Project Deliverable F

Prototype I and Customer Feedback

Marie Levin, Leila Smaili, Rafiullah Hamdard, Hanna Paik, and Yifei Li

March 5, 2023

Abstract

This document contains Project Deliverable F. The purpose of this document is to show our first prototype of the product. This prototype was created according to the client's needs/requests, and through user/technical benchmarking. Moreover, this document contains the customer feedback which we got during our second meeting with the client.

Table of Contents

1.0 Introduction

Robert Ritchie has tasked our design team to create a product that displays the specific conveyor speeds that optimize the yield of beer from his manufacturing line. Following the initial client meeting, our team developed the following problem statement.

“A need exists for Robert Ritchie and his fellow supervisors to find the optimal speeds of their beer packaging process to ensure it is “always at top efficiency (Project Background, Brightspace)”. The solution must have an attractive and straightforward interface that uses a flexible algorithm based on the V-Curve Theory to report optimized speeds of each unit and make recommendations on how to achieve them.”

From this problem, we developed design criteria and metrics that will be used to measure our design’s ability to solve our client’s problem. With research, benchmarking, and brainstorming, the first conceptual design was developed and it was presented during the second client meeting.

After we presented our presentation to the client in our second meeting and he expressed interest in our conceptual design, our team developed the first prototype of the product. This prototype was created based on the feedback that we got from the client meeting, following the client’s needs, and meeting the design criteria.

2.0 Feedback from the client:

After we presented our conceptual design to the client during the second meeting, the client displayed interest in our conceptual design. The client promised that he would provide us with the following raw data:

1: Speed of the filler station (cans per minute).
2: The speed of the conveyors (cans per minute).
3: The preset speeds of different units.

We presented the client with two possible designs:

1: Windows software
2: Mobile application

The client was more interested in mobile applications than windows software. Because he and his team currently use iPads in their office, it was clear that an iOS app was much more interesting to him than an android device. In addition, the client stated that he wants to be able to carry the device around the production line. Then, he can change the can filler speeds and type the desired input speed into the device, thus he should be able to get the best output and optimize his production line.

3.0 Objectives of our first prototype

Based on the feedback that we got from the client in our second meeting, our team developed the first prototype. Moreover, this prototype will be used in our future design choices and will improve the design and product.

Our goals for creating our first prototype are

1: To learn more and understand better our conceptual design

2: To get constructive feedback from the client

3: To reduce the risk and uncertainty associated with a specific aspect of the design

4: To measure the overall performance

3.1 Prototype of Pure UI

In the previous deliverable, the final conceptual design had been created.

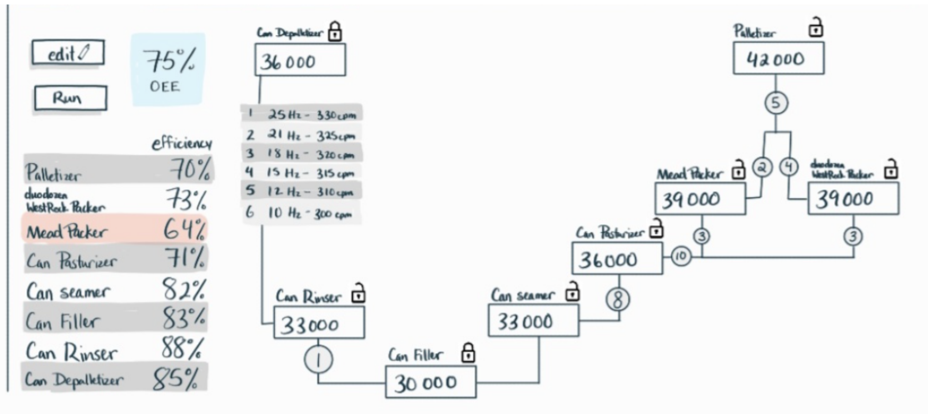


Figure 1 Final Conceptual Design

In this deliverable, a sample interface was created using Xcode based on the final conceptual design. At the moment, this is purely a temporary UI prototype according to the design plan. This prototype currently has no ability to do arithmetic and provide results. The goal of the UI is to test its capability and ability for user interaction. Its function will be perfected later based on the design plan.

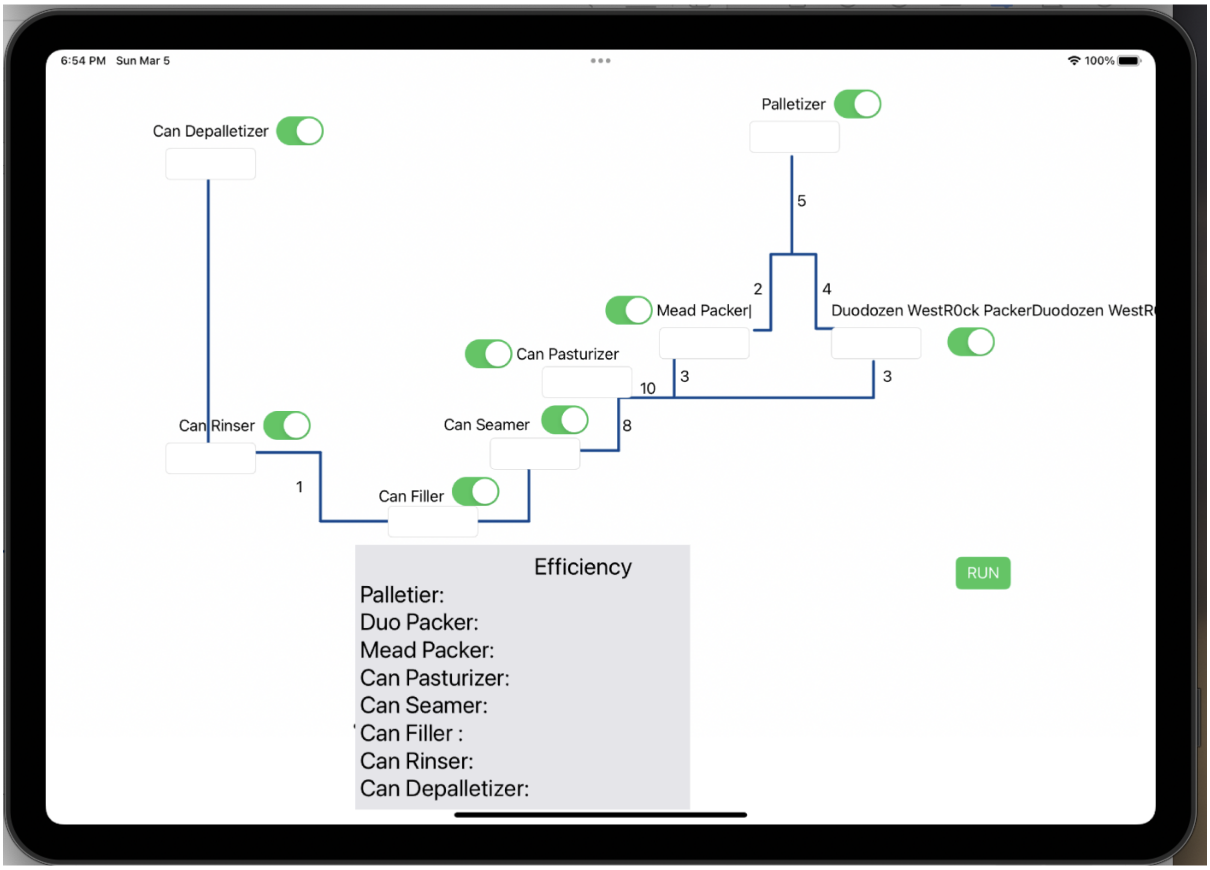


Figure 1 Screenshot of simulation of the interface

3.1.1 Critical Component

Since it is just a pure UI, little engineering and science knowledge has been involved.

1: Production line display.
It helps users to visually connect V-theory to the conveyor system so that the UI is easier to understand.

2: Table
All desired data (optimal data) will show in the table so that users are able to distinguish the system efficiency quickly.

3: Switch
A “lock” is turned on when the user no longer needs to change the data. This is to prevent accidental edits.

4: Editable Text area (input: Speed of the conveyor)
Keyboard enter available.

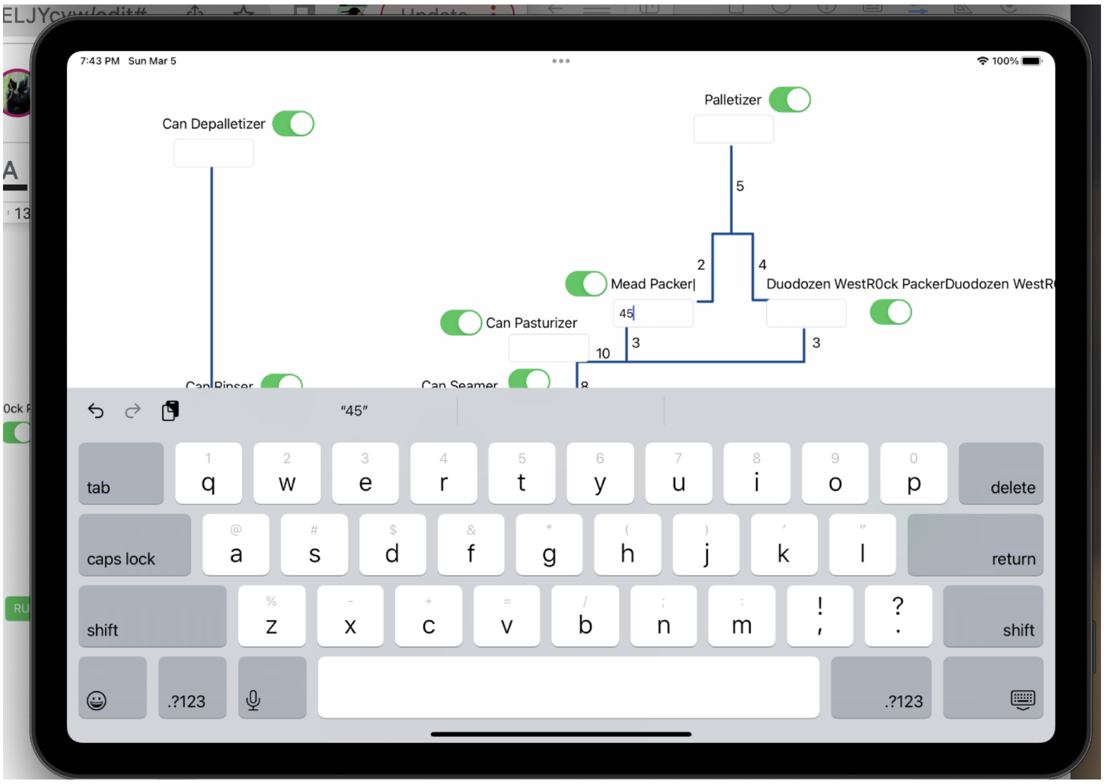


Figure 3 Editable Text area

Users are able to input original data so that the system can calculate the desired optimal data of the system through an installed algorithm.

3.2 Prototype of V-curve Theory

$$SpeedOfDesiredUnit = \left(a_1 (r)^{n-1}\right)(x)^{n_1 \times \frac{a_1}{200}} = \begin{cases} \left(a_1(r)^{n-1}\right)(x)^{\frac{n^\wedge 2}{2} \times \frac{a_1}{200}} : beforeFiller \\ \left(a_1(r)^{n-1}\right)(x)^{\frac{n^\wedge 2}{2} \times \frac{a_1}{200} + 0.2} : afterFiller \end{cases}$$

This prototype is a proof of concept for our algorithm that will be implemented into our application to calculate the speeds of each unit in our production line given the filler speed.

This prototype was created to ease the process of building the back end of our application, instead of working straight with XCode and attempting to use Objective C to create the formulas that would calculate the speeds of each Unit. This Prototype outlines the formula that needs only to be translated into Objective C to build the back end of our application. The prototype breaks down the construction of the back end of our application into two parts; calculating the formulas and translating them into Objective C.

3.2.0 How the Algorithm was Derived

This prototype is an algorithm derived from the V-Curve Theory presented by LineView. They state that the V-curve Theory uses the slowest unit on a line and all the units before and after the line are 10% faster than the unit before or after it (LineView 2022). This is evident in figure 1:

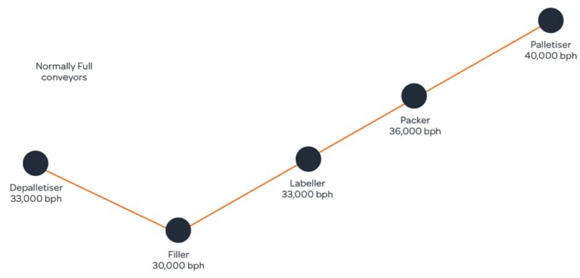


Figure 1: “Here's a shot of what a V-Curve might look like in a typical production line” (LineView 2022)

This 10 percent incrementation is a Geometric Series as seen in the equation below:

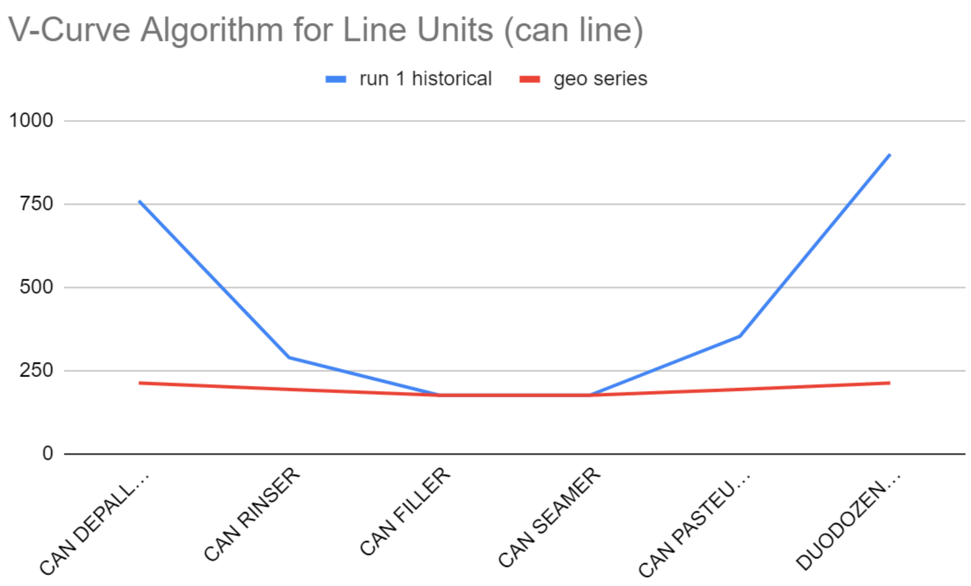
$$\sum_{n=1}^{\infty} a_1 (r)^{n-1} = (fillerSpeed)(1.10)^{distanceFromFiller}$$

The Table Below shows this equation implemented with the historical filler speed of Run 1 and Run 2.

Table 1: Geometric Series

	Distance from Filler	Run 1	Run 2
CAN DEPALLETIZER	2	175(1.10)^2 =211.75	200(1.10)^2 =242
CAN RINSER	1	175(1.10)^1 =192.5	200(1.10)^1 =220
CAN FILLER	0	175	200
CAN SEAMER	0	175	200
CAN PASTEURIZER	1	175(1.10)^1 =192.5	200(1.10)^2 =220
DUODOZEN WESTROCK PACKER	2	175(1.10)^2 =211.75	200(1.10)^2 =242

This equation alone is not enough to calculate accurate speeds as the farther one gets from the filler the less accurate this equation becomes when comparing it to real data. This can be seen in the graph below:



The difference in distribution is similar to that of a power series as shown below:

$$\sum_{n_1=?}^{\infty}(C_n)(x)^{n_1} = (GeoSeriesValue)_{distanceFromFiller}\left(\frac{fillerSpeed}{100}\right)^{powerIncermentValue}$$

Power incrementation (n_1) is not one-to-one with the distance from filler, therefore through trial and error the relationship between the n_1 of the power series and the n of the Geo series was found to be:

$\frac{n^2}{2}$ before the filler and $\frac{n^2}{2}+0.2$ after the filler. The 0.2 increase was derived from the fact that after the Filler, the Can Seamer adds a greater distance from the filler and affects the rest of the production line.

This formula as a whole is a combination of a Geometric and Power Series, which are the critical components of this algorithm prototype.

$$SpeedOfDesiredUnit = a_1(r)^{n-1}(x)^{n_1} = \begin{cases} a_1(r)^{n-1}(x)^{\frac{n^2}{2}} & :beforeFiller \\ a_1(r)^{n-1}(x)^{\frac{n^2}{2}+0.2} & :afterFiller \end{cases}$$

a_1: Filler Speed
r: 1.10 (10% increment)
n: distance from filler (i.e the CAN RINSER is 1 away from the filler)
x: Filler Speed / 100 (value derived from experimentation)

This formula was derived from experimenting during run 2, which has a filler speed of 200cpm. This means that a conversion is necessary when using other filler speeds.

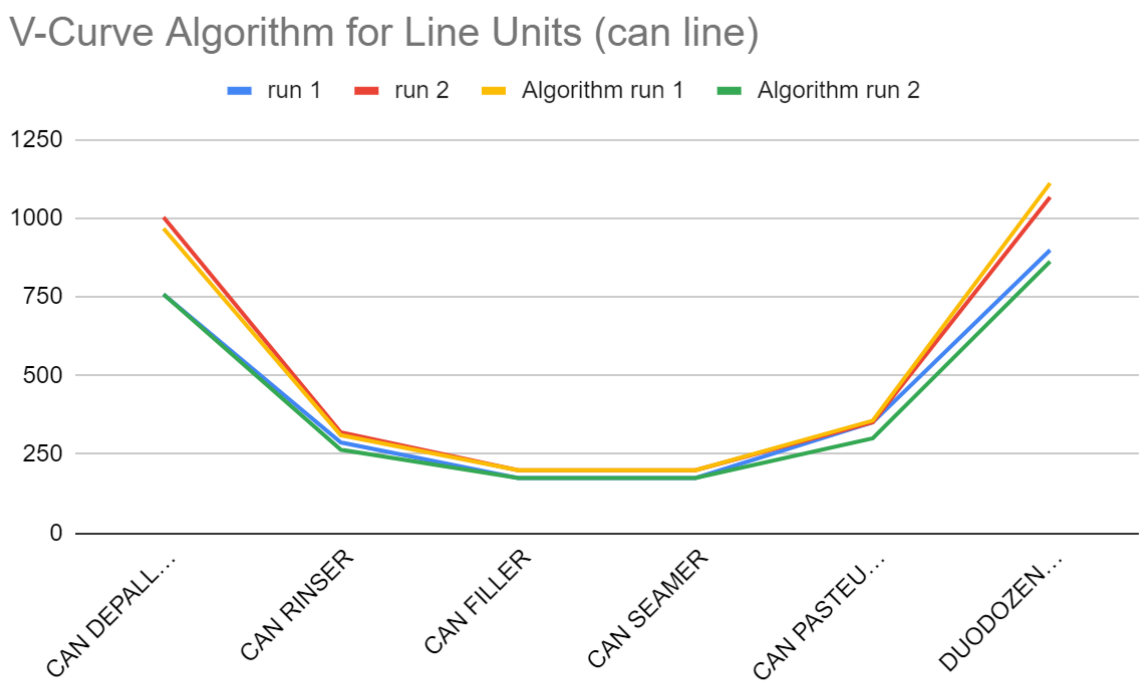
The final formula is shown below:

$$SpeedOfDesiredUnit = \left(a_1(r)^{n-1}\right)(x)^{n_1 \times \frac{200}{a_1}} = \begin{cases} \left(a_1(r)^{n-1}\right)(x)^{\frac{n^2}{2} \times \frac{200}{a_1}} & :beforeFiller \\ \left(a_1(r)^{n-1}\right)(x)^{\left(\frac{n^2}{2} \times \frac{200}{a_1}\right)+0.2} & :afterFiller \end{cases}$$

The table below shows the full process of implementing the algorithm on the historical data of the canning line.

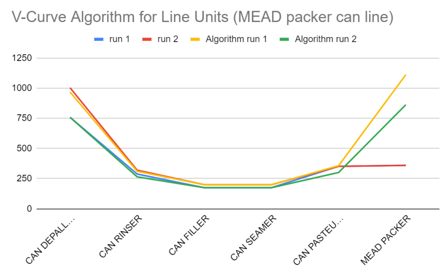
Table 2: Full Algorithm Process

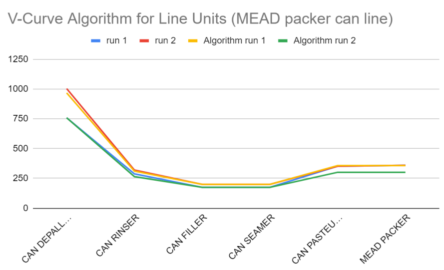
	Unit	run 1	run 2	Geo Series on run 2	Algorithm on run 2	Geo Series on run 1	Algorithm on run 1
MSB-PCKG-CANL-CAN DEPAL	CAN DEPALLETIZER	759.02064	1004.0352	242	968	211.75	758.49054
MSB-PCKG-CANL-CAN RINSER	CAN RINSER	288.0045	319.8489	220	311.12698	192.5	264.8271
MSB-PCKG-CANL-CAN FILLER	CAN FILLER	175	200	200	200	175	175
MSB-PCKG-CANL-CAN SEAMER	CAN SEAMER	175	200	200	200	175	175
MSB-PCKG-CANL-CAN PASTEURIZER	CAN PASTEURIZER	352.1616	352.1616	220	357.39105	192.5	301.2043
MSB-PCKG-CANL-CAN PACKER	DUODOZEN WESTROCK PACKER	899.136	1067.724	242	1111.94	211.75	862.67839
(fixed value)							



This is a graphical representation of each historical run and the derived speeds from the algorithm. Visually, the algorithm is close enough to the historical run to be considered an accurate algorithm. From this visual observation the algorithm passes the historical reference test.

The graph above shows the line with only the WESTROCK packer running. The MEAD packer is very different from the WESTROCK because it doesn't run at high speeds. Instead it runs at virtually the same speed as the pasteurizer. Therefore such a case would require our application to allow the user to adjust if the unit will continue the algorithm or use the previous value.





Unit	run 1	run 2	Geo Series on run 2	Algorithm on run 2	Geo Series on run 1	Algorithm on run 1
CAN DEPALLETIZER	759.02064	1004.0352	242	968	211.75	758.49054
CAN RINSER	288.0045	319.8489	220	311.12698	192.5	264.8271
CAN FILLER	175	200	200	200	175	175
CAN SEAMER	175	200	200	200	175	175
CAN PASTEURIZER	352.1616	352.1616	220	357.39105	192.5	301.2043
MEAD PACKER	360	360	220	357.39105	192.5	301.2043

4.0 Prototyping Test Plan

4.1 General Test plan:

Test ID	***Test Objective*** ***(Why)***	***Description of Prototype used and of Basic Test Method*** ***(What)***	***Description of Results to be Recorded and how these results will be used (How)***	***Estimated Test duration and planned start date*** ***(When)***
1	To test whether the function is correct or not. (Main part of the project)	Numerical: Excel Prototype: Algorithm on Excel	If the algorithm matches the test cases that we created from historical data.	4-7th march
2	To improve the user interface	Analytical: TestFilght	If each function of the application works Is the app able to run? Yes -> record If no, improve.	6-10th march
	To test the uncertainty and to check the application if it works or no	Experimental: To show the client	Client feedback: If they love it	13-17th march

4.2 Detailed Test Plan:

Test ID	***Test Objective*** ***(Why)***	***Description of Prototype used and of Basic Test Method*** ***(What)***	***Description of Results to be Recorded and how these results will be used (How)***	***Estimated Test duration and planned start date*** ***(When)***
1	[Deploy Test] The objective is to test if we need to buy a developer account to deploy an application for our client. And test the feasibility of using Xcode.	Prototype Xcode tut will be used. An application that will most likely be a tutorial of how to use/ getting started with Xcode will be attempted to be downloaded on an Ipad without a developer account.	**Recorded:** Fail **Response:** If the test fails then a developer account will be used to deploy the real thing. **developer account will be bought when application is fully ready**	26-27th Feb
2	[Pure UI Test] The objective of the test is to make sure the program can be interacted with without issue before focusing on its ability to perform its task.	Test the functionality and ease of use of the UI.(refer to 5.1) Prototype UI Shell will be used. A prototype of the UI that does not perform any real calculations or simulations.	**Recorded:** Score 5/5 students : pass **Response:** Some suggestions have been given and will be taken into account for final UI design (see 5.1) UIkit will continue to be used	4 - 7th March
3	[UI aesthetics Test] The objective is to test the aesthetics of the UI with the user.	Test the aesthetics of the UI with client feedback. Prototype UI Shell will be used. A picture of the UI will be sent to the client for feedback.	**Record:** A number from one to ten will be recorded on the client's likability of the UI colour scheme, Formatting and Aesthetics. Score > 8/10 : pass Score <8/10 : fail **Response:** Ask for feedback and fix the UI. Iterate until it passes.	6-10th March Need time for the Client to respond.
4	[Algorithm Test] The objective of this Test is to prove	Test of the algorithm that predicts the optimal speeds.	**Record:** Margin of Error: 10% Pass. **Response:**	4-7th March

	our ability to calculate the optimal speeds before implementing it into the application.	Prototype ExcelA will be used. An excel spreadsheet, that displays all the elements and functions of the algorithm and when given the historical data it is able to replicate the optimal speeds within a small percentage of error.	Algorithms will be expanded to include the conveyors in the future.	
5	[Integration Test] The objective of this test is to make sure that the display of the input and output is well formatted on the UI.	Test of the algorithm's integration into the UI.Prototype R1 will be used. A first rendition of the application will be created that includes the UI and the algorithm working together, the focus will be on the formatting of the input and output numbers and other variables on the UI and the ability of the user to change aspects of the production line and the algorithm still calculates things accordingly to test cases made from Excel prototype.	**Record:** The Pass or Fail of the applications ability to correctly format the UI and output correct speeds of three test cases derived from the Excel prototype. 3 Passes = Pass < 3 Passes = Fail **Response:** Reference working Excel algorithms and find out what's different or causing problems. Iterate until 3 passes.	9-10th March
6	[Input Error Test] The objective of testing wrong inputs is to make sure that the inputs cannot exceed the	Test of the UI ability to catch error inputs. Prototype R2 will be used. On the second rendition of the application where invalid errors are caught,	**Record:** Number of invalid inputs that were stopped Number of invalid inputs that were accepted The ratio between invalid inputs caught	13-14th March

	max speeds of the conveyors.	a comprehensive collection of bad inputs will be tested on each possible place where the user can do something wrong.	and accepted must be 90% caught to pass. **Response**: catch invalid inputs that were accepted and iterate until pass	
7	[Multiple Tab Test] The objective of this test would be to make sure that the program is able to handle multiple sets of inputs and display each accordingly.	Test of the applications ability to simulate multiple production lines and save and load the users edits to each production line. Prototype R3 will be used. On the third rendition of the application where multiple lines can be added. Multiple production lines will be added to the application. They will be edited, saved and loaded at selected points throughout a simulated use of the application.	**Record:** Based on the information of different production lines. Three production lines will be tested to run on multiple simulations at once. If they can be created, saved and loaded in the right places during regular usage of the application then the test is passed. **Response:** Figure out what went wrong and fix the code and iterate the test until it passes.	13-14th March
8	[Stress Test] The objective of the test would be to figure out and test the limits of the application so measures can be put in place to make sure that the user knows the limit of the application	A stress test on the amount of data can be added to the program before it loses functionality or UI formatting. Prototype R3 will be used. On the Third rendition of the application. Production lines will continuously be added to the application until it crashes. Conveyors	**Record:** The limit to the number of production lines, conveyors and units and numbers that can be inputted into the system before the UI gets messed up or the application crashes.	15-16th March

		and Units will continuously be added to a single line until the UI formatting becomes illegible or the program crashes. Extremely large numbers will be added to the filler speed or other inputs until the application crashes or the UI formatting becomes illegible.	If the limits < the theoretical values that would be needed to run the application under normal conditions (determined from the information we gathered for multiple tab tests) then it fails. **Response:** Find the source of the data storage limit and work around or buy more storage. Fix how data is saved and loaded if there is a formatting issue. Iterate until pass.	
9	[Random Person Ease of Use Test] The objective would be to make sure that the program and final user manual is comprehensive for the client.	A test on the ease of use of random person ability to follow an instruction manual and the application. Prototype R3 will be used. The application and a user manual will be sent to a person with no knowledge of beer production. They will be required to complete a set of instructions from the user manual. They will be asked to give feedback on the ease of use.	**Record:** The feedback and the percentage of tasks that the person was able to complete will be recorded. **Response:** Use feedback to improve the user manual and UI if failed. Iterate until pass.	March 18-20th.

4.2 UI Test plan (detailed)

The objective of the UI Test plan is to check

1: Are potential users able to access and use the app smoothly?

Why: App capability is essential for this project.

2: Is the interface logic appropriate and intuitive-accessible?

Why: User interaction is necessary.

3: Is the App able to run on the Ipad? (client requirement)

Why: The client has Ipads as working equipment.

4: Does the user like the interface formatting?

Why: Beauty is also significant.

When: Before the due date of Deliverable F.

How: In order to test the app, the UI has been shown to random students. Unfortunately, we do not have time to show this UI to clients (clients are satisfied with the final conceptual design).

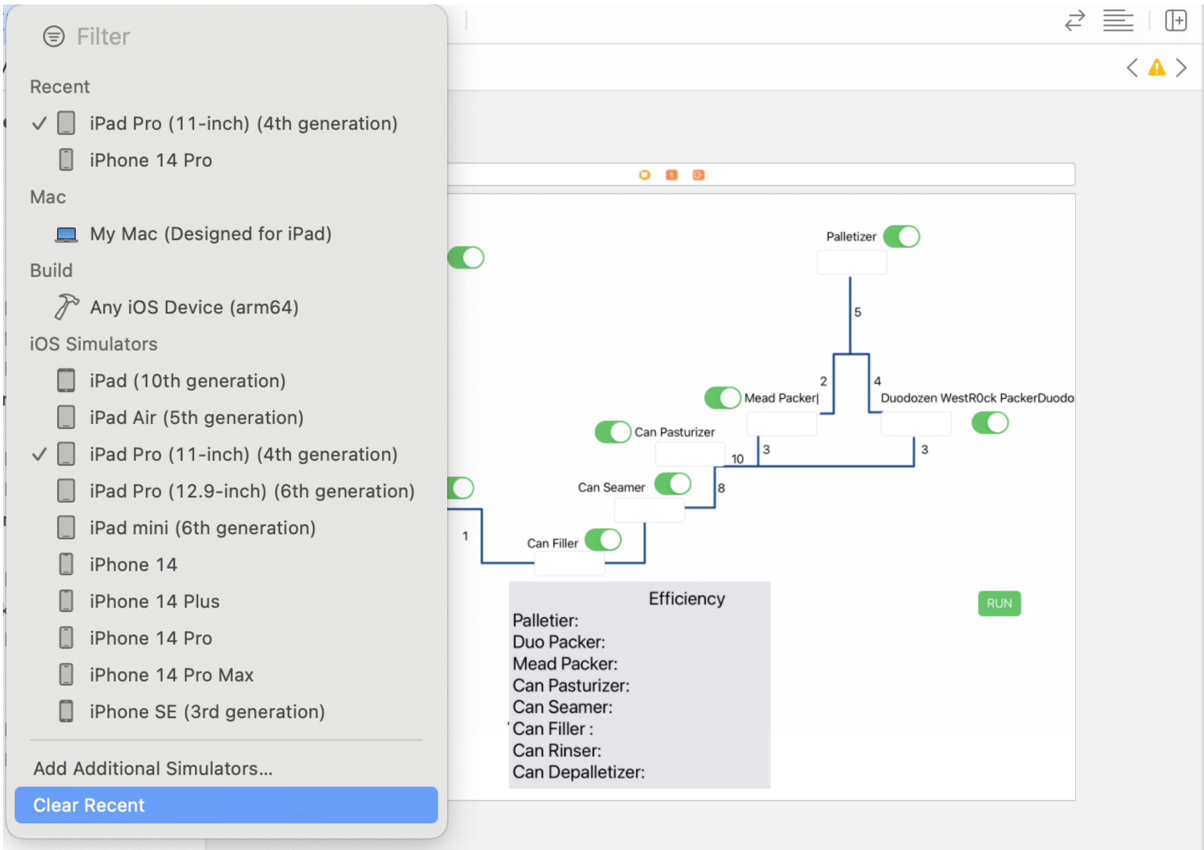
5.0 General Analysis of Critical Subsystems

Reference 3.2.0 for algorithm Critical subsystems

Subsystem #	***Subsystem***	***Components & Materials***	***Properties***
1	Xcode	● Source Editor ● Assistant Editor ● Version Editor ● Interface Builder ● Simulator ● Integrated Build System ● Compilers ● Graphical Debugger	● Supports multiple platforms ● Free for Apple users ● User-friendly interface ● Best application testing
2	Testflight	● Creating app ● Testing app ● Optimising app	● Clear process ● Easy testing ● Automatically records data
3	Excel	● Cells ● Rows ● Columns ● Menu bar ● Formula bar	● Storing Data ● Performing calculation ● Data analysis ● Graphs and charts ● Basic formulas and coding

5.1 UI (detailed)

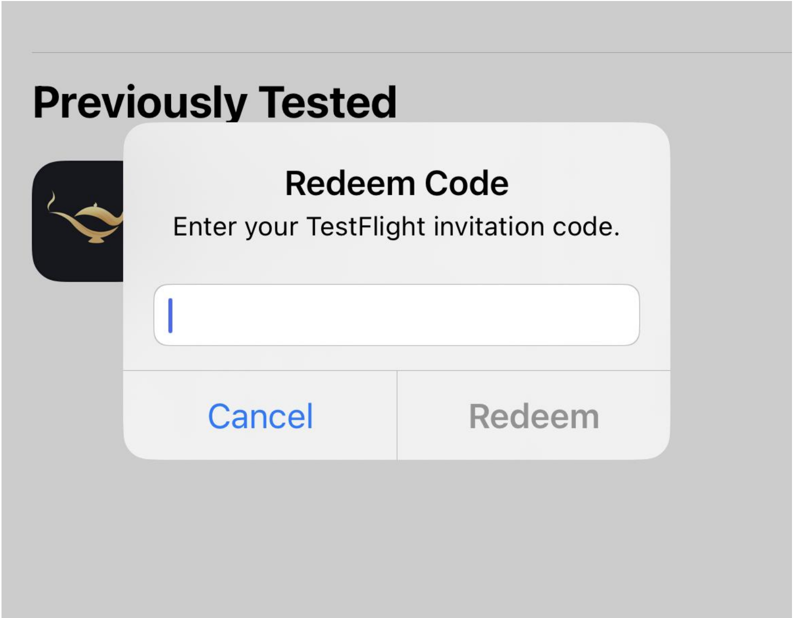
- Supports multiple platforms
  This UI (app) can be accessed through any device that installed the IOS system. Available devices are shown in the following figure.

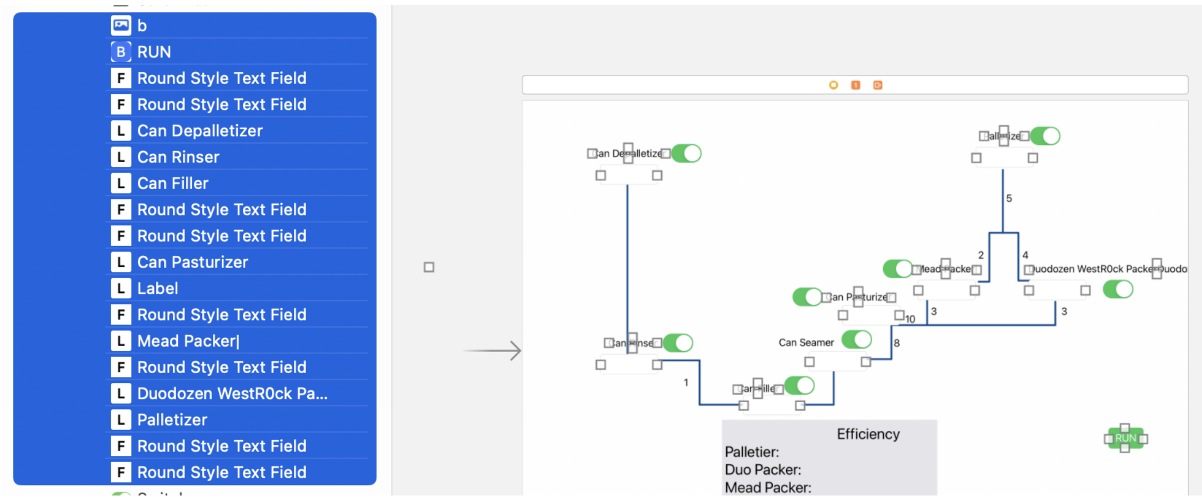


- Free for Apple users

Once the app is published onto TestFlight, users are able to free download it with a redeem code.

Only the user with a redeem code can download and access the app, it raises the security of the app.





- User-friendly interface

-

- The production line is corresponding to the V-curve theory. Users are able to visually see the numerical relationship of the speed of the conveyors. This helps in reducing the error-speed type-in.
- Best application testing (Please see in feedback and comment part).

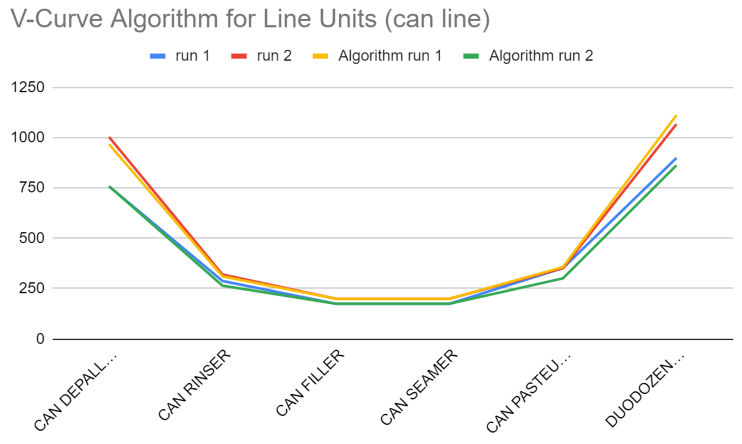
6.0 Feedback and Comments

This part is to show the collected feedback and comments on the prototypes.

6.1 UI

Question	Student A	Student B	Student C	Student D
Are you able to access and use the app smoothly?	YES	YES	YES	YES
Is the interface logic appropriate and intuitive accessible?	The production line makes sense.	I really like the logic of the production line.	I like the design of the V-theory corresponding to the production line.	YES.
Does the user like the interface formatting?	Student A likes the design of the production line.	Student B thinks the table looks a little bit abrupt.	Student C loves the overall design. However, she thinks there are too many switches.	Student D thinks the Table can be placed to the right of the production line.
Is the app able to run on the Ipad?	YES	YES	YES	YES

6.2 Algorithm



Through visual representation, It is clear that the algorithm is able to derive unit speeds that are very close to the historical data. From this observation the algorithm passes the historical data test. Next week our team will work on expanding the algorithm to solve for conveyor speeds aswell.

Wrike Snapshot:

https://www.wrike.com/frontend/ganttchart/index.html?snapshotId=o7djfrcgqKbFJpT7HOIRJjcpBmwneGtp%7CIE2DSNZVHA2DELSTGIYA

7.0 References

(n.d.). Checking link. Retrieved March 5, 2023, https://statics.teams.cdn.office.net/evergreen-assets/safelinks/1/atp-safelinks.html

Ekren, Kemal. "What Is Xcode and How to Use It?" Netguru, 25 November 2022, https://www.netguru.com/blog/what-is-xcode-and-how-to-use-it

How to Use TestFlight For Beta Testing: Benefits & Pre-Requisites- Testrig. (n.d.). Testrig Technologies. Retrieved March 4, 2023, from https://www.testrigtechnologies.com/ios-application-beta-testing-using-testflight

Is the V-Curve the best option for setting accumulation line speed? (n.d.). LineView Solutions. Retrieved March 5, 2023, from https://news.lineview.com/is-the-v-curve-theory-the-best-option-for-setting-accumulation-line-speed

Microsoft Excel Spreadsheet Software. (n.d.). Microsoft. Retrieved February 19, 2023, from https://www.microsoft.com/en-us/microsoft-365/excel

Xcode 14 Overview. (n.d.). Apple Developer. Retrieved February 19, 2023, from https://developer.apple.com/xcode/