

Deliverable K: User Manual

Chanelle Menard, 300287290
Elisha Khandaker, 300308377
Jonathan Aguilera, 300190572
Rachelle Choi, 300281171
Sorosh Benvidi, 300143234

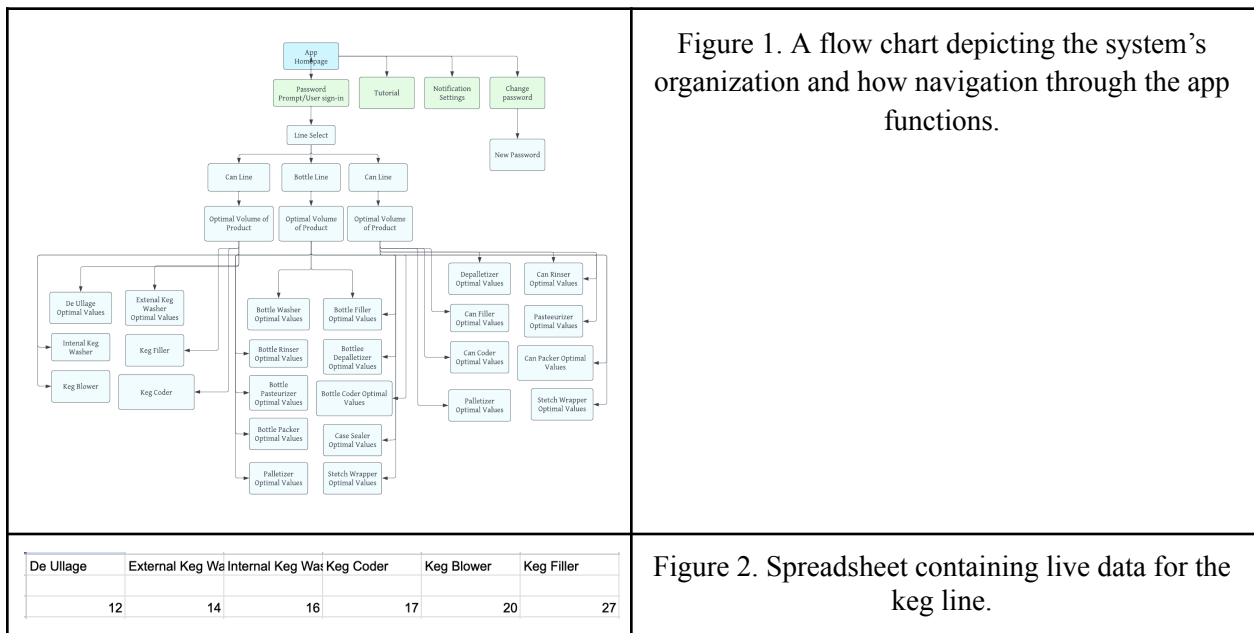
April 11th, 2023

Table of Contents

1 Introduction	1
Document Structure and Purpose	1
Security, Safety, and Privacy Considerations	2
Simplifying Assumptions	2
2 Overview	3
3 Getting started	4
3.1 Configuration Considerations	4
3.2 User Access Considerations	4
3.3 Accessing/setting up the System	5
3.3.1 System Setup	5
3.3.2 Password Change	5
3.3.3 System Access	6
3.4 System Organization & Navigation	6
3.4.1 The Welcome Page	7
3.4.2 Home Page	7
3.4.3 Systems Settings	7
3.4.4 Line Select and User-Input	7
3.4.6 Optimal Range of Speeds	7
3.5 Exiting the System	7
4 Using the System	8
4.1 Sign In	8
4.2 Line select	8
4.2.1 Can Line	8
4.2.1.1 User input	8
4.2.1.2 Can Line Data	8
4.2.2 Bottle Line	8
4.2.2.1 User input	8
4.2.2.2 Bottle Line Data	8
4.2.3 Keg Line	9
4.2.3.1 User input	9
4.2.3.2 Keg Line Data	9
4.3 System Settings	9
4.3.1 Notifications	9
4.3.2 Change Password	9
4.3.3 Tutorial	9
5 Troubleshooting & Support	10
5.1 Error Messages or Behaviors	10
5.2 Special Considerations	10
5.3 Maintenance	10

5.4 Support	11
6 Product Documentation	12
6.1 Databases & Libraries	12
6.1.1 BOM (Bill of Materials)	12
6.1.2 Equipment List	12
6.1.3 Instructions	12
6.2 UX and UI Design	14
6.2.1 BOM (Bill of Materials)	14
6.2.2 Equipment list	15
6.2.3 Instructions	15
6.3 Code Implementation	24
6.3.1 BOM	24
6.3.2 Equipment List	24
6.3.3 Instructions	24
6.4 Unused Components	36
6.5 Testing & Validation	38
7 Conclusions and Recommendations for Future Work	42
8 Bibliography	43
9 APPENDIX I: Design Files	44

List of Figures



Can Depalletizer	Can Rinser	Can Filler	Can Pasteurizer	Can Coder	Can Packer	Palletizer	Stretch Wrapper
152	162	175	200	228	243	272	282

Figure 3. Spreadsheet containing live data for the can line

Bottle Depalletize	Bottle Rinser	Bottle Filler	Bottle Pasteurize	Bottle Coder	Bottle Packer	Case Sealer	Palletizer	Stretch Wrapper
375	337	300	357	392	412	447	496	542

Figure 4. Spreadsheet containing live data for the bottle line

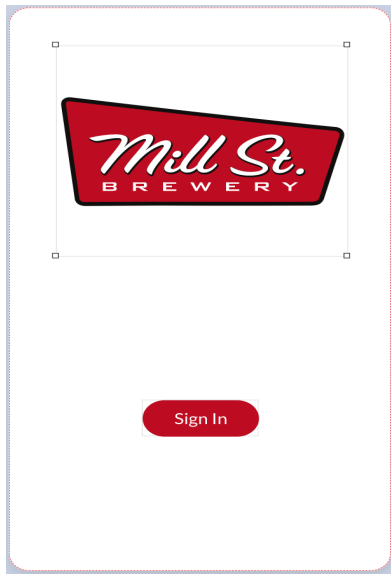


Figure 5. The homepage of the app that users see upon opening the app.

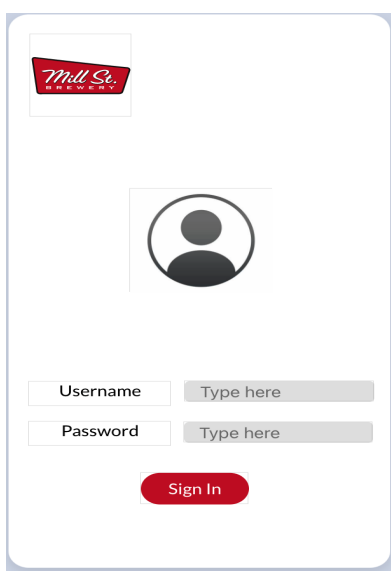


Figure 6. The sign-in page that users are taken to once clicking on the sign-in button.

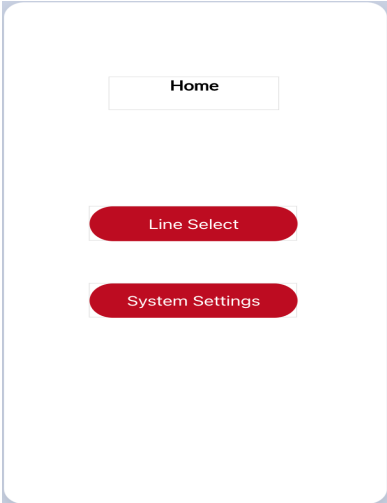


Figure 7. The homepage which users are able to view upon successfully signing in.

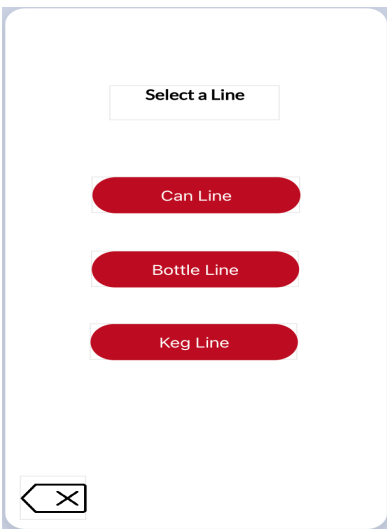


Figure 8. The Select line page which users will navigate to once they have selected this option from the home page.

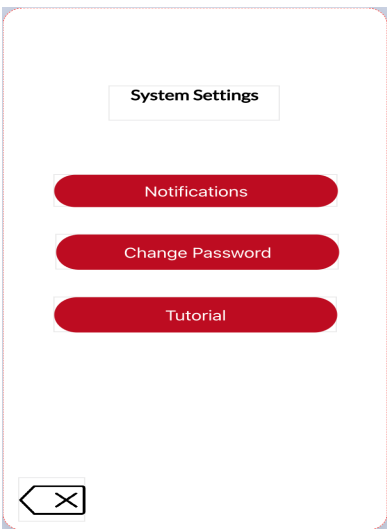


Figure 9. The systems settings page that users will navigate to once selecting the 'system's settings' button from the home page.

Username

Password:

Confirm password:

Figure 10. The change password page which users can access once selecting on this option from the system settings.

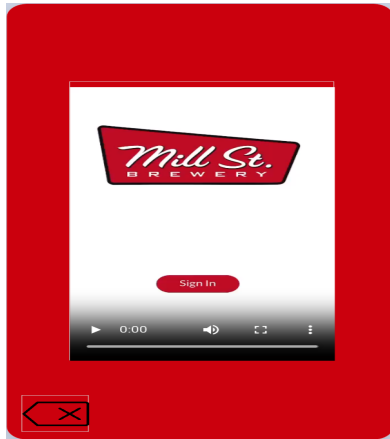
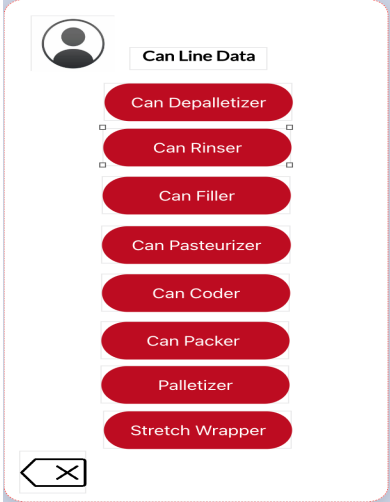
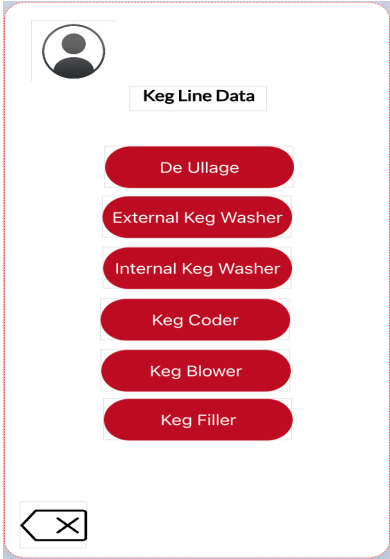
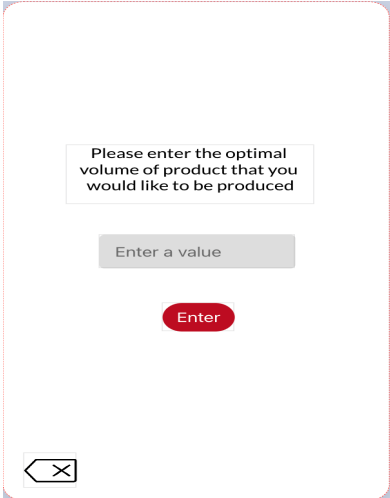


Figure 11. The tutorial page that shows users a video of how to navigate through the app. This is accessible from the system settings.

Figure 12. The Bottle line Data that is available from the select line page.

	<p>Figure 13. The Can line Data that is available from the select line page.</p>
	<p>Figure 14. The Keg line Data that is available from the select line page.</p>
	<p>Figure 15. The users are prompted to enter the ideal volume of bottles they would like to produce within a given day.</p>

Please enter the optimal volume of product that you would like to be produced

Enter a value

Enter

Navigation icons: back and close

Figure 16. The users are prompted to enter the ideal volume of cans they would like to produce within a given day.

Please enter the optimal volume of product that you would like to be produced

Enter a value

Enter

Navigation icons: back and close

Figure 17. The users are prompted to enter the ideal volume of kegs they would like to produce within a given day.

Bottle Depalletizer Optimal Speed

Speed

368 <=> 365 <=>

Navigation icons: back and close

Figure 18. The optimal speed range for the bottle depalletizer.

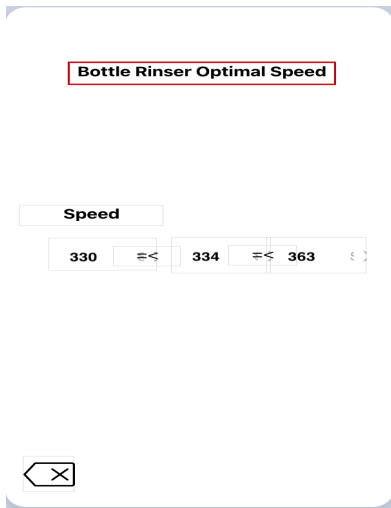


Figure 19. The optimal speed range for the bottle rinsers.

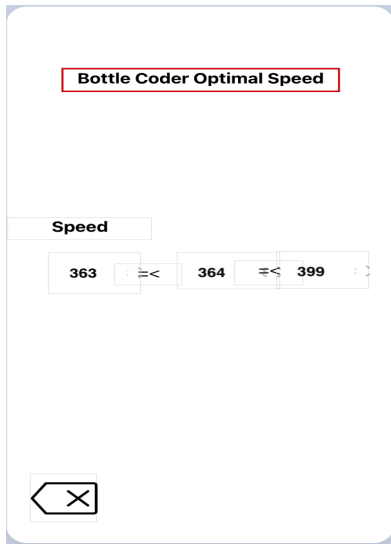


Figure 20. The optimal speed range for the bottle coder.

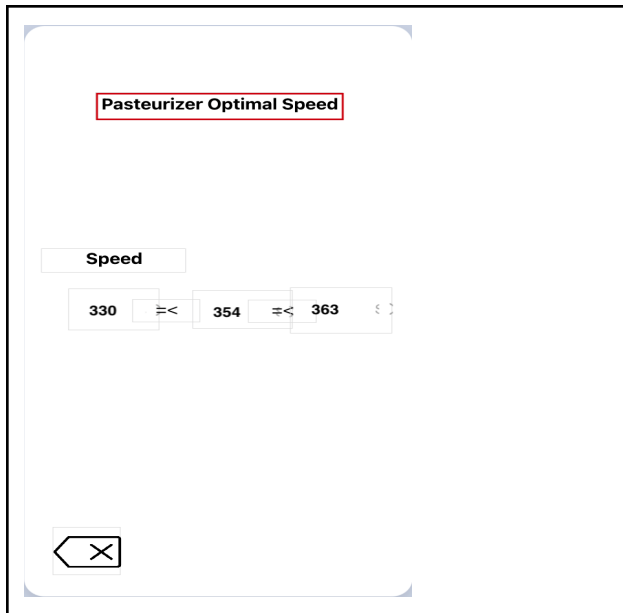


Figure 21. The optimal speed range for the bottle

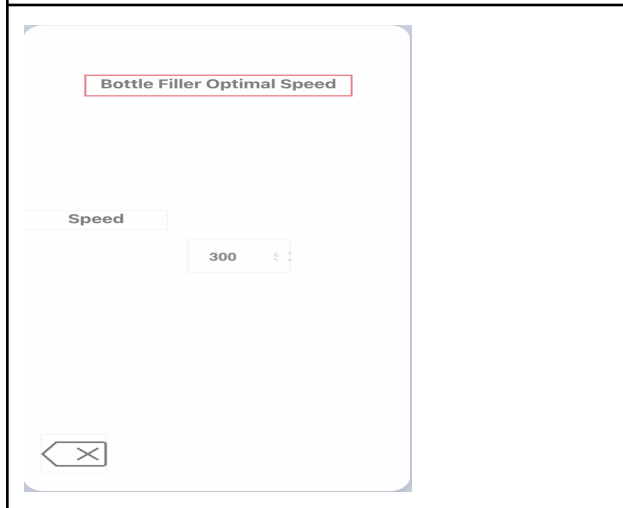


Figure 22. The optimal speed range for the bottle filler.

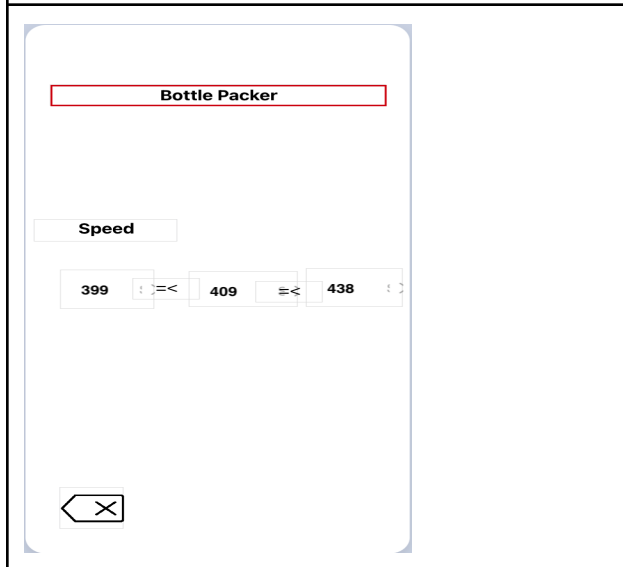


Figure 23. The optimal speed range for the bottle packer.

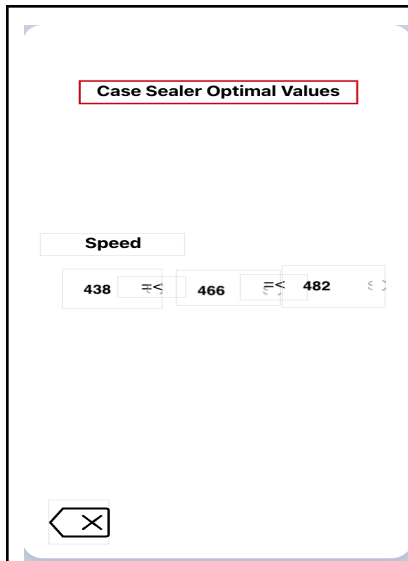


Figure 24. The optimal speed range for the bottle case sealer.

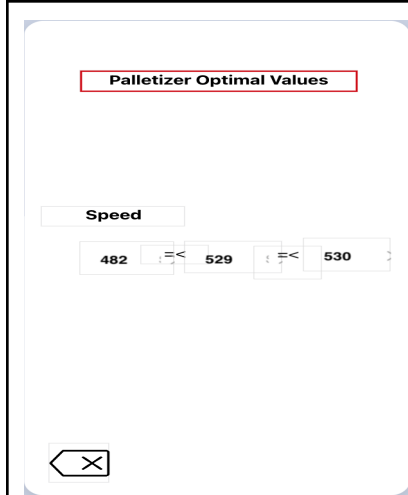


Figure 25. The optimal speed range for the bottle palletizer.

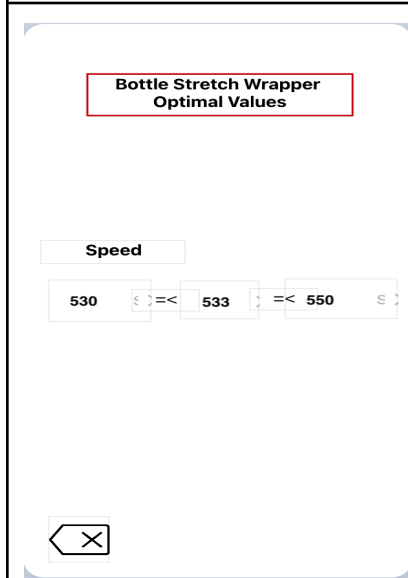


Figure 26. The optimal speed range for the bottle stretch wrapper.

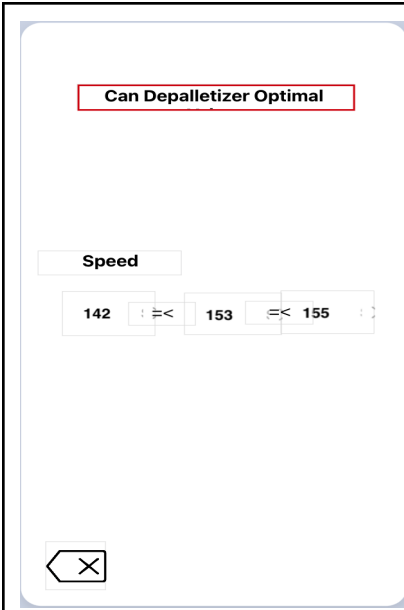


Figure 27. The optimal speed range for the can depalletizer.

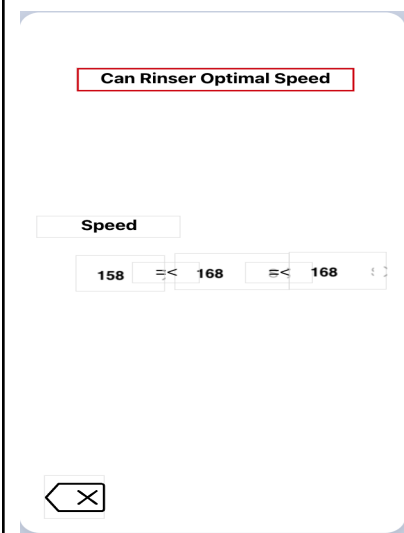


Figure 28. The optimal speed range for the can rinser.

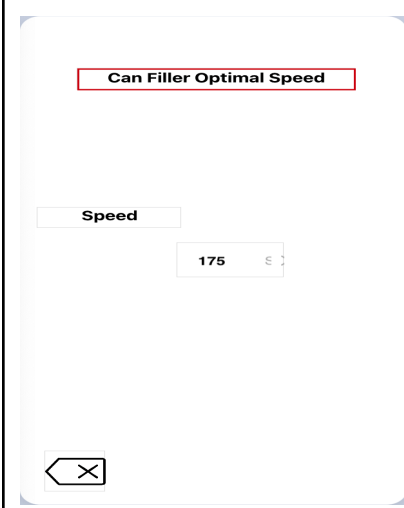


Figure 29. The optimal speed range for the can filler.

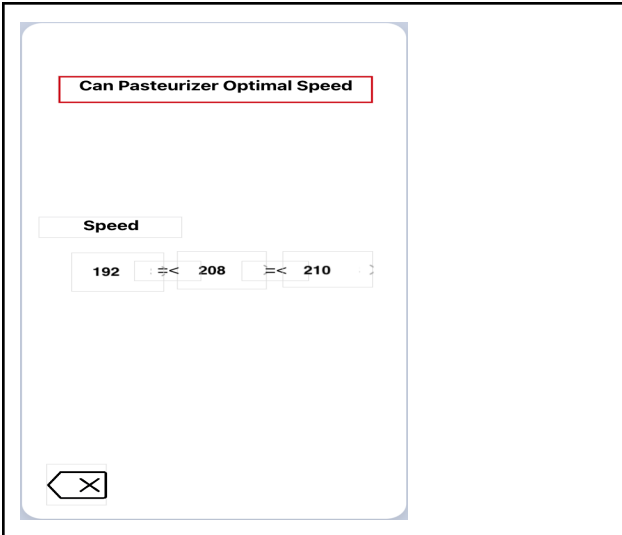


Figure 30. The optimal speed range for the can pasteurizer.

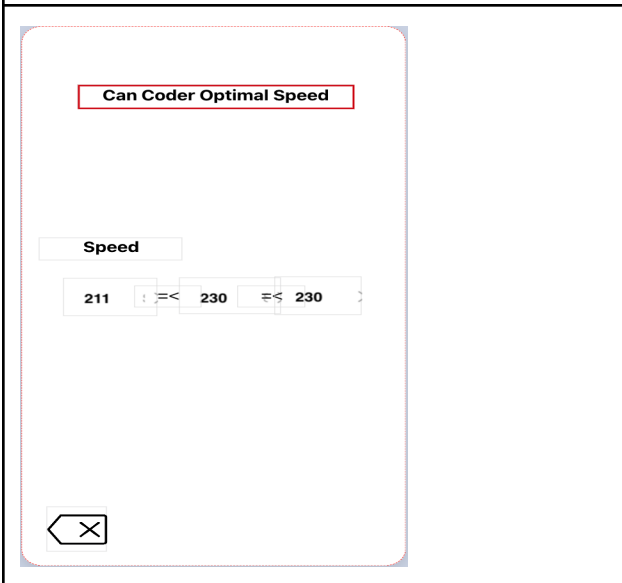


Figure 31. The optimal speed range for the can coder.

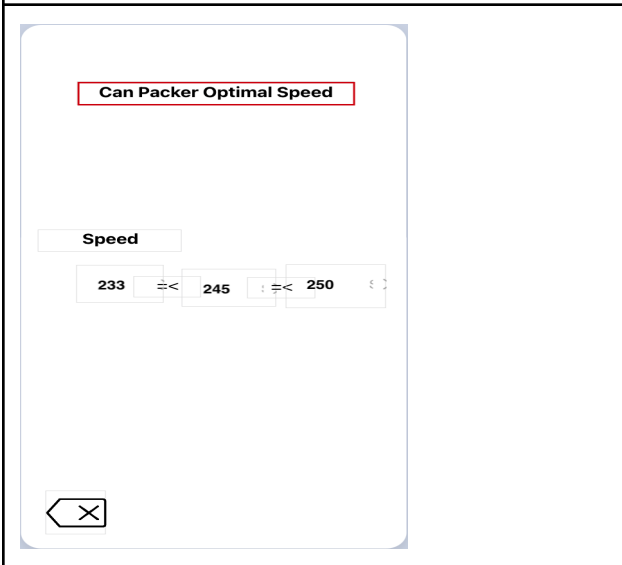


Figure 32. The optimal speed range for the can packer.

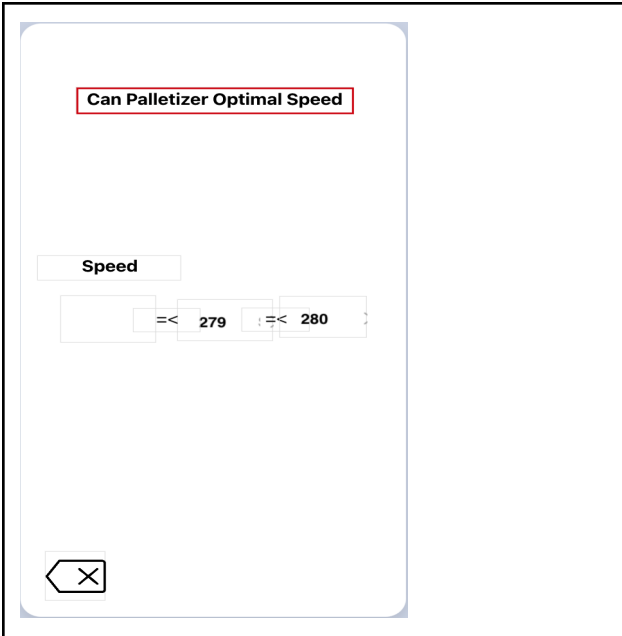


Figure 33. The optimal speed range for the can palletizer.

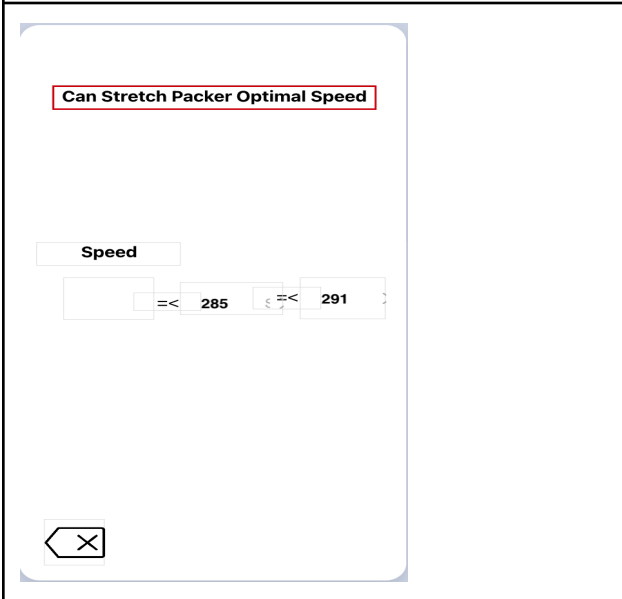


Figure 34. The optimal speed range for the can stretch packer.

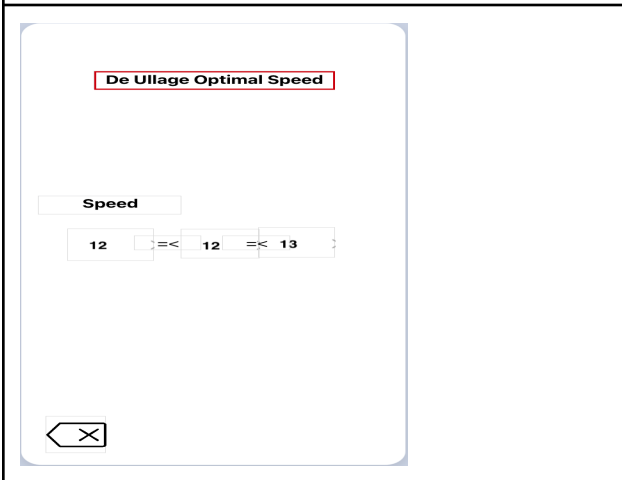


Figure 35. The optimal speed range for the De Ullage Machine.

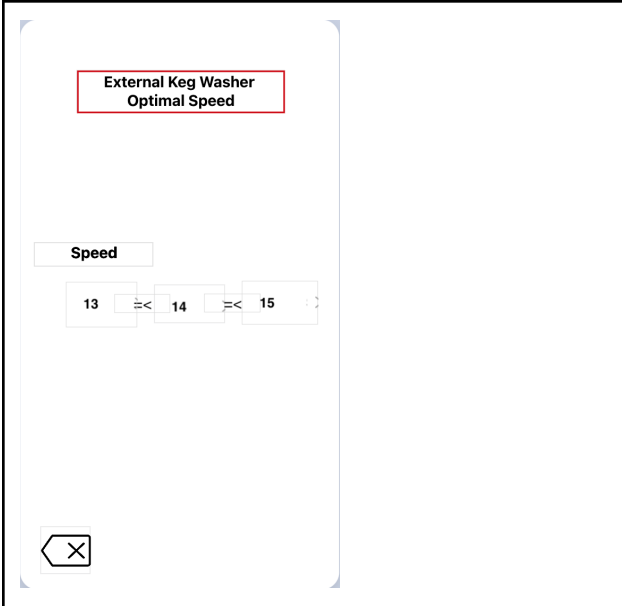


Figure 36. The optimal speed range for the external keg washer.

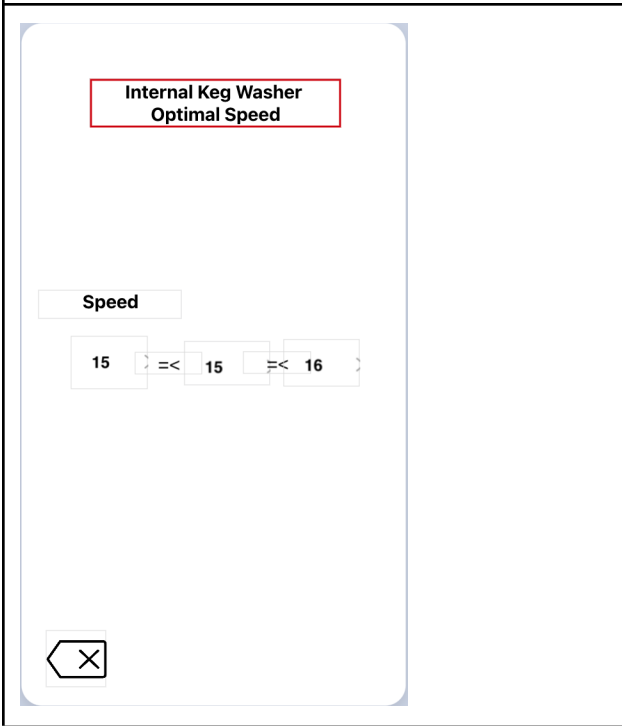


Figure 37. The optimal speed range for the internal keg washer.

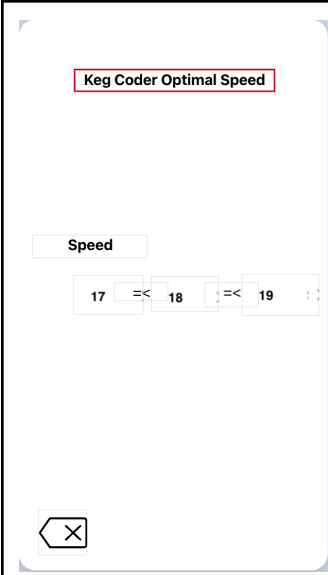
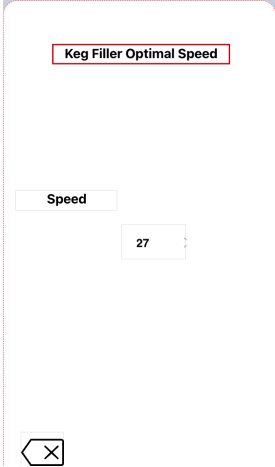
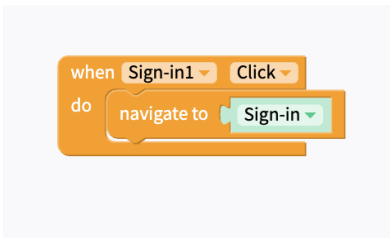
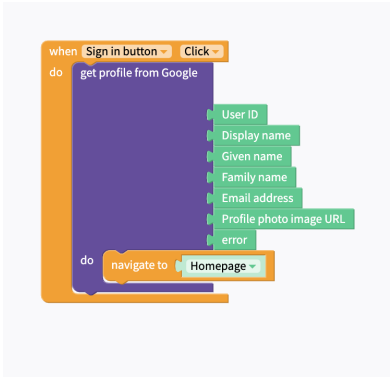
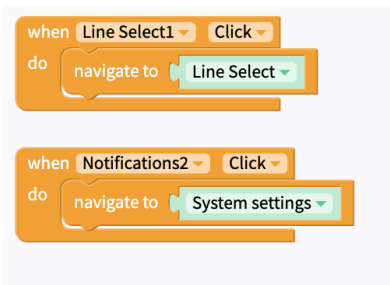


Figure 38. The optimal speed range for the keg coder.



Figure 39. The optimal speed range for the keg blower.

 <p>The screenshot shows a mobile application screen titled "Keg Filler Optimal Speed". Below the title is a label "Speed" followed by a text input field containing the number "27". At the bottom left, there is a back arrow icon and a close 'X' icon.</p>	<p>Figure 40. The optimal speed range for the keg filler.</p>
 <p>The code block consists of an orange "when" block with "Sign-in1" and "Click" dropdowns, and an orange "do" block with a "navigate to" block containing a "Sign-in" dropdown.</p>	<p>Figure 41. Code blocks representing the function of the sing-in button on the welcome page.</p>
 <p>The code block starts with an orange "when" block for "Sign in button" and "Click". The "do" block contains a "get profile from Google" block with a list of data fields: User ID, Display name, Given name, Family name, Email address, Profile photo image URL, and error. Below this is another orange "do" block with a "navigate to" block set to "Homepage".</p>	<p>Figure 42. Code blocks that store user's password and username information to compare with those stored in the manager's database.</p>
 <p>There are two code blocks. The first is an orange "when" block for "Line Select1" and "Click" with a "do" block containing "navigate to" and "Line Select". The second is an orange "when" block for "Notifications2" and "Click" with a "do" block containing "navigate to" and "System settings".</p>	<p>Figure 43. Code blocks that navigate to the line select page and system settings pages once they have been selected on the home page.</p>

```

when Conveyor Data2 Click
do navigate to Can Prompted Input

when Conveyor Data1 Click
do navigate to Bottle Prompted Input

when Conveyor Data3 Click
do navigate to Keg Prompted Input

when Image42 Click
do navigate to Homepage

```

Figure 44. Code blocks that navigate to the prompted input pages for all production lines including can, bottle, and keg lines in addition to a backspace button.

```

when Notifications1 Click
do navigate to Change password

when Image43 Click
do navigate to Homepage

when Notifications3 Click
do navigate to Tutorial Page

```

Figure 45. Code blocks for the systems settings page that bring users to the change-password page, the home page and the tutorial page.

```

when Button1 Click
do navigate to Homepage

when Image41 Click
do navigate to System settings

when Image42 Click
do navigate to Change password
do navigate to Tutorial Page
do navigate to Help Page
do navigate to Home Page
do navigate to Logout Page

```

Figure 46. Code blocks that demonstrate how the changing-password option functions.

```

when Image45 Click
do navigate to System settings

```

Figure 47. Code block that navigates to the systems settings of the app.

```

when Palletizer Click
do navigate to Bottle Separator

when Empty Bottle Filter Click
do navigate to Bottle Packager

when Pasteurizer Click
do navigate to Bottle Pasteurizer

when Labeler Click
do navigate to Bottle Coder

when Case Packer Click
do navigate to Bottle Braker

when Depalletizer Click
do navigate to Case Sealer

when Filter Click
do navigate to Bottle Filler

when Bottle Packager Click
do navigate to Bottle Braker

when Bottle Pasteurizer Click
do navigate to Bottle Pasteurizer

when Image41 Click
do navigate to Line Select

```

Figure 48. Code blocks for the bottle line machines that take the user to the optimal range of values for that specific machine.

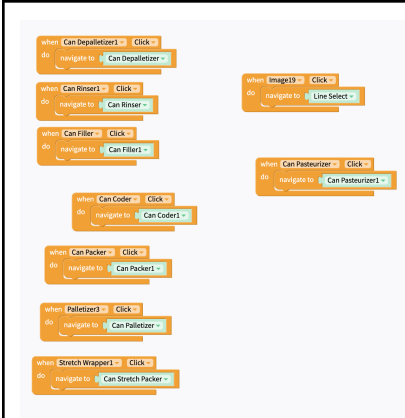


Figure 49. Code blocks for the can line machines that take the user to the optimal range of values for that specific machine.

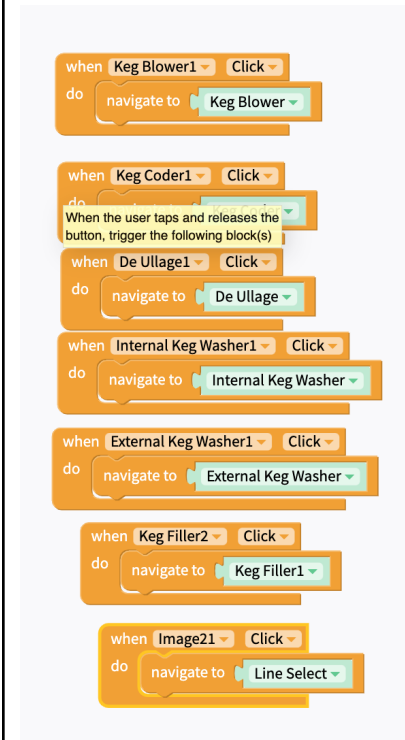


Figure 50. Code blocks for the keg line machines that take the user to the optimal range of values for that specific machine.

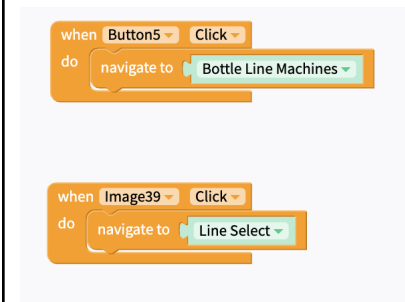


Figure 51. Code blocks on the user prompted input that take the users to the bottle line once an optimal volume of product has been entered.

when Button8 Click
do navigate to Keg Line Machines

when Image41 Click
do navigate to Line Select

Figure 52. Code blocks on the user prompted input that take the users to the keg line once an optimal volume of product has been entered.

when Button7 Click
do navigate to Can Line Machines

when Image40 Click
do navigate to Line Select

Figure 53. Code blocks on the user prompted input that take the users to the can line once an optimal volume of product has been entered.

when Image8 Click
do navigate to Bottle Line Machines

when Bottle Depalletizer Opens
do call Data_Viewer_List1's Refresh Data
then do when Refresh Data is done

when Data_Viewer_List24 Item Click
do vibrate
call Alert3's Show with output wasConfirmed
then do if wasConfirmed do

when Data_Viewer_List25 Item Click
do vibrate
call Alert2's Show with output wasConfirmed
then do if wasConfirmed do

Figure 54. Code blocks which refresh the simulated spreadsheet containing live data for the bottle depalletizer.

Figure 55 shows four Scratch code blocks:

- Block 1:** A 'when Image9 Clicked' event block followed by a 'do' block containing a 'navigate to Bottle Line Machines' block.
- Block 2:** A 'when Bottle Rinser Opens' event block followed by a 'do' block containing a 'call Data_Viewer_List2's Refresh Data' block and a 'when Refresh Data is done' block.
- Block 3:** A 'when Data_Viewer_List27 Item Clicked' event block with a 'row id' variable. The 'do' block contains: 'vibrate', 'call Alert5's Show' with 'with output' and 'wasConfirmed' outputs, and a 'then do' block with an 'if wasConfirmed' block.
- Block 4:** A 'when Data_Viewer_List26 Item Clicked' event block with a 'row id' variable. The 'do' block contains: 'vibrate', 'call Alert4's Show' with 'with output' and 'wasConfirmed' outputs, and a 'then do' block with an 'if wasConfirmed' block.

Figure 55. Code blocks which refresh the simulated spreadsheet containing live data for the bottle rinser.

Figure 56 shows four Scratch code blocks:

- Block 1:** A 'when Image10 Clicked' event block followed by a 'do' block containing a 'navigate to Bottle Line Machines' block.
- Block 2:** A 'when Bottle Coder Opens' event block followed by a 'do' block containing a 'call Data_Viewer_List3's Refresh Data' block and a 'when Refresh Data is done' block.
- Block 3:** A 'when Data_Viewer_List28 Item Clicked' event block with a 'row id' variable. The 'do' block contains: 'vibrate', 'call Alert6's Show' with 'with output' and 'wasConfirmed' outputs, and a 'then do' block with an 'if wasConfirmed' block.
- Block 4:** A 'when Data_Viewer_List29 Item Clicked' event block with a 'row id' variable. The 'do' block contains: 'vibrate', 'call Alert7's Show' with 'with output' and 'wasConfirmed' outputs, and a 'then do' block with an 'if wasConfirmed' block.

Figure 56. Code blocks which refresh the simulated spreadsheet containing live data for the bottle coder.

```

when Image11 Click
do
  navigate to Bottle Line Machines

when Bottle Pasteurizer Opens
do
  call Data_Viewer_List4's Refresh Data
  then do when Refresh Data is done

when Data_Viewer_List31 Item Click
do
  vibrate
  call Alert10's Show
  with output
  then do if wasConfirmed
  do

when Data_Viewer_List30 Item Click
do
  vibrate
  call Alert9's Show
  with output
  then do if wasConfirmed
  do
  
```

Figure 57. Code blocks which refresh the simulated spreadsheet containing live data for the bottle pasteurizer.

```

when Image12 Click
do
  navigate to Bottle Line Machines

when Bottle Filler Opens
do
  call Data_Viewer_List5's Refresh Data
  then do when Refresh Data is done
  
```

Figure 58. Code blocks which refresh the simulated spreadsheet containing live data for the bottle filler.

```

when Image13 Click
do
  navigate to Bottle Line Machines

when Bottle Packer Opens
do
  call Data_Viewer_List6's Refresh Data
  then do when Refresh Data is done

when Data_Viewer_List32 Item Click
do
  vibrate
  call Alert11's Show
  with output
  then do if wasConfirmed
  do

when Data_Viewer_List33 Item Click
do
  vibrate
  call Alert12's Show
  with output
  then do if wasConfirmed
  do
  
```

Figure 59. Code blocks which refresh the simulated spreadsheet containing live data for the bottle packer.

```

when Image14 Click
do
  navigate to Bottle Line Machines

when Case Sealer Opens
do
  call Data_Viewer_List7 Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List34 Item Click
do
  vibrate
  call Alert13 Show with output wasConfirmed
  then do if wasConfirmed do

when Data_Viewer_List35 Item Click
do
  vibrate
  call Alert14 Show with output wasConfirmed
  then do if wasConfirmed do
  
```

Figure 60. Code blocks which refresh the simulated spreadsheet containing live data for the bottle case sealer.

```

when Image22 Click
do
  navigate to Bottle Line Machines

when Bottle Palletizer Opens
do
  call Data_Viewer_List8 Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List36 Item Click
do
  vibrate
  call Alert15 Show with output wasConfirmed
  then do if wasConfirmed do

when Data_Viewer_List37 Item Click
do
  vibrate
  call Alert16 Show with output wasConfirmed
  then do if wasConfirmed do
  
```

Figure 61. Code blocks which refresh the simulated spreadsheet containing live data for the bottle palletizer.

```

when Image23 Clicked
do
  navigate to Bottle Line Machines

when Bottle Stretch Wrapper Opens
do
  call Data_Viewer_List5's Refresh Data
  then do
    when Refresh Data is done

when Data_Viewer_List38 Item Clicked
do
  vibrate
  call Alert17's Show
  with output
  wasConfirmed
  then do
    if wasConfirmed
    do
      call Data_Viewer_List39's Refresh Data

when Data_Viewer_List39 Item Clicked
do
  vibrate
  call Alert18's Show
  with output
  wasConfirmed
  then do
    if wasConfirmed
    do
      call Data_Viewer_List39's Refresh Data
  
```

Figure 62. Code blocks which refresh the simulated spreadsheet containing live data for the bottle stretch wrapper.

```

when Image24 Clicked
do
  navigate to Can Line Machines

when Can Depalletizer Opens
do
  call Data_Viewer_List10's Refresh Data
  then do
    when Refresh Data is done

when Data_Viewer_List40 Item Clicked
do
  vibrate
  call Alert19's Show
  with output
  wasConfirmed
  then do
    if wasConfirmed
    do
      call Data_Viewer_List41's Refresh Data

when Data_Viewer_List41 Item Clicked
do
  vibrate
  call Alert20's Show
  with output
  wasConfirmed
  then do
    if wasConfirmed
    do
      call Data_Viewer_List41's Refresh Data
  
```

Figure 63. Code blocks which refresh the simulated spreadsheet containing live data for the can depalletizer.

```

when Data_Viewer_List43 Item Clicked
do
  vibrate
  call Alert20's Show
  with output
  wasConfirmed
  then do
    if wasConfirmed
    do
      call Data_Viewer_List43's Refresh Data

when Image25 Clicked
do
  navigate to Can Line Machines

when Can Rinsers Opens
do
  call Data_Viewer_List11's Refresh Data
  then do
    when Refresh Data is done

when Data_Viewer_List42 Item Clicked
do
  vibrate
  call Alert19's Show
  with output
  wasConfirmed
  then do
    if wasConfirmed
    do
      call Data_Viewer_List42's Refresh Data
  
```

Figure 64. Code blocks which refresh the simulated spreadsheet containing live data for the can rinser.

```

when Image26 Clicked
do
  When the user taps inside the bounds of the image and releases, trigger the following block(s)

when Can Filler1 Opens
do
  call Data_Viewer_List12's Refresh Data
  then do
    when Refresh Data is done
  
```

Figure 65. Code blocks which refresh the simulated spreadsheet containing live data for the can filler.


```

when Image27 Click
do
  navigate to Can Line Machines

when Can Pasteurizer1 Opens
do
  call Data_Viewer_List13 Refresh Data
  then do
    when Refresh Data is done

when Data_Viewer_List44 Item Click
do
  vibrate
  call Alert21 Show
  with output
  then do
    if wasConfirmed
    do

when Data_Viewer_List45 Item Click
do
  vibrate
  call Alert22 Show
  with output
  then do
    if wasConfirmed
    do
  
```

Figure 66. Code blocks which refresh the simulated spreadsheet containing live data for the can depalletizer

```

when Image28 Click
do
  navigate to Can Line Machines

when Can Coder1 Opens
do
  call Data_Viewer_List14 Refresh Data
  then do
    when Refresh Data is done

when Data_Viewer_List46 Item Click
do
  vibrate
  call Alert23 Show
  with output
  then do
    if wasConfirmed
    do

when Data_Viewer_List47 Item Click
do
  vibrate
  call Alert24 Show
  with output
  then do
    if wasConfirmed
    do
  
```

Figure 67. Code blocks which refresh the simulated spreadsheet containing live data for the can coder.

```

when Image29 Click
do
  navigate to Can Line Machines

when Can Packer Opens
do
  call Data_Viewer_List15 Refresh Data
  then do when Refresh Data is done

when Data_Viewer_List48 Item Click
do
  vibrate
  call Alert25 Show with output
  then do if wasConfirmed
do

when Data_Viewer_List49 Item Click
do
  vibrate
  call Alert26 Show with output
  then do if wasConfirmed
do
  
```

Figure 68. Code blocks which refresh the simulated spreadsheet containing live data for the can packer.

```

when Image30 Click
do
  navigate to Can Line Machines

when Can Palletizer Opens
do
  call Data_Viewer_List16 Refresh Data
  then do when Refresh Data is done

when Data_Viewer_List51 Item Click
do
  vibrate
  call Alert23 Show with output
  then do if wasConfirmed
do

when Data_Viewer_List54 Item Click
do
  vibrate
  call Alert27 Show with output
  then do if wasConfirmed
do
  
```

Figure 69. Code blocks which refresh the simulated spreadsheet containing live data for the can palletizer.

```

when Image31 Click
do
  navigate to Can Line Machines

when Can Stretch Packer Opens
do
  call Data_Viewer_List17 Refresh Data
  then do when Refresh Data is done

when Data_Viewer_List52 Item Click
do
  vibrate
  call Alert29 Show with output
  then do if wasConfirmed
do

when Data_Viewer_List53 Item Click
do
  vibrate
  call Alert30 Show with output
  then do if wasConfirmed
do
  
```

Figure 70. Code blocks which refresh the simulated spreadsheet containing live data for the can stretch wrapper.

```

when Image32 Click
do
  navigate to Keg Line Machines

when De Ullage Opens
do
  call Data_Viewer_List18 Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List54 Item Click
do
  vibrate
  call Alert31 Show
  with output
  then do if wasConfirmed
  do

when Data_Viewer_List55 Item Click
do
  vibrate
  call Alert40 Show
  with output
  then do if wasConfirmed
  do
  
```

Figure 71. Code blocks which refresh the simulated spreadsheet containing live data for the keg De Ullage machine.

```

when Image34 Click
do
  navigate to Keg Line Machines

when External Keg Washer Opens
do
  call Data_Viewer_List19 Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List56 Item Click
do
  vibrate
  call Alert32 Show
  with output
  then do if wasConfirmed
  do

when Data_Viewer_List57 Item Click
do
  vibrate
  call Alert33 Show
  with output
  then do if wasConfirmed
  do
  
```

Figure 72. Code blocks which refresh the simulated spreadsheet containing live data for the external keg washer.

```

when Image35 Click
do
  navigate to Keg Line Machines

when Internal Keg Washer Opens
do
  call Data_Viewer_List20 Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List58 Item Click
do
  vibrate
  call Alert34 Show
  with output
  then do if wasConfirmed
  do

when Data_Viewer_List59 Item Click
do
  vibrate
  call Alert35 Show
  with output
  then do if wasConfirmed
  do
  
```

Figure 73. Code blocks which refresh the simulated spreadsheet containing live data for the internal keg washer.

```

when Image36 Click
do
  navigate to Keg Line Machines

when Keg Coder Opens
do
  call Data Viewer_List21 Refresh Data
  then do When Refresh Data is done

when Data Viewer_List60 Item Click
do
  vibrate
  call Alert36 Show
  with output wasConfirmed
  then do If wasConfirmed
  do
  
```

Figure 74. Code blocks which refresh the simulated spreadsheet containing live data for the keg coder.

```

when Image37 Click
do
  navigate to Keg Line Machines

when Data Viewer_List53 Item Click
do
  vibrate
  # If Start Button is Clicked
  Returns the selected row's id when an item is selected
  when output wasConfirmed
  then do If wasConfirmed
  do

when Keg Blower Opens
do
  call Data Viewer_List22 Refresh Data
  then do When Refresh Data is done

when Data Viewer_List52 Item Click
do
  vibrate
  call Alert38 Show
  with output wasConfirmed
  then do If wasConfirmed
  do
  
```

Figure 75. Code blocks which refresh the simulated spreadsheet containing live data for the external keg blower.

```

when Image38 Click
do
  navigate to Keg Line Machines

when Keg Filler1 Opens
do
  call Data Viewer_List23 Refresh Data
  then do When Refresh Data is done
  
```

Figure 76. Code blocks which refresh the simulated spreadsheet containing live data for the external keg filler.

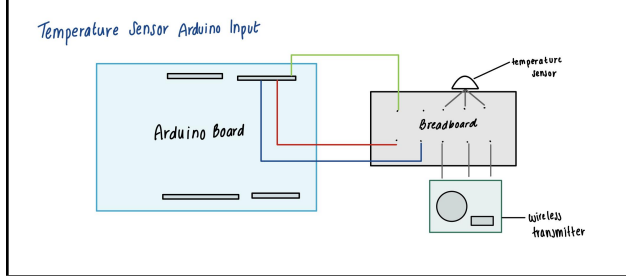


Figure 77. A Temperature Sensor Arduino Uno Hardware Component (reference. Deliverable E)

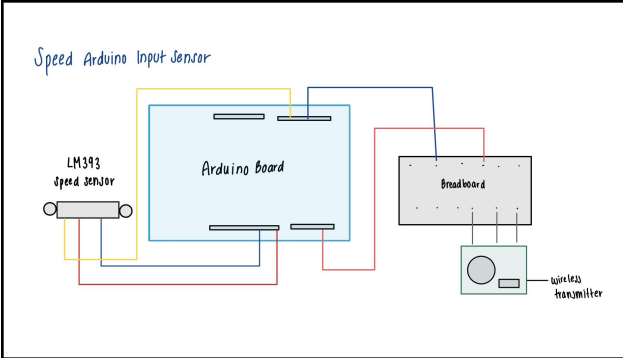


Figure 78. A Speed Sensor Arduino Uno Hardware Component (reference. Deliverable E)

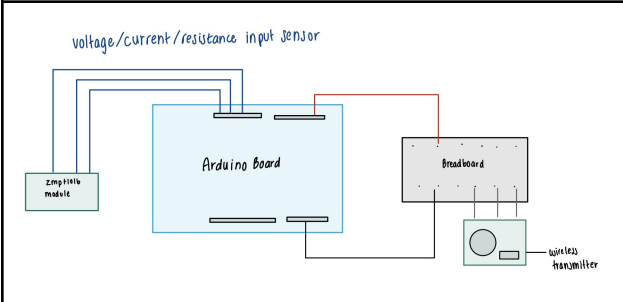


Figure 79. A Voltage Sensor Arduino Uno Hardware Component (reference. Deliverable E)

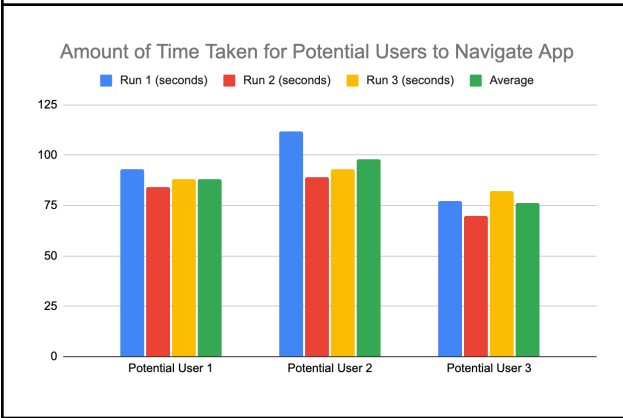


Figure 80. Graphical Representation of the Analytical Model to gauge users' ability to navigate through the app. (reference. Deliverable H)

List of Tables

Table 1. Acronyms	vii
Table 2. Glossary	vii
Table 3. Referenced Documents	10
Table 4. User Restrictions	11

List of Acronyms and Glossary

Table 1. Acronyms

Acronym	Definition
UI	User interface
UX	User experience

Table 2. Glossary

Term	Definition
Bottleneck	A station in the production lines of each product - cans, kegs, and bottles that is the filler station of the production line and is the determining factor in how much product is able to be produced.

1 Introduction

Document Structure and Purpose

Presented with the task of increasing production speed by 2% for the Mill Street Brewery production lines, our group, C8, was prompted to provide a solution that was able to interpret and analyze input data, and output accurate recommendations for users to make to their system to optimize production speed. Through following the steps of the design process which include empathizing with the client, defining the issue, ideating a solution, formulating prototypes, and testing our prototypes, we were able to create a solution that effectively fulfilled our client's requirements.

This document contains an overview of the product, a generalized walkthrough of the product from startup to exit, a detailed description of each user function and feature, a description of recovery and error correction procedures, an explanation of how the final prototype was built, and possible work that can be conducted for future prototypes. The purpose of this document is to provide a foundation for other students and the client to completely replicate the final product. This document will relay the process that was constructed in order to create the final fully-functioning prototype that was presented to the client.

This document is organized in a way that aims to be entirely comprehensive to new potential users or clients who have an interest in expanding on the work done to create the final project. As such, the document first provides a general overview of the project. In this section, the problem that the group was presented with is revisited and the fundamental requirements of the user are explained. Additionally, the key features of the product that distinguish our product from others are explained and the reasons for their implementation are justified. The major functions of the product are briefly explained and the construction of the system is provided. Finally, a block diagram explaining the way in which the system works is provided and a brief explanation of the transition between pages is given. The first subsection of this selection explains the conventions of the final product in which stylistic and command syntax are justified. The second subsection identifies the precautions that must be taken by the user before using the prototype, including but not limited to a copy of permissions that must be obtained prior to using the device.

In the following section, a generalized walkthrough of the system is provided from startup to exit. This section provides a comprehensive insight into the final product with figures and labels given. This section also defines user access considerations required for the product to function. In this section, the types of inputs and outputs are given in addition to the databases that the product takes information from. This selection also provides a comprehensive explanation how to access the system using a user-prompted password and username. The organization of the system is also explained in which a flow chart is used to describe the progression of the user through the app. Finally, a comprehensive explanation detailing how to properly exit the app is given.

The following section provides an explanation of how to use the system. In this selection, each page of the app is provided with a detailed explanation of their function. Subsections of this section include the sign-in page, the can line, the keg line, the bottle line, and the system settings. These subsections also have their own respective subsections that are relevant to their function.

The next section details troubleshooting and error-correction protocol which need to be followed in order to address potential issues that arise in the system. These contingency plans will detail the exact issue that may occur within the system and then provide a procedure to follow in order to correctly mitigate the issue without causing further damage to the system. Subsections for this section include error messages or

behaviors, maintenance and support. Error messages or behaviors details the appropriate notifications that will be sent to the user if an issue arises. Maintenance details the appropriate upkeep that must be done to the app in order to guarantee that the app runs at optimal capacity. Finally, support details emergency responses that will be taken by the team should such a situation arise. In this subsection, contact information will be explicitly stated for potential users to use in the instance of an emergency.

The following section provides a thorough explanation of how the final prototype was built. This section is split into its appropriate categories including the software's user interface, the UX design, the databases used, and the calculations that were made in order to fulfill the client's needs. The materials that were used to complete the product are listed and the corresponding bill of materials is created. Instructions for how each subsystem works is also provided. Finally, the testing and validation methods for each prototype subsystem is provided.

The last subsection concludes the document by providing a summary of the relevant lessons learned in addition to possible extensions that can be made on future prototype iterations to develop the work done on the current prototype. The avenues that would provide the most productive construction on the current prototype will be outlined and appropriate action plans will be constructed.

Security, Safety, and Privacy Considerations

This user manual is intended to only be viewed by the client and potential users of the product as approved by the client. Any other party attempting to view the contents of this document is strictly prohibited unless special permission is granted by the client. This document contains sensitive information regarding our client's plans to increase production and therefore the appropriate security and privacy considerations must be taken into consideration.

Simplifying Assumptions

The assumptions that were made in order to simplify the calculations and implementation of our solution include the following:

- Every production line including the can, bottle, and keg lines have a bottleneck station which is the filler station.
- Every machine on each production line runs for exactly 12 hours a day without any downtime.
- The volume of product that the user would like to produce within a given day is immutable.

2 Overview

The problem at hand is the need to increase production speed for Mill Street Brewery production lines by 2%. This is an important problem to solve because increasing production speed will result in higher output and greater revenue for the company. The fundamental needs of the user include a solution that can accurately interpret and analyze input data, and provide recommendations for users to optimize their production speed. The solution must also be user-friendly and easy to use.

What differentiates our product from others is our approach to solving the problem. We followed the design process which includes empathizing with the client, defining the issue, ideating a solution, formulating prototypes, and testing our prototypes. Our product also features a comprehensive explanation of how to use the system, troubleshooting and error-correction protocol, and a thorough explanation of how the final prototype was built. The key aspects that make our product better are its user-friendliness, comprehensiveness, and the fact that it was built with the specific needs of Mill Street Brewery in mind.

The key features of the product include the ability to interpret and analyze input data and provide accurate recommendations to optimize production speed for the Mill Street Brewery production lines. The major functions of the product include data input, data analysis, and recommendation output. The system is constructed using a web-based architecture and can be accessed through a user-prompted password and username.

The system requires access to the production line data through a direct interface with the brewery's systems. To ensure proper functioning of the system, users must follow specific conditions such as providing accurate and complete input data, and following the troubleshooting protocols when required.

The block diagram of the system includes a data input module that collects input data from the brewery's systems, a data processing module that analyzes the input data and a recommendation output module that provides accurate recommendations to optimize production speed. The system's design ensures that the user can easily navigate through the GUI and access the required modules without any difficulty.

3 Getting started

3.1 Configuration Considerations

Databases and Libraries Required for Configuration

- +
 - +
 - +
- Bottle Line Data
Can Line Data
Keg Line Data

The appropriate configuration of the system involves interconnecting the libraries and databases that are to be provided to the software where the data can then be used in calculations with user-input in order to create the optimal range of values. As verified by the client during client meetings, sensors that record speed data are established in the facility where all production lines run. The data that is collected by the sensors is then delivered to the person in charge of operating the software. The format of the information is delivered in a JSON file, and must therefore be converted into a .csv file for the information to be readable to the software. Once this conversion has taken place, the information must be organized as such: each production line is represented by their own spreadsheet. In these spreadsheets, the name of the machine is written in the first row. The second row is the speed of the machines as recorded by the sensors. Once these values have been entered, the spreadsheets must be linked to the software through inputting data lines in the software implementation. Once this configuration is complete, the app is fully functional and ready for use.

3.2 User Access Considerations

There are a few important users and/or groups of users that would regularly be using the system. The table below (table 4) demonstrates in detail who these users are and how they are restricted by their accessibility within the app.

Table 4. User Restrictions

Users and/or User Groups	Accessibility Restrictions
The 27 employees working at Mill Street Brewery (morning shift vs evening shift)	BrewMetrics has a ‘password settings’ section that is also password protected where managers are able to change the password of the system if needed. Morning and evening employees are not able to change the password to the entire system because they are not provided with the password needed to access that section. Morning and Night employees also do not have access to previous shifts optimization data through the app.
Managers and supervisors at Mill Street Brewery	Managers and supervisors do not have access to optimization data history from when they are not in the brewery.
IT/Maintenance individuals	IT/maintenance individuals do not have access to the system without sign in password assistance. They also can not change the password or access optimization data from when they were not in the brewery.

3.3 Accessing/setting up the System

The app BrewMetrics is available on the Google play store as well as the App store, so the following steps will make sure setting up the system occurs smoothly on any platform. This section of the manual will describe to the users how to initially use the system. Only managers and supervisors are able to set up the system and change the password, whereas management and employees are able to access the system and use it for day to day analyses/recommendations.

3.3.1 System Setup

The following list of steps is for the manager/supervisor to set up the system.

Step 1. Download “BrewMetrics” from the Google play store if you are using a tablet, or from the App store if you are using an iPad.

Step 2. Open up the application and click the “system settings” button. Then click the “Change Password” button.

Step 3. Enter a username and password that you will provide to all employees at the brewery and click “Save Changes”.

Step 4. At this point you can try exiting out of the app and refreshing it by clearing it from device history and then reopening it. You should now be prompted to the sign in page.

Step 5. Enter the username and password you just created into the designated sign in sections and click “Sign in”.

Step 6. At this point the system set up process has been completed and you are ready to start using BrewMetrics.

3.3.2 Password Change

The following list of steps are for managers/supervisors trying to change the password to the system.

Step 1. Open the app and click “Sign In”.

Step 2. Enter the username and password and click “Sign In”.

Step 3. Click the “System Settings” button.

Step 4. Click the “Change Password” button.

Step 5. Enter the already set username. Then enter the new desired password and confirm it by entering the same password again in the last section.

Step 6. Click “Save Changes”. Now the process for changing the password is completed.

3.3.3 System Access

The following list of steps are for employees trying to access the system.

Step 1. On the tablets or iPads provided to you at the brewery click on the “BrewMetrics” icon to open the app.

Step 2. You will be prompted to the user sign in page. Enter the username and password provided to you by the manager/supervisor.

Step 3. At this point you have access to the application and are now viewing the homepage.

Step 4. By clicking the “System Settings” button you can then click the “Tutorial” button and watch a video that visualizes exactly how the app works.

3.4 System Organization & Navigation

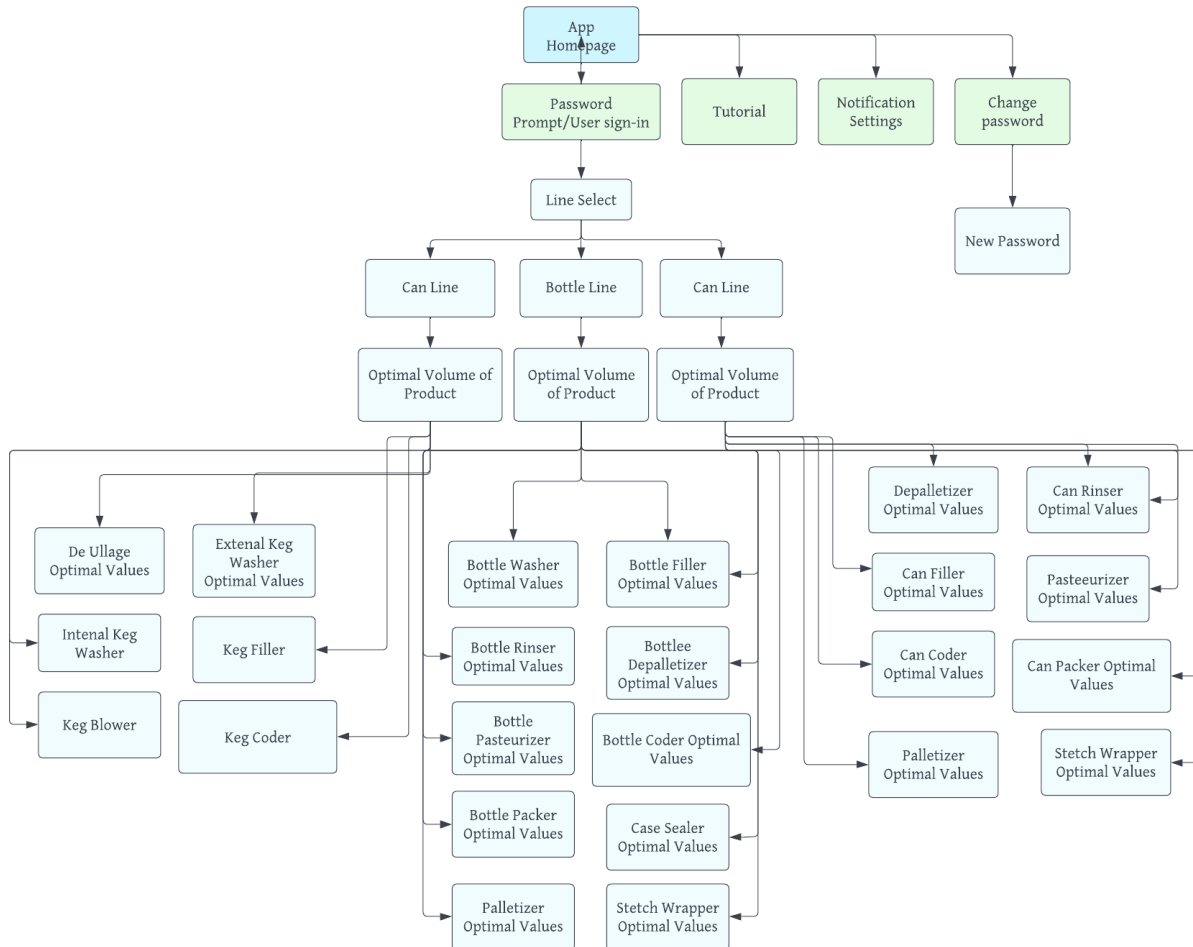


Figure 1. A flow chart depicting the system’s organization and how navigation through the app functions.

3.4.1 The Welcome Page

The welcome page of the app is what the user views once they download the app. This page prompts the user to sign into the app with a 'sign-in' button displayed underneath the logo of the client. Once selected, the user will then be prompted to enter any relevant login information. If this information is entered and it coincides with login information stored in the IT operator's system, then the login process is successful and the user is allowed to navigate through the app.

3.4.2 Home Page

The home page is what the user is brought to after a successful sign in. Here they are prompted to select either the line select button, or the systems settings button. These two options were included to prioritize readability in addition to making the app easier to navigate for the user.

3.4.3 Systems Settings

Once the user has selected system settings from the home page they have access to settings that maximize their comfort on the app. Here, they are able to alter the notifications that they receive from the app in terms of frequency and volume, watch a video tutorial to become well acclimated to the app's interface, and they are given the ability to change their password if they feel the need to do so concerning security reasons.

3.4.4 Line Select and User-Input

The line select option is what the user is able to see from the home page. Once this option is selected, the user will then be prompted to look at data for a specific production line. These options include the can line, bottle line, and the keg line. Once one of these lines are selected, the user is then prompted to enter an ideal volume of product that they would like to produce for that given work day. Once this value is entered, they will be taken to view a comprehensive list of machines available within that specific production line.

3.4.6 Optimal Range of Speeds

Once the optimal volume of product is entered, users will be given the opportunity to select certain machines from the production line. Once a specific machine is selected, they will be able to view the optimal range of speeds for that specific machine. Users are able to navigate back and forth to different machines after having entered an ideal volume of product for that specific day. If they would like to enter an ideal volume of product for a different production line, they are able to do so by simply clicking the back button and navigating to a different production line from the 'select line' button.

3.5 Exiting the System

The system can be exited in any of the pages because there are no in app processes that needs to be performed before exiting. When exiting out of the system the password will automatically reset and users will have to re-enter the username and password when opening the app again. To exit out of the app, depending on the device, close the app like you would for any other app. For most tablets there is a home button to click at the bottom of the screen, here there is also a button that gives the option to clear the app from device history as well by swiping up. For older iPads there is a circle home button that you need to click to exit the system, double clicking this button will allow you to clear the app from device history by swiping up. For newer iPads at the bottom of the screen, swipe up to exit the system. Also from the bottom of the screen, swipe up and hold to be able to clear it from device history by swiping up again.

4 Using the System

4.1 Sign In

This is the initial page that is opened when a user opens the app. When the sign in button is clicked the user is brought to a new page that prompts them to input their username and password so that they may access the full app.

4.2 Line select

When the Line select button is clicked the user is brought to a new page that contains the can, bottle, and keg line buttons.

4.2.1 Can Line

When the Can Line button is clicked the user is brought to the can user input page.

4.2.1.1 User input

Here the user is prompted to “please enter the optimal volume of product that you would like to be produced”. They would then enter the number of cans they would like to produce that day and click the enter button which brings the user to the can line data page.

4.2.1.2 Can Line Data

Here the user can see a list of buttons that have the can line machines on them. These machines are the depalletizer, rinser, filler, pasteurizer, coder, packer, palletizer, and stretch wrapper. The user can click on any of the machines to view the data for that specific one. For example if the user was to click the Can Rinser button they would see a minimum speed, optimal speed, and maximum speed for the rinser.

4.2.2 Bottle Line

When the Bottle Line button is clicked the user is brought to the bottle user input page.

4.2.2.1 User input

Here the user is prompted to “please enter the optimal volume of product that you would like to be produced”. They would then enter the number of bottles they would like to produce that day and click the enter button which brings the user to the data page.

4.2.2.2 Bottle Line Data

Here the user can see a list of buttons that have the bottle line machines on them. These machines are the depalletizer, rinser, filler, pasteurizer, coder, packer, case sealer, palletizer, and stretch wrapper. The user can click on any of the machines to view the data for that specific one. For example if the user was to click the Bottle Packer button they would see a minimum speed, optimal speed, and maximum speed for the packer.

4.2.3 Keg Line

When the Keg Line button is clicked the user is brought to the keg user input page.

4.2.3.1 User input

Here the user is prompted to “please enter the optimal volume of product that you would like to be produced”. They would then enter the number of kegs they would like to produce that day and click the enter button which brings the user to the data page.

4.2.3.2 Keg Line Data

Here the user can see a list of buttons that have the keg line machines on them. These machines are the de ullage, external keg washer, internal keg washer, coder, blower, and filler. The user can click on any of the machines to view the data for that specific one. For example if the user was to click the Keg Coder button they would see a minimum speed, optimal speed, and maximum speed for the coder.

4.3 System Settings

When the system settings button is clicked the user is brought to a new page which contains the notifications, change password and tutorial buttons.

4.3.1 Notifications

When machines are not running at optimal speeds the app displays a message telling the user to either slow down or speed up the machine. When a machine is running too slow the message displayed is “Speed up! The *machine name* is running below the minimum optimal speed” and when it is going too fast the message says”Slow down! The *machine name* exceeds the maximum optimal speed”.

4.3.2 Change Password

When the change button password is clicked the user is brought to a new page where there are fields to enter their username, new password and confirm the new password. In order for the app to save the new password, the user name should be entered properly and both passwords should match.

4.3.3 Tutorial

When the tutorial button is clicked the user is brought to a new page where a tutorial video automatically starts playing. The video describes to the user how to use all of the different features of the app. The user can pause, rewind, or skip forward in the video.

5 Troubleshooting & Support

5.1 Error Messages or Behaviors

5.1.1 Internal Server Error: This indicates that there is a problem with the server, which could be due to a coding error, hardware malfunction, or configuration issue. To resolve the issue, you should get in touch with the website administrator or try accessing the website at a later time.

5.1.2 Crashing or freezing: This can happen in a device for various reasons like lack of memory, incompatible software, or a virus. To fix this issue, one can restart the device and remove any conflicting software or viruses.

5.1.3 Hardware failure: Parts of a system may break due to physical damage, aging, or overheating. The corrective action is to replace the faulty hardware component.

5.1.4 Slow performance: This behavior can be due to insufficient memory, malware, outdated software or hardware, or system overload. The corrective action is to optimize the system performance, such as removing malware, updating software, or upgrading hardware.

5.2 Special Considerations

5.2.1 Data collection: Ensuring the accuracy and reliability of data input into the system is essential as any problems with data collection may lead to erroneous analysis and recommendations.

5.2.2 Equipment compatibility: The system needs to be compatible with the equipment used in the brewery. This includes the type of conveyor systems and sensors used to measure speed.

5.2.3 Integration with existing systems: The system should be designed to integrate with existing brewery systems and software to ensure seamless operation.

5.2.4 Real-time monitoring: The system should be capable of monitoring production line speed in real-time. This will enable quick identification and resolution of any issues that arise.

5.2.5 Security: The system should be secure to prevent unauthorized access and data breaches.

5.3 Maintenance

5.3.1 Calibration: If the prototype includes sensors or other measurement devices, it may require regular calibration to ensure accurate readings. Calibration should be performed according to the manufacturer's instructions or industry standards.

5.3.2 Software updates: If the prototype includes software, it may require regular updates to fix bugs or security issues, or to add new features or functionality. Updates should be installed according to the manufacturer's instructions.

5.3.3 Testing: Regular testing of the prototype is important to ensure that it is still functioning as intended. This may involve running tests or simulations to check for proper operation or performance or conducting user testing to identify any usability issues.

5.3.4 Password management: Passwords should be strong, unique, and changed regularly. Passwords should not be shared among different accounts or stored in unsecured locations such as sticky notes or spreadsheets.

5.4 Support

For emergency contact, you can email the production support team for help regarding the app:

Chanelle Menard: cmena097@uottawa.ca

Elisha Khandaker: ekhan082@uottawa.ca

Jonathan Aguilera: jagui077@uottawa.ca

Rachelle Choi: rchoi076@uottawa.ca

Sorosh Benvidi: shoss111@uottawa.ca

If a problem has been identified, please contact any member of the production support team. Add the subject “Emergency Assistance and System Support” in the email so we can resolve the issue faster.

6 Product Documentation

6.1 Databases & Libraries

6.1.1 BOM (Bill of Materials)

Table 1. Bill of Materials for Subsystem 1 - Databases and Libraries

Item	Cost	Link
Thunkable Premium	\$45	https://x.thunkable.com/projects/63fb9f93f165ed2407403950/a90ac201-d688-46d8-aaf1-a3e8d99c4232/designer

6.1.2 Equipment List

Table 2. Equipment List for Subsystem 1 - Databases and Libraries

Item	Link
Google Spreadsheets	+ Bottle Line Data + Can Line Data + Keg Line Data

6.1.3 Instructions

The databases that were used to transfer data to the app were provided by the client, however, this data had to be restructured in order to better cohere to the interface that was used to create the app. It was essential that the app building software used to create the user interface was capable of collecting, interpreting, and analyzing data from this database in order to make accurate calculations.

1. The client provides a database with live data that is collected using sensors that have already been implemented on the floor of the brewery.
2. This live data is then transferred to three separate spreadsheets. In each of these spreadsheets is information that conveys the current speeds of each machine for the can line, keg line, and bottle line. An example of how the spreadsheets are formatted can be seen in figures 2-4.

De Ullage	External Keg Wa	Internal Keg Was	Keg Coder	Keg Blower	Keg Filler
12	14	16	17	20	27

Figure 2. Spreadsheet containing live data for the keg line

Can Depalletizer	Can Rinser	Can Filler	Can Pasteurizer	Can Coder	Can Packer	Palletizer	Stretch Wrapper
152	162	175	200	228	243	272	282

Figure 3. Spreadsheet containing live data for the can line

Bottle Depalletize	Bottle Rinser	Bottle Filler	Bottle Pasteurize	Bottle Coder	Bottle Packer	Case Sealer	Palletizer	Stretch Wrapper
375	337	300	357	392	412	447	496	542

Figure 4. Spreadsheet containing live data for the bottle line

3. Once this live data is filtered into the spreadsheet, a calculation will be done using data that was provided by the client in the earlier stages of product development. In order for these calculations to be completed, the following assumptions were made:
 - a. The machines run for at least 12 hours a day without any breaks.
 - b. The amount of product that the user would like to create within a given span of time remains consistent for the entire time the machines are running.
 - c. The machines do not experience major blockages or malfunctions.
4. The calculations are produced upon receiving data for the filler stations of every production line. The data for the production lines includes the amount of product per hour (for kegs) and per minute (for cans and bottles). Using this data, the theoretical yield of the product was calculated.
5. Once the theoretical yield was calculated, a ratio for the filler speed by the volume of product created was made. This ratio is then able to be multiplied by the user-input ideal amount of product they would like to make for the day, and the optimal speed for the filler station of that production line is calculated.
6. The machines that precede and succeed the filler have their speeds calculated by using calculations that adhere to the V-curve theory. This theory suggests that all production lines have their own bottle neck that is the slowest running machine in the line. In order to create the most optimal procedure to fulfill the production line's objectives, all machines that precede and succeed this bottle neck must be running at a speed that is 10% faster than the bottleneck. Therefore, as a result of this theory all machines that come before and after the bottleneck will be calculated by increasing 10% of the speed from the filler station.
7. Once all these calculations have been processed by software, it will then be displayed on a screen in the form of an inequality:
 - a. **Minimum Value <= Optimal Value <= Maximum Value**

Sample Calculation

Information provided by the client:

- Filler speed for the can line: 300 cans/minute

1. Calculating the theoretical yield of product

$$300 \cdot 60 = 18000 \text{ cans per hour}$$

$$18000 \cdot 12 = 216000 \text{ cans per day/in a 12-hour shift}$$

$\therefore 216000 \text{ cans is the theoretical yield of a 12 hour work day}$

2. Calculating the ratio to apply to the user-entered volume

$$\frac{300 \text{ cans per minute}}{216000 \text{ cans per shift}} = 1.388 \cdot 10^{-3}$$

3. Calculating the filler speed for a user-entered ideal volume of product

- Ex. User entered value: 300000 cans

$$(1.388 \cdot 10^{-3})(300000) = 417 \text{ cans per minute}$$

∴ In order to produce 300000 cans within the 12 hour work day, the filler station should be producing 417 cans per minute.

4. Calculating the machines preceding and succeeding the filler station in the production line

Table 3. Can Line Machines Optimal Speeds Sample Calculations

Can Depalletizer	Can Rinser	Can Filler	Can Pasteurizer	Can Coder	Can Packer	Palletizer	Stretch Wrapper
505	459	417	459	505	556	612	673

Table 4. All provided and calculated sample data.

Bottle Line Ratio	Can Line Ratio	Keg Line Ratio
1.388 x 10 ⁻³	1.388 x 10 ⁻³	0.083
Optimal volume of product	Optimal volume of product	Optimal volume of product
216,000	126,000	300
Filler Speed	Filler Speed	Filler Speed
300	175	25

6.2 UX and UI Design

6.2.1 BOM (Bill of Materials)

Table 5. Bill of Materials for Subsystem 2 - UX and UI Design

Item	Cost	Link
Thunkable Premium	\$45.00	https://x.thunkable.com/projects/63fb9f93f165ed2407403950/a90ac201-d688-46d8-aaf1-a3e8d99c4232/designer

6.2.2 Equipment list

Table 6. Equipment List for Subsystem 2 - UI and UX Design

Item	Link
Google Spreadsheets	+ Bottle Line Data + Can Line Data + Keg Line Data

6.2.3 Instructions

The UI and UX designs for the final solution were implemented using the features available on the app-building software that was utilized. The app-building software that was selected was chosen based on its user-friendly interface and surplus of resources that could be used to learn the interface in more depth. Using these resources, we were able to create a comprehensive product for the client. The UI and UX features were tested rigorously throughout the testing phase of the design process to ensure that it was up to standard for potential users.

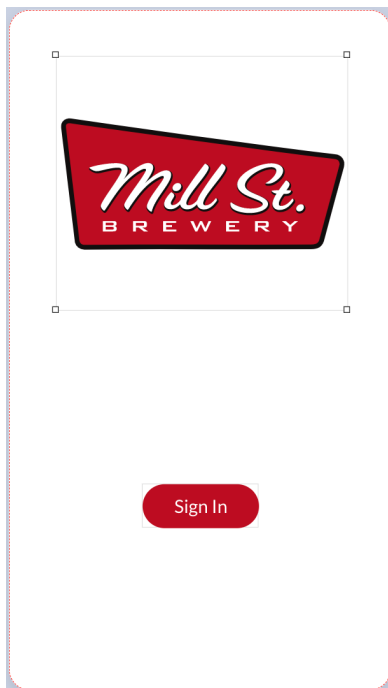


Figure 5. The homepage of the app that users see upon opening the app.

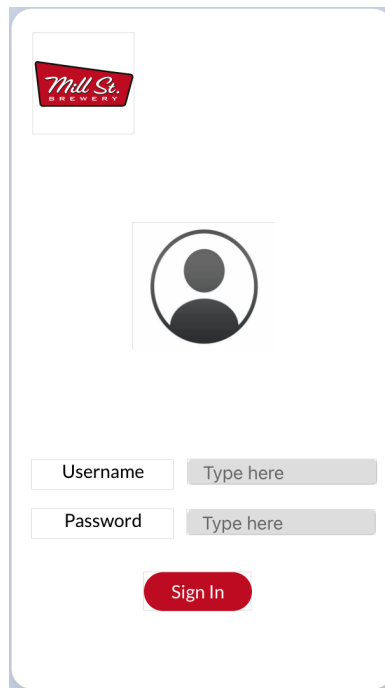


Figure 6. The sign-in page that users are taken to once clicking on the sign-in button.

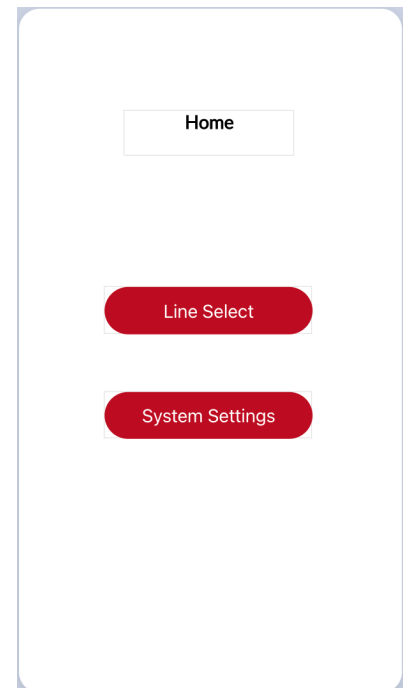


Figure 7. The homepage which users are able to view upon successfully signing in.

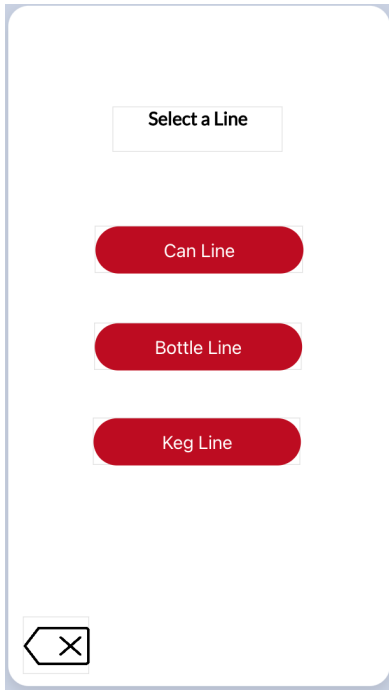


Figure 8. The Select line page which users will navigate to once they have selected this option from the home page.

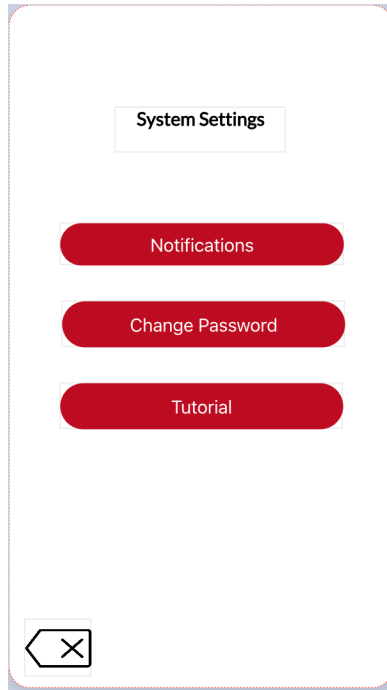


Figure 9. The systems settings page that users will navigate to once selecting the 'system's settings' button from the home page.

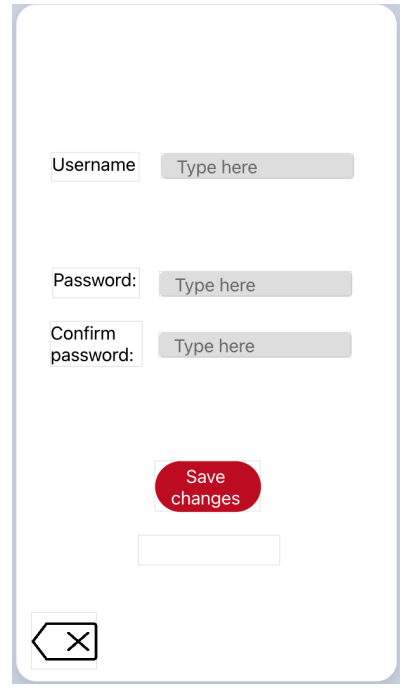


Figure 10. The change password page which users can access once selecting on this option from the system settings.

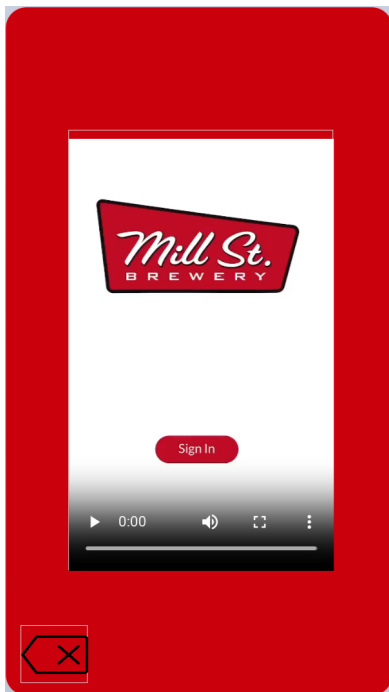


Figure 11. The tutorial page

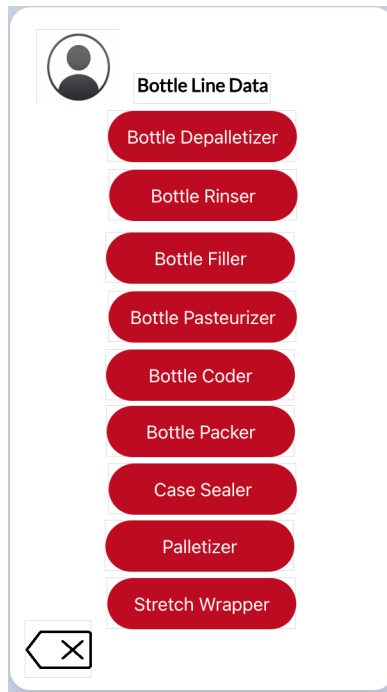


Figure 12. The Bottle line Data

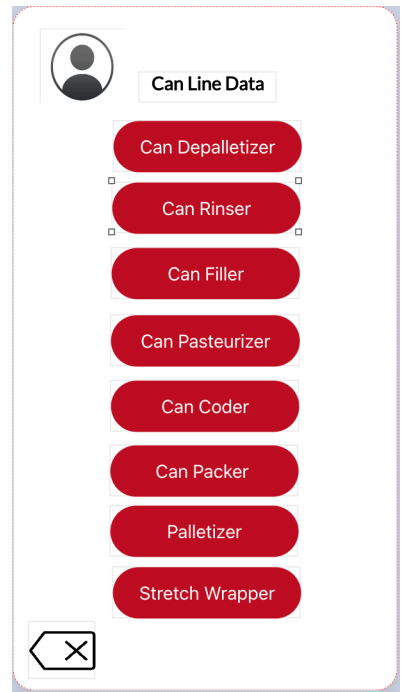


Figure 13. The Can line Data

that shows users a video of how to navigate through the app. This is accessible from the system settings.

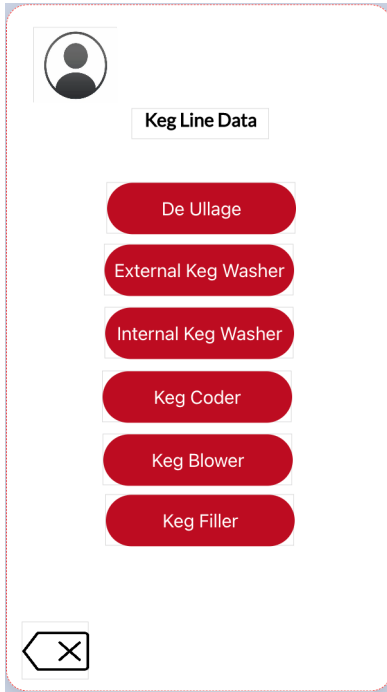


Figure 14. The Keg line Data that is available from the select line page.

that is available from the select line page.

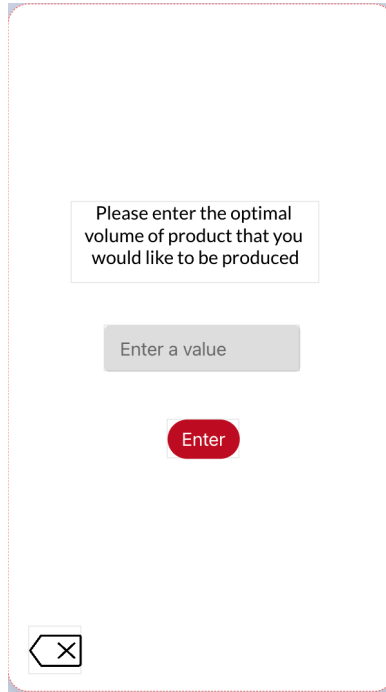


Figure 15. The users are prompted to enter the ideal volume of bottles they would like to produce within a given day.

that is available from the select line page.

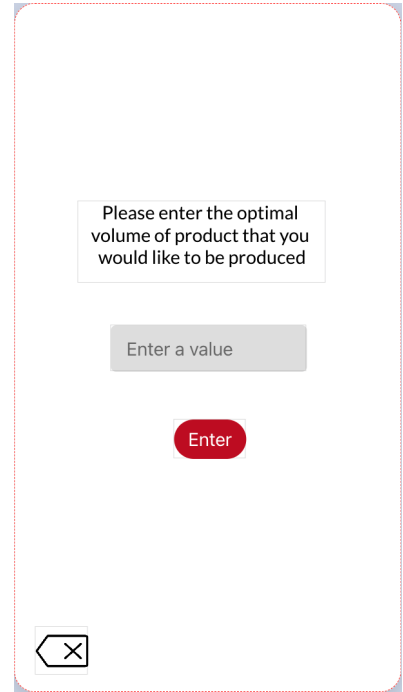


Figure 16. The users are prompted to enter the ideal volume of cans they would like to produce within a given day.

Please enter the optimal volume of product that you would like to be produced

Enter a value

Enter

Figure 17. The users are prompted to enter the ideal volume of kegs they would like to produce within a given day.

Bottle Depalletizer Optimal Speed

Speed

368 ≤ 365 ≤

Figure 18. The optimal speed range for the bottle depalletizer.

Bottle Rinsers Optimal Speed

Speed

330 ≤ 334 ≤ 363 ≤

Figure 19. The optimal speed range for the bottle rinsers.

Bottle Coder Optimal Speed

Speed

363 ≤ 364 ≤ 399 ≤

Figure 20. The optimal speed range for the bottle coder.

Pasteurizer Optimal Speed

Speed

330 ≤ 354 ≤ 363 ≤

Figure 21. The optimal speed range for the bottle

Bottle Filler Optimal Speed

Speed

300 ≤

Figure 22. The optimal speed range for the bottle filler.

pasteurizer.

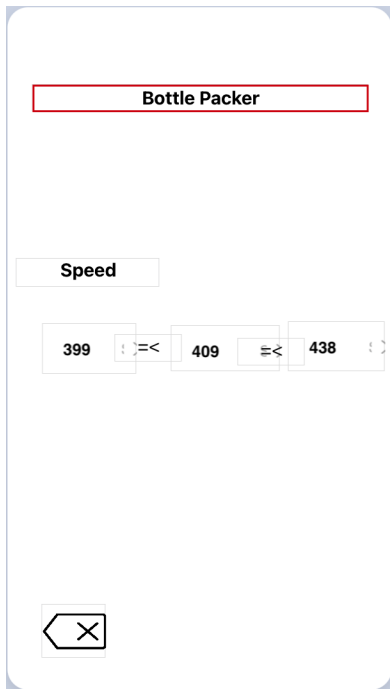


Figure 23. The optimal speed range for the bottle packer.

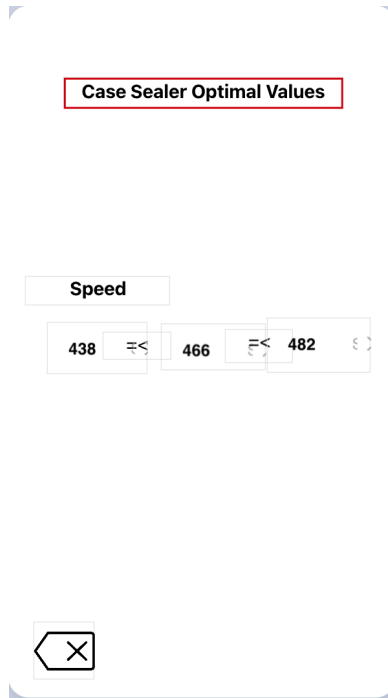


Figure 24. The optimal speed range for the bottle case sealer.

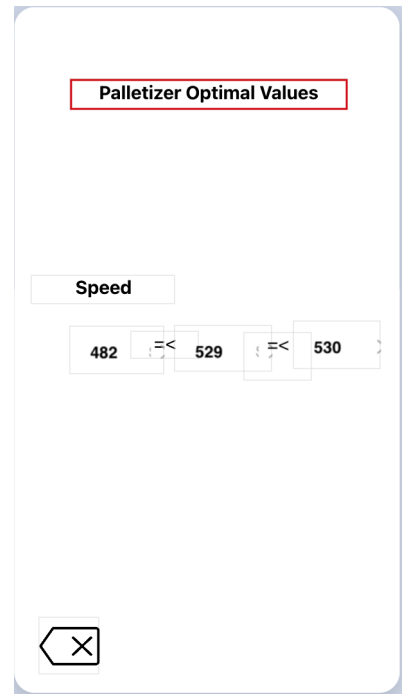


Figure 25. The optimal speed range for the bottle palletizer.

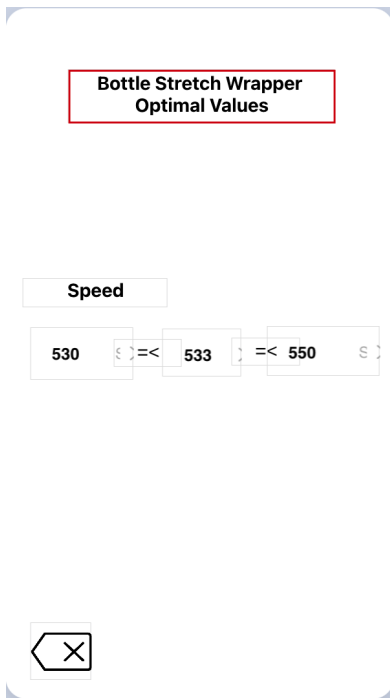


Figure 26. The optimal speed

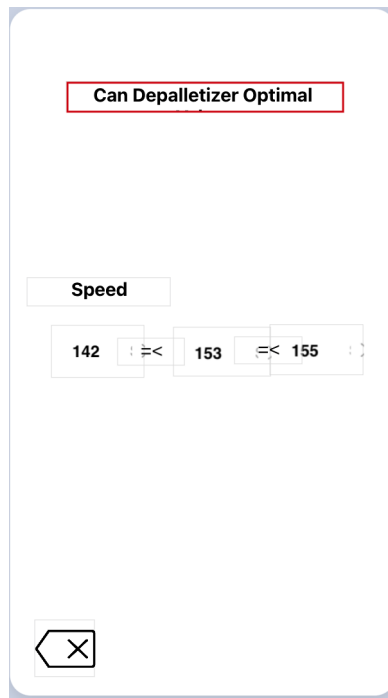


Figure 27. The optimal speed

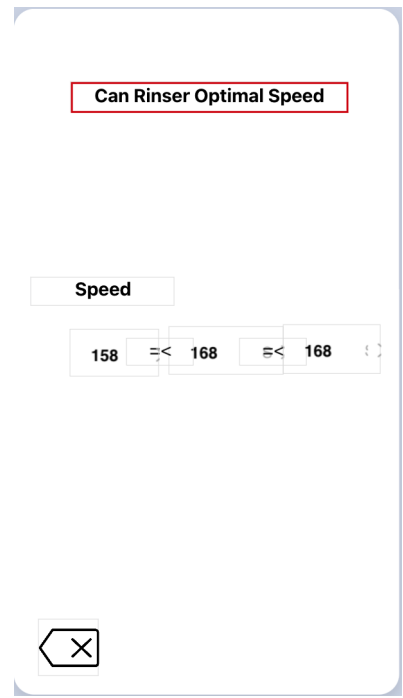


Figure 28. The optimal speed

range for the bottle stretch wrapper.

range for the can depalletizer.

range for the can rinser.

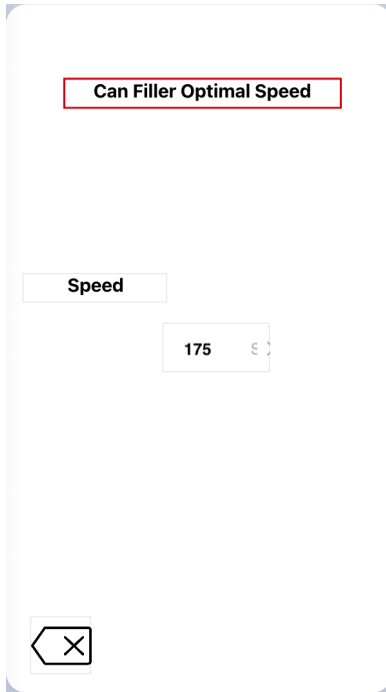


Figure 29. The optimal speed range for the can filler.

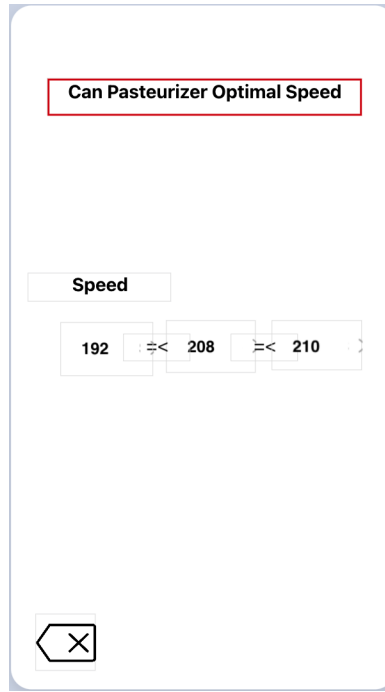


Figure 30. The optimal speed range for the can pasteurizer.

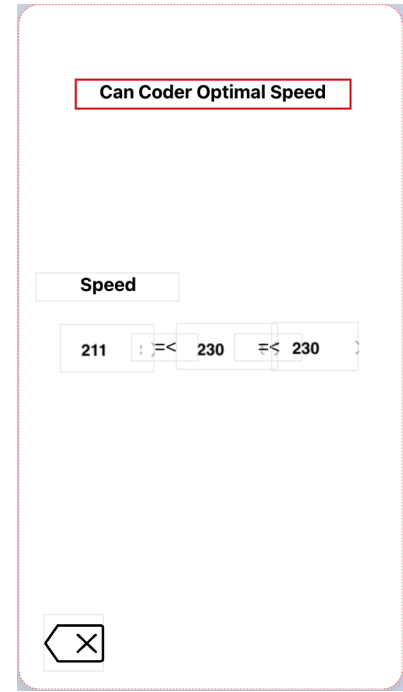


Figure 31. The optimal speed range for the can coder.

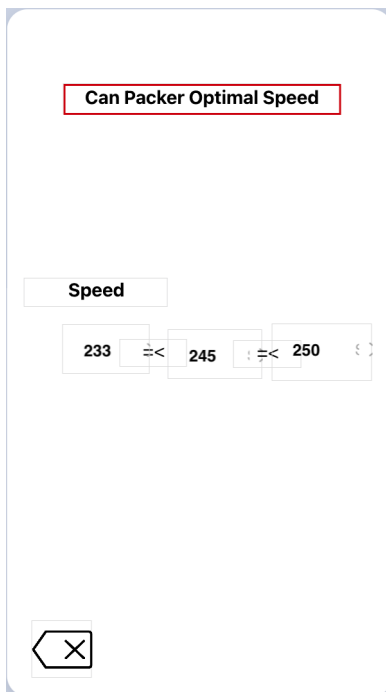


Figure 32. The optimal speed

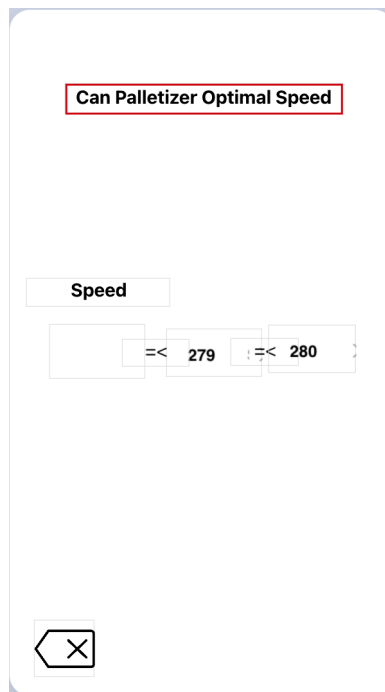


Figure 33. The optimal speed

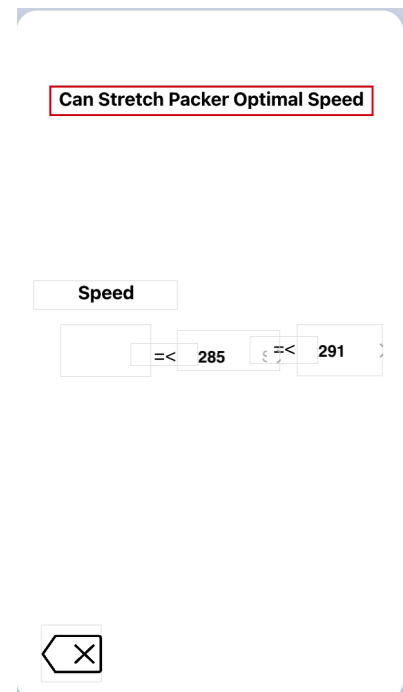


Figure 34. The optimal speed

range for the can packer.

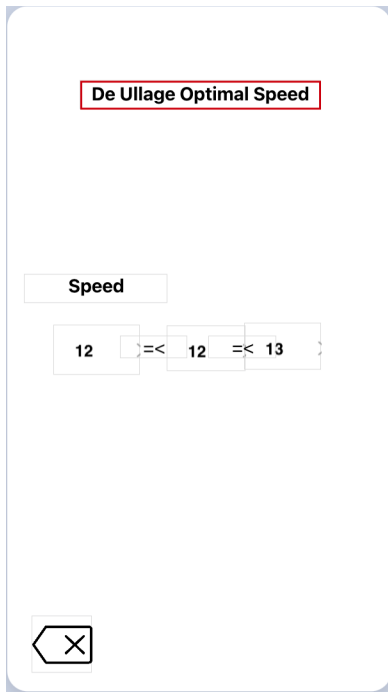


Figure 35. The optimal speed range for the De Ullage Machine.

range for the can palletizer.

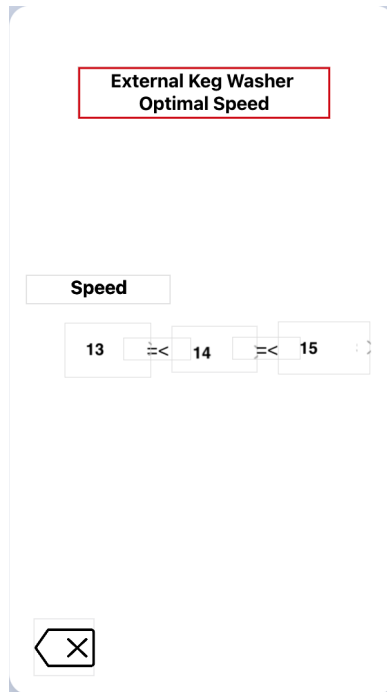


Figure 36. The optimal speed range for the external keg washer.

range for the can stretch packer.

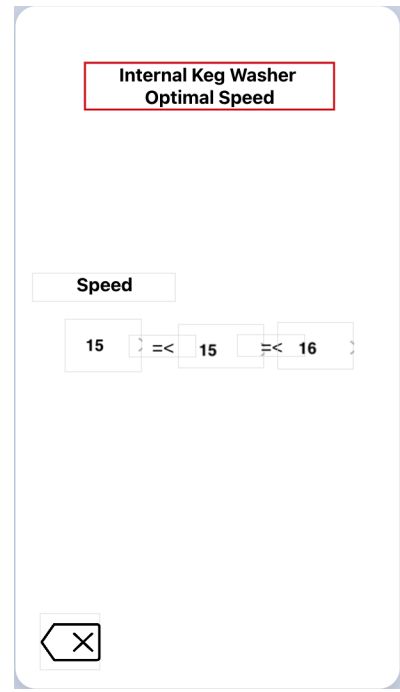


Figure 37. The optimal speed range for the internal keg washer.

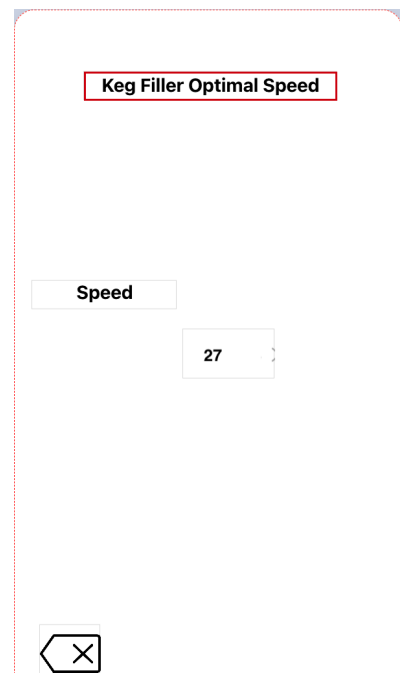
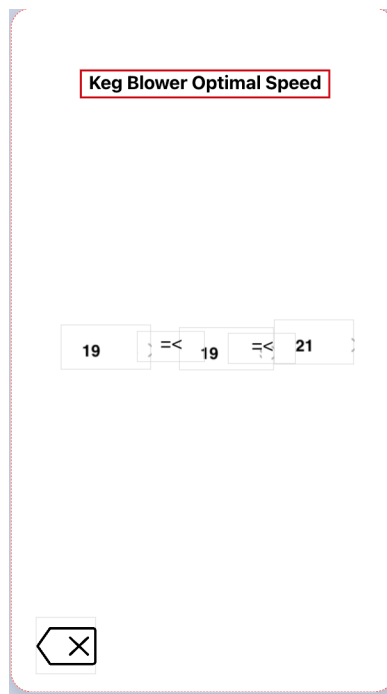
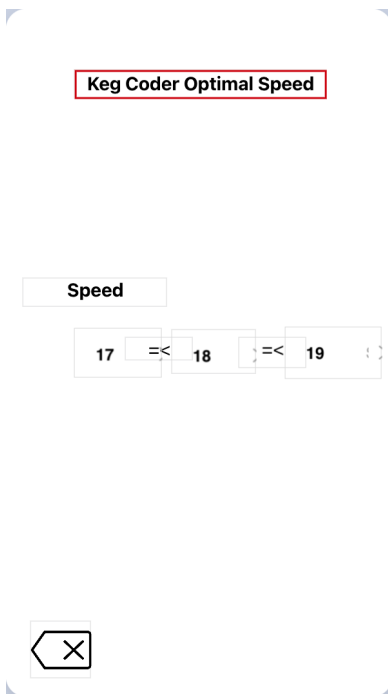


Figure 38. The optimal speed range for the keg coder.

Figure 39. The optimal speed range for the keg blower.

Figure 40. The optimal speed range for the keg filler.

Many of the pages within the app contain very similar stylistic traits. The goal of the user design was to prioritize readability and clarity of the information being delivered. This was to satisfy the user requirements of the app being highly accessible to already existing employees in addition to new employees who may be hired.

Creating the Welcome Page:

1. Save a .png image of the client's logo from a web browser and upload it into the app building software.
2. Create a button and name it 'sign-in'.
3. Recolor the button red in order to match the logo of the client.

Creating the Sign in page

1. Use the downloaded logo .png image and place it in the corner of the screen.
2. Download a profile picture icon from your primary web browser and drag it into the middle of the screen.
3. Take a text box and name it 'Username'.
4. Text a text box and name it 'Password'.
5. Take two text input boxes and place them next to the username and password text boxes. This is where the user will be prompted to enter their username and password.

Creating the Homepage/Select Line Page/System Settings Page

1. Drag buttons into the page from the features select icons on the left-hand side of the screen into the center of the screen and space them an equal distance below each other.
2. Label the buttons appropriately for their function corresponding to the purpose of the page.
 - a. Line select and system settings are the buttons specifically for the homepage.
 - b. Can Line, Keg Line, and Bottle Line are buttons for the select line page.
 - c. Notifications, change password, and tutorial are all buttons for the system settings tab.
3. Insert a back button in the bottom left corner for the user to exit the page for all pages..

Creating the Change Password Page

1. Create four text blocks which read: 'Username', 'Password', 'New Username', and 'Confirm Password'.
2. Create a corresponding text input block for each of the text blocks.
3. Create a button and name it 'Save Changes'. Color this button red once complete.
4. Insert a back button in the bottom left corner for the user to exit the page.

Creating the Tutorial Page

1. Create a new page and change the background of it to red to contrast the video background.
2. Insert a video in the center of the page detailing how to navigate through the app. This will provide users with a comprehensive understanding of how to use the app and reduce the amount of time that is needed to train new users.

3. Insert a back button in the bottom left corner for the user to exit the page.

Creating the Bottle Line Data, Can Line Data, and Keg Line Data Pages

1. Insert a .png image of a profile photo in the top left-hand corner of the screen.
2. Drag a text block into the top half of the screen and center it. Depending on which screen you are working on, title it appropriately (Bottle Line Data, Can Line Data, Keg Line Data).
3. Insert the same amount of buttons that that specific production line has. For instance, the bottle line has 9 machines, so insert 9 buttons onto the screen.
4. Name each of the buttons corresponding to what machine they apply to. When selected, the user will be able to navigate to speed data pertaining to that specific machine.
5. Insert a back button in the bottom left corner for the user to exit the page.

Creating the User-Entered Optimal Volume of Product Page

1. Create three separate pages for each respective production line.
2. Input a text block that reads “Please enter the optimal volume of product that you would like to be produced”.
3. Create a text input block and place it under the text.
4. Place a red button underneath the text input block and name it ‘Enter’. Once this button is pressed, the optimal volume of product will be stored and used in calculations to calculate the optimal speed that the filler station should run at.
5. Insert a back button in the bottom left corner for the user to exit the page.

Creating the Optimal Range of Values Pages for all Machines in Each Production Line.

1. Create the title of the page by inserting a text block and writing, “(Name of Machine) Optimal Speed”.
2. Outline the text block in red to catch the attention of the potential user.
3. Create another text block which reads ‘Speed’ and place it in the upper half of the screen slightly to the left.
4. Insert three data list features from the components list.
 - a. On the left-most side of the screen, select on the stack icon labeled ‘data’.
 - b. Import the spreadsheets created in the data and libraries instructions selections into Thinkable. The libraries included should be Can line data, Bottle line data, and Keg line data with corresponding minimum and maximum values.
 - c. Once imported, click on one of the data lists and select ‘minimum value’ for the minimum value pertaining to that specific machine’s production line. Repeat for maximum and optimal values as well.
 - d. Repeat until all machines for every production line have values for the minimum, optimal, and maximum speed displayed.
5. Insert a back button in the bottom left corner for the user to exit the page.

6.3 Code Implementation




6.3.1 BOM

Table 5. Bill of Materials for Subsystem 3 - Code Implementation

Item	Cost	Link
Thunkable Premium	\$45.00	https://x.thunkable.com/projects/63fb9f93f165ed2407403950/a90ac201-d688-46d8-aaf1-a3e8d99c4232/designer

6.3.2 Equipment List

Table 6. Equipment List for Subsystem 3 - Code Implementation

Item	Link
Google Spreadsheets	 Bottle Line Data  Can Line Data  Keg Line Data

6.3.3 Instructions

The code implementations for the final product were constructed using code-blocking techniques which were available on the app-building software that was utilized to display the final product, which is Thunkable. The code-blocking aspect of the final product was a learning curve as no member in the group was extensively familiar with the process of using code blocks to produce functions in the app. Resources such as forums, online tutorials, and simple experiments with the software were used to become familiarized with the interface. The code blocks provide rather self-explanatory explanations for how they work on their labels, however, more research was needed to understand the full applications of each block.

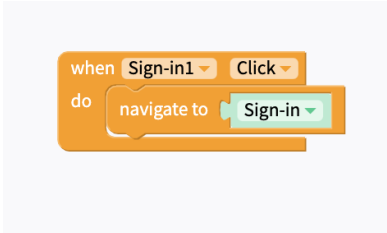


Figure 41. Code blocks representing the function of the sing-in button on the welcome page.

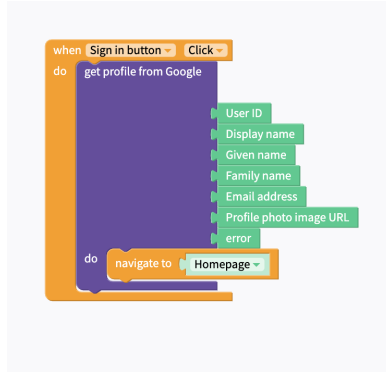


Figure 42. Code blocks that store user's password and username information to compare with those stored in the manager's database.

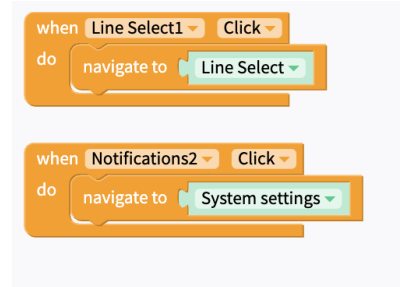


Figure 43. Code blocks that navigate to the line select page and system settings pages once they have been selected on the home page.

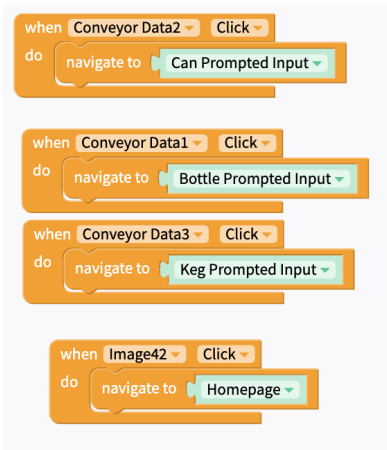


Figure 44. Code blocks that navigate to the prompted input pages for all production lines including can, bottle, and keg lines in addition to a backspace button.

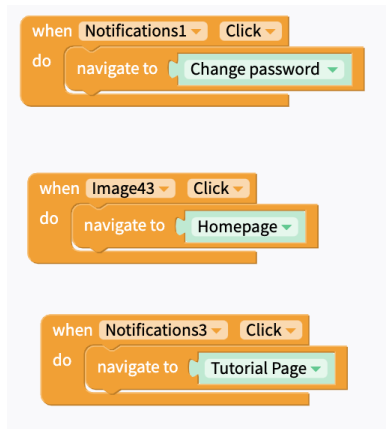


Figure 45. Code blocks for the systems settings page that bring users to the change-password page, the home page and the tutorial page.

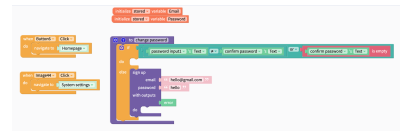


Figure 46. Code blocks that demonstrate how the changing-password option functions.


```

when Button7 Click
do
  navigate to Can Line Machines

when Image40 Click
do
  navigate to Line Select

```

Figure 53. Code blocks on the user prompted input that take the users to the can line once an optimal volume of product has been entered.

```

when Image8 Click
do
  navigate to Bottle Line Machines

when Bottle Depalletizer Opens
do
  call Data_Viewer_List1's Refresh Data
  then do when Refresh Data is done

when Data_Viewer_List24 Item Click row id
do
  vibrate
  call Alert3's Show with output
  then do if wasConfirmed
do

when Data_Viewer_List25 Item Click row id
do
  vibrate
  call Alert2's Show with output
  then do if wasConfirmed
do

```

Figure 54. Code blocks which refresh the simulated spreadsheet containing live data for the bottle depalletizer.

```

when Image9 Click
do
  navigate to Bottle Line Machines

when Bottle Rinser Opens
do
  call Data_Viewer_List2's Refresh Data
  then do when Refresh Data is done

when Data_Viewer_List27 Item Click row id
do
  vibrate
  call Alert5's Show with output
  then do if wasConfirmed
do

when Data_Viewer_List26 Item Click row id
do
  vibrate
  call Alert4's Show with output
  then do if wasConfirmed
do

```

Figure 55. Code blocks which refresh the simulated spreadsheet containing live data for the bottle rinser.

```

when Image10 Click
do
  navigate to Bottle Line Machines

when Bottle Coder Opens
do
  call Data_Viewer_List3's Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List28 Item Click
do
  vibrate
  call Alert6's Show
  with output wasConfirmed
  then do if wasConfirmed
  do

when Data_Viewer_List29 Item Click
do
  vibrate
  call Alert7's Show
  with output wasConfirmed
  then do if wasConfirmed
  do
  
```

Figure 56. Code blocks which refresh the simulated spreadsheet containing live data for the bottle coder.

```

when Image11 Click
do
  navigate to Bottle Line Machines

when Bottle Pasteurizer Opens
do
  call Data_Viewer_List4's Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List31 Item Click
do
  vibrate
  call Alert10's Show
  with output wasConfirmed
  then do if wasConfirmed
  do

when Data_Viewer_List30 Item Click
do
  vibrate
  call Alert9's Show
  with output wasConfirmed
  then do if wasConfirmed
  do
  
```

Figure 57. Code blocks which refresh the simulated spreadsheet containing live data for the bottle pasteurizer.

```

when Image12 Click
do
  navigate to Bottle Line Machines

when Bottle Filler Opens
do
  call Data_Viewer_List5's Refresh Data
  then do When Refresh Data is done
  
```

Figure 58. Code blocks which refresh the simulated spreadsheet containing live data for the bottle filler.

```

when Image13 Click
do
  navigate to Bottle Line Machines

when Bottle Packer Opens
do
  call Data_Viewer_List6's Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List32 Item Click
do
  vibrate
  call Alert11's Show
  with output wasConfirmed
  then do if wasConfirmed
  do

when Data_Viewer_List33 Item Click
do
  vibrate
  call Alert12's Show
  with output wasConfirmed
  then do if wasConfirmed
  do
  
```

```

when Image14 Click
do
  navigate to Bottle Line Machines

when Case Sealer Opens
do
  call Data_Viewer_List7's Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List34 Item Click
do
  vibrate
  call Alert13's Show
  with output wasConfirmed
  then do if wasConfirmed
  do

when Data_Viewer_List35 Item Click
do
  vibrate
  call Alert14's Show
  with output wasConfirmed
  then do if wasConfirmed
  do
  
```

```

when Image22 Click
do
  navigate to Bottle Line Machines

when Bottle Palletizer Opens
do
  call Data_Viewer_List8's Refresh Data
  then do When Refresh Data is done

when Data_Viewer_List36 Item Click
do
  vibrate
  call Alert15's Show
  with output wasConfirmed
  then do if wasConfirmed
  do

when Data_Viewer_List37 Item Click
do
  vibrate
  call Alert16's Show
  with output wasConfirmed
  then do if wasConfirmed
  do
  
```

Figure 59. Code blocks which refresh the simulated spreadsheet containing live data for the bottle packer.

The code blocks for Figure 59 include:

- when Image23 Click: navigate to Can Line Machines
- when Bottle Stretch Wrapper Opens: call Data_Viewer_List5 Refresh Data, then do when Refresh Data is done
- when Data_Viewer_List38 Item Click: row id, do vibrate, call Alert17 Show with output, then do if wasConfirmed
- when Data_Viewer_List39 Item Click: row id, do vibrate, call Alert18 Show with output, then do if wasConfirmed

Figure 60. Code blocks which refresh the simulated spreadsheet containing live data for the bottle case sealer.

The code blocks for Figure 60 include:

- when Image24 Click: navigate to Can Line Machines
- when Can Depalletizer Opens: call Data_Viewer_List9 Refresh Data, then do when Refresh Data is done
- when Data_Viewer_List40 Item Click: row id, do vibrate, call Alert19 Show with output, then do if wasConfirmed
- when Data_Viewer_List41 Item Click: row id, do vibrate, call Alert20 Show with output, then do if wasConfirmed

Figure 61. Code blocks which refresh the simulated spreadsheet containing live data for the bottle palletizer.

The code blocks for Figure 61 include:

- when Image25 Click: navigate to Can Line Machines
- when Data_Viewer_List43 Item Click: row id, do vibrate, call Alert21 Show with output, then do if wasConfirmed
- when Can Rinser Opens: call Data_Viewer_List11 Refresh Data, then do when Refresh Data is done
- when Data_Viewer_List42 Item Click: row id, do vibrate, call Alert22 Show with output, then do if wasConfirmed

Figure 62. Code blocks which refresh the simulated spreadsheet containing live data for the bottle stretch wrapper.

The code blocks for Figure 62 include:

- when Image26 Click: When the user taps inside the bounds of the image and releases, trigger the following block(s)
- when Can Filler Opens: call Data_Viewer_List12 Refresh Data, then do when Refresh Data is done

Figure 63. Code blocks which refresh the simulated spreadsheet containing live data for the can depalletizer.

The code blocks for Figure 63 include:

- when Image27 Click: navigate to Can Line Machines
- when Can Pasteurizer Opens: call Data_Viewer_List13 Refresh Data, then do when Refresh Data is done
- when Data_Viewer_List44 Item Click: row id, do vibrate, call Alert21 Show with output, then do if wasConfirmed
- when Data_Viewer_List45 Item Click: row id, do vibrate, call Alert22 Show with output, then do if wasConfirmed

Figure 64. Code blocks which refresh the simulated spreadsheet containing live data for the can rinser.

The code blocks for Figure 64 include:

- when Image28 Click: navigate to Can Line Machines
- when Can Coder Opens: call Data_Viewer_List14 Refresh Data, then do when Refresh Data is done
- when Data_Viewer_List46 Item Click: row id, do vibrate, call Alert23 Show with output, then do if wasConfirmed
- when Data_Viewer_List47 Item Click: row id, do vibrate, call Alert24 Show with output, then do if wasConfirmed

Figure 65. Code blocks which

The code blocks for Figure 65 include:

- when Image26 Click: When the user taps inside the bounds of the image and releases, trigger the following block(s)
- when Can Filler Opens: call Data_Viewer_List12 Refresh Data, then do when Refresh Data is done

Figure 66. Code blocks which

The code blocks for Figure 66 include:

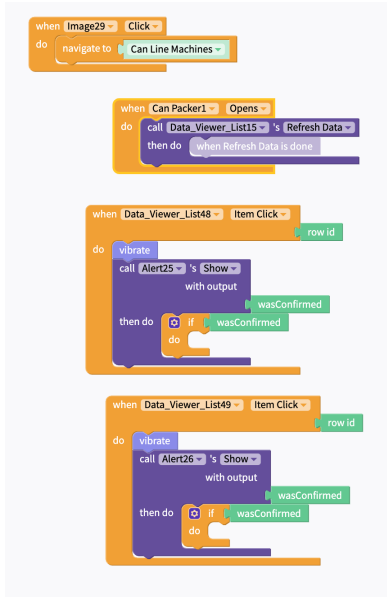
- when Image27 Click: navigate to Can Line Machines
- when Can Pasteurizer Opens: call Data_Viewer_List13 Refresh Data, then do when Refresh Data is done
- when Data_Viewer_List44 Item Click: row id, do vibrate, call Alert21 Show with output, then do if wasConfirmed
- when Data_Viewer_List45 Item Click: row id, do vibrate, call Alert22 Show with output, then do if wasConfirmed

Figure 67. Code blocks which

The code blocks for Figure 67 include:

- when Image28 Click: navigate to Can Line Machines
- when Can Coder Opens: call Data_Viewer_List14 Refresh Data, then do when Refresh Data is done
- when Data_Viewer_List46 Item Click: row id, do vibrate, call Alert23 Show with output, then do if wasConfirmed
- when Data_Viewer_List47 Item Click: row id, do vibrate, call Alert24 Show with output, then do if wasConfirmed

refresh the simulated spreadsheet containing live data for the can filler.

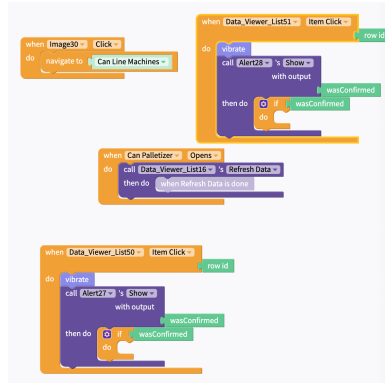


The code blocks in Figure 68 are as follows:

- when Image29 Clicked: do navigate to Can Line Machines
- when Can Packer Opens: do call Data_Viewer_List15 Refresh Data then do when Refresh Data is done
- when Data_Viewer_List48 Item Clicked: do vibrate call Alert25 Show with output wasConfirmed then do if wasConfirmed do
- when Data_Viewer_List49 Item Clicked: do vibrate call Alert26 Show with output wasConfirmed then do if wasConfirmed do

Figure 68. Code blocks which refresh the simulated spreadsheet containing live data for the can packer.

refresh the simulated spreadsheet containing live data for the can depalletizer.

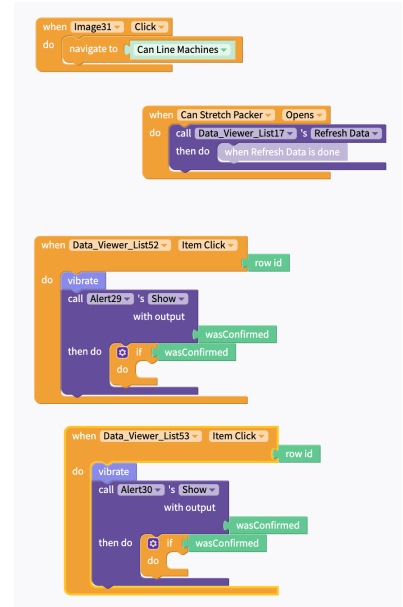


The code blocks in Figure 69 are as follows:

- when Image30 Clicked: do navigate to Can Line Machines
- when Data_Viewer_List51 Item Clicked: do vibrate call Alert28 Show with output wasConfirmed then do if wasConfirmed do
- when Can Palletizer Opens: do call Data_Viewer_List16 Refresh Data then do when Refresh Data is done
- when Data_Viewer_List50 Item Clicked: do vibrate call Alert27 Show with output wasConfirmed then do if wasConfirmed do

Figure 69. Code blocks which refresh the simulated spreadsheet containing live data for the can palletizer.

refresh the simulated spreadsheet containing live data for the can coder.



The code blocks in Figure 70 are as follows:

- when Image31 Clicked: do navigate to Can Line Machines
- when Can Stretch Packer Opens: do call Data_Viewer_List17 Refresh Data then do when Refresh Data is done
- when Data_Viewer_List52 Item Clicked: do vibrate call Alert29 Show with output wasConfirmed then do if wasConfirmed do
- when Data_Viewer_List53 Item Clicked: do vibrate call Alert30 Show with output wasConfirmed then do if wasConfirmed do

Figure 70. Code blocks which refresh the simulated spreadsheet containing live data for the can stretch wrapper.

Figure 71. Code blocks which refresh the simulated spreadsheet containing live data for the keg De Ullage machine.

Figure 72. Code blocks which refresh the simulated spreadsheet containing live data for the external keg washer.

Figure 73. Code blocks which refresh the simulated spreadsheet containing live data for the internal keg washer.

Figure 74. Code blocks which refresh the simulated spreadsheet containing live data for the keg coder.

Figure 75. Code blocks which refresh the simulated spreadsheet containing live data for the external keg blower.

Figure 76. Code blocks which refresh the simulated spreadsheet containing live data for the external keg filler.

Numerous of the pages within the app contain similar coding implementations. This continues to enhance the readability and clarity of the system while building it.

Welcome Page Code Implementation

1. Use a “do when” instruction block and insert a “navigate to” instruction block inside it.
2. Insert that when “sign-in1” is “clicked” it will navigate to the “Sign-in” page.
3. Now when the “sign in” button is clicked it will direct the user to the sign in page of the app.

Sign In Page Code Implementation

1. Use a “do when” block and insert the desired information.
2. Insert that when “Sign in button” is “clicked” it will get profile information stored from google so it can then gain access to the home page.
3. Insert the “get profile from Google and do” block inside the “do when” block.
4. Insert the following blocks into the “get profile from Google” block: “User ID, Display name, Given name, Family name, Email address, Profile photo image URL, error” which will attain the saved profile information from google and display it on the screen.
5. Insert the “navigate to” block into the “do” section of the “get profile from Google and do” block, and navigate it to the “Homepage”.
6. Now when users enter the username and password, the code views the saved data and signs them in, which then navigates them to the home page of the app.

Homepage Code Implementation

1. Use a “do when” block and insert the desired information.
2. Insert that when “Line Select1” is “clicked” it will navigate to the Line Select page of the app.
3. Insert a “navigate to” block into the “do when” block, and insert the “Line Select” block.
4. Now when the Line Select button is clicked the app will navigate to the “Line Select” page.
5. Use another “do when” block and insert the desired information.
6. Insert that when “Notifications2” is “clicked” it will navigate to the “System settings” page of the app.
7. Insert a “navigate to” block into the “do when” block, and insert the “System settings” block.
8. Now when the System settings button is clicked the app will navigate to the “System settings” page.

Line Select Code Implementation

1. Use a “do when” block and insert the desired information.
2. Insert that when “Conveyor Data1” is “clicked” it will navigate to the “Bottle Prompted Input” page of the app.
3. Insert a “navigate to” block inside the “do when block” and insert the “Bottle Prompted Input” block.
4. Use another “do when” block and insert the desired information.
5. Insert that when “Conveyor Data2” is “clicked” it will navigate to the “Can Prompted Input” page of the app.
6. Insert a “navigate to” block inside the “do when block” and insert the “Can Prompted Input” block.
7. Use a third “do when” block and insert the desired information.

8. Insert that when “Conveyor Data3” is “clicked” it will navigate to the “Keg Prompted Input” page of the app.
9. Insert a “navigate to” block inside the “do when block” and insert the “keg Prompted Input” block.
10. Now when Can Line, Bottle Line, or Keg Line is clicked the app will navigate to its corresponding page.
11. Use a fourth “do when” block and insert the desired information.
12. Insert that when “Image42” is “clicked” the app will navigate to the previous page.
13. Insert a “navigate to” block into the “do when” block and insert the “Homepage” block.
14. Now when the backspace button is clicked the app will navigate to the previous page (the Homepage in this case).

System Settings Code Implementation

1. Use a “do when” block and insert the desired information.
2. Insert that when “Notifications1” is “clicked” the app will navigate to the “Change password” page of the app.
3. Insert a “navigate to” block inside the “do when” block and insert the “Change password” block.
4. Now when the “Change password” button is clicked the app will navigate to the Change password page.
5. Use a second “do when” block and insert the desired information.
6. Insert that when “Notifications3” is “clicked” the app will navigate to the “Tutorial Page” page of the app.
7. Insert a “navigate to” block inside the “do when” block and insert the “Tutorial Page” block.
8. Now when the “Tutorial Page” button is clicked the app will navigate to the Tutorial page.
9. Use a third “do when” block and insert the desired information.
10. Insert that when “Image43” is “clicked” the app will navigate to the “Homepage” page of the app.
11. Insert a “navigate to” block inside the “do when” block and insert the “Homepage” block.
12. Now when the backspace button is clicked the app will navigate to the Homepage.

Change Password Code Implementation

1. Use two “initialize variable” block and insert the desired information
2. Initialize the first block to the “stored” “email” variable.
3. Initialize the second block to the “stored” “Password” variable.
4. Use a “do when” block and insert that when “Button6” is “clicked” the app will navigate to the Homepage.
5. Then insert a “navigate to” block into the “do when” block and insert the “Homepage” block.
6. Now when the “Save Changes” button is clicked the app will navigate back to the Homepage
7. Use another “do when” block and insert that when “Image44” is “clicked” the app will navigate to the System settings page.
8. Then insert a “navigate to” block into the “do when” block and insert the “System settings” block.
9. Now when the backspace button is clicked the app will navigate to the System settings page.
10. Use a “to” block and insert the “change password” block.
11. Insert an “if, do, else” block into the “to” block.
12. Inset a “or” block into the if section and enter the following blocks:

- a. If “password input1, text” does not equal to “confirm password, text” or if “confirm password, text” is empty do nothing (do section stays empty because password protocols were not followed).
- b. Into the “else” section insert the “sign up” block.
- c. Enter the email address into the “email” section and enter the password into the “password” section.
- d. Lastly add in an “error” button in case email address or password is entered incorrectly.

Tutorial Page Code Implementation

1. Use a “do when” block and insert the desired information.
2. Insert that when “image45” is “clicked” the app will navigate to the System settings page.
3. Insert a “navigate to” block into the “do when” block and insert the “System settings” block.
4. Now when the backspace button is clicked the app will navigate to the System settings page.

Bottle Line Machines Code Implementations

1. Use a “do when” block and insert the desired information.
2. Insert that when “image16” is “clicked” the app navigates to the Line Select page.
3. Insert a “navigate to” block inside the “do when” block and insert the “Line Select” block. Now when the backspace button is clicked the app will navigate to the Line Select page.
4. Use another “do when” block and insert that when “Palletizer” is “clicked” the app navigates to the Bottle Depalletizer page.
5. Insert a “navigate to” block inside the “do when” block and insert the “Bottle Depalletizer” block.
6. Repeat steps 4 and 5 eight more times for the following information:
 - a. When “Empty Bottle Rinser” is “touched down”, navigate to the “Bottle Packer” page.
 - b. When “Pasteurizer” is “clicked”, navigate to the “Bottle Pasteurizer” page.
 - c. When “Labeler” is “clicked”, navigate to the “Bottle Coder” page.
 - d. When “Case Packer” is “clicked”, navigate to the “Bottle Rinser” page.
 - e. When “Depalletizer” is “clicked”, navigate to the “Case Sealer” page.
 - f. When “Filler” is “clicked”, navigate to the “Bottle Filler” page.
 - g. When “Stretch Wrapper” is “clicked”, navigate to the “Bottle Stretch Wrapper” page.
 - h. When “Bottle Palletizer1” is “clicked”, navigate to the “Bottle Palletizer” page.

Can Line Machines Code Implementations

1. Use a “do when” block and insert the desired information.
2. Insert that when “image19” is “clicked” the app navigates to the Line Select page.
3. Insert a “navigate to” block inside the “do when” block and insert the “Line Select” block. Now when the backspace button is clicked the app will navigate to the Line Select page.
4. Use another “do when” block and insert that when “Can Pasteurizer” is “clicked” the app navigates to the Can Pasteurizer page.
5. Insert a “navigate to” block inside the “do when” block and insert the “Can Pasteurizer1” block.
6. Repeat steps 4 and 5 seven more times for the following information:
 - a. When “Can Depalletizer1” is “clicked”, navigate to the “Can Depalletizer” page.
 - b. When “Can Rinser1” is “clicked”, navigate to the “Can Rinser” page.
 - c. When “Can Filler” is “clicked”, navigate to the “Can Filler1” page.

- d. When “Can Coder” is “clicked”, navigate to the “Can Coder1” page.
- e. When “Can Packer” is “clicked”, navigate to the “Can Packer1” page.
- f. When “Palletizer3” is “clicked”, navigate to the “Can Palletizer” page.
- g. When “Stretch Wrapper1” is “clicked”, navigate to the “Can Stretch Wrapper” page.

Keg Line Machines Code Implementations

1. Use a “do when” block and insert the desired information.
2. Insert that when “image21” is “clicked” the app navigates to the Line Select page.
3. Insert a “navigate to” block inside the “do when” block and insert the “Line Select” block. Now when the backspace button is clicked the app will navigate to the Line Select page.
4. Use another “do when” block and insert that when “Keg Blower1” is “clicked” the app navigates to the Keg Blower page.
5. Insert a “navigate to” block inside the “do when” block and insert the “Keg Blower” block.
6. Repeat steps 4 and 5 five more times for the following information:
 - a. When “Keg Coder1” is “clicked”, navigate to the “Keg Coder” page.
 - b. When “De Ullage1” is “clicked”, navigate to the “De Ullage” page.
 - c. When “Internal Keg Washer1” is “clicked”, navigate to the “Internal Keg Washer” page.
 - d. When “External Keg Washer1” is “clicked”, navigate to the “External Keg Washer” page.
 - e. When “Keg Filler2” is “clicked”, navigate to the “Keg Filler1” page.

Can Prompted Input Code Implementations

1. Use a “do when” block and insert the desired information.
2. Insert that when “Button5” is “clicked” the app will navigate to the Can Line Machines page.
3. Insert a “navigate to” block inside the “do when” block and insert the “Can Line Machines” block.
4. Now when the user enters a value and clicks enter the app will navigate to the Can Line Machine page.
5. Use another “do when” block and insert the desired information.
6. Insert that when “Image39” is “clicked” the app will navigate to the Line Select page.
7. Insert a “navigate to” block inside the “do when” block and insert the “Line Select” block.
8. Now when the backspace button is clicked the app will navigate to the Line Select page.

Keg Prompted Input Code Implementations

1. Use a “do when” block and insert the desired information.
2. Insert that when “Button8” is “clicked” the app will navigate to the Keg Line Machines page.
3. Insert a “navigate to” block inside the “do when” block and insert the “Keg Line Machines” block.
4. Now when the user enters a value and clicks enter the app will navigate to the Keg Line Machine page.
5. Use another “do when” block and insert the desired information.
6. Insert that when “Image41” is “clicked” the app will navigate to the Line Select page.
7. Insert a “navigate to” block inside the “do when” block and insert the “Line Select” block.
8. Now when the backspace button is clicked the app will navigate to the Line Select page.

Bottle Prompted Input Code Implementations

1. Use a “do when” block and insert the desired information.
2. Insert that when “Button7” is “clicked” the app navigates to the Bottle Line Machines.
3. Insert a “navigate to” block inside the “do when” block and insert the “Bottle Line Machines” block.
4. Now when the user enters a value and clicks enter the app will navigate to the Bottle Line Machines page.
5. Use another “do when” block and insert the desired information.
6. Insert that when “Image40” is “clicked” the app navigates to the Line Select page.
7. Insert a “navigate to” block inside the “do when” block and insert the “Line Select” block.
8. Now when the user clicks the backspace button the app will navigate to the Line Select page.

Bottle Depalletizer, Bottle Rinser, Bottle Coder, Bottle Pasteurizer, Bottle Filler, Bottle packer, Case Sealer, Bottle Palletizer, Stretch Wrapper Code Implementations

1. Use a “do when” block and insert a “navigate to” block inside it.
2. Insert that when “Image8” is “clicked” the app navigates to the “Bottle Line Machines” page.
3. Now when the backspace button is clicked the app navigates to the Bottle Line Machines page.
4. Use another “do when” block and insert a “call then do” block inside it.
5. Insert that when “Bottle Depalletizer” “opens” the corresponding data viewer list refreshes data.
6. Use another “do when” block and insert a “vibrate” and “call then do” block into it.
 - a. Insert that when the corresponding data viewer list is clicked the app knows to vibrate.
 - b. Insert that the corresponding alert will show if the condition is confirmed.
7. Use another “do when” block and insert a “vibrate” and “call then do” block into it.
 - a. Insert that when the corresponding data viewer list is clicked the app knows to vibrate.
 - b. Insert that the corresponding alert will show if the condition is confirmed.
8. Repeat steps 1 to 7 for every single bottle line machine with its correct corresponding data viewer lists, alerts, conditions, and buttons.

Can Depalletizer, Can Rinser, Can Filler1, Can Pasteurizer1, Can Coder1, Can Packer1, Can Palletizer, Can Stretch Packer Code Implementations

1. Follow steps 1 to 8 from the bottle section above for every single can line machine, making sure to use the correct corresponding data viewer list, alerts, conditions, and buttons.

De Ullage, External Keg Washer, Internal Keg Washer, Keg Coder, Keg Blower, Keg Filler1 Code Implementation

1. Follow steps 1 to 8 from the bottle section above for every single keg line machine, making sure to use the correct corresponding data viewer list, alerts, conditions, and buttons.

6.4 Unused Components

Temperature Sensor Arduino Input

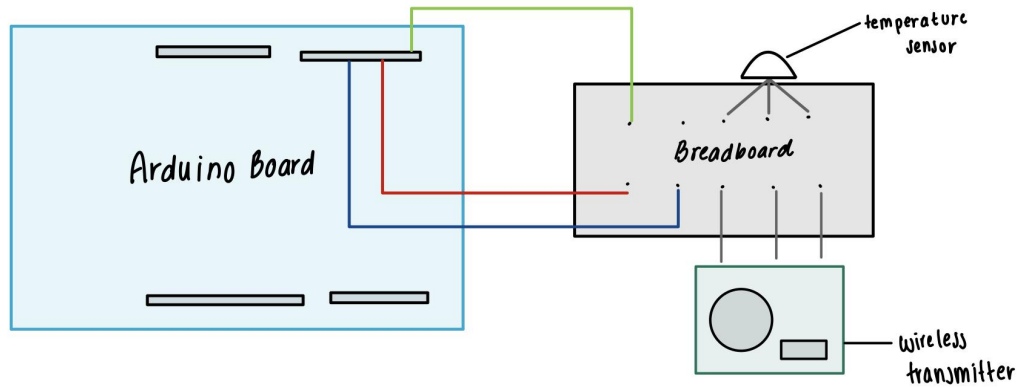


Figure 77. A Temperature Sensor Arduino Uno Hardware Component (reference. Deliverable E)

Speed Arduino Input Sensor

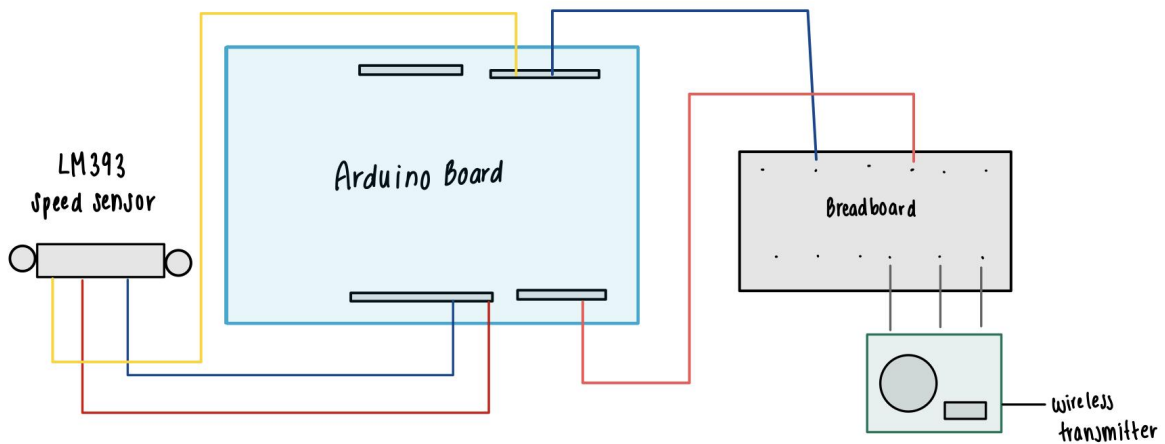


Figure 78. A Speed Sensor Arduino Uno Hardware Component (reference. Deliverable E)

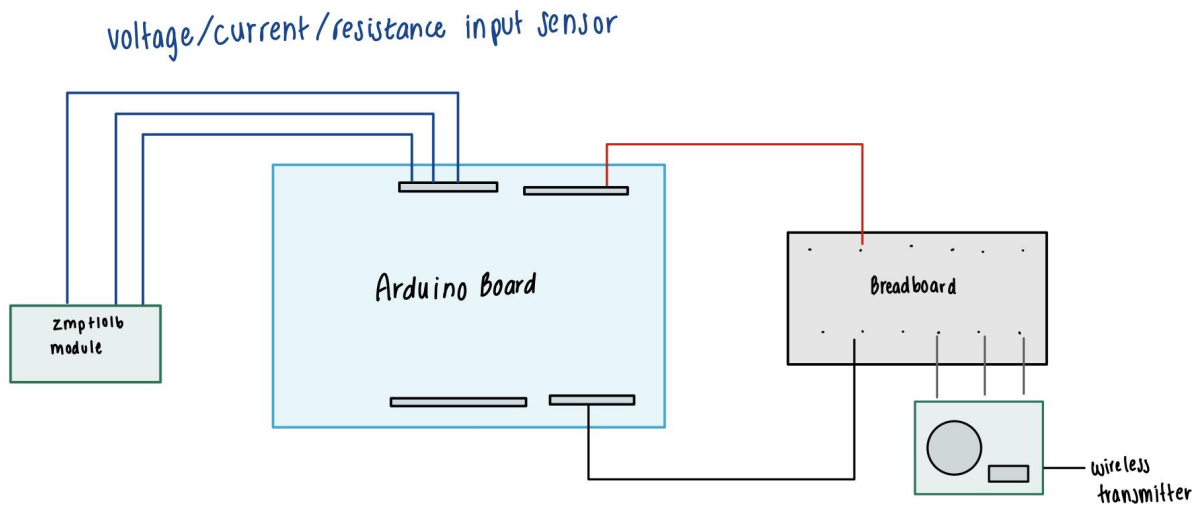


Figure 79. A Voltage Sensor Arduino Uno Hardware Component (reference. Deliverable E)

As observed in Figures 77-79, all physical hardware systems were eliminated from the final system. This executive decision was made as a result of prioritizing the cost-effectiveness and adaptability of our system. It was important for our client that our final product maintained a dynamic ability to adapt to changes on the floor of the production lines. Resultantly, these hardware components were removed completely.

6.5 Testing & Validation

Table 7. Tabulated Prototype III Test Plan with Results and Analysis (reference. Deliverable H)

Objective	Stopping Criteria	Unit of Measurement & Acceptable Fidelity/Simplifying Assumptions
To gauge user perception of the overall app.	For the third and final prototype, it is essential that user perception of the overall system is positive. The test that will be conducted to evaluate this component will consist of an interview with a potential user that has been engaged throughout the entirety of the prototyping process. An appropriate stopping criteria for this component will consist of a short survey given to the user. If the user rating is above 70% by the time survey results are given, then the stopping criteria will be deemed to have been successful. This test will evaluate mainly non-technical components such as the aesthetic of the system, and overall functionality. The user will be given time to look over and navigate the app before being given the survey.	<ul style="list-style-type: none"> • Unit of measurement: The user's overall rating of the app collected via the survey. • Fidelity: This is the third and final prototype of the system, and therefore, it should be fairly similar to that of the final product, therefore there should be very little fidelity. Considering that the app is being tested by an employee of the Mill Street Brewery, this test should very closely resemble the impression of another employee.
Results and Analysis		

Through a survey that was given to potential users, it was determined that the overall user perception of the app is positive. After interviewing a total of ten potential users, the average rating that was given to the app based on functionality, design, and user interface was 8.7/10. This rating correlated with 87% positive user perception of the app. This exceeds the outlined stopping criteria by 17%, therefore allowing us to draw the conclusion that the app was successful. This test provided us with information that allowed us to further gauge user perception on the app. Notable positive comments received from the potential users regarded the app's navigable and easy to use interface, and the cohesive color scheme which added to the overall aesthetic of the app. Some arrears of improvement that were stated by the potential users noted that there were some errors when a value was input into the app. This brought attention to a potential issue if a user were to accidentally input a value that was too large. There was no preparation for this event prior to the app being tested, so we were not prepared for a very large volume of product to be entered. This allowed the team to be more responsive to potential user mishandlings of the app, and the appropriate contingency plans were implemented accordingly. This test was very significant in highlighting last-minute changes that needed to be made to the product.

To analyze the input of information and how the system is able to handle different volumes of information for different machines.

This test will analyze how the input of information into the system is taken, and processed. The efficiency of this intake of information will be tested based on how the app is able to deliver an accurate notification output given certain file inputs with varying information that indicates that different machines are not operating at optimal capacity. This is a continuation of test II from the prototype II test plan in which the test had failed as a result of the sheer magnitude of information that had to be output. In order to circumvent the same outcome, changes will be made to the overall method in which the system takes in, and outputs information.

- **Unit of measurement:** the amount of test cases passed.
- **Fidelity:** The frequency of information being passed to the system is still unknown and will have to be verified with the client. Until then, the system will be receiving information at a consistent rate which may be different in a real-world instance in which information will be passed to the system sporadically.

Results and Analysis

In these tests, the app was given spreadsheets with different ranges that produced random numbers that would be expected in a real life implementation of the product. In these different spreadsheets, certain values were expected to set off the notification system by deliberately exceeding the maximum or succeeding the minimum value. In these instances, the system was tested on its ability to send notifications appropriately for the machines that were falling out of the range of values. The results from this test demonstrated that there were issues that arose when multiple machines were falling out of their optimal range of values. This was problematic as it modeled a very real potential issue that may occur, as one machine falling out of the range of optimal values will have a snow-ball effect in which the machines following that machine will also fall out of their optimal range of values. In order to mitigate this issue, more extensive coding will be done in the app to ensure that it can compensate for pushing multiple notifications at once. The method in which multiple notifications will be delivered will need to be orderly, so it will be ensured that the notifications are sent sequentially, starting at the root machine so the employee is able to address the problem in the most efficient way possible. Overall, these tests were very insightful and allowed us to make changes to our final product in order to ensure that it is fully functional in all circumstances.

To analyze the integration of all subsystems within the app.

This test will analyze how all the systems in the app will work together to create a cohesive product. This is a continuation of prototype II test III. This test was unsuccessful as the calculations that are implemented to connect the user-entered optimal amount of product they would like to be produced during that day and the input of the files for machines regarding their speed/temperature/energy consumption were not compatible. In this test case, the calculations will be reviewed and reevaluated to create an equation that is

- **Unit of measurement:** range of optimal values.
- **Fidelity:** The files provided by the client with the information regarding energy consumption, temperature, and speed increase the fidelity of this test exponentially, so there is very little deviation from this test with comparison to the actual test real-world implementation of this

	able to produce an optimal range of values. The stopping criteria will be when the values create a reasonable range of values for the temperature, speed, and energy consumption to fall under, as verified by the client.	system.
--	--	---------

Results and Analysis

In this test, the integration of all app subsystems were tested. This test was a continuation of the third test in the second prototype test plan. This test plan was overall unsuccessful as the calculations were not considering client input and input data to give optimal value recommendations. Resultantly, the optimal range was not accurate. In order to mitigate this issue, extensive troubleshooting was done in the form of watching tutorials and consulting online forums in order to gain more insight from experienced users of the software Thinkable. Through this trouble shooting, this final test was able to be successful and accurate optimal ranges were calculated and displayed to the user. In order to test the success of this implementation, test files that were used in the second prototype test were reused to ensure that the optimal value range was accurate. All test cases passed, demonstrating data that was consistent with optimal data given by the client. Overall, this test strengthened the app’s overall performance by ensuring that the information that was being given to the user was accurate.

Table 8. Tabulated Raw Data demonstrating Analytical Data for User Navigability (**reference. Deliverable H**)

	Run 1 (seconds)	Run 2 (seconds)	Run 3 (seconds)	Average
Potential User 1	93	84	88	88.3
Potential User 2	112	89	93	98
Potential User 3	77	70	82	76.3
Overall Average				87.5

Amount of Time Taken for Potential Users to Navigate App

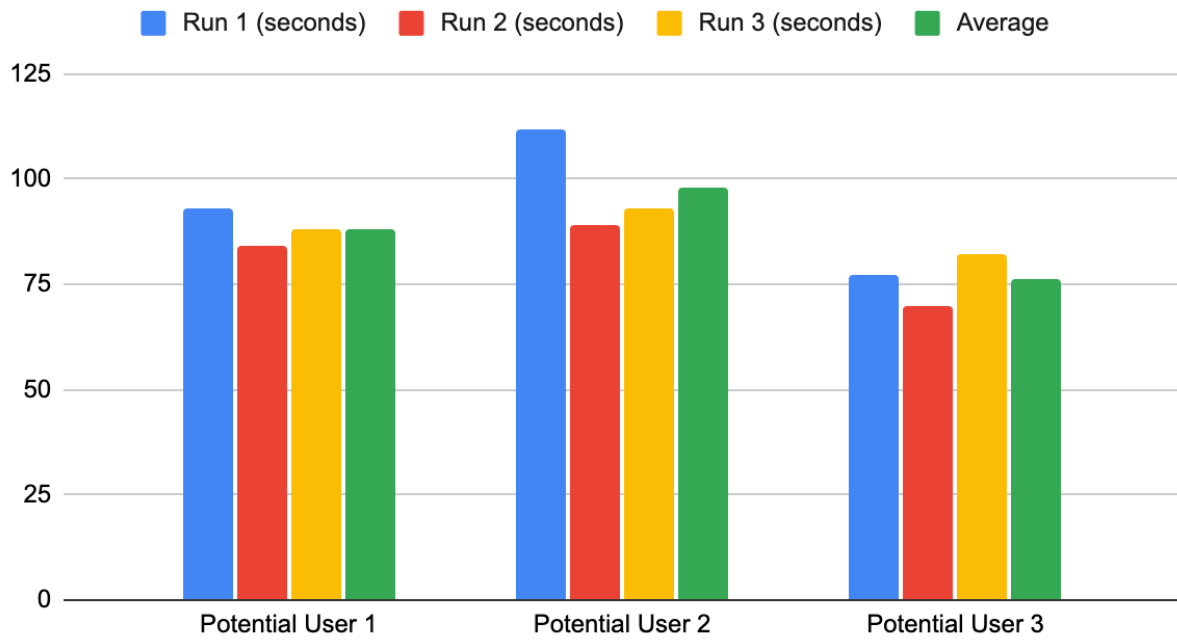


Figure 80. Graphical Representation of the Analytical Model to gauge users' ability to navigate through the app. (reference. Deliverable H)

7 Conclusions and Recommendations for Future Work

In conclusion, our group C8 was tasked with increasing production speed by 2% for the Mill Street Brewery production lines. Through following the steps of the design process, we were able to create a solution that effectively fulfilled our client's requirements. We have presented a comprehensive document that contains an overview of the product, a generalized walkthrough of the product from startup to exit, a detailed description of each user function and feature, a description of recovery and error correction procedures, an explanation of how the final prototype was built, and possible work that can be conducted for future prototypes.

Our experience working on this project has taught us the importance of empathizing with the client and clearly defining the problem before proceeding with ideation and prototyping. We have also learned the significance of thorough testing and validation methods to ensure the final product meets the client's requirements.

Moving forward, we suggest that future groups prioritize further developing the machine learning capabilities of the system to improve accuracy and efficiency. Additionally, exploring the integration of real-time data and analytics can further optimize production speed.

If we had a few more months to work on this project, we would have liked to expand the scope of our user testing to include more diverse user groups to ensure that the product is accessible and intuitive to all users. We also would have liked to explore the possibility of integrating remote monitoring and control capabilities to allow for greater flexibility and convenience in managing the production lines.

In summary, our work on this prototype has been a valuable learning experience and we hope that our document serves as a foundation for future groups to continue and improve upon our work. We encourage future groups to prioritize further developing the machine learning capabilities, exploring the integration of real-time data and analytics, expanding user testing, and exploring the possibility of remote monitoring and control capabilities.

8 Bibliography

No external sources were cited.

APPENDICES

9 APPENDIX I: Design Files

Table 3. Referenced Documents

Document Name	Document Location and/or URL	Issuance Date
Deliverable A	☰ Deliverable A - Team Contract	04/10/2023
Deliverable B	☰ Deliverable B - Needs Identificatio...	04/10/2023
Deliverable C	☰ Deliverable C - Design Criteria and...	04/10/2023
Deliverable D	☰ Deliverable D: Conceptual Design	04/10/2023
Deliverable E	☰ Deliverable E - Project Plan and Co...	04/10/2023
Deliverable F	☰ Deliverable F - Prototype I and Cus...	04/10/2023
Deliverable G	☰ Deliverable G - Prototype II and C...	04/10/2023
Deliverable H	☰ Deliverable H - Prototype III and C...	04/10/2023