Performance Livestock Analytics

Final Project Document

Team Number: DEC1708
Client: Dustin Balsley
Advisor: Daji Qiao

Team Members/Roles:
Michael Rhodas - Team Leader
Rachel Hartman - Communication Leader
Jacob Johnson - Key Idea Holder
Jeffrey Murray - Key Idea Holder
Ken Kohl - Webmaster

Team Email: dec1708@iastate.edu
Team Website: http://dec1708.sd.ece.iastate.edu/index.html

Revised: 12/05/2017, v4.0

Contents

1. INTRODUCTION

Our project was to create a web application for tracking the health of groups of cattle. Our application will be used by farmers and veterinarians to keep a long term history of costs, medication supplies, and use associated medicating and vaccinating cattle. Our application will store vaccination/medication procedures that occur on a farm. The goal of this project is to have this application be integrated with the client's pre-existing product to be market as an additional feature to be used in their production.

1.1 Project Statement

Create a web application for Performance Livestock Analytics. Our web application, Cattle Health, tracks the health of cattle over their lifetime. Our application is focused audience is cattle farmers as they give medications to their cattle. We will create an intuitive web application for tracking veterinary medical information for cattle farmers. This application will help our clients record and monitor medical treatment and recovery information for their animals and facilitate data-driven analysis and decision making to help our users make more informed economic and medical decisions.

1.2 Project Purpose

To create an application that allows our customer to improve their bottom line. We will do this by providing the means to track the health of their cattle and determine cost of medicating/vaccination a cattle for its lifetime. This project would benefit our clients by giving them better metrics of profitability and animal care analysis for their livestock. Our product will in turn help make our clients more efficient, profitable, and reduce operational costs. By analyzing and tracking data, cattle farmers will have the data to drive their decisions in real time in the field.

1.3 Project Goals

For this project we are striving to create a veterinary medicine companion app to help the clients manage the health of cattle as well as make data driven decision about medical treatment. Our application will help cattle farmers track the recovery or non-recovery of their animals and would provide support for monitoring of various health indicators, medical treatment data, and cost analysis. The application will include systems for secure access to the Performance Beef database for information tracking, a method of data entry for more efficient and effective application of medical treatment, and monitoring tools that allow our clients to quickly see how their animals are recovering as well as make informed decision about continuing treatment. To achieve this project we have identified the following project goals:

- Create a simple and intuitive user interface and data entry model that is easy to learn and painless to use in the field
- Develop methods of communication with the existing Performance Beef database systems in order to use and improve the information made available to our clients
- Build a information review system from synthesized data that allows our clients to make more effective and informed decisions about their animal health

In the document that follows, we will outline our project deliverables moving into our market research survey. After describing our market research, we will deliver our design decisions and assessments in section 4. Next, we will move on to our project requirements outlined in section 5. From there we will discuss the challenges we faced and the testing we did to ensure a sound product in sections 6 and 7 . Lastly, the document will end with an summary of our project and the references and appendices used to create this document.

2. DELIVERABLES

In order to accomplish the project goals described above we will create a fully featured Web application . This application will include:

- An easy to use data entry system that allows our clients to retrieve data for each head of cattle as well as add additional health and treatment information for later analysis
- Systems of connected access to Performance Beef for data retrieval and updates as well as methods of offline caching
- A home for synthesized and meaningful data that that allows our clients to see and learn from their treatment and health records
- Easy record keeping for medications administration, cost, and amount

2.1 Implementation Details

While planning our web application development we looked at many different frontend and backend components for our project. We looked at angular.js, Sass, Bootstrap, React, React Native, node.js, and JQuery. The technologies we choose to use for our development were Firebase, React, Bootstrap, HTML5, CSS, and Javascript as these were the technologies best for our application. Bootstrap allows quick development of forms, buttons, and layouts all readily available for development. We choose to use Bootstrap because it allowed quick uniform development and the licensing issue did not concern our client. React also allowed for quick responsive development.

3. MARKET SURVEY

We will now present an overview of the research conducted for our project. In order to further understand the audience for our project our team research other products in the same market of cattle health. Also, we researched the IoAHT (Internet of Animal Health Things) to grasp how the internet was being used for livestock production as that is our application's purpose.

3.1 Related Products

In the market of web-based applications, there does appear to be an application, CattleMax [4], which accomplishes very similar health data tracking systems as proposed in our application. The primary difference is that the CattleMax application appears to that CattleMax is a broader health tracking application it tracks its cattle health using spreadsheets instead of a database application like our application. Our application utilizes real time data with database tracking of the medications given to cattle. Tracking the data in a spreadsheet allows you to get information quickly, however it does not provide you a complete transaction history as our product does.

3.2 Related Literature

While there are a number of existing products available as web applications that serve different veterinary purposes, there are no existing products which accomplish what we are proposing. A number of existing apps available list existing FDA approved medications [1], [2], however, these products do not provide tracking for individual animals and medications or doses of the drugs administered. The strive to apply the internet of things to livestock production is an ever growing need. “Internet of Animal Health Things” or IoAHT for short, is a term given to how the Internet of Things connects with the animal health industry to improve the industry. The demand for meat keeps rising. If the demand keeps going up, farmers have to keep up as well. The emphasis here is that the cattle market has grown rapidly in recent years and with it expected to continue to grow, there is a need to monitor the health of animals without having a farmer inspect each animal which becomes impractical with larger herds. By farmers using software tools to maximize their efficiency, their methods will become more effective, making more meat available per farm. Not to mention the profits the farmer is also bound to gain. If there were tools to track animal's health in real time, then conclusions might be drawn that the operation would become more efficient. Another article, The Internet of Animal Healthy Things: IoAHT, suggests that IoAHT can be broken down into different categories sections such as wildlife, pets, and farm animals. Each section would use the IoT to accomplish different tasks and concerns. A wearable device would be the next step for our product in the farm animal category. If Performance Livestock, were to use this technology the data would tell us if the cow is getting sick before it is actually sick which would allow PLA to start treatment earlier ie. tracking health more efficiently. This allows the

farmer to track vitals for all the animals that they are raising in a more seamless way that provides the detailed background information that consumers demand today. The challenge that is being faced with this concept today is that there is no defined industry standard for how to approach storage of this type of data.

4. DESIGN

The project has changed quite significantly since the first semester of work. We started off creating an iOS application programmed using the Swift language. After a discussion with our client from Performance Livestock Analytics at the beginning of the semester, we decided it was best to look into a multi-platform solution. This brought us to designing a modern web application because then the application will only require an internet browser like Google Chrome.

The application also had a slight change in procedure for entering in cattle health data but otherwise remained the similar. At first, we were going to enter cattle data in a streamlined process, one cow after another. Notes would be made if a cow required medical attention. In the application currently, this process is not streamlined but entered in manually if a cow needs tracking.

Knowing what the goal of the next project was, we needed to make a decision for the database, front-end, and back-end. We used Firebase, a real-time database from Google in the last semester and decided to continue using this. By having a database that updates in real time, we let it act as our back-end as well. Last, for the front-end, we chose to use ReactJS, a javascript library developed by Facebook. We chose this because some members on the team have used it before, it is straightforward to learn for the rest of the team, and it is a modern resource used to develop web applications.

For the layout and style of our application, we looked at Performance Livestock Analytic's current web app and based it off that. The client provided us with a CSS style sheet to show us the coloring and such. From there, members of the team were assigned to develop different panels of the application starting with the login screen and basic layout for the home screen.

The application in its near final state now provides farmers with the ability to easily track their cattle's health concerns. They can create their own custom medications, groups of cattle, and then apply those medications to respective groups and individuals in that group. A history of all medical related operations can be viewed on the history panel. The plan for the future is to create data analytic visuals for the farmer to view costs, medical supplies, operations per day, and any other valuable visuals for them to use.

4.1 Proposed System Block Diagram

Our previous semester in senior design, we created an iOS app that contained many of the features you see in this semester's application. There was one main story that we made functional in the last semester's application. This was called the "My Medications" story. The purpose of this story was to have users create custom medications and be able to import FDA medication data to their data entry form. This story was complete for the iOS application, but needed to be redone for this semester's project because we switched platforms to a web application (JS/HTML5) base. See figure one below for more details. The "Record Keeping" story has been removed with a different method put into place explained below. One key difference is that there is no more scanning of cattle's tags, rather it is all entered manually.

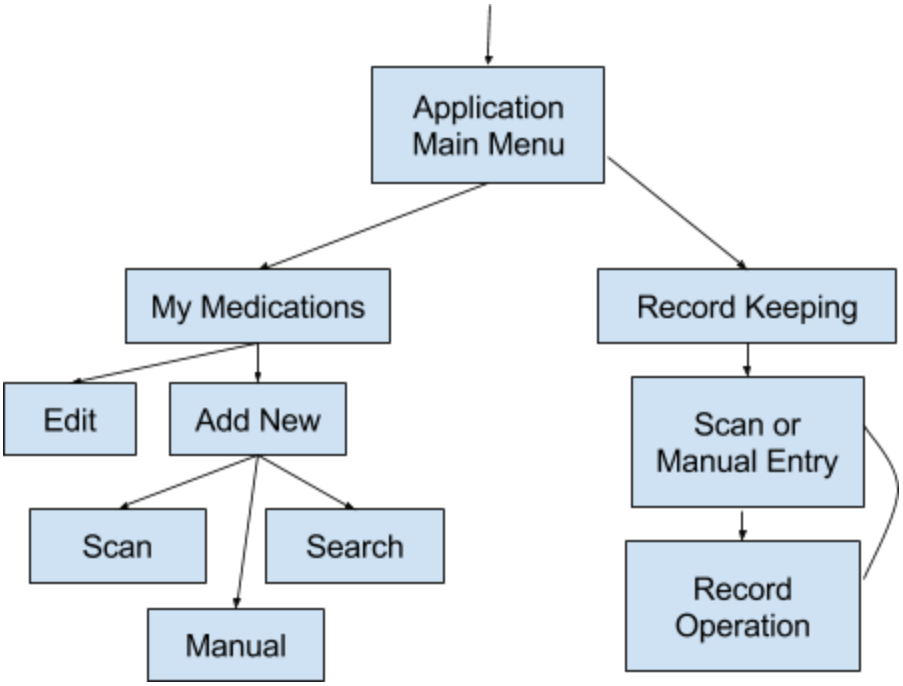


Figure One - Semester One Block Diagram

The flow of our current application can be thought of as a single page web application. This entails the synchronization of all the data between the switching of panels without having the browser reload or refresh the page.

When first visiting our application, there is a login screen for authorization. This login will connect the user's medications, groups, and operations to the application. It will also auto login if there is data in the cache of the browser. If you are not currently a user, there is an option to register as a user with the email and password filled in. See figure Two on the next page.

Figure Two - Login View

The option to sign out is always at the top right of the screen, which brings you to the login screen again. The first screen that appears after login is the history panel. This panel is a nice overview of the operations completed to the current date. See figure three below for more details.

Figure Three - History View

The other panels we currently have are Apply Medications, My Medications, and My Groups. The switching of these panels is done with the left navigation bar, and can be switched at any time. My Medications and My Groups are similar panels. My Medications is used to create either custom medications or import a medication from the FDA to apply to a group of cattle or an individual cow. When a medication is selected, you are able to edit and delete it with the buttons to the right in the row. See figure four below for more details and figure five below for an example of the medication editor form.

Figure Four - My Medications View

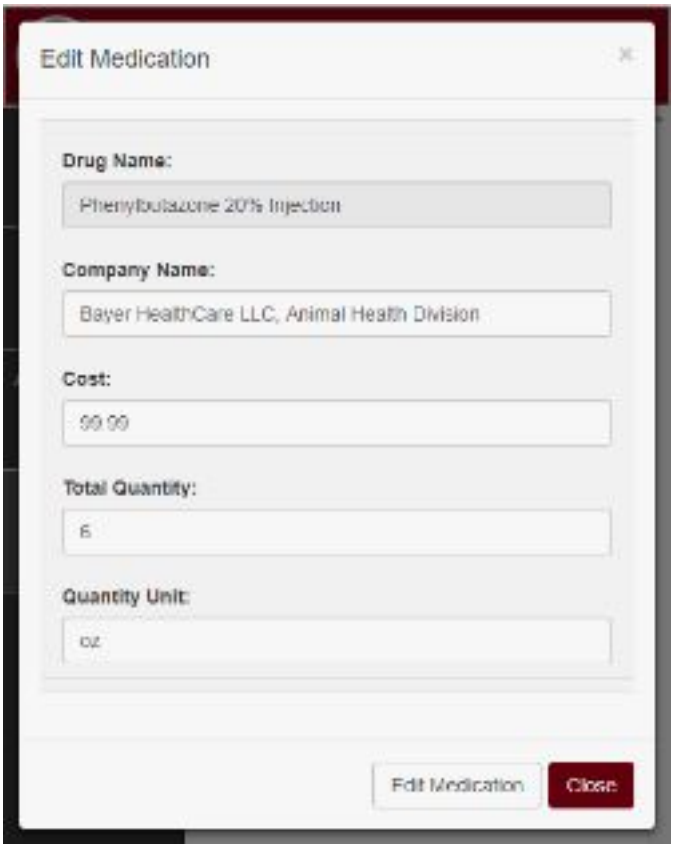


Figure Five - Medication Editor

My Groups, see figure six below, is a way to easily create a group of cattle for applying medications to. These groups will eventually be imported from Performance Livestock Analytic's database, but for now we make temporary entries for functionality. This page behaves just like My Medications.

Figure Six - My Groups View

Apply Medications is the core of the application currently and provides the functionality to apply a user's medications to their groups. It is a dynamic form that goes through steps to get your medication data recorded. First, you enter the Date and all the groups you'd like to apply the medication to. Next, enter in tag numbers if the operation only applies to individual cattle, otherwise enter no tag numbers to apply the medication to the whole group. Next, select all medications required from the user's medication list. Add any notes about the operation, then hit apply. After this, you are brought to the History panel to show you your newly added operation(s). See figure seven below for an example configuration of this form component.

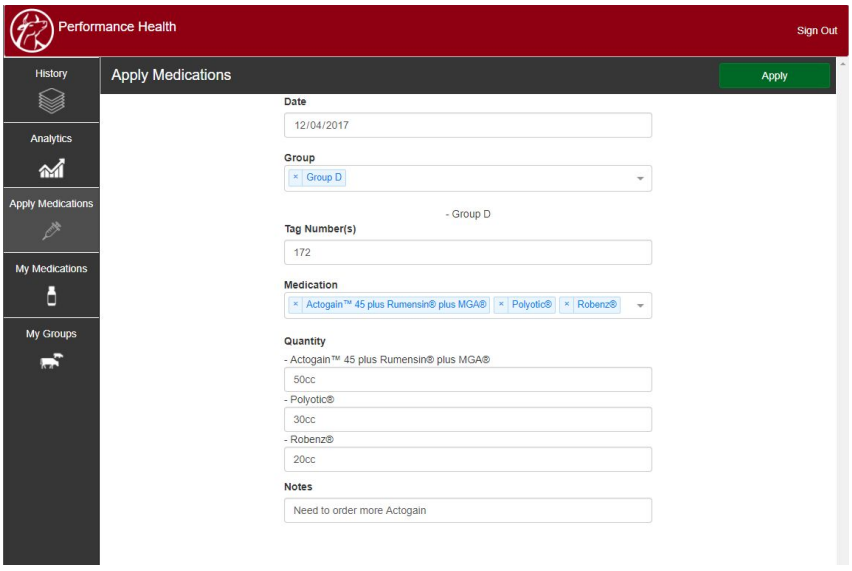


Figure Seven - Apply Medication View

With all parts put together, our block-design diagram is as follows in figure eight below.

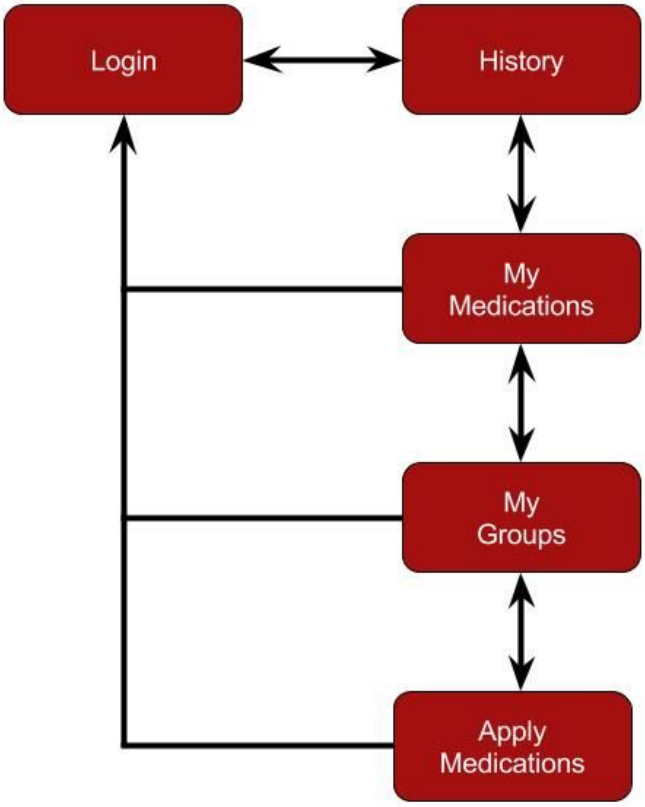


Figure Eight - Updated Block Diagram

4.2 Assessment of Proposed Methods

We made a web app using React JS and Bootstrap. The hope is that this will allow to make the application cross platform. Using the platform has gone surprisingly well. Each of us had some level of understanding developing web applications so we were able to spend more time making a quality product instead of trying to learn a language from the ground up.

5. PROJECT REQUIREMENTS / SPECIFICATIONS

Below is a summary of the functional and nonfunctional requirements and additional specifications recorded for the project.

5.1 Functional Requirements

The table below explores all of the functional requirements defined for the project.

Requirement	User Description
FR-1.0	Can log into the system.
FR-1.1	- Can create an account and log into the system.
FR-1.2	- Can logout of the system.
FR-2.0	Can manage their “my medications” list.
FR-2.1	- Can add a medication to their “my medications” list.
FR-2.2	- Can remove a medication from their “my medications” list.
FR-2.3	- Can edit an existing medication in their “my medications” list.
FR-2.4	- May add name, company, price, and quantity metadata to a medication.
FR-3.0	Can manage their farms list of groups.
FR-3.1	- Can add a group to their farm.
FR-3.2	- Can remove a group from their farm.
FR-4.0	Can “administer” a medication to a group or “subgroup.”
FR-4.1	- Can create a new administering session.
FR-4.2	- Can define the group to be administered to.
FR-4.3	- Can define the individual tag numbers in that group being administered to. A user may also select that the entire group is being administered to.
FR-4.4	- Can define a product being administered in the session.nA user can define the quantity per head being administered. A user can define additional products being administered for the session
FR-5.0	Can view history of medication usage.

FR-5.1	Can view the history of all medications administered over time on a per application (not per session) basis.
FR-6.0	Can view summary analytics for medication usage.
FR-6.1	- Can view the total quantity administered for each product used.
FR-6.2	- Can view the total cost associated with each product used.
FR-6.3	- Can view the pull rate for each of their farm's groups.
FR-6.4	- Can view which individual tags have had medications administered. A user can view how much and what has been administered to a specific group / tag number.

5.2 Non-Functional

The table below explore all of the non-functional requirements defined for the project.

Requirement	Description
NFR-1.0	Data must remain integrable when user enters it manually. Must be consistent with the data already populated on the database.
NFR-2.0	Many users and accounts should be able to interact with the system at any given time.
NFR-3.0	Many instances of a single user account should be able to interact with the system at the same time with proper interleaving.
NFR-4.0	Database CRUD operations should never fail unsafely resulting in data loss
NFR-5.0	User authentication and database operations should use encryption protocol
NFR-6.0	Must be available with SLA of 99%.
NFR-7.0	User login and authentication system should use Firebase encryption system
NFR-8.0	All new user accounts should be setup in less than 3 seconds upon signup.

5.3 Standards

During this project we followed the standards for HTML5, CSS, Bootstrap, React, and Firebase. We also modeled our project off best coding practices such as modal coding, formatting, and structure. We also modeled our project based on Airbnb React Javascript Style Guide. We referenced this guide for naming, mixins, classes, props, references, and methods during our project. For naming classes, methods, and functions we used camelcase naming convention. We

organized our project with subdirectory folders defined as actions, components, constants, css, icons, and stores. By doing so we kept our coding organized and modalized.

5.4 Contraints

1. Site will only be available through web application.
    a. Must perform on multiple web browsers.
2. Must be modularized in order for Dustin to connect to his database.
    a. Adding and removing features should be as simple as deleting/adding a module.
    b. Conformance to coding standards and best practices.

5.5 Concerns

1. High usability
    a. Basic user can navigate and perform tasks easily.
    b. A user should never get confused, or not know how to use a feature.
2. Navigation must be intuitive.
    a. Adding, scrolling, moving, inserting must be consistent throughout application.
    b. It should be fairly obvious to the user how all features function.

6. CHALLENGES

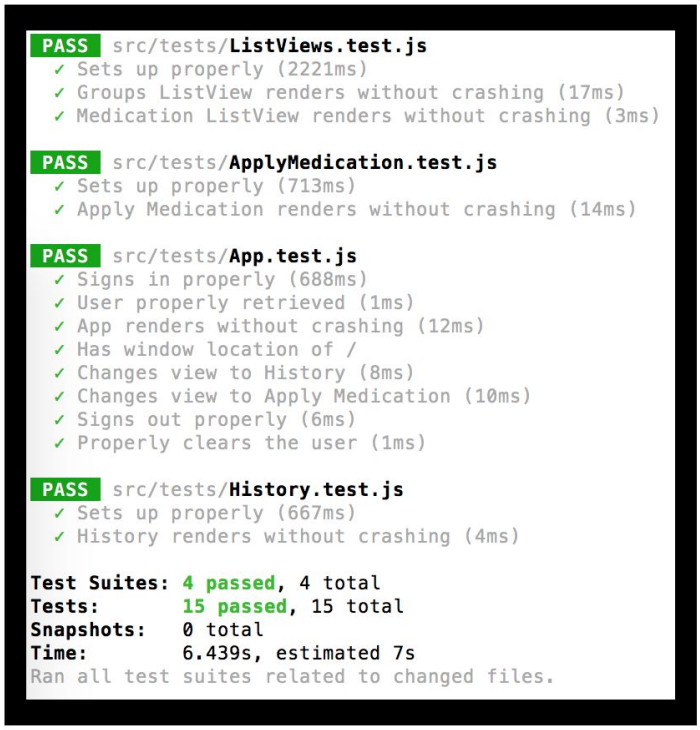
From the start of this project there have been challenges with communication and getting the requirements from the client. In the first semester it wasn't until after a month were we able to set up a meeting with the client. We had thought from the project description that this would be a web based project. We found out in the first meeting that the project would be in IOS. None of us had ever worked with IOS or written Swift code. It was also a challenge not having a consistent requirements document. It was common in both semesters that whenever we would demo a prototype to the client he would change his mind or add features that he wanted to the product.

A big challenge came in the first meeting with the client in the Fall semester. Essentially we came in expecting to continue on with the IOS app we had made the last semester. By the end of the meeting we were no longer making an IOS app and instead making a web application using React. That put us back because we essentially discarded all of the work that had previously done and starting from the ground up.

Another hurdle that we have had to consider in all of our work is system interactions with the existing Performance Beef database. We haven't had access to this database directly so it has been very important that we are clear in our API needs and consistent in how we are storing, using, and updating user data.

7. TESTING

Our current testing environment utilizes a utility called Jest to ensure the functionality of core components of our application. Jest allows us to run a suite of tests and monitor the output as we write tests or as we add new components to the application. An example of this suite is shown to the right in figure nine. This core functionality is primarily focused on the ability to sign in and sign out, retrieve data for an individual user, and render core components of the application without the application crashing. As these tests have a focus on more broad functionality than specific use cases, it is currently not comprehensive, and does not have cases to guard against individual bugs. The goal with setting up this testing environment was mainly to set up a base for testing that can be utilized by the team at Performance Livestock Analytics once we actually hand the application off.



8.FUTURE WORK

The future work for our application includes creating code documentation and handing over the project to PLA. We decided to document our code, so that PLA would be able to reference it when integrating our project with theirs. Code documentations will provide understanding and explanation when they are integrating our project. We will give them our repository and all the necessary documents need to run our project such as a how to guide, use of technologies, design approach, and many more. By doing so, we are ensuring that our project will be a viable addition to their product.

9. CONCLUSIONS

Our goal for this project was to create cattle health application for PLA. During our year working with PLA, we created two deliverables; a iOS application and an intuitive web application. Our application will allow farmers to track the health of cattle over their lifetime along with keeping track of their own medication supply and a record of when they are applying medications. Our application will integrate with PLA's to allow them to create analytics related to animal health along with giving a better overall picture of a farmer's operation in order make the client more efficient and profitable.

9. REFERENCES

1. Website Emulation

   www.performancebeef.com

2. GitLab Documentation

   https://docs.gitlab.com/ee/README.html

3. React-Bootstrap Components

   https://react-bootstrap.github.io/components.html

4. React-Bootstrap Setup:

   https://react-bootstrap.github.io/getting-started.html

5. Firebase Setup:

   https://firebase.google.com/docs/web/setup

6. Firebase Web References:

   https://firebase.google.com/docs/reference/js/

7. Airbnb React/JSX Style Guide:

   https://github.com/airbnb/javascript/tree/master/react

8. The Internet of Animal Healthy Things: IoAHT

   http://www.bluefintechnologypartners.com/2017/04/internet-animal-healthy-things-ioaht/

9. Recent Advances in Wearable Sensors for Animal Health

http://www.sciencedirect.com/science/article/pii/S2214180416301350

10. Non-Invasive Sensor Technology for the Development of a Dairy Cattle Health Monitoring System

www.mdpi.com/2073-431X/5/4/23/pdf

11. How the Internet of Things Is Impacting Animal Health Part 1: Large Animal

http://www.engagemobile.com/how-the-internet-of-things-is-impacting-animal-health-part-1-large-animal/

10. APPENDICES

The following appendices are included as reference materials for the project design, setup, and building processes.

9.1 Appendix One - Operations Manual

9.1.1 Project Setup

The first step to start developing is to download your web development environment of choice. Two nice options you might consider are Brackets and WebStorm - the former is free to everyone and the latter is free for students. Brackets if fairly simple but it's a great option for just quickly editing files and getting things going quickly. Webstorm is more complicated but also far more powerful and an industry standard for JavaScript development.

http://brackets.io https://www.jetbrains.com/webstorm/

To run the local development server for this application you will need NPM a very popular package management tool for JavaScript. This comes packaged with Node.js which we can install very easily. First, check if you already have these tools with the commands below.

$ node -v

$ npm -v

If you don't already have these tools installed do so now by visiting the website below and following the install instructions.

https://www.npmjs.com/get-npm

To run the local development server for this application you will need to download the required project dependencies. Luckily, the node package manager we just installed makes this easy. This can be done with the following command:

$ npm install

9.1.2 Project Demo

In order to run or demo the product software on must compile the program software and configure the development web server. Again, node makes this very simple. There are two available build versions of the web application at any time - the production build and the development build. To build for development use the command:

$ npm start

For a full production build without shortcuts use the command below and check the build directory for the resulting application code:

$ npm run build

9.1.3 Project Testing

To run the integration testing supplied with the project use the build in testing script activated with the command:

$ npm run test

This will individually run all the configured project tests including but not limited to the Jest implementations created within the context of our team's development parameters.

9.2 Appendix Two - Alternate Implementations

During the first semester of senior design project, we created an iOS application. We created the application from the ground up. We had most functionality done by the end of the first semester. When we were asked to switch to a web application we thought about wrapping our iOS development into a web application, but decided that it would be better to start over. After deciding the platforms we were going to use. We decided to develop our web application with the thought of wrapping it into a iOS application in mind. We took attention to the side menu and navigation page on screen size reduction. We also developed our Modals with smaller screen sizes in mind. Once we are finished with our application, it will have the availability to be wrapped into an iOS application. Due to the complete restart of our project, we had limited time to complete non functional functions.