give the report to the INTERFACEPAGE where it will be displayed to the BAHA Admin/USER. The concept(s) (User Interface, Database, Cattle Profile) evolved into (Interface Page, Controller, Verifier, Database, System, Report Generator) respectively.


User
Interface Page
Controller
Verifier
Database
System
ReportGenerator
Enter Role ID
Validate ID
Verify User
Verification Result
alt
[valid input]
Redirection
[invalid input]
Display Error
Show Error
Generate Report
UC 7: GenerateReport
RequestInfo()
Displays Form
Selects Filter Category
RequestInfo()
sendCattleInfo()
SendSpecificInfo()
SendReport()
GiveReport()
DisplayReport()

Sequence Diagram for UC 8: Add Slaughter Information

Add Slaughter information allows a slaughterhouse manager to add slaughter information to previously uploaded CATTLEBIRTHINFO and to provide a completion message when the information is successfully entered. A slaughterhouse manager can input their ID to confirm their role and choose their portal. The slaughter role ID is subsequently sent to the VERIFIER, who confirms the slaughter manager ID from the DATABASE. The SYSTEM will be informed and the SLAUGHTERHOUSEPORTAL will be displayed when the slaughter manager ID is obtained from the database. The slaughterhouse manager may then search for a cattle profile, which will cause the system to request a criterion (specific cattleID). When a SEARCHREQUEST is issued, it offers the criteria to the SYSTEM, which will obtain the necessary information from the DATABASE associated with the cattleID. The DATABASE then returns the cattle information to the SYSTEM, which then transmits the records to the CONTROLLER. After that, the CONTROLLER will DISPLAYCATTLEINFO on the INTERFACEPAGE where the slaughterhouse manager can press the ADDBUTTON and input SLAUGHTERINFO. After entering this, the SLAUGHTERINFO is stored to the SYSTEM and DATABASE (during this process UC8 is initiated). The NOTIFIER will be notified after the SLAUGHTERINFO has been successfully uploaded to the DATABASE. The slaughterhouse manager receives a successful confirmation message from the NOTIFIER on the INTERFACEPAGE, indicating that the process had completed successfully. The concept(s) (User Interface, Database, Product Profile) developed into (Interface Page, Controller, Verifier, Database, System, and Notifier) respectively.



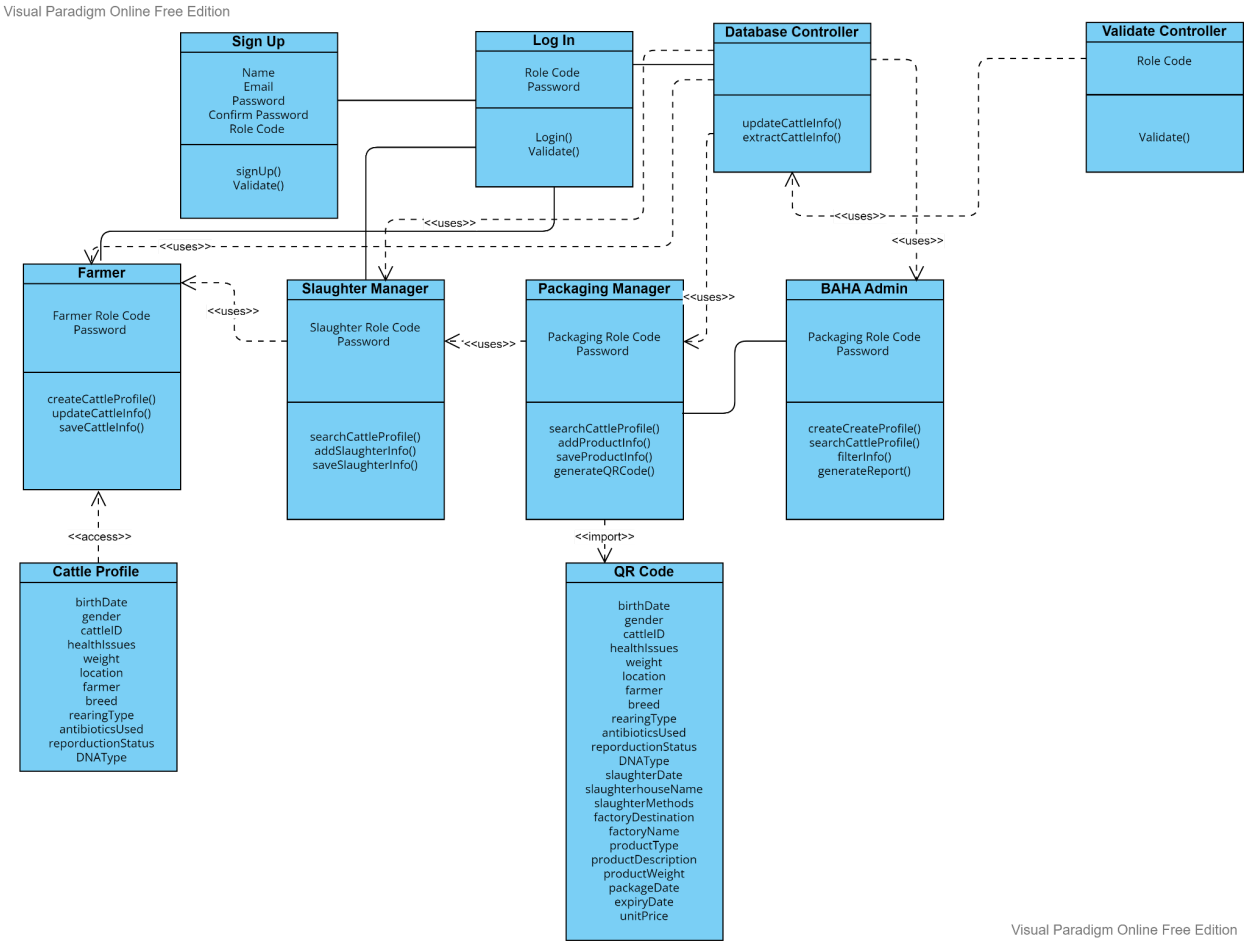
There were no changes made to our system's interaction diagrams as it was the final structure we opted to employ. Our system is built on high coupling and cohesiveness. Because of the extent to

which our components are coupled to one another, our system has strong coupling. For example, if our component that adds cattle information fails, the other components, such as those that add product and slaughter information, would be rendered inoperable since they are heavily reliant on the component that supplies the initial information required for them to function. Our system also has high cohesion because it has classes that execute a specified function rather than functions with no commonality. For example, our profile class is dedicated to providing information for various roles in our system; it also includes the opportunity for a user to change their password, but the procedure is handled by a separate class.

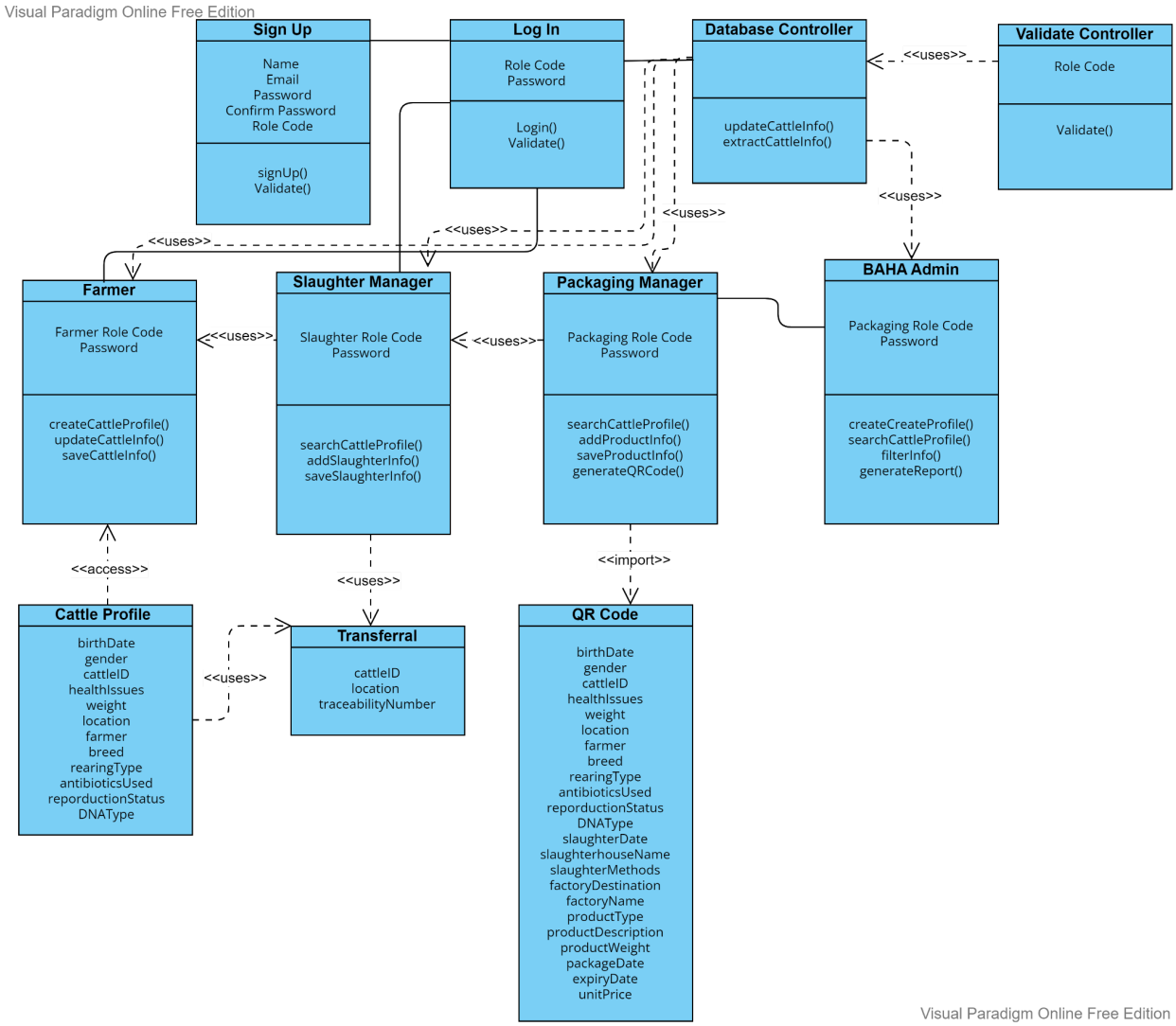
Class Diagram and Interface Specification

Class Diagram

Before Changes



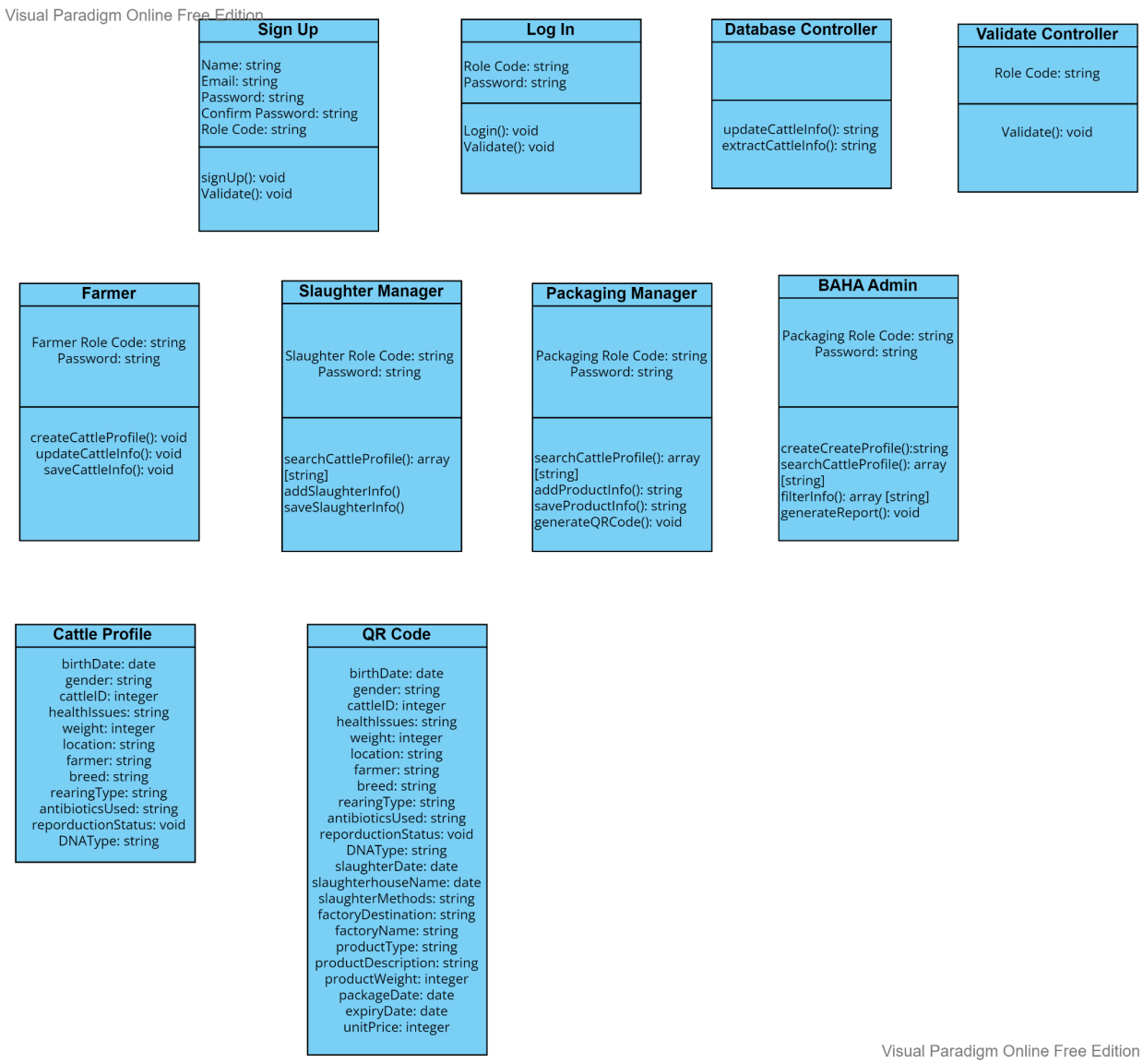
After Changes



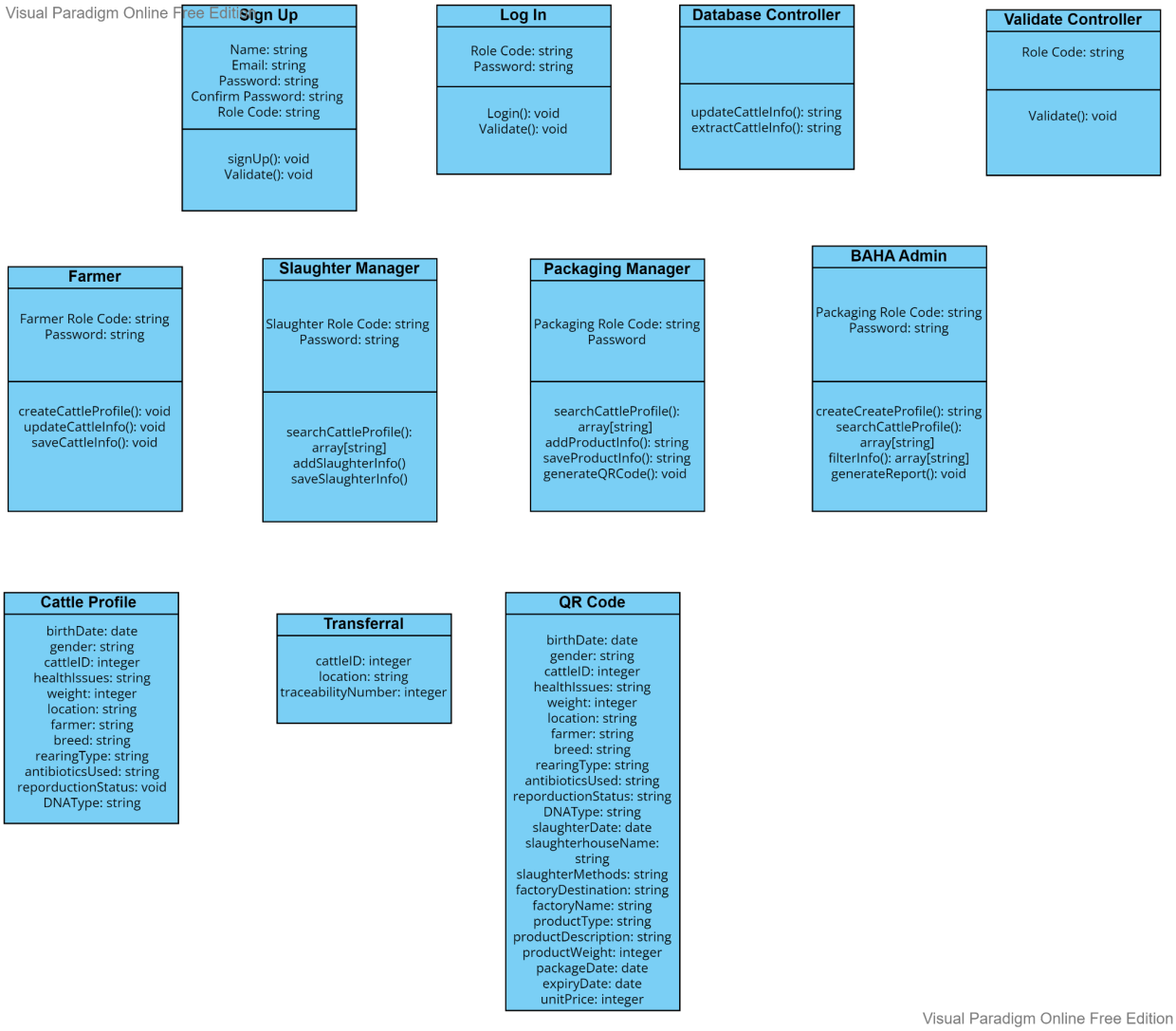
Explanation of Class Diagram Evolution: With the growth of the class diagram for our system, there were only minor alterations. The single addition was the transferral class, which was required for both the birth and slaughter classes to help with the process of passing information to the next stage and also to help with the system's safety and dependability (to not allow for any changes to be made when a cattle and its information is at a next stage).

Data Types and Operation Signatures

Before Changes



After Changes



OCL Constraint Specification

Contract Name	Farmer
Cross Reference	***UC 1 - Create Cattle Profile***
Invariants	SearchbyCattleID(), AddCattleInfo()
Precondition	None
Post condition	CattleProfileCreatedSuccessfully(), TerminateEditAccess()

Contract Name	Slaughter Manager
Cross Reference	***UC 8: Add Slaughter Information***
Invariants	SearchbyCattleID(), AddCattleInfo()
Precondition	CreateCattleProfile()
Post condition	TerminateEditAccess()

Contract Name	Packaging Manager
Cross Reference	***UC 5 - Add Product Information***
Invariants	SearchbyCattleID(), AddCattleInfo()
Precondition	CreateCattleProfile(), AddSlaughterInformation()
Post condition	TerminateEditAccess()

Contract Name	BAHA Admin
Cross Reference	***UC 7 - Generate Report***
Invariants	SearchbyCattleID()
Precondition	CreateCattleProfile(), AddSlaughterInformation(), AddProductInformation()
Post condition	None

Traceability Matrix

Domain Concepts	Sign Up	Login	Database-Controller	Validate-Controller	Farmer	Slaughter Manager	Packaging Manager	BAHA Admin	Cattle Profile	QR Code
Controller	x	x	x	x	x	x	x	x	x	x
Verifier	x	x		x					x	x
PortalDisplay	x	x	x	x	x	x	x	x	x	x
ProfileCreator			x		x				x	
EnterInfo					x	x	x	x	x	x
SaveInfo			x		x	x	x	x		
DatabaseConnection	x	x	x		x	x	x	x		
Notifier			x						x	x
Scanner							x			
QR Code							x		x	x
DisplayReport			x				x		x	x
SearchRequest					x	x	x	x		
DatabaseRetreive			x							
AddInfo			x							
DatabaseStore			x							
GenerateQR							x	x		x
QRRequest			x							x
QRDataRetreive			x							x
DisplayQR							x	x		x
ReportGenerator			x					x		
SearchFormDisplay				x	x	x	x	x		
Filter Search			x					x		
SearchRequest				x	x	x	x	x		
CriteriaSearch				x	x	x	x	x	x	
Select ID				x	x	x	x	x	x	
DisplayRecords				x	x	x	x	x		x

Sign Up -

- Deals with the sign up functionalities when a manager signs up to use the Website Application.
- Not Derived from any of the Domain Concepts however it works alongside VERIFIER in assisting to ensure that the user enters the right credentials. Also works alongside INTERFACE PAGE which collects user inputs and displays the page for viewing.

Log in -

- Deals with the login functionalities when a manager signs in to perform an operation
- Not Derived from any of the Domain Concepts however it works alongside VERIFIER in assisting to ensure that the user enters the right credentials. Also works alongside INTERFACE PAGE which collects user inputs and displays the page for viewing.

Database Controller

- Evolved from the Concept DATABASECONNECTION which handles the connection between the other Domain concepts and the database

Validate Controller

- Evolved from the Concept VERIFIER which handles the validation of user role ID and password by comparing values in the Database with the User's Input.

Farmer

- Concerned with handling different functionalities such as CREATECATTLEPROFILE(), which creates a new profile for a newly registered cattle, as well as

UPDATECATTLEINFO(), which updates the information for an already created Cattle profile and lastly SAVECATTLEINFO(), which saves the newly updated information in the database.

Slaughter Manager

- Concerned with handling different functionalities such as SEARCHCATTLEPROFILE(), which searches for a cattle profile using a unique identifier such as an ID, as well as ADDSLAUGHTERINFO(), which adds information about the slaughter of an already created cattle profile. Works after UPDATECATTLEINFO() has been implemented by the Farmer and lastly SAVESLAUGHTERINFO(), which saves the newly updated information in the database.

Packaging Manager

- Concerned with handling different functionalities such as SEARCHCATTLEPROFILE(), which searches for a cattle profile using a unique identifier such as an ID, as well as ADDPRODUCTINFO(), which adds information about the packaging of an already created cattle profile. Works after ADDSLAUGHTERINFO() has been implemented by the Slaughter Manager, another functionality that is being performed by this class is SAVEPRODUCTINFO(), which saves the newly updated information in the database and lastly GENERATEQRCODE(), which simply generates a QR Code from the cattle the user has saved in the Database.

Cattle Profile

- Concerned with the handling and storing of several details about the cattle that is being created by the Farmer

QR CODE

- Concerned with the handling and storing of several details about the cattle that is being created by all the managers. This information will be displayed to the customer when they scan the QR Code.

Algorithms and Data Structures

Algorithms

The Belize Cattle Tracker demonstrates mainly search and storing algorithms. Unlike other applications, the algorithms done by the Belize Cattle Trackers don't require complex formulas or arithmetic operations.

I. Create Profile

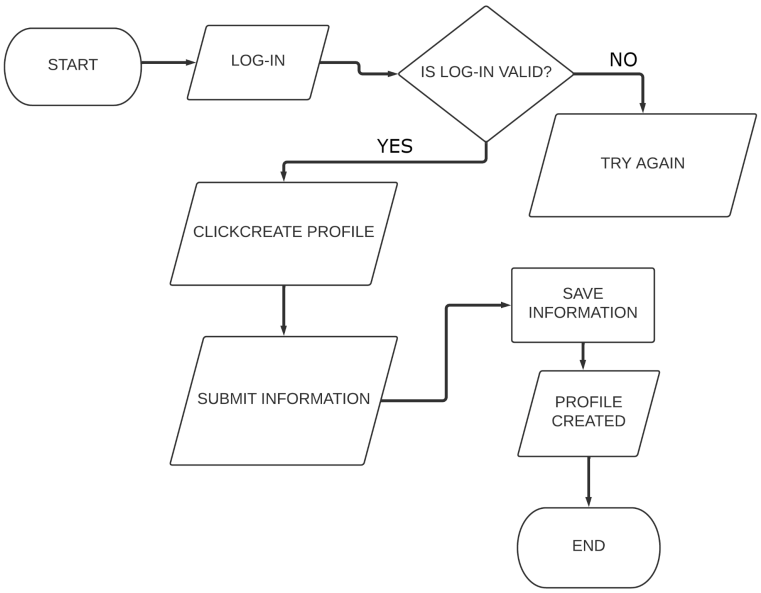


Figure 1. CreateProfile activity diagram

II. view Product Details

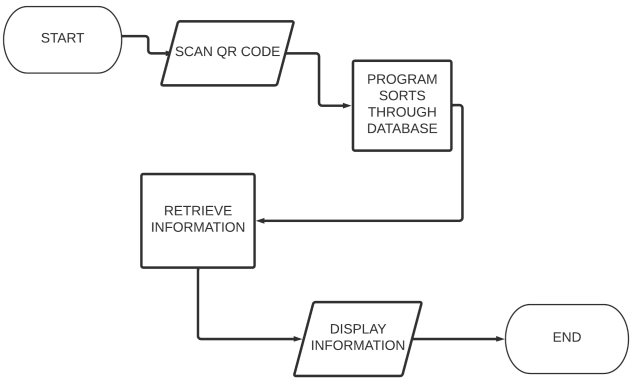


Figure 2. viewProductDetails activity diagram

III. Add Product Information

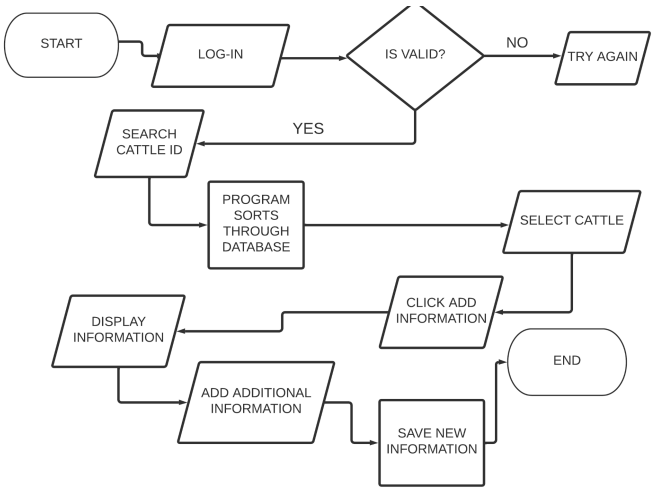


Figure 4. addProductInformation activity diagram

IV. Generate Report

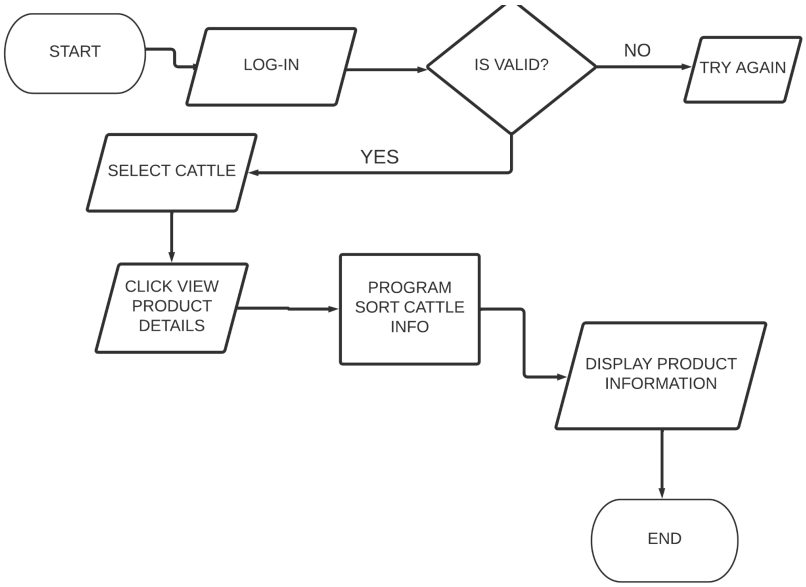


Figure 5. generateReport activity diagram

V. Add Slaughter Information

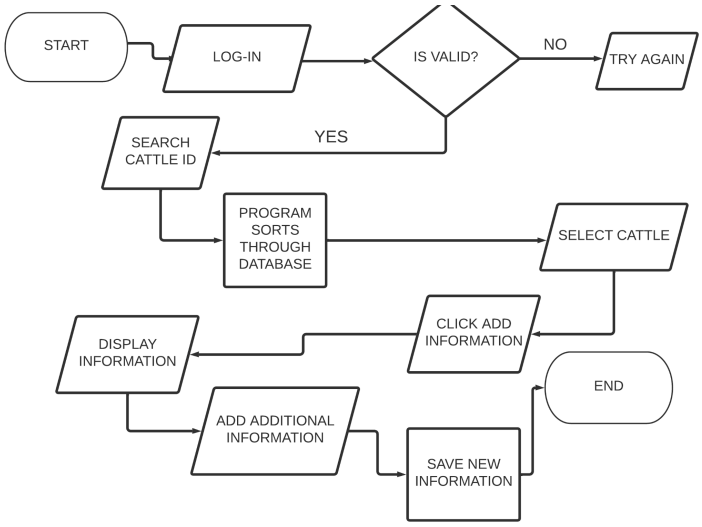


Figure 6. addSlaughterInformation activity diagram

Data Structures

The data structure that the Belize Cattle Tracker utilizes the most are arrays. Arrays are data structures that consist of a collection of elements, which are then identified by their index or key. In our case, the array's main purpose will be to hold whatever information is pulled from the database during searches. Not only do arrays provide much more flexibility in Javascript, in arrays:

- ArrayList can expand and contract dynamically.
- Elements can be added to or removed from a specific place.
- Data can be easily accessible in Array.
- Methods for Array in Javascript are very useful, for instance filter, map, pop, push, and more.
- Arrays in Javascript can contain a mix of different data types.

Concurrency

This system will use multithreading in the event that more than one user is on the website at the same time. Due to the fact that users would be able to change and fully interact with the system all at the same time, concurrency would be needed as these would be a variety of different functions taking place at the same time. By implementing multi-threading, the users would be able to simultaneously work on the system at the same time. A Farmer can be logged in at the same time as a Packaging manager and make real time changes to the database and website. In order to handle and synchronize threads and requests in real time, The server/database would be responsible for doing so. The most vital features that would need to be synchronized in our

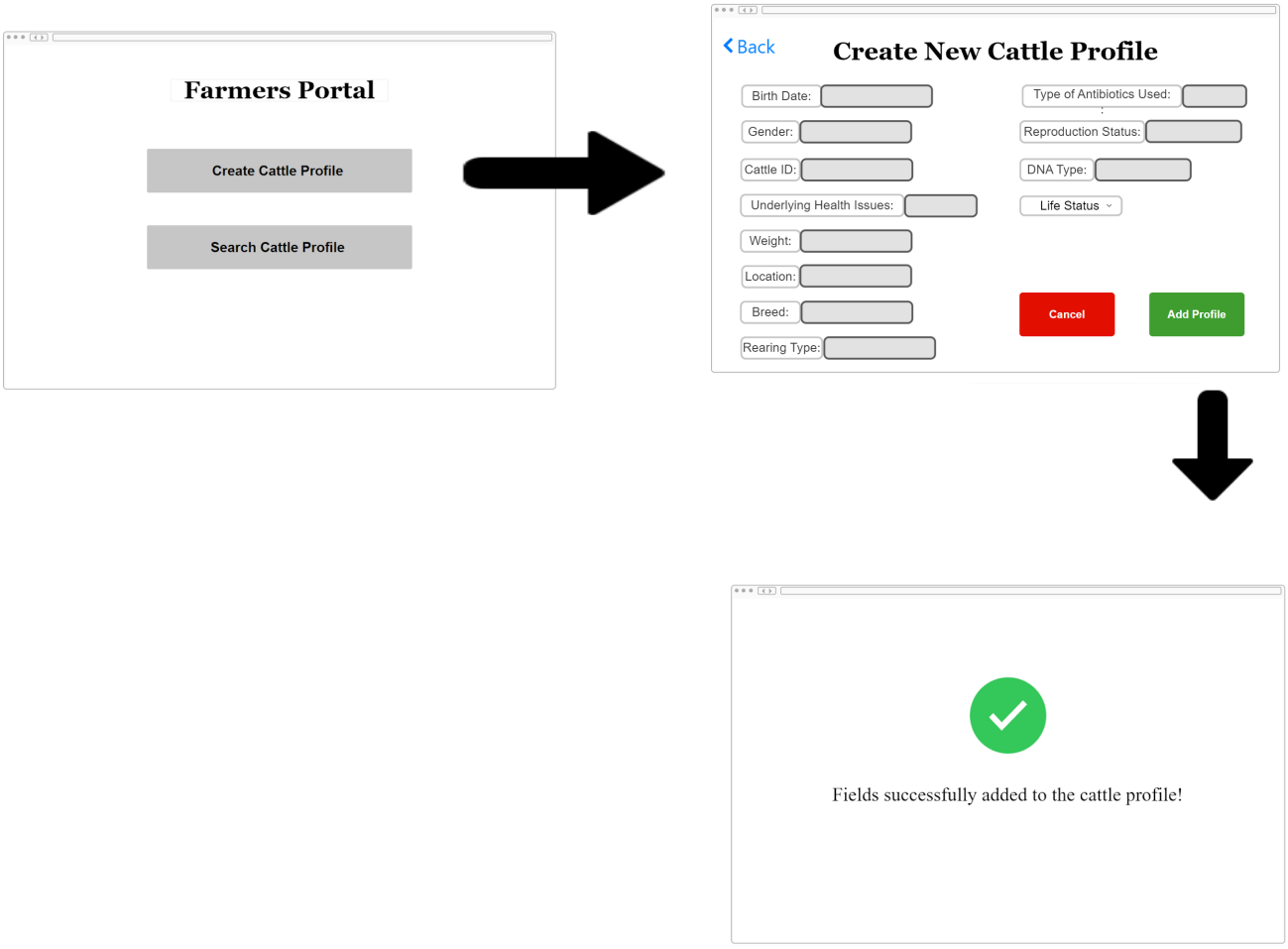
system would be the Different Roles performing insertions into the database in real time as this is vital to ensuring that there is no room for error.

User Interface Design and Implementation

Modifications and Implementation of Screens

Screen MockUps

UC-1 Create Cattle Profile

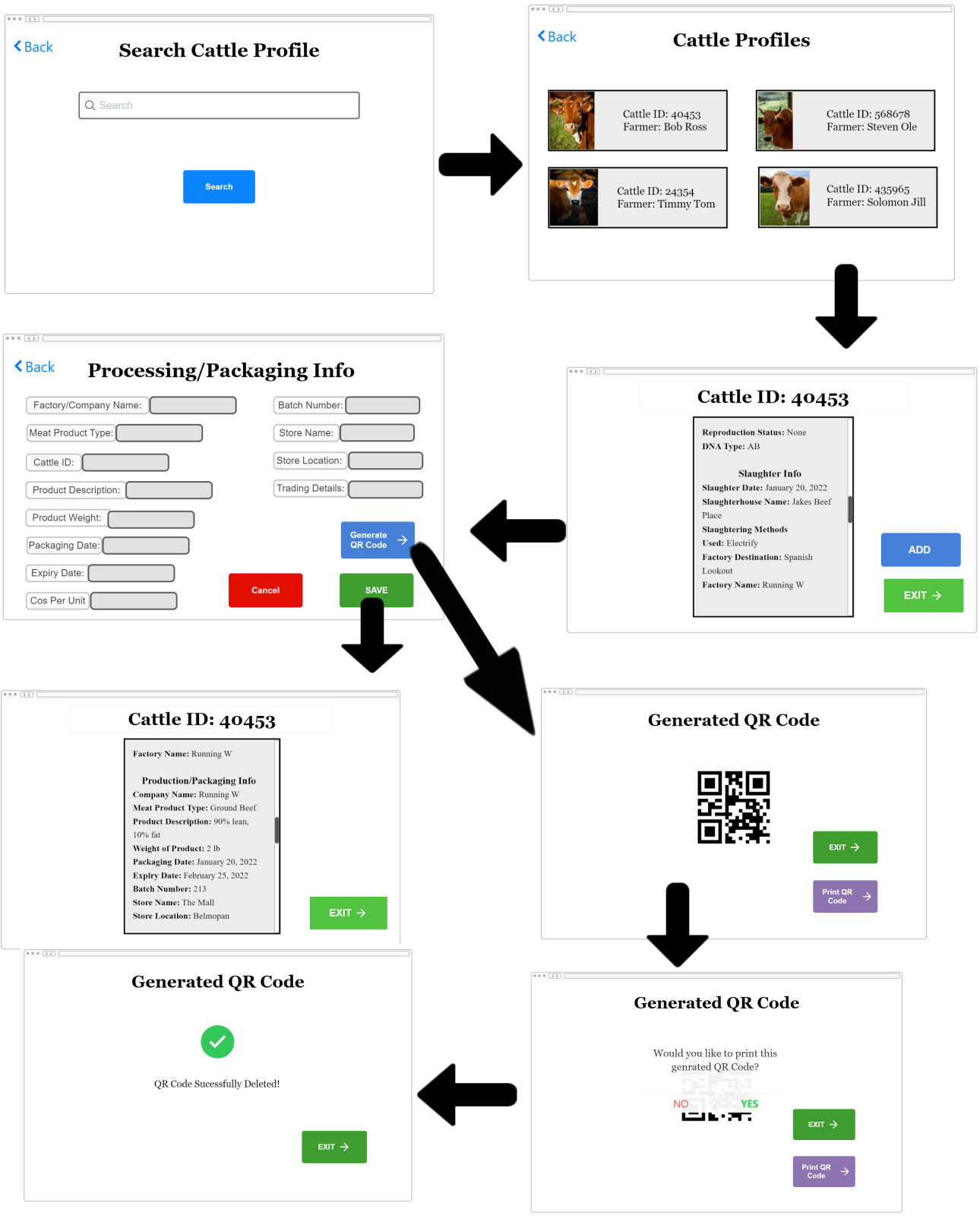


For the Create New Cattle Profile screen for UC1, a “Cancel” button was added to the interface so that farmers/BAHA can terminate their operations in case they do not want to enter in the records. A “Back” button was also added to the interface to allow the farmer to go back to a previously opened interface. Adding these assists with abiding by the user control and freedom heuristic “Users often perform actions by mistake. They need a clearly marked "emergency exit" to leave the unwanted action without having to go through an extended process (Nielson, 2020).” These types of heuristics should always be considered when developing the screens that store data because users can make mistakes sometimes on the system and they should have a way to exit the problem. Another heuristics that this abides by is the consistency and standards heuristic, “Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform and industry conventions (Nielson, 2020)..” Systems should always include a “Cancel” operation if there is an “Add” operation because industry standard applications always have to consider that users can make mistakes when operating with the system.

UC-8 Add Slaughter Information

For the Search Cattle Profile screen for UC8, a “Search” button was added to the interface so that admin users (farmers, BAHA, packaging and slaughterhouse managers) can click on the button to search for a cattle profile by its ID. In case they do not want to use their “Enter” button on their keyboard, this can serve as another option to provide user control and freedom. A “Back” button was also added to the Cattle Profiles interface and Slaughter Info interface to allow the farmer to go back to a previously opened interface. Adding these assists with abiding by the user control and freedom heuristic “Users often perform actions by mistake. They need a clearly marked "emergency exit" to leave the unwanted action without having to go through an extended process (Nielson, 2020).” Additionally, once again, to abide by the consistency and standards heuristic. a “Cancel” button was added to the Slaughter Info interface so that slaughterhouse managers can terminate their operations in case they do not want to enter in the records.

UC-5 Add Product Information

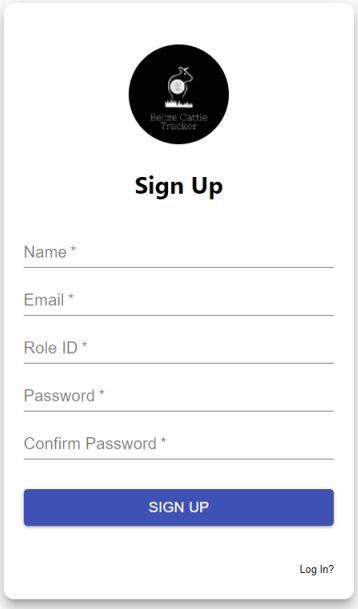


For the Search Cattle Profile screen for UC8, a “Search” button was added to the interface so that admin users (farmers, BAHA, packaging and slaughterhouse managers) can click on the button to search for a cattle profile by its ID. In case they do not want to use their “Enter” button on their keyboard, this can serve as another option to provide user control and freedom. A “Back” button was also added to the Cattle Profiles interface and Packaging/Processing Info interface to allow the farmer to go back to a previously opened interface. Adding these assists with abiding by the user control and freedom heuristic “Users often perform actions by mistake. They need a clearly marked "emergency exit" to leave the unwanted action without having to go through an extended process (Nielson, 2020).” Additionally, once again, to abide by the consistency and standards heuristic. a “Cancel” button was added to the Slaughter Info interface so that slaughterhouse managers can terminate their operations in case they do not want to enter in the records.

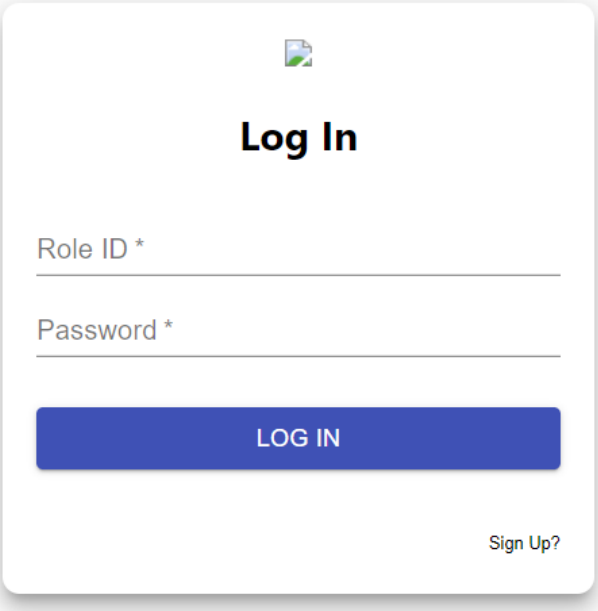
Implemented System Screens from Screen MockUps

Below are the implemented screens created based on the mockups and priority use cases. The screen format/layout is quite different as the MUI library from React was used to make the website mobile and web friendly; should in case the different admins would like to enter in the data from their mobile devices.

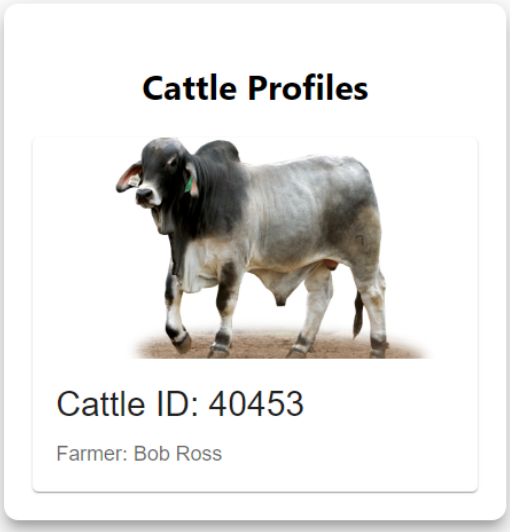
Sign Up Screen



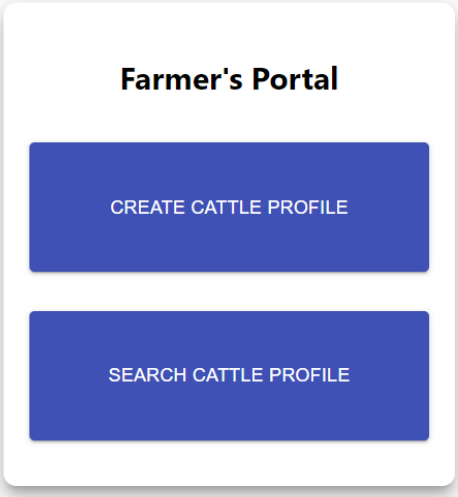
Log In Screen



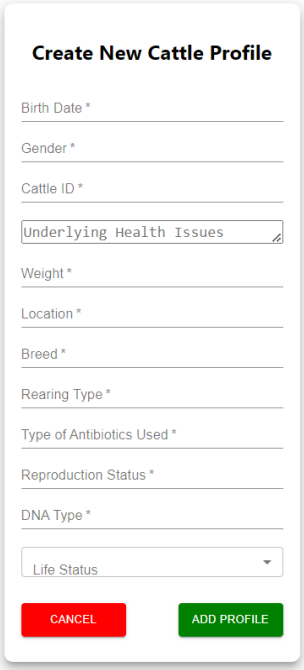
Cattle Profile Screen



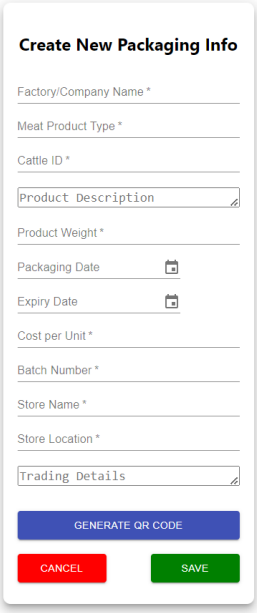
Farmer's Portal Screen



New Cattle Profile Screen

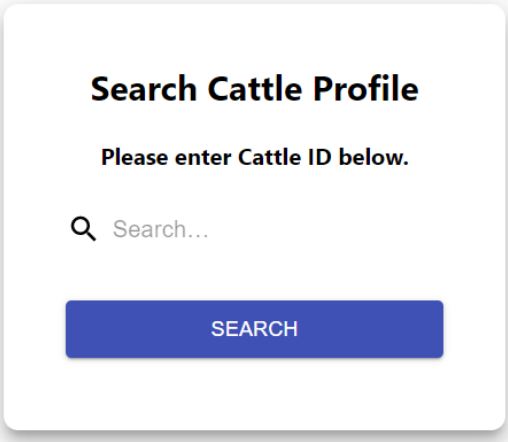


New Packaging Info Screen



New Slaughter Info Screen

Search Cattle Screen



Ease of Use

In terms of "ease of use," the system is based on both "click" and "touch" interactions between the user and the system, making it suitable for both desktop and mobile devices. The Belize Cattle Tracker System is simple to use and gives the user a sense of control, particularly when entering information about cattle or meat products. It allows the user to choose whether to exit a page or proceed to another operation (generating QR code) by clicking or touching buttons that provide the options. The pages have been designed to be as straightforward as possible, with page names, buttons, field labels, and other features condensed to provide users with only the most important information to avoid confusion. All of the system's pages have a consistent design and format (where styling and alignments are of similar structure), and complementary color schemes. This allows users to stay focused on the work at hand and avoid being sidetracked. Instead of having the user complete repeated operations, the system helps them to perform their duties more effectively, and it also offers feedback to them. For example, while generating a cattle profile, the system alerts the user that the profile was successfully created. Users who are familiar or new with mobile or desktop programs should find the Belize Cattle Tracker System to be simple to use, with little to no documentation required. Because the majority of the duties on the system include filling out forms and operating the system by clicking or touching buttons.

Design Tests

Overview of Design Tests

Functions that were used for the test cases were functionals that played a crucial part in our priority requirements and our system as a whole. There will never be a case where 100% of a project can have regression testing fully automated. Programming an automated script to handle the vast number of alternatives would be extremely tough. And that's even if it's possible. It would very certainly take far too long to justify the time savings. For the time being, that is. This is one work that humans are better suited to since they can think, react to, and compute all of the factors that machines can't yet handle.

TestCase1

Test Case	TC-1
Use Case being tested	**CreateCattleProfile**
Criteria for success/fail:	**Test that farmer can create cattle profile and was posted to the database**
Input Data:	**Alphanumeric**
Test Procedure:	
Step 1: Call function AddCattleInfo(info) with valid data.	**Success**

Step 2: Call function AddCattleInfo(info) with invalid data.	**Fail**
Step 3: Call function getCattleInfo(id) with valid data.	**Success**
Step 3: Call function getCattleInfo(id) with invalid data.	**Fail**

This Test Case will mainly test the functions of UC-1 (CreateCattleProfile). It uses Req 11, Req 10.

TestCase2

Test Case	TC-2
Use Case being tested	**ViewProductDetails**
Criteria for success/fail:	**Test that QRCode to appropriate URL**
Input Data:	**QRCode**
Test Procedure:	
Step 1: Call function ReadQRCode(info) with valid QRCode	**Success**

Step 1: Call function ReadQRCode(info) with invalid QRCode	**Fail**

This Test Case will mainly test the functions of UC-2 (ViewProductDetails). It uses Req6, Req2, Req 10. Test was UI based.

TestCase5

Test Case	TC-5
Use Case being tested	**AddProductInformation**
Criteria for success/fail:	**Test that the information for add product was posted to the database**
Input Data:	**Alphanumeric**
Test Procedure:	
Step 1: Call function addProduct(info) with valid data	**Success**
Step 2: Call function addProduct(info) with invalid data	**Fail**
Step 3: Call function	**Success**

getProductInfo(id) with valid data.	
Step 4: Call function getProductInfo(id) with invalid data.	**Fail**

This Test Case will mainly test the functions of UC-5 (AddProductInformation) with the aid of UC-1 (CreateCattleProfile). It uses Req4, Req 10.

TestCase7

Test Case	TC-7
Use Case being tested	**Generate Report**
Criteria for success/fail:	**Test that database query that best matches the actor's search criteria and retrieve the records from the database**
Input Data:	**Alphanumeric**
Test Procedure:	
Step 1: Call function filterProductInfo(info) with valid data	**Success**
Step 2: Call function filterProductInfo(info) with invalid	**Fail**

data	
Step 3: Call function getCattleInfo(Id) with valid data	**Success**
Step 4: Call function getCattleInfo(Id) with invalid data	**Fail**

This Test Case will mainly test the functions of UC-7 (GenerateReport) with the aid of UC-1 (CreateCattleProfile), UC-5 (AddProductInformation). It uses Req8, Req 10,Req 15.

TestCase8

Test Case	TC-8
Use Case being tested	**AddSlaughterInformation**
Criteria for success/fail:	**Test that information was added to the database**
Input Data:	**Alphanumeric**
Test Procedure:	
Step 1: Call function addSlaughterInfo(info) with valid data	**Success**

Step 2: Call function addSlaughterInfo(info) with invalid data	**Fail**
Step 3: Call function getSlaughterInfo(Id) with valid data	**Success**
Step 4: Call function getSlaughterInfo(Id) with invalid data	**Fail**

This Test Case will mainly test the functions of UC-8 (AddSlaughterInformation) with the aid of information from UC-1 (CreateCattleProfile), UC-5 (AddProductInformation), and UC-7 (GenerateReport). It uses Req 4, Req 10.

History of Work

Breakdown of Responsibilities- Report 2

Name	Did	Doing	Will Do
Joanne	Project Management. Did Problem Statement., Glossary, Preliminary Designs, Plan of Work.	Next phase report, attending meetings, project managing, system	System framework and design, create and populate database tables, assist with

		framework.	front end functionality, debugging.
Chimeziri m	Did System Specifications/Requirements & Global Control Flow.. Assisted with User Effort Estimation, Traceability Matrix, Problem Statement, Hardware Requirements, Project Management.	Next phase report, attending meetings.	Front end and back end functionalities, populate database, generate and assign QR codes.
Rene	Did Problem Statement, Casual Description, Identifying Subsystems, Architectural Styles Assisted with Use Case Diagrams, System Sequence Diagrams	Next phase report, attending meetings.	System framework and design, assist with front end functionality.Populating database tables.
Miguel	Assisted with System Specifications, User Effort Estimation, Identifying Subsystems	Next phase report, attending meetings.	Front end and back end functionalities, populate database, generate and assign QR codes. Debugging.
Juan	Assisted with Traceability Matrix, Did fully dressed Descriptions,	Next phase report, attending meetings.	Front end and back end functionalities, populate

	Connectors & Network Protocols		database, generate and assign QR codes.
Cameron	Did Architectural Styles and Mapping subsystems to Hardware Assisted with Sequence Diagrams and Use Case Diagrams.	Next phase report, attending meetings.	Front end and back end functionalities, populate database, generate and assign QR codes.

Breakdown of Responsibilities- Report 2

Name	Did	Doing	Will Do
Joanne	R1: Project Management. Did Problem Statement., Glossary, Preliminary Designs, Plan of Work. Assisted with Use Cases, Traceability Matrix, Fully Dressed Description, Hardware Requirements. Updated Report 1 based on feedback R2: Did Data Model and Persistent Data Storage, Class Diagrams, Data Types and Operation Signatures, User Interface Design	Next phase report, attending meetings, project managing, system framework, unit testing, database management.	System framework and design, create and populate database tables, assist with front end functionality, debugging.

	and Implementation, and Project Management. Assisted with Interaction Diagrams, conceptual model, Algorithms, Concurrency, and Data Structures.		
Chimeziri m	R1: Did System Specifications/Requirements & Global Control Flow.. Assisted with User Effort Estimation, Traceability Matrix, Problem Statement, Hardware Requirements, Project Management. Updated Report 1 based on feedback R2: Did Conceptual Model, System Operation Contracts, Traceability Matrices, Interaction Diagrams, Concurrency.	Next phase report, attending meetings, QR generation, unit testing.	Front end and back end functionalities, populate database, generate and assign QR codes.
Rene	R1: Did Problem Statement, Casual Description, Identifying	Next phase report, attending meetings,	System framework and design, assist with front

	Subsystems, Architectural Styles. Assisted with Use Case Diagrams, System Sequence Diagrams. Updated Report 1 based on feedback R2: Did Conceptual Model, Interaction Diagrams, User Interface Design and Implementation,	system framework, unit testing.	end functionality.Populating database tables.
Miguel	R1: Assisted with System Specifications, User Effort Estimation, Identifying Subsystems. R2: Did Data Structures and Design of Test.	Next phase report, attending meetings, backend functionality, unit testing.	Front end and back end functionalities, populate database, generate and assign QR codes. Debugging.
Juan	R1: Assisted with Traceability Matrix, Did fully dressed Descriptions, Connectors & Network Protocols. R2: Assisted with Conceptual	Next phase report, attending meetings, database connections, unit testing.	Front end and back end functionalities, populate database, generate and assign QR codes.

	Model (Domain Models).		
Cameron	R1: Did Architectural Styles and Mapping subsystems to Hardware Assisted with Sequence Diagrams and Use Case Diagrams. R2: Algorithms	Next phase report, attending meetings.	Front end and back end functionalities.

Breakdown of Responsibilities- Report 3

Name	Did
Joanne	R1: Project Management. Did Problem Statement., Glossary, Preliminary Designs, Plan of Work. Assisted with Use Cases, Traceability Matrix, Fully Dressed Description, Hardware Requirements. Updated Report 1 based on feedback R2: Did Data Model and Persistent Data Storage, Class Diagrams, Data Types and Operation Signatures, User Interface Design and Implementation, and Project Management. Assisted with Interaction Diagrams, conceptual model, Algorithms, Concurrency, and Data Structures. R3: Edited errors from report 1 and 2, assisted with UI creations, assisted with database

	connections, assisted with linking screens, did overall project management.
Chimeziri m	R1: Did System Specifications/Requirements & Global Control Flow.. Assisted with User Effort Estimation, Traceability Matrix, Problem Statement, Hardware Requirements, Project Management. Updated Report 1 based on feedback R2: Did Conceptual Model, System Operation Contracts, Traceability Matrices, Interaction Diagrams, Concurrency. R3/final phase: Edited errors from report 1 and 2, assisted with UI creations, assisted with QR generation.
Rene	R1: Did Problem Statement, Casual Description, Identifying Subsystems, Architectural Styles. Assisted with Use Case Diagrams, System Sequence Diagrams. Updated Report 1 based on feedback R2: Did Conceptual Model, Interaction Diagrams, User Interface Design and Implementation, R3/final phase: Edited errors from report 1 and 2, assisted with UI creations, assisted with QR generation, assisted with screen linkage.
Miguel	R1: Assisted with System Specifications, User Effort Estimation, Identifying Subsystems. R2: Did Data Structures and Design of Test.

	R3/final phase: assisted with screen linkage.
Juan	R1: Assisted with Traceability Matrix, Did fully dressed Descriptions, Connectors & Network Protocols. R2: Assisted with Conceptual Model (Domain Models). R3/final phase: assisted with database connections and authentication.
Cameron	R1: Did Architectural Styles and Mapping subsystems to Hardware Assisted with Sequence Diagrams and Use Case Diagrams. R2: Algorithms

Evolution of Milestones and Deadlines

Report 1:

- Overall mockup (digital and drawing) of system design.
- Selected platforms/interfaces for the system.
- Created basic UI screens (sign up, log in, profile).
- Completed required sections for the report, (CSR/System Requirements, Functional Requirement Specifications, System Architecture.
- Started to populate the database tables.
- Begun looking into Firebase and its authentication sector.

Report 2:

- Unit Testing of priority use cases.
- Integration testing of the system.
- Created UI screens for the priority use cases.
- Connected some of the priority use case screens to the database to allow for reading and writing to the database (for unit testing data).
- Improved some of the old UI screens to more responsive and professional looking screens (sign up, log in).
- Implemented the QR generating library.
- Populated more information to our database.

Report 3:

- Completed all UIs for the priority use case screens and some of the non functional requirements (eg: transferral).
- Connected the database to screens that needed to write and write from the database.
- Updated the database table columns to have a more general and organized structure.
- Completed edits for report 1 and 2.
- Created links between pages to assist with the flow of the system screens.
- Implemented Firebase authentication.

Key Accomplishments

- Database connections to screens that require it.
- Developed all required screens for our priority use cases.

- Easily implemented basic UI system screens (change password, etc.).
- Successful QR generation, linked to its designated data from the database.
- Formatted, professional screen designs.
- Completion of the overall system.

Possible Directions for Future Work (on this project)

- Implement a better termination process when cattle and its profile information is transferred to the next stage (slaughter or packaging).
- Implement a system/filepicker to host images (Eg: Amazon s3) so that images can be stored in our database successfully.
- Implement a more practical way of creating specified classes for specific elements (eg: buttons, navigation bars, menus, etc.).

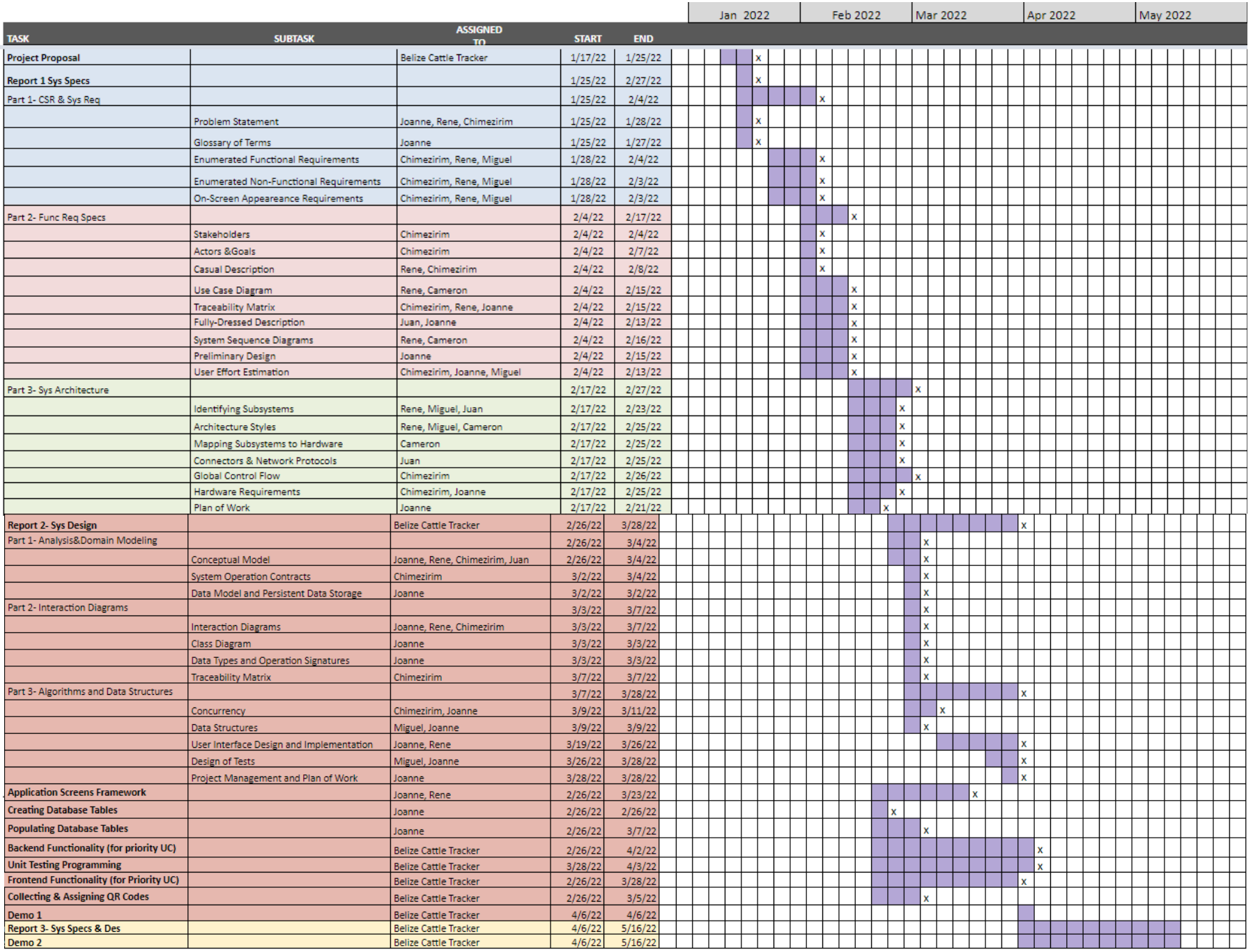
Project Management and Plan of Work

Issues Encountered

Some issues that were encountered while compiling this report was finding the proper time to work on the project together as all the members work and have daily responsibilities that they have to take care of, which means that we struggled in finding the time needed to meet and establish a deadline or make vital decisions like what programming languages we would be using or who would be doing what. There was also poor time management in certain phases of the Report process because of external factors. We were able to counter this problem and find a

solution for it by making a github repo, which all the members were able to clone and work on when they had time to do so and also setting flexible deadlines. Another problem that was encountered was coming up with the classes that would be tested. Since our project scope was mostly limited to extracting information from the Database and displaying it to the user as well as Inserting information into the database and since the React already comes with a QR library, coming up with a QR Code seemed to be made much easier and so we had trouble with that aspect of the project. However we were able to counteract this problem by doing further research as to how we could properly unit test the functionality that we already had available in our code. Another problem was finding an IDE that we would use to conduct the unit test especially since we were generally unfamiliar with how unit testing works, however after further research, we would be able to come to the conclusion that we would just continue using VSCode as it was the one that we were most comfortable with. In general even though we did encounter some difficulties with the coding process and getting firebase to connect to our react code, most of these issues were resolved after putting our heads together and working on it together to come up with a solution.

Plan of Work (Gantt Chart)



TASK	SUBTASK	ASSIGNED TO	START	END
Project Proposal		Belize Cattle Tracker	1/17/22	1/25/22
Report 1 Sys Specs			1/25/22	2/27/22
Part 1- CSR & Sys Req			1/25/22	2/4/22
	Problem Statement	Joanne, Rene, Chimezirim	1/25/22	1/28/22
	Glossary of Terms	Joanne	1/25/22	1/27/22
	Enumerated Functional Requirements	Chimezirim, Rene, Miguel	1/28/22	2/4/22
	Enumerated Non-Functional Requirements	Chimezirim, Rene, Miguel	1/28/22	2/3/22
	On-Screen Appeareance Requirements	Chimezirim, Rene, Miguel	1/28/22	2/3/22
Part 2- Func Req Specs			2/4/22	2/17/22
	Stakeholders	Chimezirim	2/4/22	2/4/22
	Actors &Goals	Chimezirim	2/4/22	2/7/22
	Casual Description	Rene, Chimezirim	2/4/22	2/8/22
	Use Case Diagram	Rene, Cameron	2/4/22	2/15/22
	Traceability Matrix	Chimezirim, Rene, Joanne	2/4/22	2/15/22
	Fully-Dressed Description	Juan, Joanne	2/4/22	2/13/22
	System Sequence Diagrams	Rene, Cameron	2/4/22	2/16/22
	Preliminary Design	Joanne	2/4/22	2/15/22
	User Effort Estimation	Chimezirim, Joanne, Miguel	2/4/22	2/13/22
Part 3- Sys Architecture			2/17/22	2/27/22
	Identifying Subsystems	Rene, Miguel, Juan	2/17/22	2/23/22
	Architecture Styles	Rene, Miguel, Cameron	2/17/22	2/25/22
	Mapping Subsystems to Hardware	Cameron	2/17/22	2/25/22
	Connectors & Network Protocols	Juan	2/17/22	2/25/22
	Global Control Flow	Chimezirim	2/17/22	2/26/22
	Hardware Requirements	Chimezirim, Joanne	2/17/22	2/25/22
	Plan of Work	Joanne	2/17/22	2/21/22
Report 2- Sys Design		Belize Cattle Tracker	2/26/22	3/28/22
Part 1- Analysis&Domain Modeling			2/26/22	3/4/22
	Conceptual Model	Joanne, Rene, Chimezirim, Juan	2/26/22	3/4/22
	System Operation Contracts	Chimezirim	3/2/22	3/4/22
	Data Model and Persistent Data Storage	Joanne	3/2/22	3/2/22
Part 2- Interaction Diagrams			3/3/22	3/7/22
	Interaction Diagrams	Joanne, Rene, Chimezirim	3/3/22	3/7/22
	Class Diagram	Joanne	3/3/22	3/3/22
	Data Types and Operation Signatures	Joanne	3/3/22	3/3/22
	Traceability Matrix	Chimezirim	3/7/22	3/7/22
Part 3- Algorithms and Data Structures			3/7/22	3/28/22
	Concurrency	Chimezirim, Joanne	3/9/22	3/11/22
	Data Structures	Miguel, Joanne	3/9/22	3/9/22
	User Interface Design and Implementation	Joanne, Rene	3/19/22	3/26/22
	Design of Tests	Miguel, Joanne	3/26/22	3/28/22
	Project Management and Plan of Work	Joanne	3/28/22	3/28/22
Application Screens Framework		Joanne, Rene	2/26/22	3/23/22
Creating Database Tables		Joanne	2/26/22	2/26/22
Populating Database Tables		Joanne	2/26/22	3/7/22
Backend Functionality (for priority UC)		Belize Cattle Tracker	2/26/22	4/2/22
Unit Testing Programming		Belize Cattle Tracker	3/28/22	4/3/22
Frontend Functionality (for Priority UC)		Belize Cattle Tracker	2/26/22	3/28/22
Collecting & Assigning QR Codes		Belize Cattle Tracker	2/26/22	3/5/22
Demo 1		Belize Cattle Tracker	4/6/22	4/6/22
Report 3- Sys Specs & Des		Belize Cattle Tracker	4/6/22	5/16/22
Demo 2		Belize Cattle Tracker	4/6/22	5/16/22

Project Progress Description (Report 1&2's Progresses)

The milestones and deadlines were adjusted in response to input from our project developers and the review procedure outlined in the management processes. This project was completely managed using WhatsApp chats and a GitHub repository set up for the team. When modifications were made, they were recorded in the revision history table located at each commit's position in the repository. As stated in reports 1 and 2, group meetings gave input that was used to regulate progress reports and the construction and implementation of user screens. From the project's start date until the present, our group holds virtual meetings over Zoom on one day of the week (typically Fridays and Saturdays) to discuss the project's progress and documentation.

From our initial idea and intended design, we wanted to implement a camera library (from React) on the public (customer) side of the website so that users can scan meat product QR codes through it, but after some research and experimenting we realized that it was deemed inefficient as the camera from the camera library is flawed for some mobile devices. We then realized that the QR codes could have been scanned with the normal camera app on a user's phone and the user would immediately be redirected to the site containing the product details. This was not only more convenient but it is more time efficient for the user because the user wouldn't have to access our public website in order to scan.

Possible new milestones include creating, designing and implementing screens and functionality for:

1. A portion/interface where farmers can only view their created cattle profiles. (UI implemented)

2. A better filtering system for BAHA so that they can filter out and generate reports based on different things such as DOB of cattle, farmer name, etc. instead of just by cattleID. (UI implemented)

3. An implementation of the non-functional requirement where farmers, slaughterhouse managers, and packaging managers get terminated from adding new information to the cattle profile after the cattle/produce is handed off to the next stage. (UI implemented)

Key accomplishments formulated until present for the project:

• Portals for some of the admin users (Eg: farmer, BAHA)

• The adding of information from each stage. (Eg: Farmer→ Slaughterhouse→ Packaging)

• Unique generating of QR codes

• Information connection between Firebase database and to the specified QR code.

• QR code scanner can now scan a QR code generated by us

• Basic functionalities of the system (log in, sign up, authentication)

Summary of Changes

Report 1 changes:

Project Description - At this stage we wanted to print labels containing the generated QR Code, but we don't have access to label machines so we would be unable to connect it to the system.

Use Case Descriptions - In report 1, we only had 3 fully dressed descriptions and these were based on the main use cases that we found to be the most important ones. Allocate Role, viewProductDetails and accessProductInformation were chosen for this phase.

Interaction diagrams - With the interaction diagrams, we had 3 use cases that we were primarily working on, which included the login, view product details, accessingInsertingProductInformation. We used the UML notations to show the flow between the screens and how it interacted with the database, notating the relationships between the interface and the database.

Report 2 changes:

Project Description - At this stage, we decided to make the production managers have the ability to print QR Code in which they can add to their packages during the packaging process. Some changes were made to the UI to reflect the changes that were made.

Use Case Descriptions - We had to make further changes to the fully dressed descriptions that we had for Report 1, where we had to split the Allocate Role use case into 3 parts which were now CreateCattleProfile, ViewProductDetails, AddProductInformation . By doing this, we now had 5 main use cases that we would mostly focus on in the Interaction Diagrams and other specifications that needed to be made.

Interaction diagrams - For the report 2, we further elaborated upon the sequence diagrams that were drawn and we linked the concept definitions that we had done earlier by expanding the different components that the system would interact with in order to show the information to the user.

Report 3 changes:

Project Description - Due to the limitations of the QR react library and the practicality of printing QR codes, we decided to allow packaging managers to download the QR codes instead so that they have freedom with the size and position of how the QR should be on the product label.

Use Case Descriptions - The Use case descriptions remained the same for this part, as no changes needed to be made.

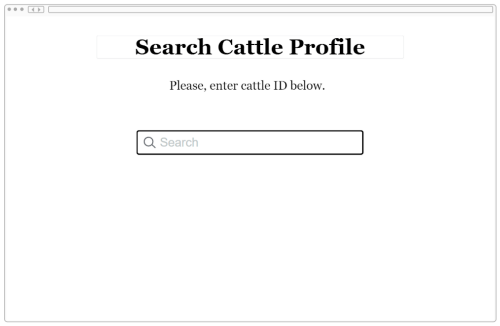
Interaction diagrams - No changes were made to the sequence diagrams, as the flow of the system mostly remained the same

Demo 1 changes (UI changes):

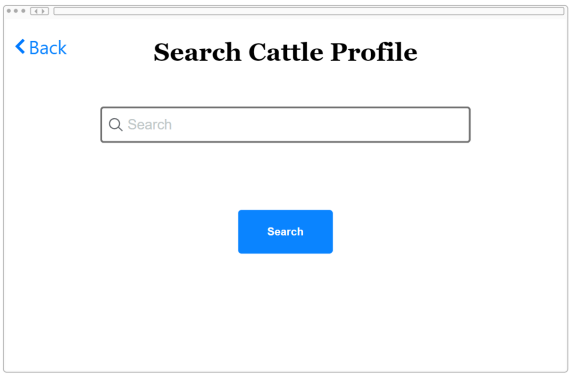
UI changes were the most prominent changes throughout the reports. While the first screens were mostly mock designs, as we went into the 2nd report, the screens started to have more of a resemblance of what the final application would look like. The changes are highlighted below :

Report 1 UI screens

Search Cattle Profile Screen



Report 2 UI Screens



The main difference between the screens shown is that there is more of an emphasis on usability, the screen in report 1 had no back button or search button which meant that the user would be stuck on that particular page or would be forced to use the back button on the browser. In terms of aesthetics, the page looks much better and cleaner as well as the fact that all the pages in the UI are now responsive and can be altered based on the screen that it is being shown on.

References

Aguilar, G. et.al (2018, 2014). Calibre: A Culinary Technology Company. [Weblog]. Retrieved March 1, 2022, from http://doit.ub.edu.bz/pluginfile.php/7777/mod_resource/content/1/REPORT2.pdf

Delaney, J. (2018,August 23). *Join Collections in Firestore.* Retrieved March 31, 2022, from https://kevintuck.co.uk/regression-testing-can-never-be-fully-automated/

Garbutt, E. (2017). InDClass: Student Attendance Tracker Using QR Code. Retrieved February 28, 2022 from https://drive.google.com/file/d/1ekLmikoZQGVXNNhDOfT0CRXrM8isXR_W/view?ts=6238eaf5

Google. (n.d.). Google. Retrieved May 15, 2022, from https://firebase.google.com/

Google. (n.d.). *Firebase authentication | firebase documentation*. Google. Retrieved May 15, 2022, from https://firebase.google.com/docs/auth

Nielsen, J. (2020, November 15). 10 Usability Heuristics for User Interface Design. Retrieved March 17, 2022 from https://www.nngroup.com/articles/ten-usability-heuristics/

Real time systems. GeeksforGeeks. (2022, January 12). Retrieved February 27, 2022, from https://www.geeksforgeeks.org/real-time-systems/

Software Architecture Patterns: Layered. Towards Data Science. (2017, August 28). Retrieved February 27, 2022, from https://towardsdatascience.com/software-architecture-patterns-98043

Stackoverflow. (n.d). Firebase Storage How to store and Retrieve images [closed]. Retrieved May 15, 2022, from https://stackoverflow.com/questions/13955813/firebase-storage-how-to-store-and-retrieve-images#:~:text=Yes%2C%20you%20can%20store%20and,URL%20generated%20for%20the%20image.

Tasks in real time systems. GeeksforGeeks. (2022, February 16). Retrieved February 27, 2022, from https://www.geeksforgeeks.org/tasks-in-real-time-systems/

The React Component Library You always wanted. MUI. (n.d.). Retrieved May 15, 2022, from https://mui.com/

Tuck, K. (2020, February 26). *Regression testing can never be fully automated*. Kevin Tuck. Retrieved March 31, 2022, from https://kevintuck.co.uk/regression-testing-can-never-be-fully-automated/

Types of Architectural Patterns. GeeksforGeeks. (2021, October 27). Retrieved February 27, 2022, from https://www.geeksforgeeks.org/types-of-software-architecture-patterns/

What is bandwidth - definition, meaning & explanation. Verizon Fios. (n.d.). Retrieved February 27, 2022, from https://www.verizon.com/info/definitions/bandwidth/

Wikimedia Foundation. (2022, February 11). *Real-time computing*. Wikipedia. Retrieved February 27, 2022, from https://en.wikipedia.org/wiki/Real-time_computing