S.P.I.C.E.

Final Design Report

Salty Seniors

John Paul Bartsch

Caleb Herrera

Carlos Zapata III

Kile Zimmermann

Department of Computer Science

Texas A&M University

May 1st, 2023

Table of Contents

1 Executive summary

The S.P.I.C.E. project is a solution to a growing problem in the United States. Approximately 27% of adults over 60 live alone. This is in distinct contrast to other countries which average 16%. [1] Considering that this demographic will only continue to grow, it's crucial that we develop tools that make everyday activities, such as cooking, more convenient and accessible. The S.P.I.C.E. project aims to address this growing need by developing a platform that simplifies the spice measurement process. This platform was distinguished based on three major design objectives:

- **Accessibility:** As an accessibility tool, SPICE's accessibility features must be useful and intuitive for everyone regardless of their physical capabilities. Considering motor control may not be the only disability SPICE users have, it is important that its features reach a wide array of needs. For example, large buttons were much easier to both press and see, and making sure that text is large enough to be read comfortably from a distance ensured ease of use. Additionally a touch screen made the user experience more intuitive, as there were no additional buttons that needed to be pressed in order to operate the SPICE platform. Overall the goal of accessibility was to incorporate features that would ensure ease of use.

- **Convenience:** This is an absolutely critical aspect of the S.P.I.C.E. project, as it ensured that the tool could be easily integrated into a user's daily routine. For older adults, convenience is especially important, as they may have difficulty with manual dexterity or memory. As a result the S.P.I.C.E. platform was designed in a way that was intuitive and straightforward, with no need for an overly convoluted manual. Additionally, the actual spice dispensing process was done in a timely manner as to not cause a roadblock in the cooking process. The idea is that S.P.I.C.E. would be used both before and during cooking. Imagine a user is cooking something on a skillet and needs a specific mix of spices to season their meal. The S.P.I.C.E. platform would be able to dispense that mix at the press of a button, leaving the user to attend the stove and make sure that their food does not burn. Convenience ultimately loops back to accessibility, as a convenient system will also ensure accessibility to more people.

- **Easy Maintenance:** Throughout the design process, the S.P.I.C.E. team strived to make all components of the S.P.I.C.E. tool easily detachable and washable. This ensured that users could swap spices and maintain the device without encountering difficulties. By making these components removable, users could disassemble and clean the device without encountering any complications, thereby minimizing the time and effort required for maintenance. This ensures that the platform is accessible to older adults that may struggle with other complex maintenance routines.

The final spice platform was able to fulfill these design objectives with validation from the SPICE development team. Although there were a few issues along the way, every implementation challenge came with a fix. The first challenge was part fabrication, as a large portion of this project was in the mechanical components. Creating parts that fit together well took a number of iterations, but eventually led to a housing that could be assembled and disassembled with ease. These parts were then validated by adding DC motors and testing that they could spin with the dispensing spiral attached. This leads me to the next design challenge, friction. The motors used on this project did not have an attached gearbox, and as a result, did not have very much torque. This kept the dispensing mechanism from working initially, but this would eventually get worked through over more iteration. After the physical components were complete, the next major design challenge was integrating the User Interface on the Raspberry Pi with motor control on the Arduino. This portion of the project underwent extensive testing, and eventually culminated in the complete S.P.I.C.E. functionality.

Overall, the development process had a number of bumps along the way, all of which required critical thinking from the team as a whole to resolve. Being able to work through these problems in a time efficient manner was crucial to the success of the project. Having focus areas for each member to work on gave us a wide range of expertise and allowed the team to work on multiple parts of the project in parallel. If the project had been completely roadblocked at anypoint, the S.P.I.C.E. project would likely not have been a success.

2 Project background

As mentioned previously, it is estimated that close to 27% of adults older than sixty in the United States live by themselves. This is in stark contrast to foreign countries which average about 16%. Based on economic and cultural factors, older adults in the United States are significantly more likely to live by themselves or with one other person (usually a spouse of approximately the same age). [1] Based on a 2022 study conducted by Dr. Laurence Knott, age accounts for approximately 1-2% loss in functional ability per year [2]. This comes as a result of changing body composition, reduction in bone mass and blood volume, and other physiological changes. Due to frailty with age, problems can begin to arise as these adults begin to gradually lose motor control.

There are also adults under sixty but still struggle with motor function disabilities. Approximately 39 million Americans in all age groups suffer from some kind of motor impairment disability. [3] These two groups may find it difficult to complete tasks requiring precise movements. One of the most important of those tasks being cooking, which can require various kinds of precise motor control. It's easy to take for granted the ease of measuring out something small like a tablespoon or teaspoon. However, not everyone can easily make these smaller measurements, especially as they age. The scope of the S.P.I.C.E. project encompasses spice measurement and mixing, both as an accessibility and convenience tool.

2.1 Needs statement

Fine motor control can decline with age, but also comes as a result of various disabilities. [3] In a country where one in four adults have some form of motor impairment and older adults are more likely to live alone, there's a serious need for tools that can make tasks easier for motor impaired adults. [1] In order to meet this need, S.P.I.C.E. aims to make cooking easier and more convenient for households where measuring spices for recipes or manually dispensing them may be difficult.

2.2 Goal and objectives

The goal of this project is to reduce the strain and inconvenience of precisely measuring quantities of spices for people with motor disabilities when cooking.

The following is a list of objectives that were taken into consideration throughout the S.P.I.C.E. design process:

- Accessibility
    - The system should include an interface that is easy to use.
    - The system UI should be intuitive and naturally guide users to operations they want.
    - The distribution of spices must be accurate to $\pm$ 0.1 teaspoon.
    - The system should be capable of dispensing a minimum measurement of 1 teaspoon
- Convenience
    - The system must support at least four different spices with different spice mixes.
    - The overall design must have a reasonable level of water resistance.
    - The system must be able to properly function in the temperature range 65-75°F(18-23°C).

    - The system must fit in a 45x45x45cm box.
- Easy Maintenance
    - The system must be easy to disassemble and maintain.
    - Cleaning spice housings must be easy and intuitive.
    - Spice dispensers must remain unclogged even after prolonged use.

2.3 Design constraints and feasibility

The most significant constraint of the project was the budget. This project aimed to be an affordable accessibility solution for people with declining motor function. There were no similar solutions available for purchase, so there were no commercial competitors and products available for a price reference. However, the available $200 budget was used as a guideline to keep the project at a reasonable price for the first of its kind officially on market.

From an ergonomic perspective, the system needs to exist and operate comfortably in a kitchen environment. As with most of the previously mentioned solutions, this system needs to function at an average room temperature of 65-75°F (18-23°C), withstand some moisture levels, and fit within a reasonably chosen size constraint measuring 45x45x45cm.

Lastly, there are a few quality of life constraints that will influence this project. To be successful, this product needs to accurately dispense a minimum spice measurement of one teaspoon with a tolerance of plus or minus one-tenth of a teaspoon. To remain relevant to the accessibility goal, it also needs to operate through a touch screen interface featuring large buttons and an intuitive design that is learnable with minimal effort.

For the most part, this project was feasible. However, the team encountered difficulties with implementing conversions between teaspoons dispensed and the time it takes for motors to spin. Otherwise, the team was still able to get the system to dispense spices, which is one of the goals of S.P.I.C.E.; to prevent a user from having to manually dispense by hand.

2.4 Literature and technical survey

While S.P.I.C.E. is not the first of its kind by far, the intentions behind it are what make this project unique. There have been several takes on this style of project already, each with their own pros and cons. *TasteTro [4],* The *Automatic Spice Dispenser [5], Spicer [6], MeasureMINT [7],* and *Fab Academy Final Project [8]* all feature similar designs to S.P.I.C.E.'s original solution. However unlike the past solutions, S.P.I.C.E. is intended as an accessibility tool. With this in mind, this project emphasizes focus on a simplified user experience.

TasteTro [4]

TasteTro is the most commercial implementation similar to this project. It's also been commercially advertised since January 2018, and has received multiple patents in the US, Canada, and Australia. The most substantial difference between TasteTro and S.P.I.C.E. is their options for distribution. S.P.I.C.E. incorporates both recipe and individual spice selection capabilities, whereas TasteTro appears to operate exclusively on predetermined recipes. One aspect of TasteTro that S.P.I.C.E. may take inspiration from would be the spice storage and distribution mechanism, which is a clever cartridge based design. Another admirable aspect of TasteTro's design is its simple three button user interface.

Figure 1: TasteTrio Colorways

Automatic Spice Dispenser [5]

Jeffrey Wang was a mechanical engineering student at UCLA. His senior design project, Automated Spice Dispenser, is very similar to the S.P.I.C.E. It includes a dispenser for up to ten spices, a maximum dispensing time of 10 seconds, and is accurate to within a tenth of a gram. The project also includes pre-programmed spice mixes. Some secondary goals of Jeffrey's project were low spice warnings, manual spice dispensing and a user interface. The rough model of his product is shown below in figure 1:

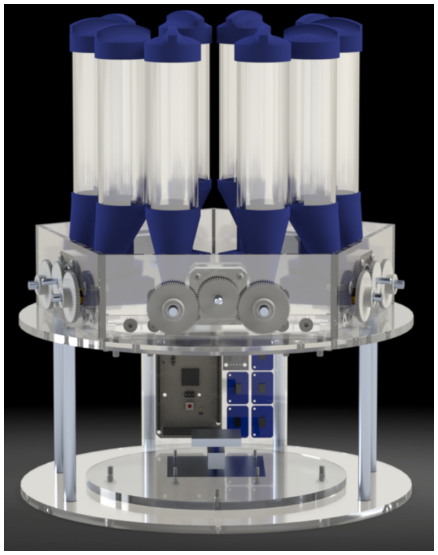


Figure 2: Wang's visualized project design for the Automatic Spice Dispenser

A point for S.P.I.C.E. to improve upon could be the user interface since this dispenser does not have any kind of user interface. Additionally, other improvements could include the addition of spice mixes programmed for specific meals. There was more focus on the fabrication of components and mechanisms since Wang's project was a mechanical engineering based capstone. As computer engineering students, it would be in our best interest to emphasize on the electrical and software components of our project.

Spicer [6]

The "Spicer" is one of the most exhaustive senior design projects we have found related to automated spice dispensing. It has vast functionality and includes many ideas that we were planning on implementing in our own project. It is capable of holding up to 8 spices.

This senior project has an extensive hardware and software design. The hardware design included three microcontrollers that controlled various aspects of the device. The software design included a mobile app which allows users to interact with the device via their smartphone. One of the biggest feats this project accomplished is utilizing machine learning to let users control the device with hand gestures.

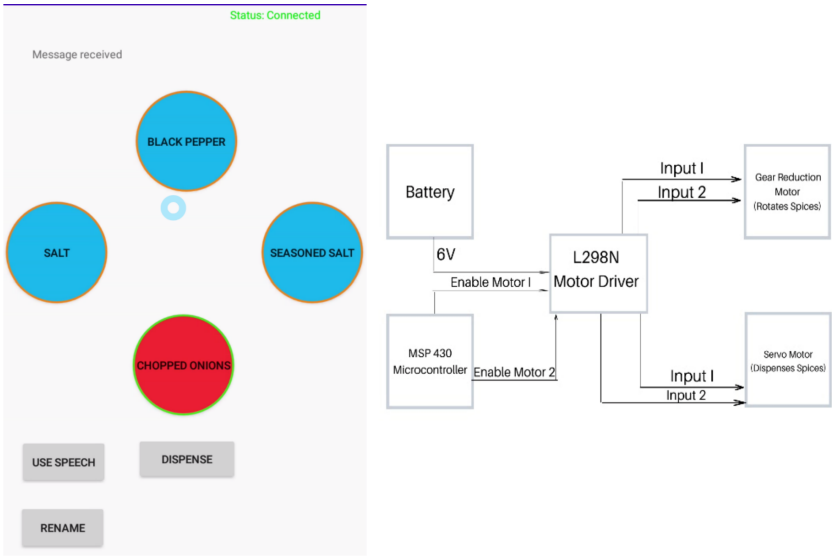


Figure 3: Main UI of the Spicer (left) and flowchart of the hardware design (right)

The team working on this project set ambitious goals, many of which were not fully completed in the first prototype. However, by the second prototype the team was able to implement most features they wanted into the project. One thing S.P.I.C.E. can improve on is the ability to web scrape recipes online and figure out what spices are needed. The team originally planned for "Spicer" to be able to do this but were not able to fully implement this feature at the end of the project.

MeasureMINT [7]

The MeasureMINT is an automatic spice dispenser that is capable of dispensing single quantities of spices based on the user's preference. It can hold up to twelve spices and can dispense spices in various quantities based on user input with a touchscreen, which is similar to S.P.I.C.E.'s proposed functionality. Some future goals that the creator mentioned were voice recognition and the ability to dispense multiple spices at a time. The creator's product is shown below in figure 4:

Figure 4: MeasureMINT

There is a recipe option shown on the UI in the demonstration for the product, but it is uncertain how exactly it works and what components are involved in that aspect of the MeasureMINT. The S.P.I.C.E.'s goal is specifically to account for recipes and potentially include a networking component such that the product can scrape data from a website and dispense spices based on spice keywords that it is able to retrieve. Another improvement would be related to the UI functionality. The demonstration shows that it may be time consuming for the user to input their data into the system, and one particular goal of S.P.I.C.E. is to make the seasoning process efficient and not tedious.

The proposed S.P.I.C.E. design is expected to be faster than MeasureMINT, as it is planned to have a more efficient process for dispensing spices. It will also have more unique functionality compared to MeasureMINT, especially with the web scraping component for recipes.

Fab Academy Final Project [8]

Fab Academy is a 20 week program with a specialization in prototyping and planning with a distributed education model. One of Muhammed's final projects was a spice dispenser as shown below.

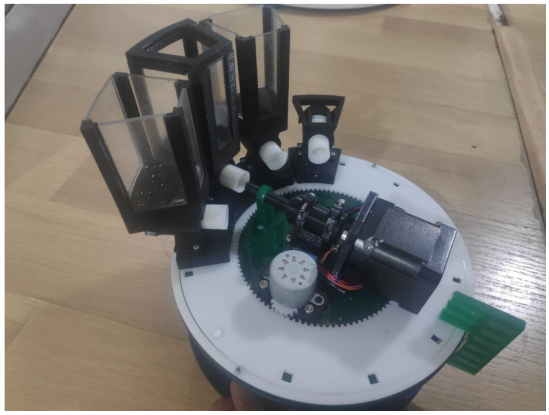


Figure 5: Fahiz Spice Dispenser

The project shown in figure 5 is similar to the measureMINT project, as it uses a single motor to rotate spices and a sliding shaft to connect a motor to the dispensing mechanism. Some downsides to this

product are that it rotates and dispenses slowly, doesn't contain a UI, and no specific measurements are taken into account.

Conclusion
Whilst these products have similar components to what S.P.I.C.E. plans to incorporate, S.P.I.C.E. intends to resolve some of the issues these products were experiencing or incorporate the features that some of these products were missing. For instance, there were some products that did not incorporate accurate measurements or a UI, and there were other products that were just inefficient. S.P.I.C.E.'s goal is to ensure that those problems are addressed alongside taking into account the needs of cooking.

2.5 Evaluation of alternative solutions

- Use rotating mechanism to house spices
    - The original idea for S.P.I.C.E.'s housing system was a linear design where each spice had its own distribution unit and all units connected with tubing to guide distributed spice to the desired location. After researching alternative solutions, using a circular, carousel-style, design appears to be the most efficient choice for organization. Although a circular design will require more space, it will also simplify software development aspects by eliminating the need for multiple distribution points.

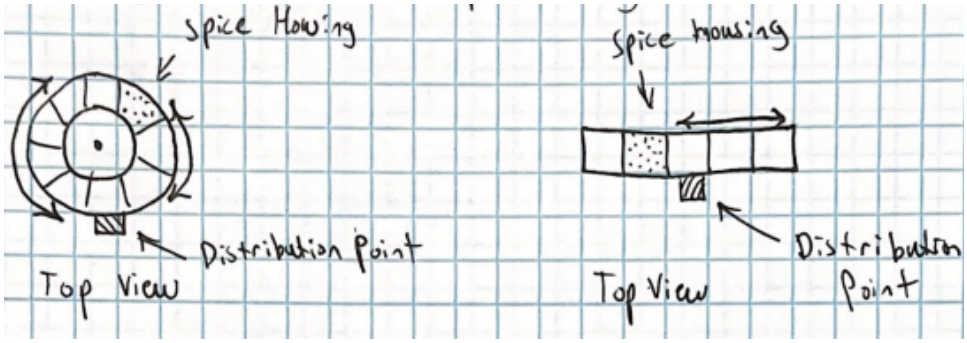


Figure 6: Rough sketch of alternate mechanisms

- Use corkscrew mechanism for spice distribution
    - The corkscrew dispensing mechanism was used in multiple projects found during the technical survey. In the Automatic Spice Dispenser [] , this system utilized a motor and 3D printed corkscrew for each individual spice housing. This approach would increase the overall cost of our prototype and is not advisable. However, the measureMINT and FabAcademy projects both improved on this design by utilizing a sliding shaft that could be connected to and retract from each spice housing. This reduced the overall number of motors necessary to implement their spice dispensing systems. While this seems like the most viable option, the mechanical design aspect of this mechanism may prove challenging in tandem with the rotating spice mechanism.

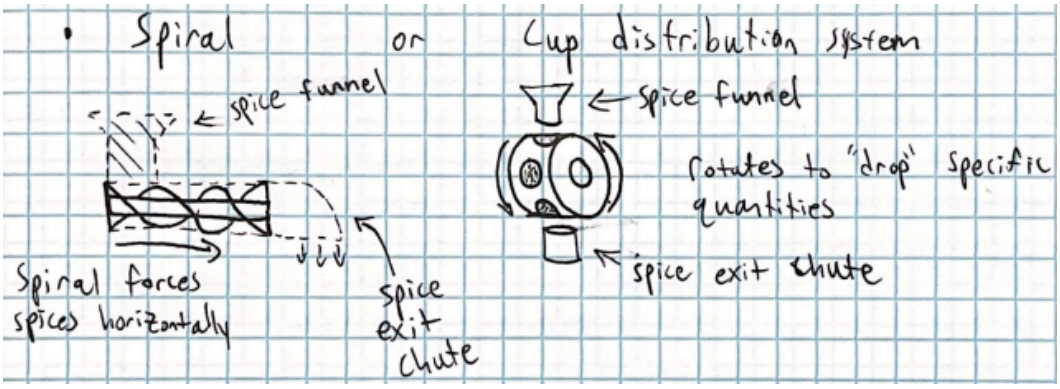


Figure 7: Rough sketch of distribution mechanism ideas

- Support recipe options and individual spice options
    - The ability to support recipe options would be a very useful feature for those who need multiple spices mixed together. This functionality will parse a recipe that is input by the user and figure out all the spices required. Parsing recipes will save time because the user won't have to run S.P.I.C.E. multiple times if they need multiple spices. This route also saves effort on the user end since the user doesn't have to know exactly what spices they need because the device will figure it out for them. Both options could be implemented into the device for maximum versatility. By adding an option to input recipes, S.P.I.C.E. will have an advantage over other products of this nature since many of them only dispense spice based on what the user wants.

- Mobile application vs. interface attached to device
    - Using a mobile application for controlling the device has a major upside in the form of convenience. Users don't have to be right in front of the machine in order to use it. A downside to this option is since the product is geared towards elderly or those with motor impairments, it could be the case that they don't own a smartphone or find smartphones hard to use. By using an interface attached to the device it eliminates the need for an external device to control the S.P.I.C.E. It is slightly less convenient since the user will have to be in front of the device. Lastly, the team concluded the interface on the device will be easier to implement versus a mobile application. Therefore, it has been decided that S.P.I.C.E. will have an interface attached to the device.

- Voice recognition capability
    - The final alternative option would be to have a voice recognition feature alongside the UI. The user can simply tell the machine what spices they want to dispense or what recipes they want to follow, as well as the spice measurements they wish to take (less button-pressing for measuring spices). A significant advantage to this feature is efficiency, as it would greatly reduce the need to be constantly pressing buttons to input data into the system. One particular goal of S.P.I.C.E. is that it should be easy to use, which this feature is intended to achieve. However, a particular disadvantage to this feature is that it doesn't consider people who are deaf, as they would not be able to speak into the system, and it might also be a difficult implementation.

3 Final design

3.1 System description

The high level block diagram for S.P.I.C.E. is shown below in figure 6:

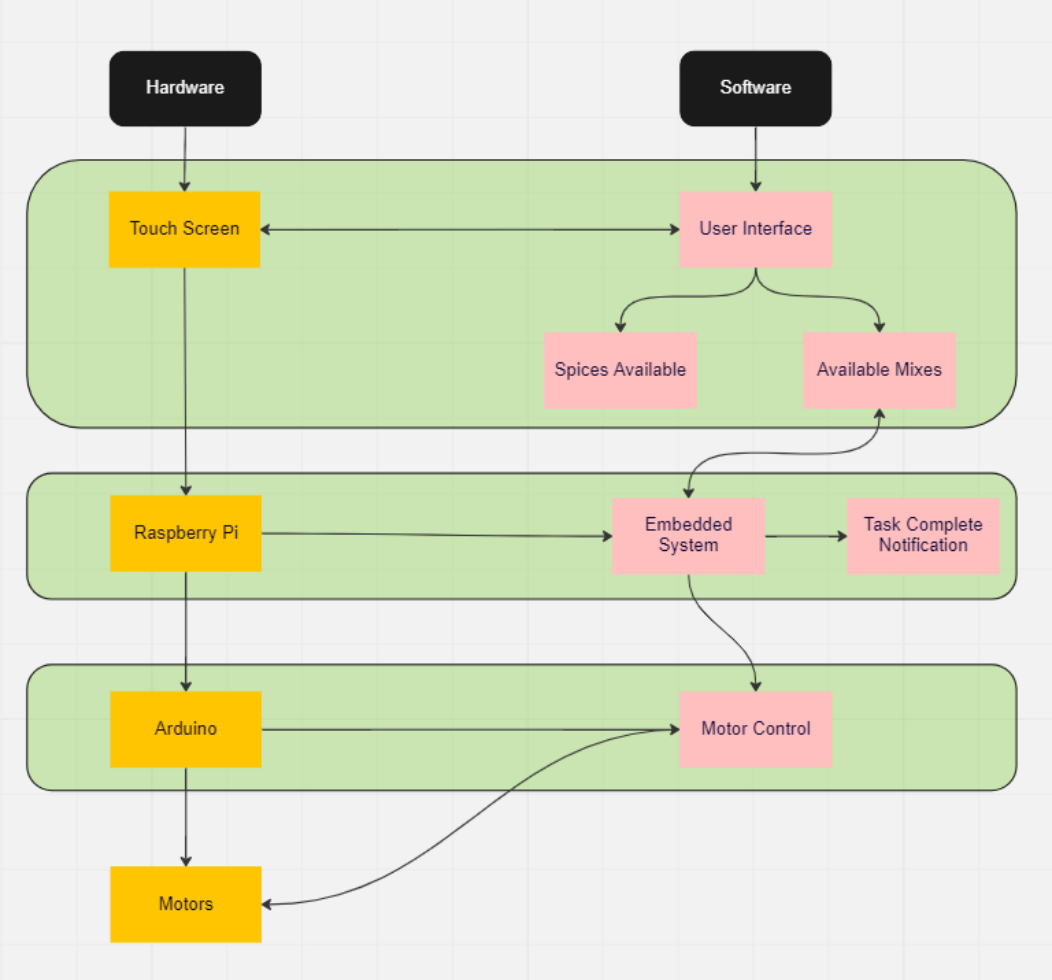


Figure 6: High Level Block Diagram

Touch Screen:

The touch screen acts as the main point of contact for users. It displays the user interface, and allows for users to interact with S.P.I.C.E. This is a key component of the project and differentiates it from a few of the others from the literature review.

Interface Buttons:

An alternative solution to the touch screen interface is physical buttons. This gives tactile feedback to users, making the interface more responsive.

Raspberry Pi:

The Raspberry Pi acts as the brains of the device. It gives the user interface functionality on the back-end. The Raspberry Pi also processes human interactions and sends the data to the Arduino which then controls the servo motors.

Arduino:

The Arduino serves as the controller for all the servos and motors throughout the device. It receives instructions from the Raspberry Pi on which servo motors to activate depending on the user input from the touch screen interface.

Motors:

The orientation motor controls which spice is being dispensed at a given time, while the distribution motors drive the actual mechanism that accurately dispenses each individual spice for the user. Orientation and distribution are used as identifiers in this case and do not relate to any technical term for a type of motor. These motors receive instructions exclusively from the Arduino microcontroller.

User Interface:

The User Interface (UI) is a key feature for accessibility and also serves as a point of interaction for users. Along with the touch screen, the UI provides all the necessary functionality needed for the user to record their data into S.P.I.C.E. for desired results. The user should be able to choose between two different options: “Recipes” and “Spices”. The “Recipes” option is designed specifically for recipes shown in the UI with pre-selected spices available in the system. Once a recipe is selected by the user, the UI displays a summary of the recipes to be dispensed. The user can then confirm that those are the correct spices, and then the system dispenses the spices as instructed. The “Spices” option accommodates users who just want to dispense custom quantities of spices or multiple different spices of their choosing, which is entirely independent of recipes. Upon selection, the UI prompts the user to select the spices they intend to dispense. After the user selects their spices, they can input the amount in teaspoons of each spice they want to dispense from the machine. The measurement feature is just a front-end feature for now since the team was unable to implement the backend for the UI to communicate with the Arduino with this specific feature. A diagram of the UI and its functionality is shown below in figure 7:

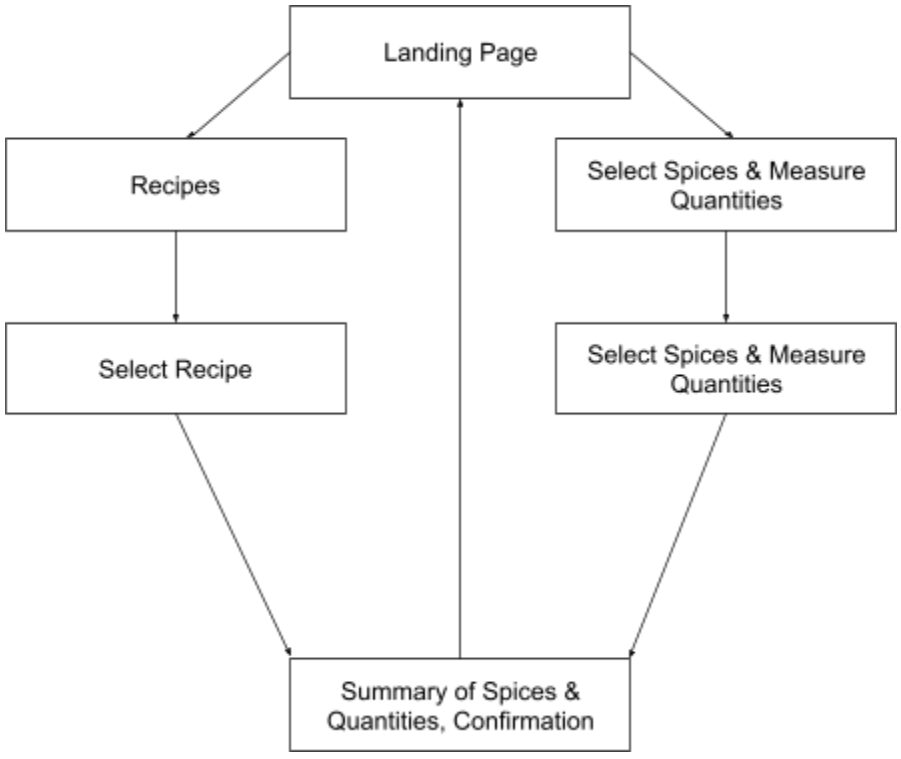


Figure 7: S.P.I.C.E. User Interface

This project is broken into two major elements: the hardware and the software. This project is executed with as little as: a motor controller, motors, well designed spice containers, and a carousel mechanism.

The goal is to utilize a raspberry pi to handle human interactions through the touch screen module. Those interactions would then be translated into necessary instructions for the Arduino to interpret and relay to the corresponding motors.

3.2 Complete module-wise specifications

Circuit Diagram

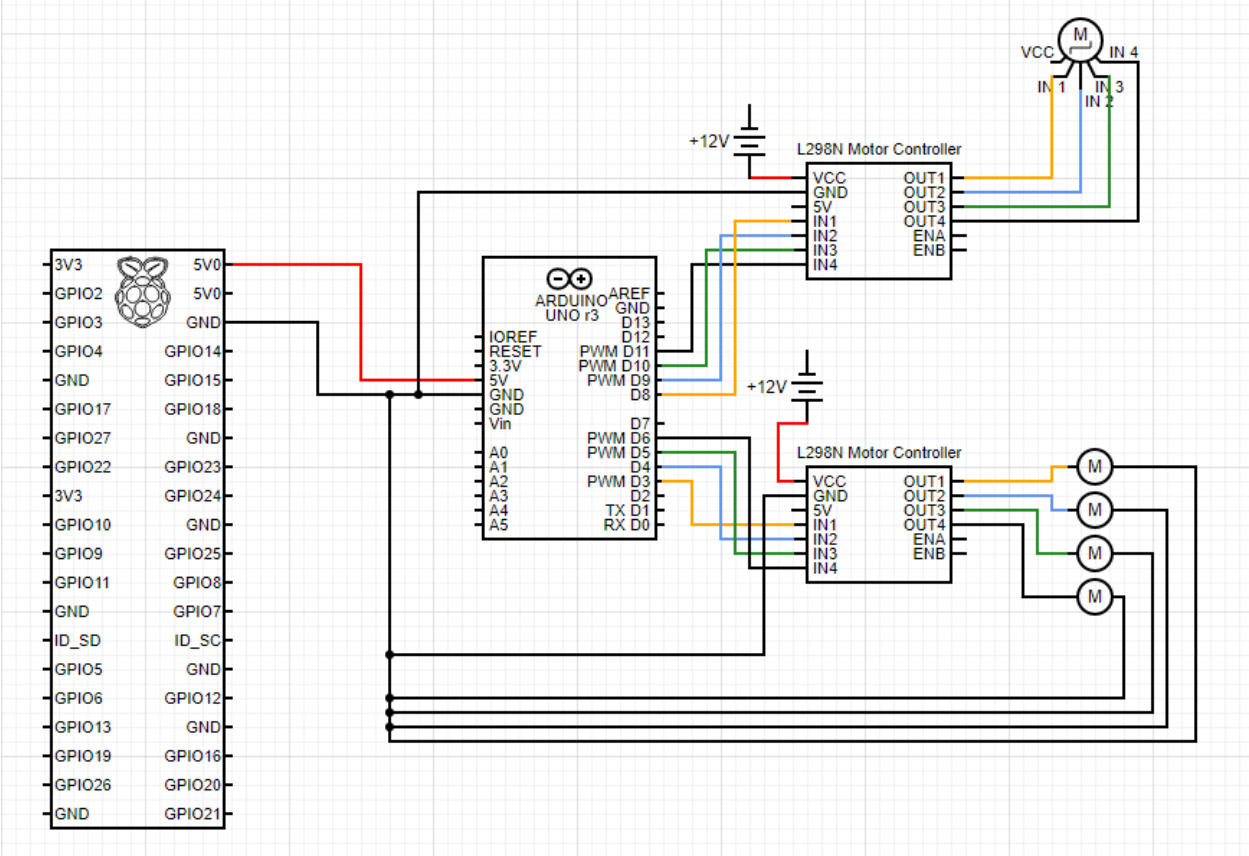


Figure 8: Complete Hardware Implementation

- The circuit diagram shown above in figure 8 features a Raspberry Pi, an Arduino, two motor controllers, 4 DC motors, and a stepper motor. The Arduino is basically responsible for receiving signals from whatever is operating the Java code, whether it be the Raspberry Pi shown above or a regular laptop. Those signals will then be sent through the motor controllers, which are responsible for controlling certain DC motors or the stepper motor.

Interfaces and pin-outs

- The team decided to use a 7 inch 1024x600 touch screen to support the user interface. The touch screen will be powered through an HDMI to micro-HDMI cable and will operate under 5V. The touch screen is shown below in figure 9:

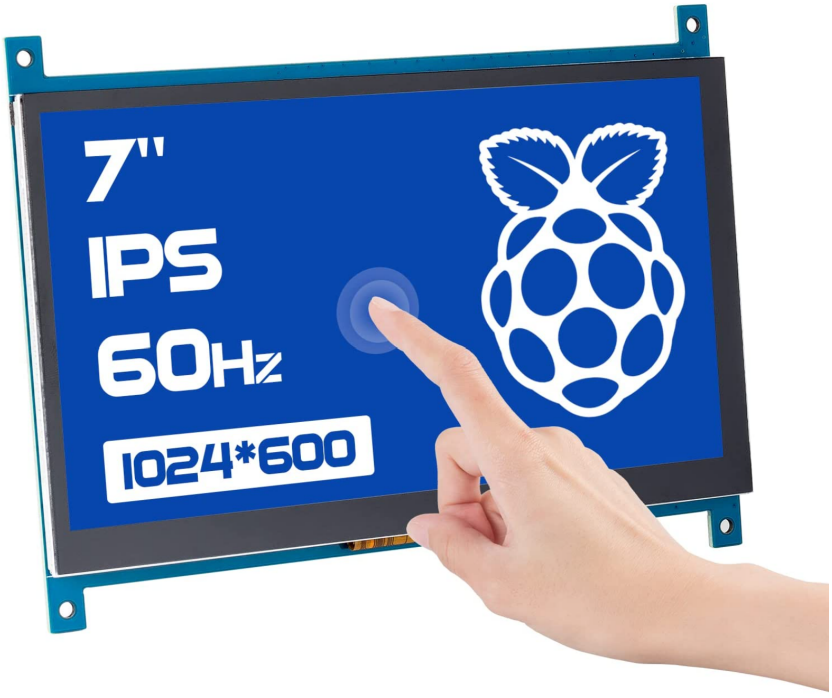
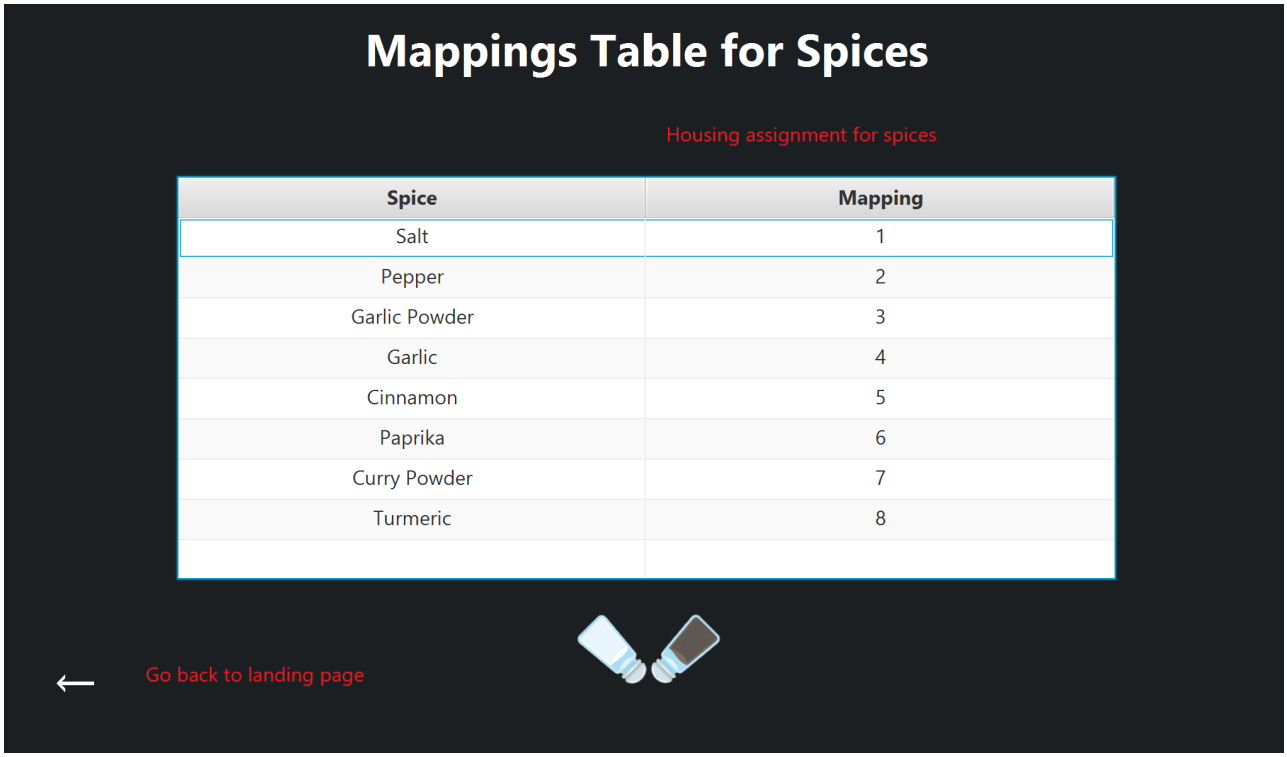
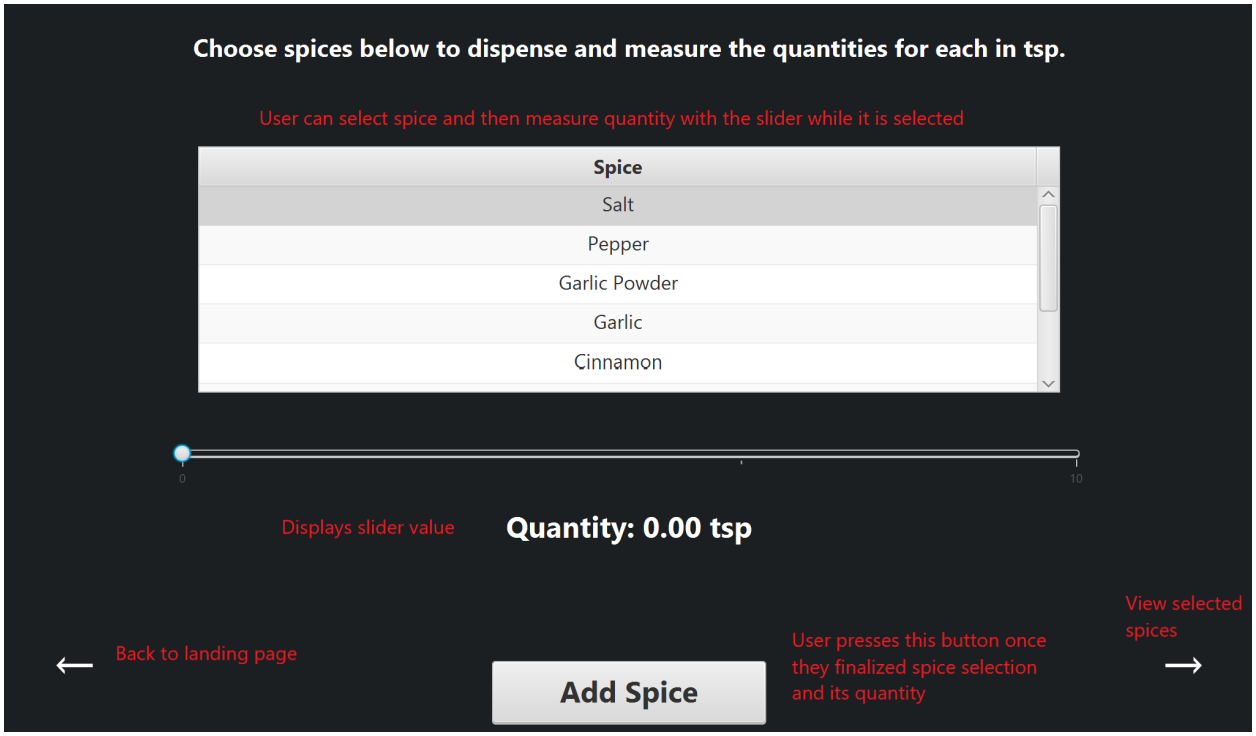
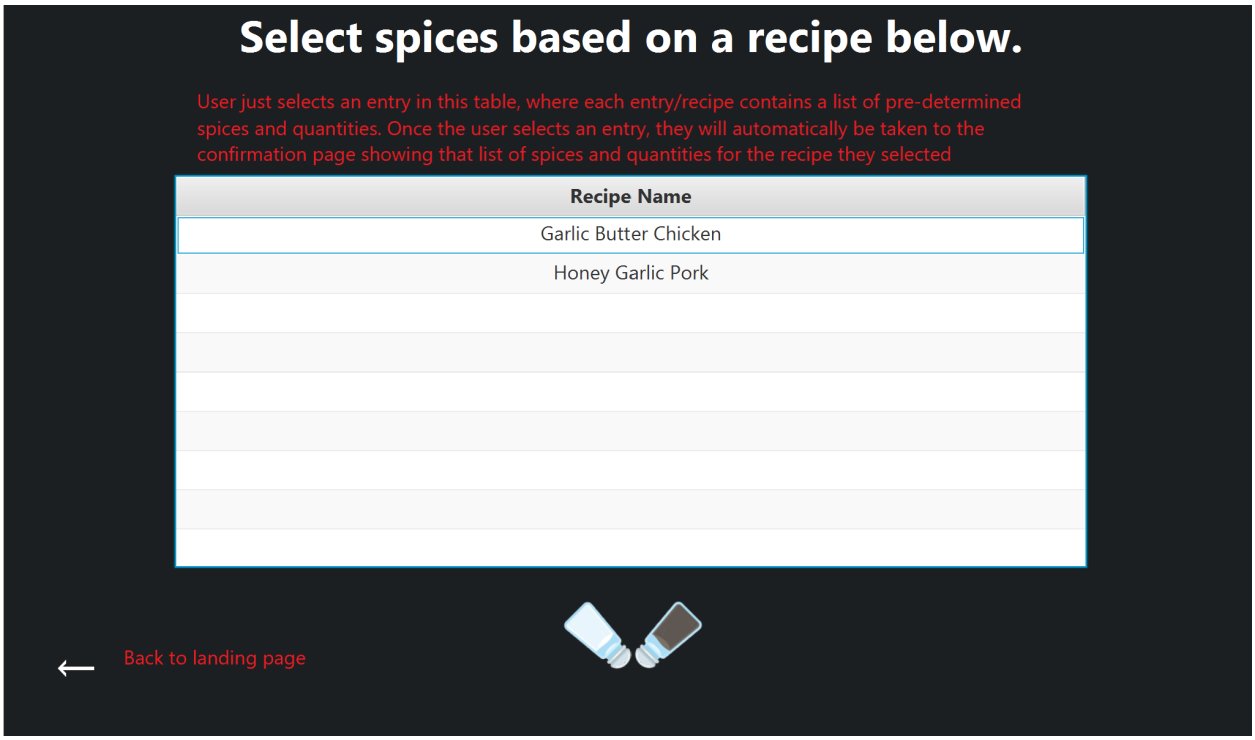


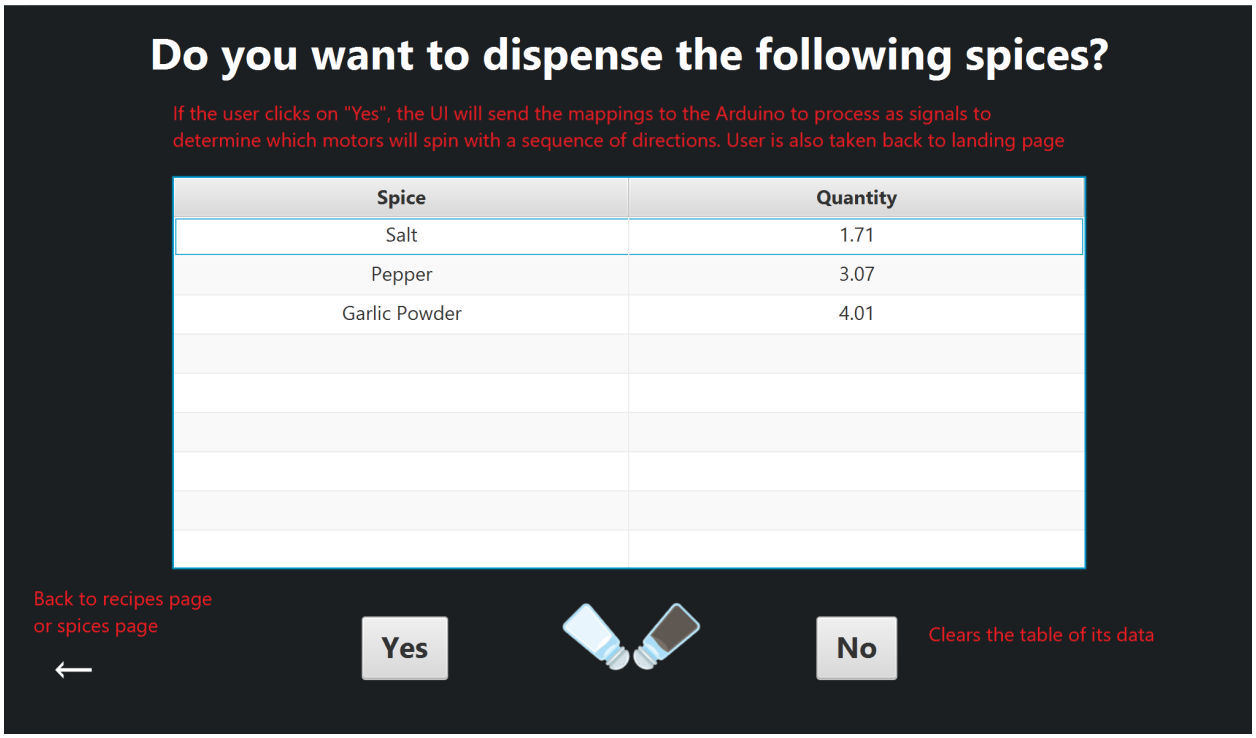
Figure 9: Touch Screen Component

Software Processes with UI

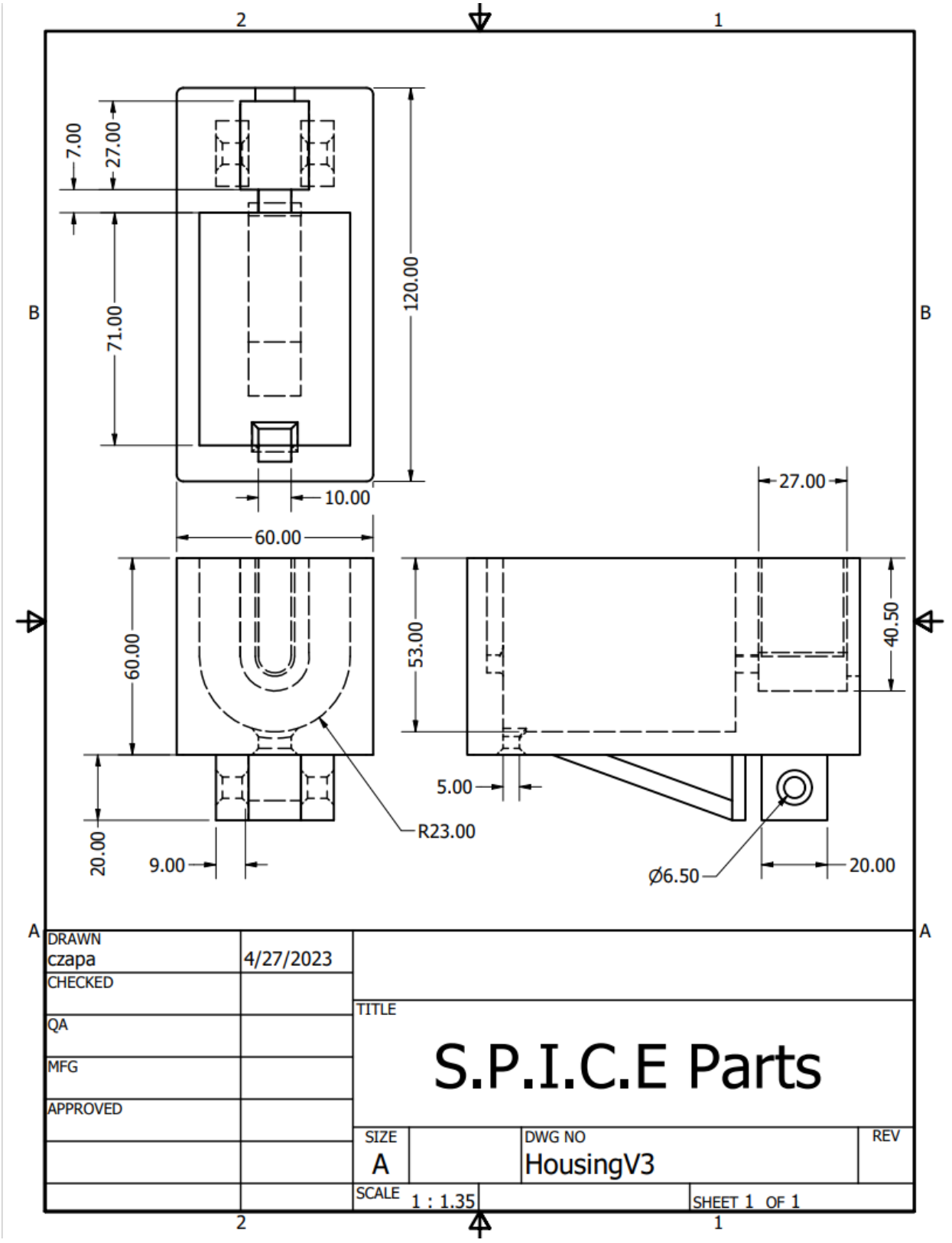


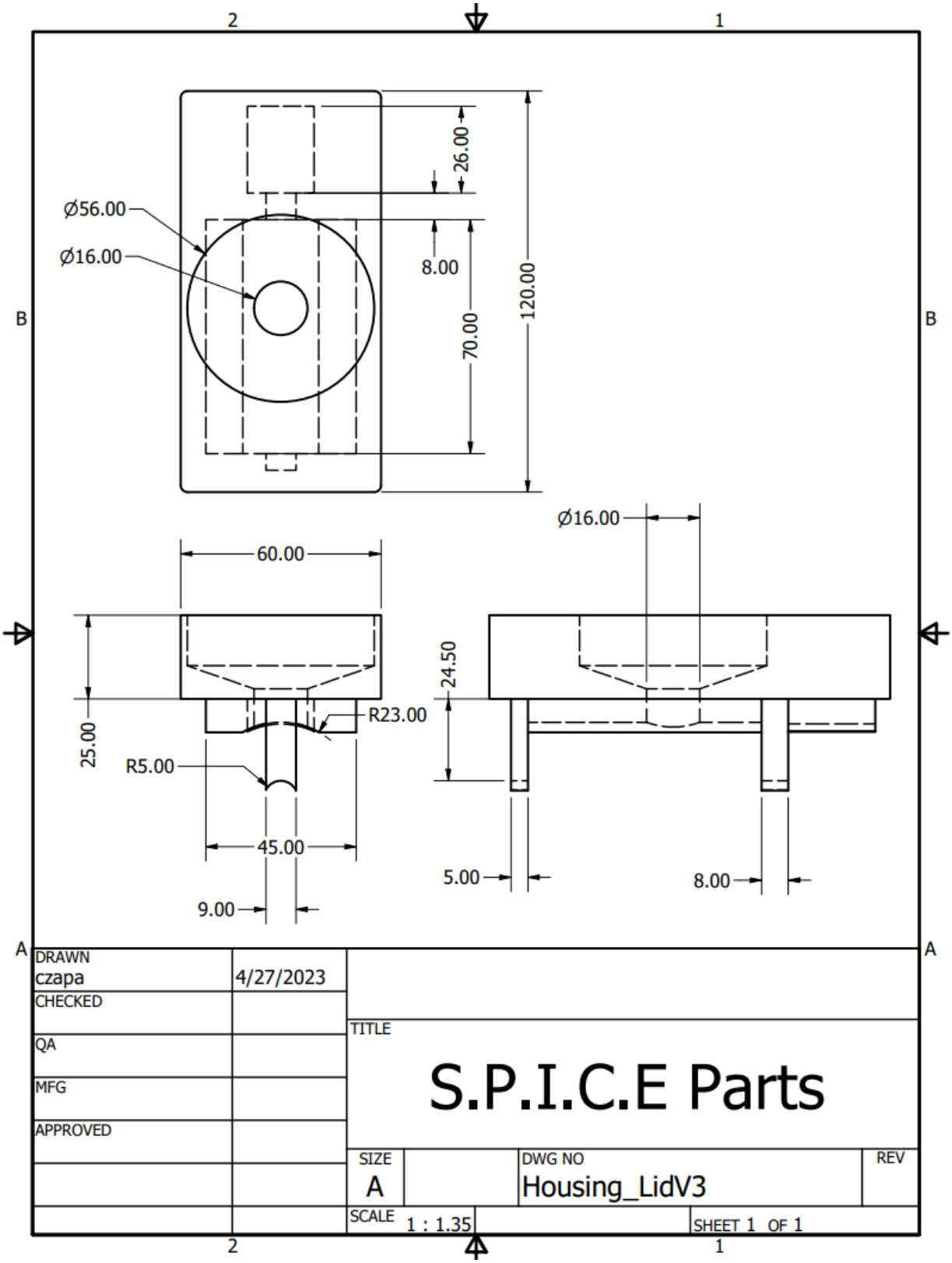

Choose spices below to dispense and measure the quantities for each in tsp.
User can select spice and then measure quantity with the slider while it is selected
Spice
Salt
Pepper
Garlic Powder
Garlic
Cinnamon
0
10
Displays slider value
Quantity: 0.00 tsp
Back to landing page
Add Spice
User presses this button once they finalized spice selection and its quantity
View selected spices

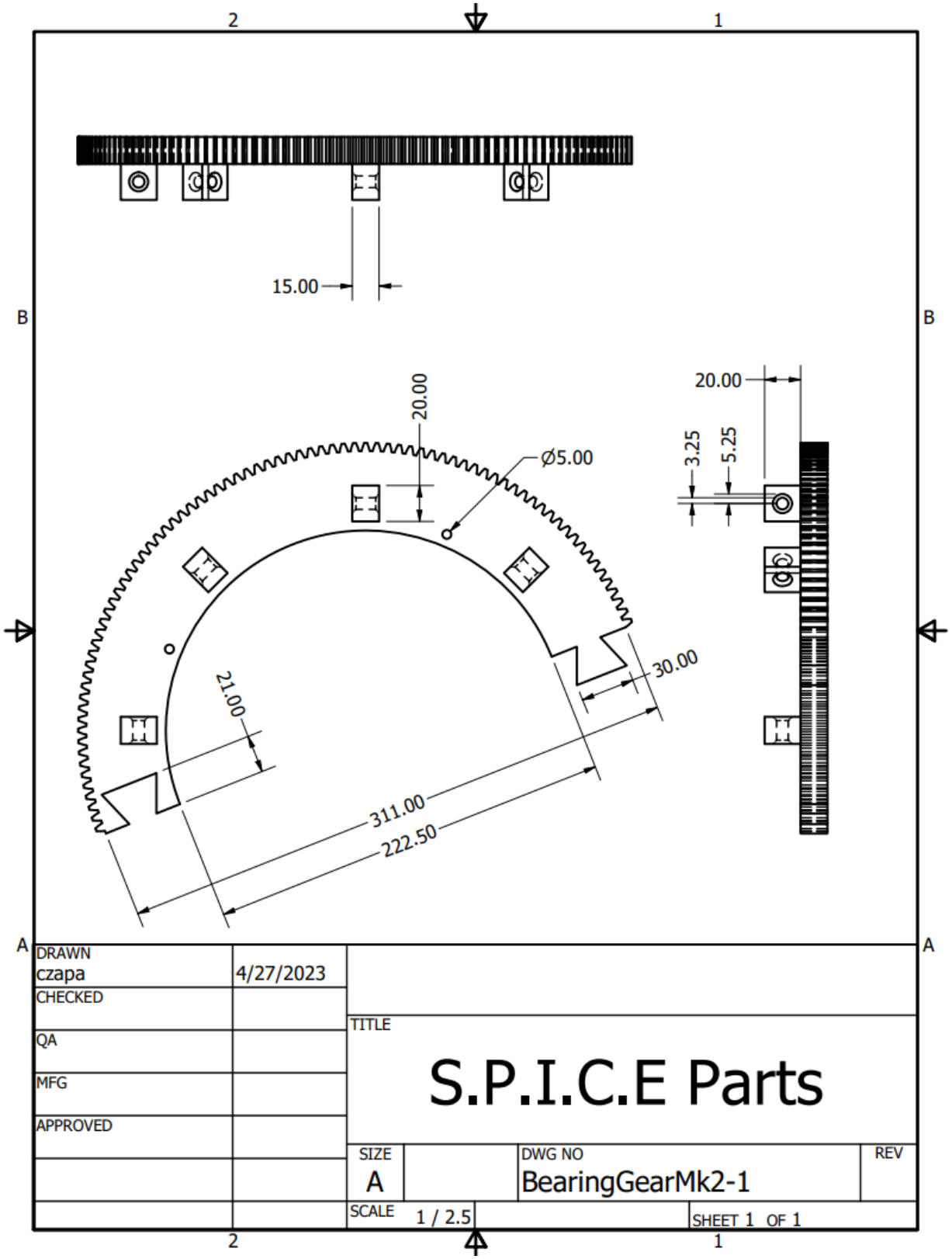

Select spices based on a recipe below.
User just selects an entry in this table, where each entry/recipe contains a list of pre-determined spices and quantities. Once the user selects an entry, they will automatically be taken to the confirmation page showing that list of spices and quantities for the recipe they selected
Recipe Name
Garlic Butter Chicken
Honey Garlic Pork
Back to landing page

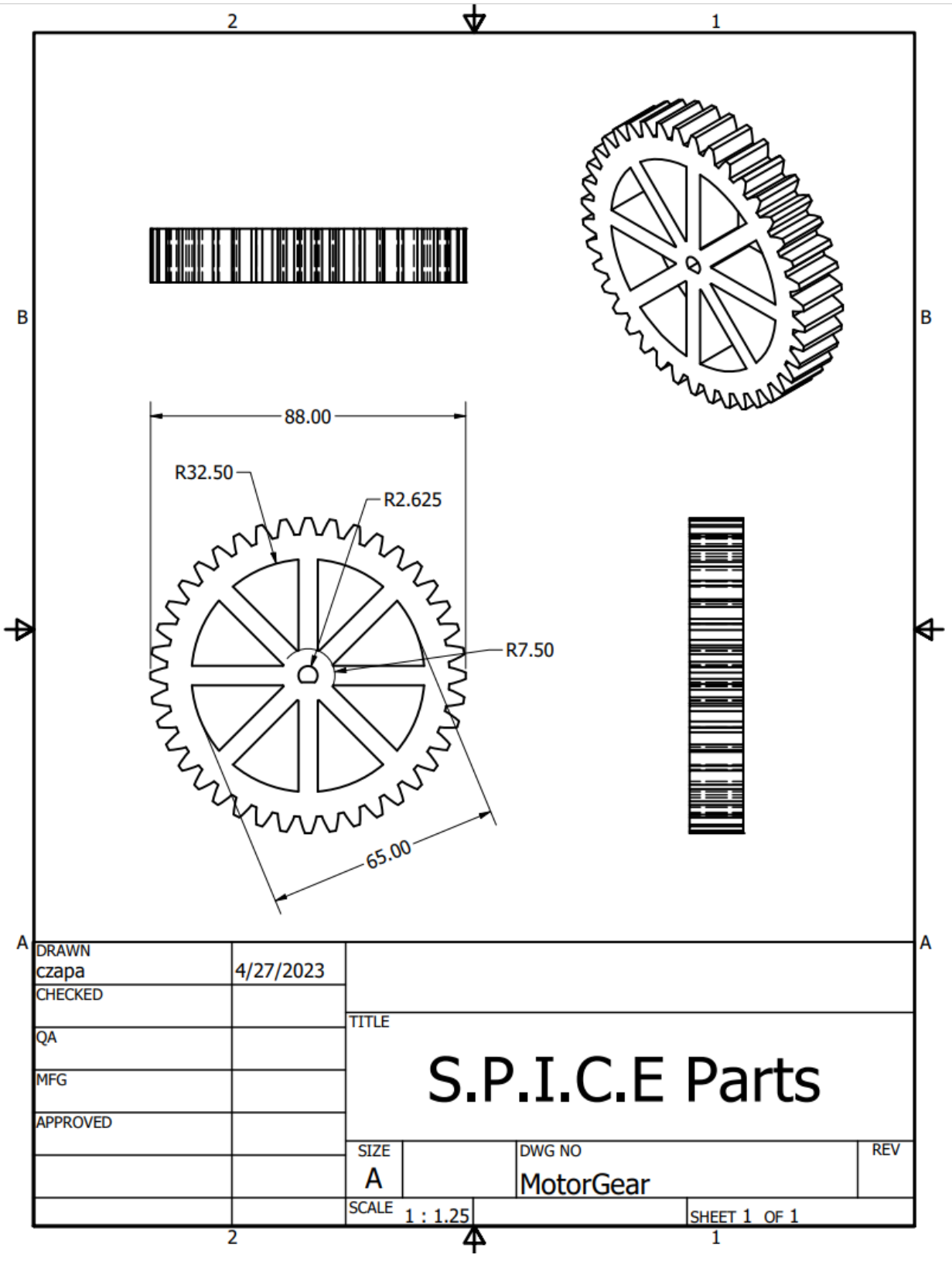

Do you want to dispense the following spices?
If the user clicks on "Yes", the UI will send the mappings to the Arduino to process as signals to determine which motors will spin with a sequence of directions. User is also taken back to landing page
Spice
Quantity
Salt
1.71
Pepper
3.07
Garlic Powder
4.01
Back to recipes page or spices page
←
Yes
No
Clears the table of its data

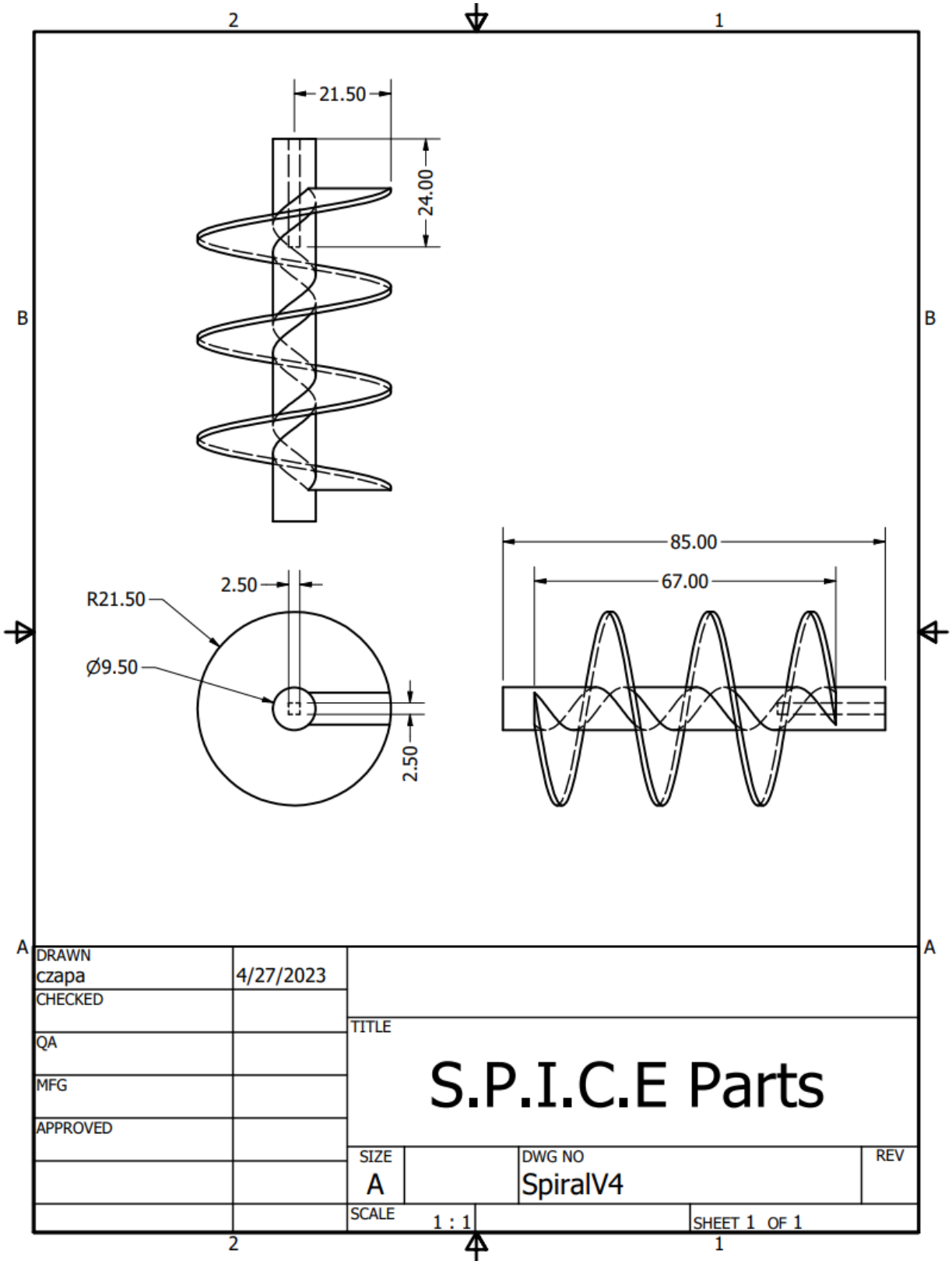
SPICE Component Drawings

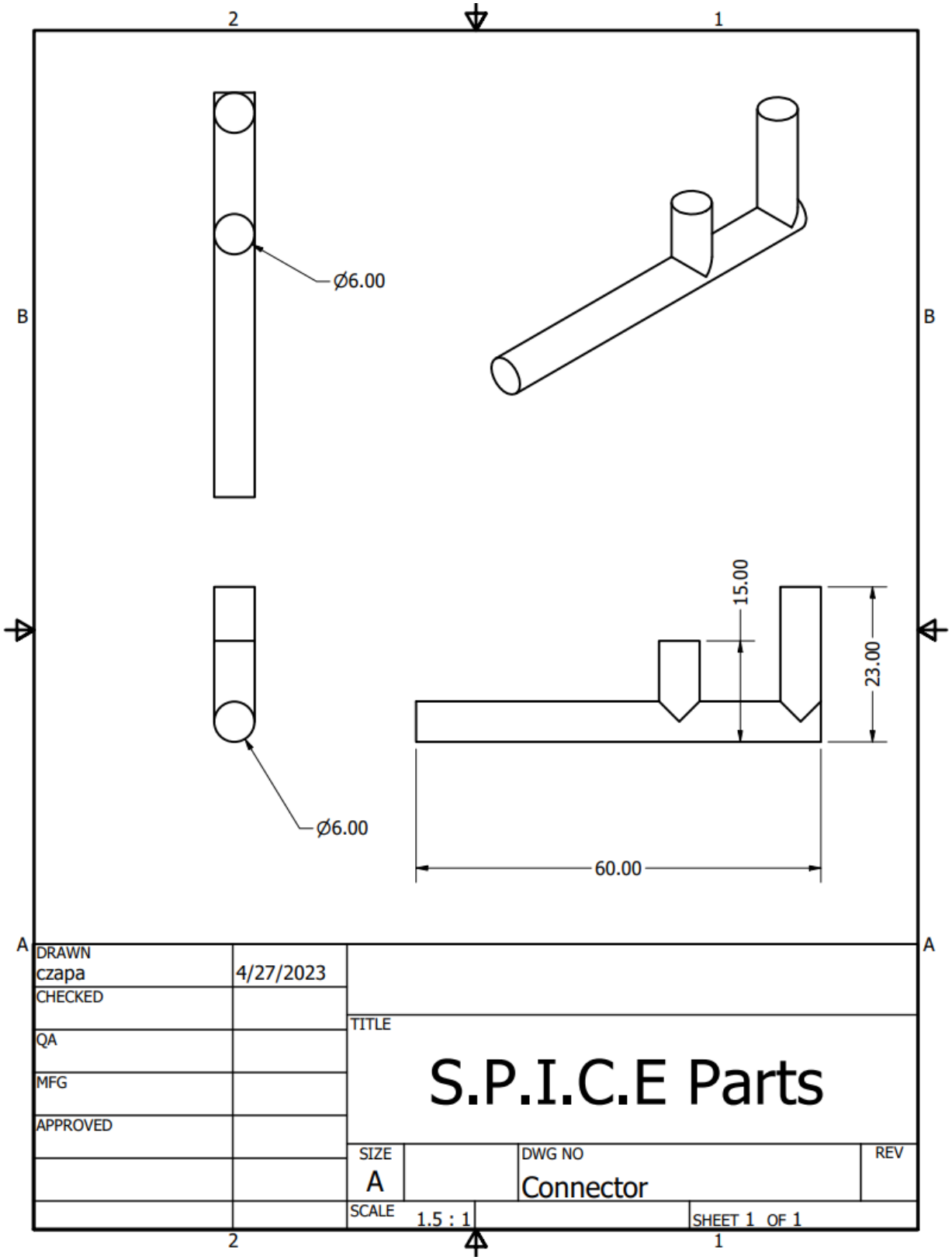



2
1
B
A
Ø56.00
Ø16.00
26.00
8.00
70.00
120.00
60.00
25.00
R5.00
R23.00
45.00
9.00
Ø16.00
24.50
5.00
8.00
DRAWN
czapa
4/27/2023
CHECKED
QA
MFG
APPROVED
TITLE
S.P.I.C.E Parts
SIZE
A
DWG NO
Housing_LidV3
REV
SCALE
1 : 1.35
SHEET 1 OF 1


2
1
B
A
15.00
20.00
Ø5.00
21.00
30.00
311.00
222.50
20.00
3.25
5.25
DRAWN
czapa
4/27/2023
CHECKED
QA
MFG
APPROVED
TITLE
S.P.I.C.E Parts
SIZE
A
DWG NO
BearingGearMk2-1
REV
SCALE
1 / 2.5
SHEET 1 OF 1


2
1
B
B
88.00
R32.50
R2.625
R7.50
65.00
A
A
DRAWN
czapa
4/27/2023
CHECKED
QA
MFG
APPROVED
TITLE
S.P.I.C.E Parts
SIZE
A
DWG NO
MotorGear
REV
SCALE
1 : 1.25
SHEET 1 OF 1
2
1


21.50
24.00
85.00
67.00
2.50
R21.50
Ø9.50
2.50
DRAWN
czapa
4/27/2023
CHECKED
QA
MFG
APPROVED
TITLE
S.P.I.C.E Parts
SIZE
A
DWG NO
SpiralV4
REV
SCALE
1 : 1
SHEET 1 OF 1



Parts:

- Raspberry Pi Touch Screen (Quantity: 1)
- 1 kg Brown PLA 1.75 mm (Quantity: 1)

- 6 Pack DC Motor (Quantity: 1)
- Fiesta Brand Spice Containers (Quantity: 8)
- Bearing Table (Quantity: 1)
- 20 kg Servo Motor (Quantity: 1)
- Stepper Motor (Quantity: 1)
- 4 Pack L298N Motor Controller (Quantity: 1)

3.3 Approach for design validation

The validation process for S.P.I.C.E. consisted of multiple stages throughout the duration of development. This method is essential for successful development in the sense that waiting until the project is "finished" to perform testing could cost time if a major issue or bug is discovered as it would be difficult to pinpoint the exact location of the bug due to there being many components to examine. This method is organized in such a way that there are multiple factors to consider when designing this product based on the objectives and constraints the team set forth for this project. Such factors include accuracy, environment, and usability/accessibility.

The first round of tests consisted of ensuring that the UI meets the team's standards and constraints, as well as testing the hardware and mechanics of the system. The UI was tested for accessibility and usability by recording the average time it would take to input data into the system as well as testing the functionality of each of the buttons and ensuring that they do what they are supposed to. The hardware testing process included tests to ensure that the base components themselves are working properly (motor, touchscreen, Raspberry Pi). This aspect is important to consider, or else there wouldn't be a working product since S.P.I.C.E. relies heavily on hardware to operate.

Durability was another factor to perform tests on. In other terms, the team performed tests on the product in the environment it is designed to operate in. These tests were based on temperature and if the system doesn't physically malfunction at a normal kitchen temperature (65-75°F). The validation process in this category consisted of observing the behavior of the motors in the system. Since one of the main aspects of S.P.I.C.E. is for it to be a household kitchen product, durability is an important factor to take into consideration when validating the design of the product.

Accuracy is another important feature to test. This required the hardware to be synchronized with the UI. Tests were performed on the system with button presses on the UI and motors. Accuracy was ensured in such a way that the correct motors would spin based on what spices the user wants to dispense. This was done through a mapping system, where each container would have its own mapping. If a user were to put a certain spice in queue for the system to dispense, the spice would basically have a pre-assigned mapping that would help the Arduino to determine which motor should be spun to dispense the proper spice. Unfortunately, the team was unable to implement conversion between teaspoons dispensed and the time it takes for motors to spin. Overall, the team was able to dispense the correct spices based on user input, but was not able to implement the backend for the measurement feature between the UI and Arduino code.

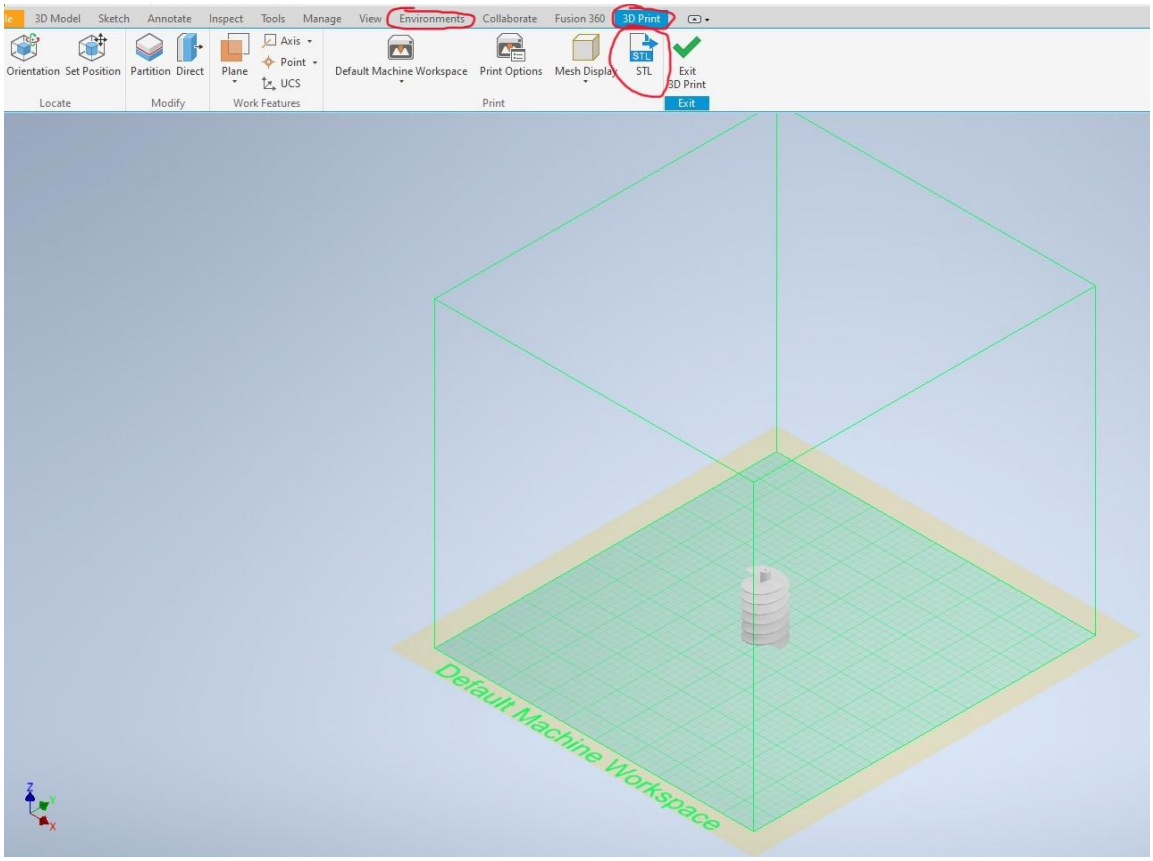
Finally, there was a round of testing with the fully integrated system. While the components of the system would be individually validated beforehand, it is still crucial to test S.P.I.C.E. to its best potential to ensure that it is still a product that satisfies the team's goals and solves the main problem identified by the team despite any hardships the team may have encountered. More specifically, S.P.I.C.E. should function in a way that it makes cooking enjoyable for people who are living alone and those with motor

disabilities. This means that it should be usable, accessible, operational, and accurate. These four main qualities will be accounted for when executing the final rounds of testing.

4 Implementation notes

3D Modeling

The housings, lids, spirals, and gears for this project were created in the following major steps: Computer-Aided Design (CAD) modeling, STL file exportation, slicing, 3D printing, and finally support layer removal and minor sanding. The CAD portion of this project was done using Autodesk Inventor and any necessary modifications can be made to the provided 3D model files on that same application. Autodesk inventor allowed the team to design individual pieces and virtually assemble the project before printing to ensure that all parts were appropriately dimensioned. The same application allowed the models to be directly exported to STL files by opening any model and choosing *Environments>3D Print>STL*, then saving to the desired location as seen below.



This STL file can then be uploaded to any slicing software to be simulated on the print bed. It's crucial to set the correct orientation and scaling at this step or else printed parts may not fit the design constraints. Once the desired orientation is set and the necessary settings have been changed in the software to match the target 3D printer, then the piece can be sliced (exported as gcode). The code can then be moved to the printer for 3D printing via SD card or over wifi if the printer is capable. This iteration of the project was printed using PLA filament due to its price point and accessibility as one of the most commonly used materials for 3D printers. However, any food-safe material can be used so long as the printer supports it. Depending on the model, material, and resolution, the print could take quite a while to print and may fail if not properly set up. For this reason, the team outsourced the printing step to

the student print shop at the Texas A&M Fischer Engineering Design Center (FEDC). The FEDC was able to produce high quality prints in a timely fashion. The final step of the printing process required removing support materials left over from the 3D printing process and carefully sanding down any areas where necessary.

During the design process of the printed components, it was vital to double check the tolerances of each part. Our team used a 2mm tolerance to account for the size of the nozzle on the 3D printers. This also gave the parts a bit of wiggle room that could then have additional material added to ensure a snug fit. For some components, mainly the ones with circular features, it was better to over-do the tolerances rather than trying to cut it close. It's important to keep in mind that not every part will print exactly the same, and thus, tolerances give the component more room for error.

Another important consideration when printing is the orientation of the model on the print bed. The orientation can have a significant impact on the quality of the print and the level of detail that can be achieved. It is essential to analyze the model and determine the best orientation that will ensure all small details are captured correctly. For example, a part that requires more precise printing, such as the lid to the housings, should have all parts that are not going to be connected to supports facing directly upwards. This ensures that support material does not attach and add an extra layer to that part of the print.

Checking the fit of parts before printing is critical, as it can save a considerable amount of time and material. Before printing a set of components that will fit together, it is recommended to create a virtual model to check how the parts fit together. This can be done using the 3D modeling software of choice. This step can help identify any issues with the design and make necessary adjustments before printing. Reducing the overall number of prints not only saves money, but a lot of time and headache. For this reason, double-checking the measurement accuracy is essential. It is recommended to verify the dimensions of parts that will need to fit into 3D printed components using a digital caliper or a measuring tool to ensure they match the intended design. This step can help avoid errors and ensure that the printed parts are accurate and fit together correctly.

Fabrication of Base

The base can be constructed from any material, however there are a few things to take into consideration. First and most importantly, the bearing table. Make sure that regardless of the material the base can house the bearing table with a little bit of extra room for the driving motor. If the driving gear (in our case the one attached to the stepper motor) is pressed too close to the bearing gear, it will cause a great deal of friction. This can increase the necessary operating power of the drive motor or make the bearing table turn unreliably. To address this, the base was designed with sufficient room for the motor to be adjusted. The second thing to take into account is the screen. Make sure that the base has enough clearance for the screen to be clearly visible and accessible. If the housings overhand too far over the screen, it may make the UI more inconvenient to access. Lastly, make sure that the material used is heavy enough to hold down the platform while the bearing table is moving. If the base is too light, the driving motor will move the entire platform rather than just the rotating table.

User Interface Implementation

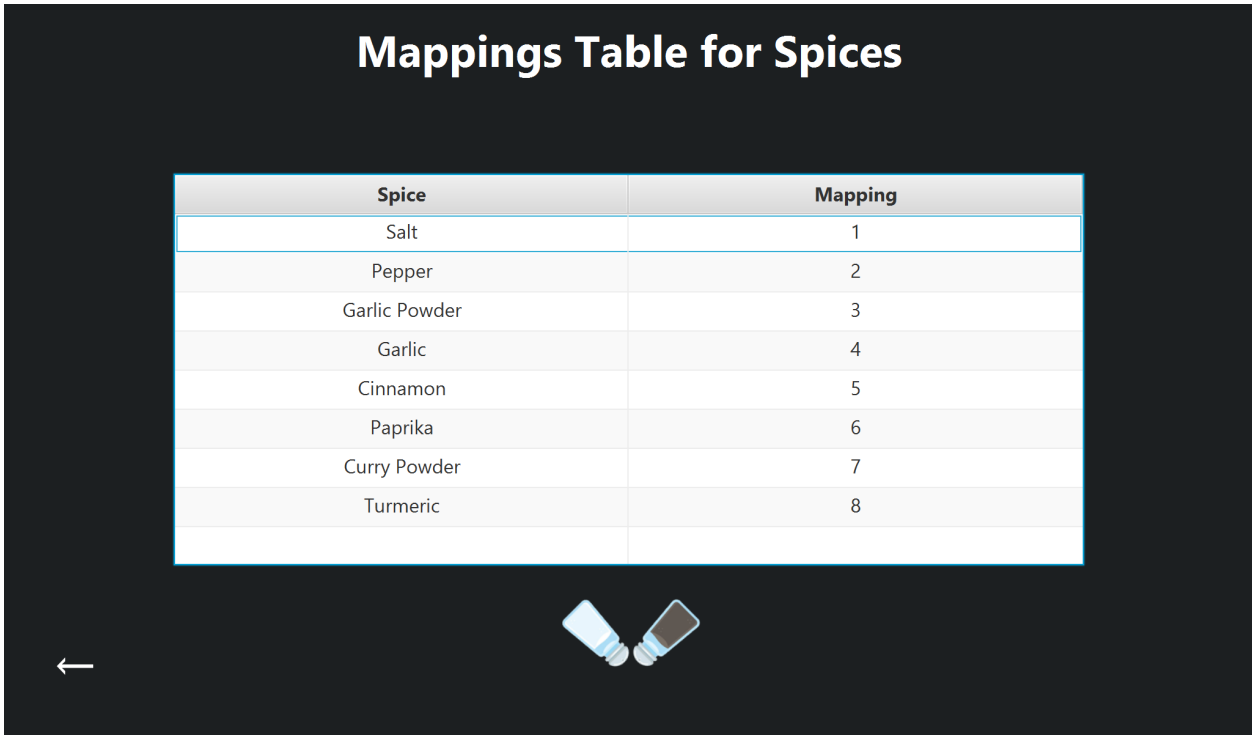
A successful implementation of this project hinges on the torque available from all of the motors. During testing, the hobbyist motors turned out to be far too weak to reliably dispense spices. We also had an issue spinning the stepper motor which was one of the most important components for this project to work. Initially the design was just going to use 9V batteries, but the team found out that the stepper required much more power to spin properly and on top of that to power the rest of the system we needed a bigger power source. For demonstration purposes we were able to hook up an adjustable power supply to our system to at least see that the motors, with enough power, could actually do their jobs. After calibration, the team found out that providing around 2A of current and 12V was enough to power the whole system reliably. A consequence of not being able to dispense spices reliably was not being able to implement a conversion system between teaspoons of spice to how much time to run a motor for. Initially, our idea was to run multiple tests for dispensing spices and see the rate at which spices can be dispensed. This rate was going to be used on the backend to reliably dispense accurate amounts of spices based on how much the user wanted. We also didn't take into consideration that different spices will have different rates which made it a lot harder to generalize the system because users theoretically get to choose which spices they want in their product.

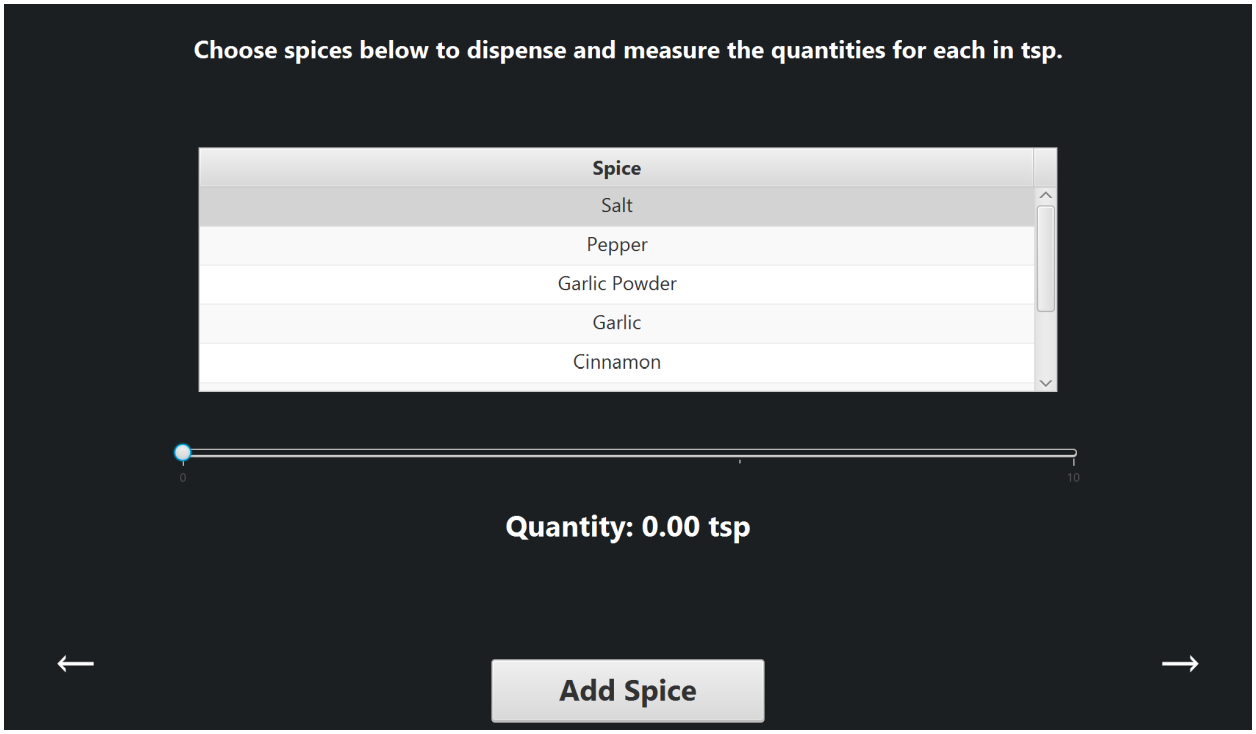
The user interface (UI) for this project is designed for a 1024x600 touchscreen, and uses JavaFX and a serial communication library (jSerialComm) to handle communications between Java and an Arduino. Since the UI was designed to work on a touchscreen and to be easy to use, the team made sure that there would not be any typing involved with a keyboard to input data into the system. One particular issue the team encountered when using the touchscreen was that it would sometimes lose its touch functionality, but restarting the Pi easily fixes that issue. Another minor issue came up whenever we tried to change the resolution of the touch screen to portrait instead of landscape. The screen would change orientations, however, the touch detection would stay on landscape. This distorted the touch functionality on the screen beyond usability. The solution was to simply update the UI to be presentable in landscape rather than portrait. The team used the Liberica Java Development Kit (JDK) to run the UI on the Raspberry Pi. In order to set this JDK up on the Pi and to get the appropriate JavaFX .jar files, the following commands below were executed in the command prompt:

```
$ cd /home/pi
$ wget https://download.bell-sw.com/java/13/bellsoft-jdk13-linux-arm32-vfp-hflt.deb
$ sudo apt-get install ./bellsoft-jdk13-linux-arm32-vfp-hflt.deb
$ sudo update-alternatives --config javac
$ sudo update-alternatives --config java
```

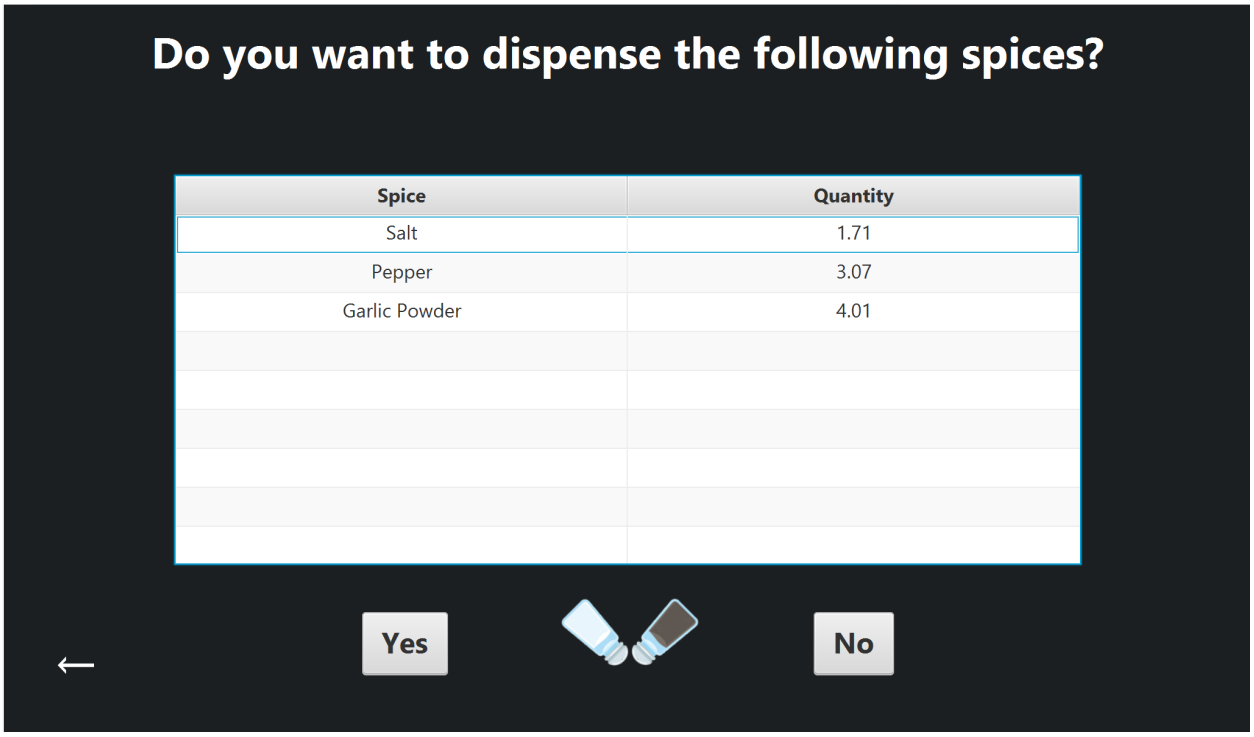
The version can also be checked to ensure that Java and JavaFX were installed correctly by running the command "*java -version"*. After the JDK is installed, the application can be compiled with the .jar files and then run with "*java <name of file>*". The final UI design for S.P.I.C.E. is shown below and should look like this when being used on the 1024x600 touchscreen:


Welcome to S.P.I.C.E.
Select an option below to proceed.
Spices
Select spices of your choice.
Recipes
Select spices based off a recipe.
Mappings
View mappings for each spice.
X


Mappings Table for Spices
Spice
Mapping
Salt
1
Pepper
2
Garlic Powder
3
Garlic
4
Cinnamon
5
Paprika
6
Curry Powder
7
Turmeric
8
←

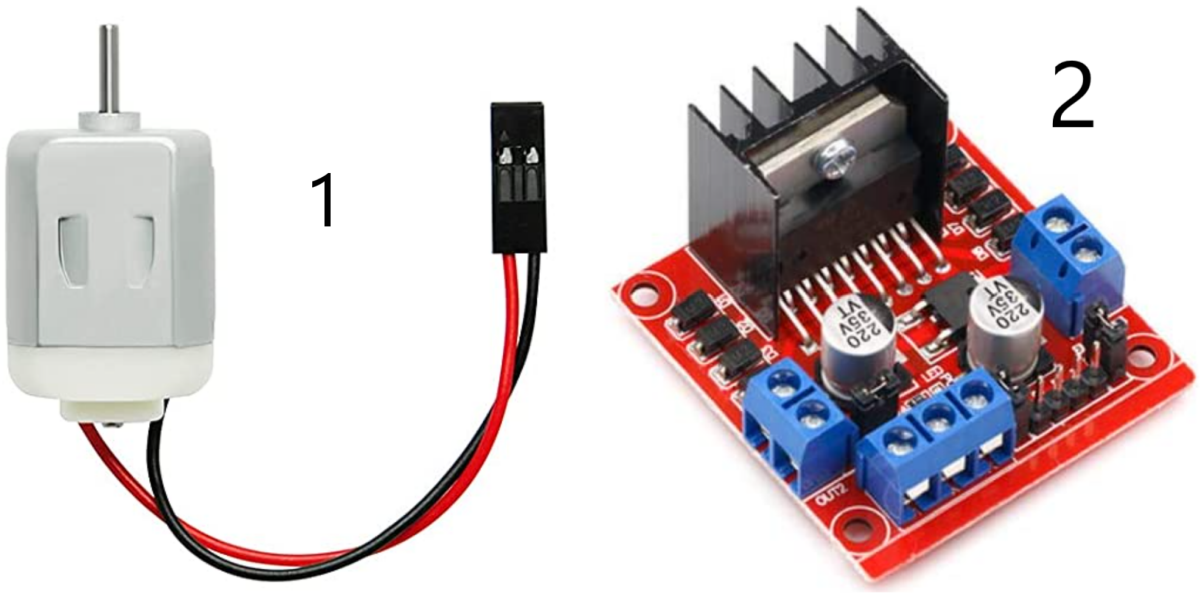

Choose spices below to dispense and measure the quantities for each in tsp.
Spice
Salt
Pepper
Garlic Powder
Garlic
Cinnamon
0
10
Quantity: 0.00 tsp
Add Spice


Select spices based on a recipe below.
Recipe Name
Garlic Butter Chicken
Honey Garlic Pork


Do you want to dispense the following spices?
Spice
Quantity
Salt
1.71
Pepper
3.07
Garlic Powder
4.01
Yes
No

Embedded Systems

The embedded system for this project was designed using four main components. These were: a Raspberry Pi, an Arduino, DC motors, and a stepper motor. The Raspberry Pi's responsibility was to act as the brain for the whole system by providing data to the Arduino whenever a user wanted to dispense spices. The Arduino did all of the motor control throughout the system. It spun specific motors based on the data sent by the Pi. For the actual wiring of the system, originally we had planned to utilize multiple motor controllers for all the motors used in the project. However, this idea did not take into consideration how much space will be needed. To reduce the amount of space required, the team decided to use one motor controller for the stepper motor and another motor controller for all the DC motors. It was found out that one motor controller can control a total of 4 motors at a time if they all share the same ground pin instead of the recommended 2 motors per controller. For the software side of the system, the main component was a serial communication channel between the Pi and the Arduino. This allowed both devices to send data to each other which provided insight on which functionality to activate based on what the user wanted. Simple messages were used to make it easy to detect which functionality to run. The serial communication aspect was integrated into the UI to allow the use of button events to make our product work. However, a "debug" mode was also implemented which allowed us to directly send messages into the serial channel without the need of buttons. This allowed us to isolate specific problems and made testing easier. In order to run the system, one needs to make sure that the Arduino and the Pi are both running their respective code bases. Due to some difficulty running our serial channel code on the Pi, a laptop was used as a substitute. The DC motors, motor controllers, and stepper motor that were used in this project are shown in figure 10 below:



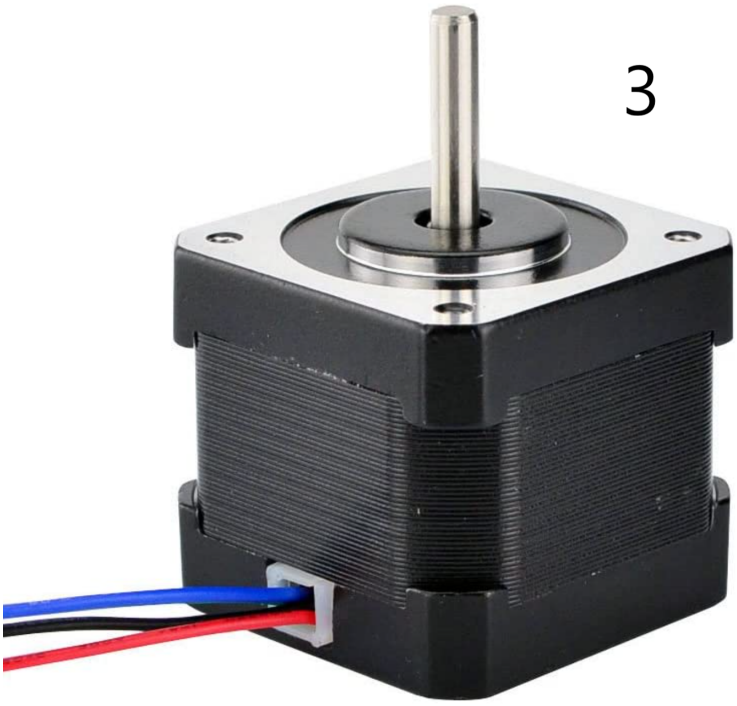


Figure 10: DC Motor (1), Motor Controller (2), Stepper Motor (3)

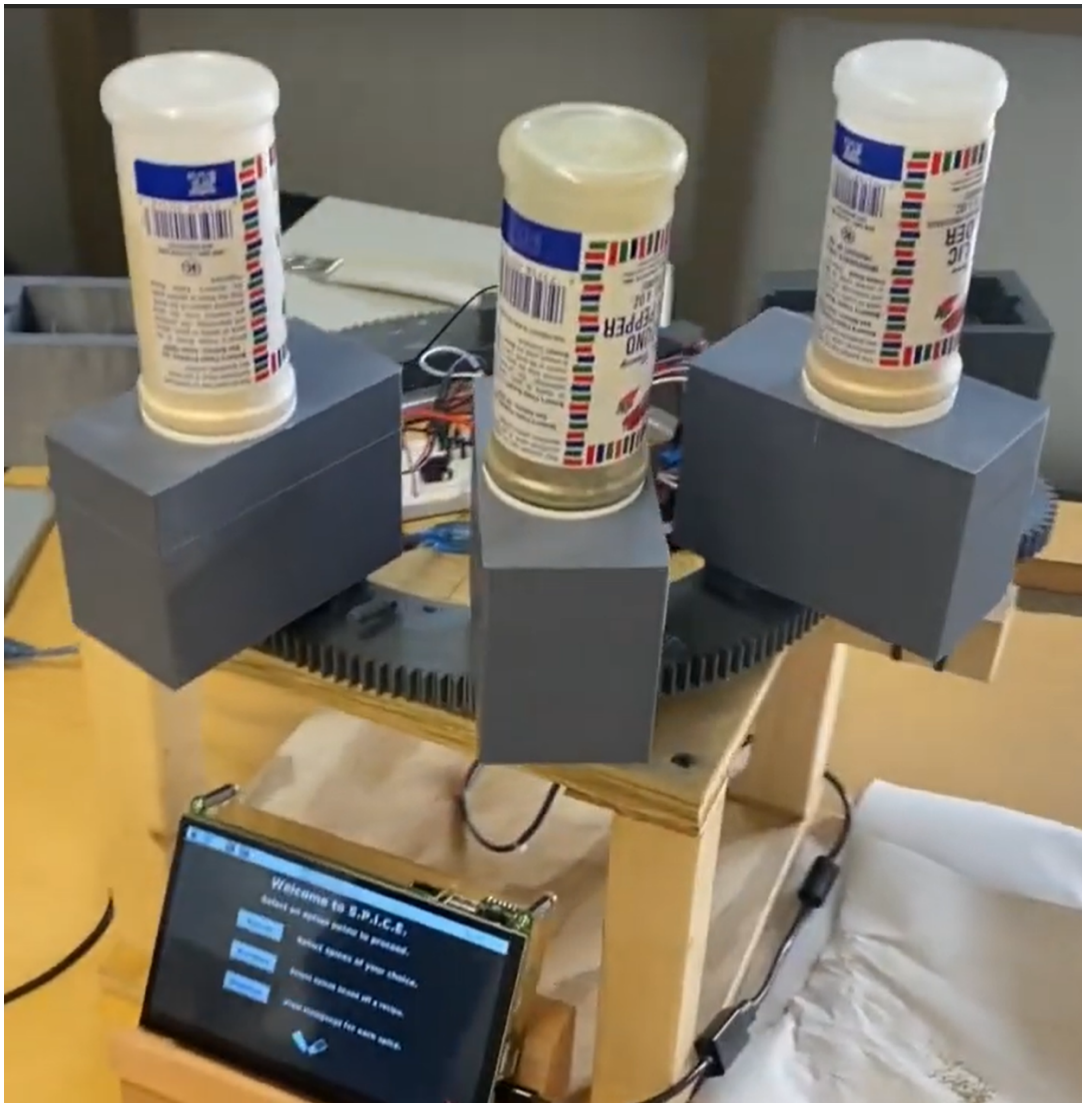
5 Experimental results

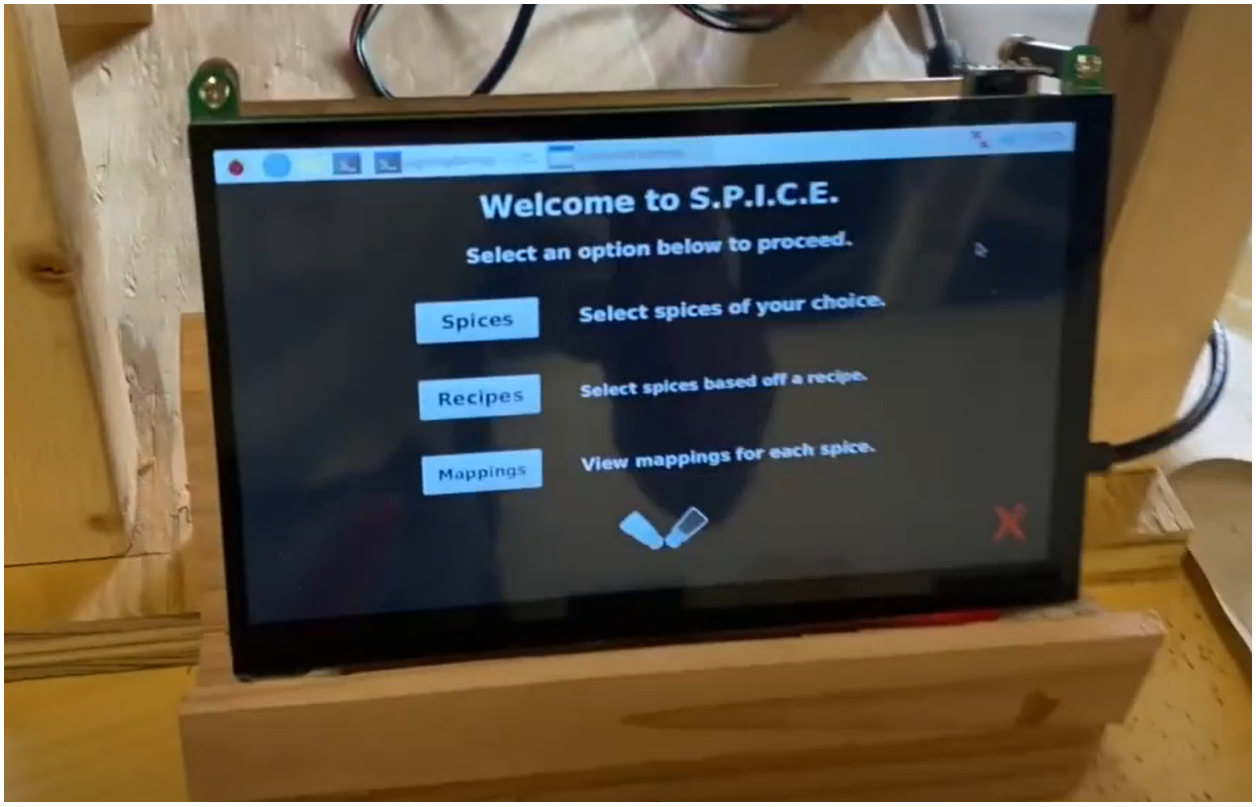
The team performed a variety of different tests to validate S.P.I.C.E.'s functionality as much as possible despite some difficulties the team encountered. The UI was tested in conjunction with the touchscreen since the team intended for the UI to be displayed there. Firstly, the team made sure that the UI was able to be displayed on the touchscreen, and once that was validated, buttons were then tested to make sure that they would be responsive to the user tapping on the screen. Appearance was also taken into consideration, as the team had to make some adjustments with the resolution in the Java code to make the UI compatible with the touchscreen being used for the project. The dispensing mechanism was also tested for each housing. The team encountered some difficulties with some of the spirals not spinning, and that was due to the lid of the housing blocking part of the spiral, which prevented it from spinning and therefore dispensing properly. Therefore, adjustments were made for some of the lids of the housings to resolve this issue. Afterwards, the team tested the DC motors with the covered housings to make sure they would spin, with or without spices inside. Finally, the team tested the serial communications between the Arduino and Java with a laptop. This test was accomplished through user input into a terminal, where there would be specific keywords set on the Arduino code for the team to be able to interact with from the Java code. More specifically, the team observed that the motors were doing what they were supposed to based on a keyword that would be typed into the terminal ("left" or "right" for example), where code would be written on the Arduino side to program how the keywords would operate the motors. The

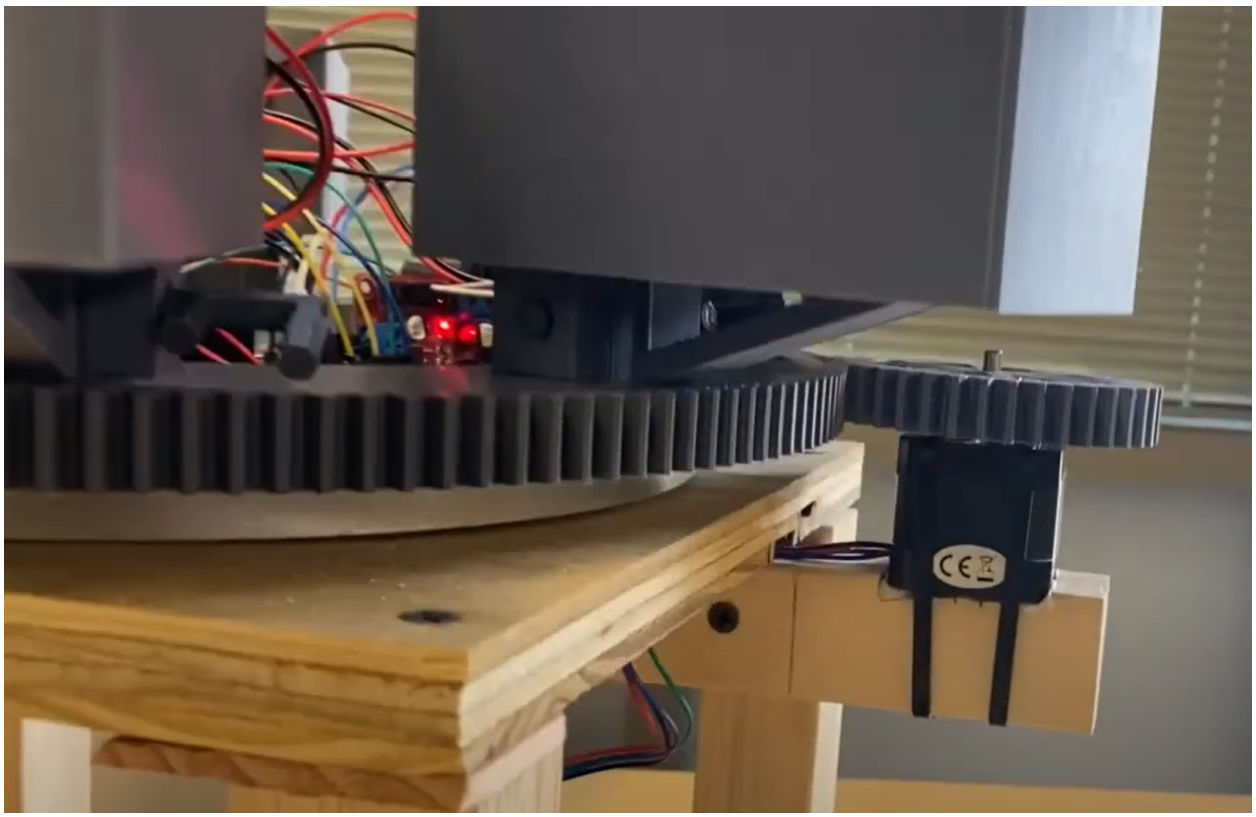
keywords were taken in as strings, and some main keywords that were used were “left”, “right”, “1”, “2”, “3”, “4”, where the numbers would represent containers. If a number was typed into the terminal, a specific spiral for that respective container would spin. The team also discovered that the stepper motor required a certain amount of voltage to be able to spin the main gear that was holding the housings, which turned out to be 12V. Any voltage less than that number would not be enough for the motor to spin the entire gear.

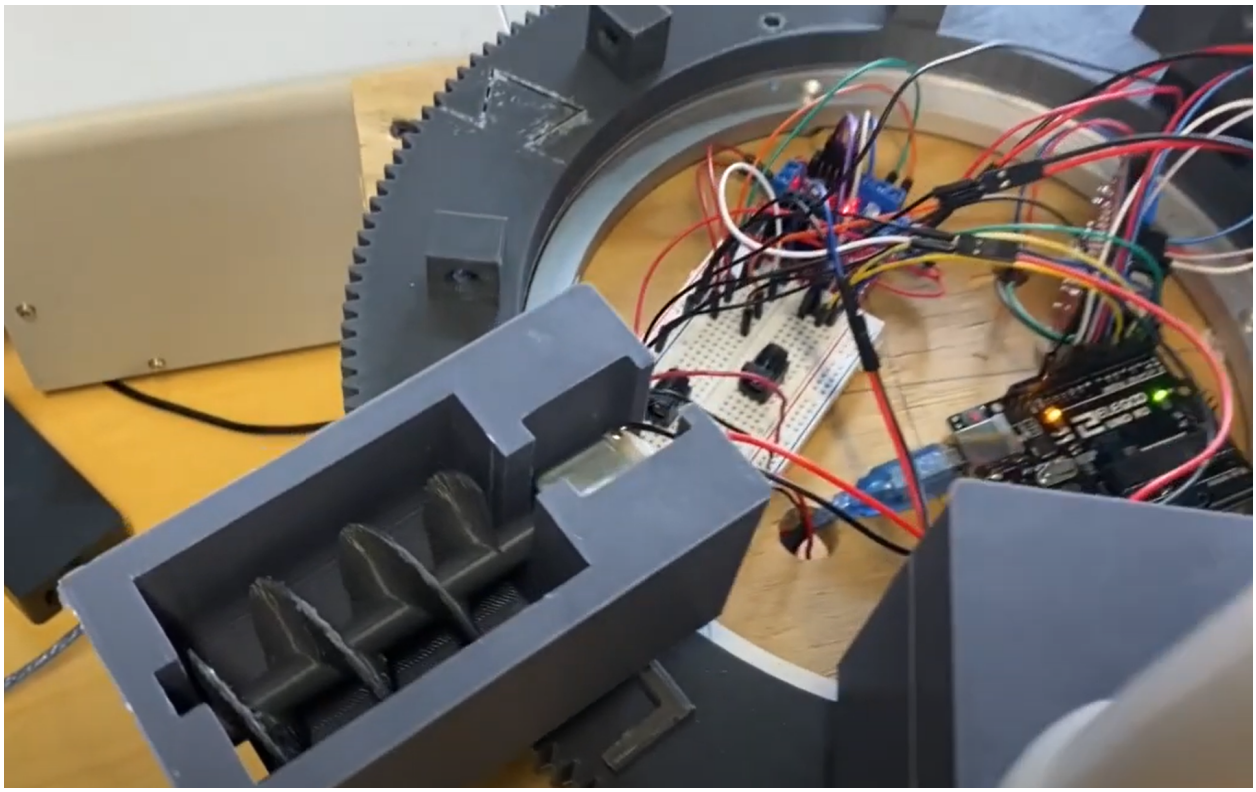
Overall, S.P.I.C.E. required more rigorous testing than the team expected. The dimensions of the UI had to be adjusted accordingly for the Raspberry Pi, since some of the text with the labels and buttons were being cut off the page for the Pi but not on the laptop. Friction was also an issue with the housings that the team did not consider earlier, as putting too much spice into a housing would be too much for the spiral and DC motor to handle. Therefore, smaller quantities had to be stored inside the housings for the motors to be able to spin and dispense accordingly. The stepper motor also required a considerable amount of voltage to be able to operate with the main gear. This wasn’t apparent until demo week, and so the team had to use an adjustable power supply to supply the required amount of voltage (12V) to get the motor to spin successfully. There were also issues with running the serial communication library on the Raspberry Pi, and so a laptop was used to circumnavigate this problem, where the library was successfully able to be used in conjunction with the UI buttons. In summary, there had to be several last-minute changes and fixes to the system.

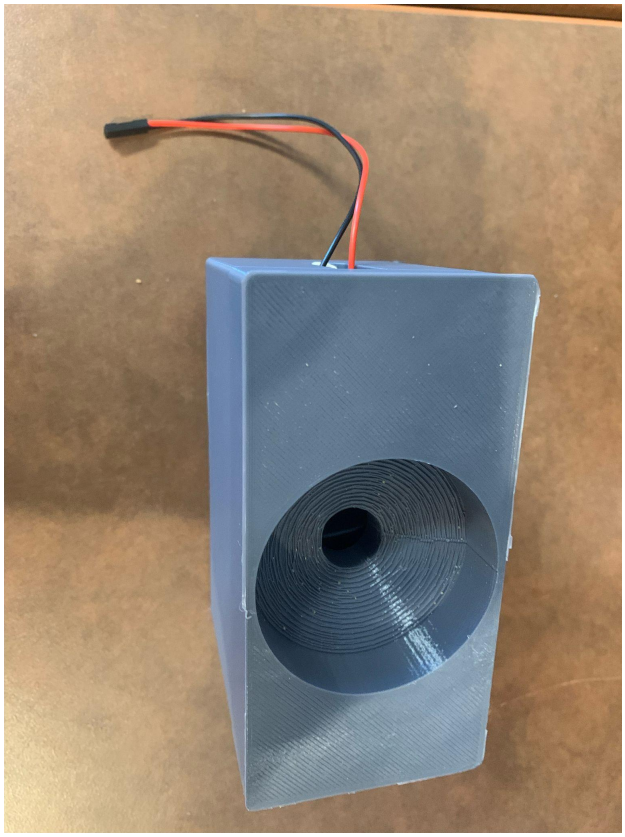
Pictures

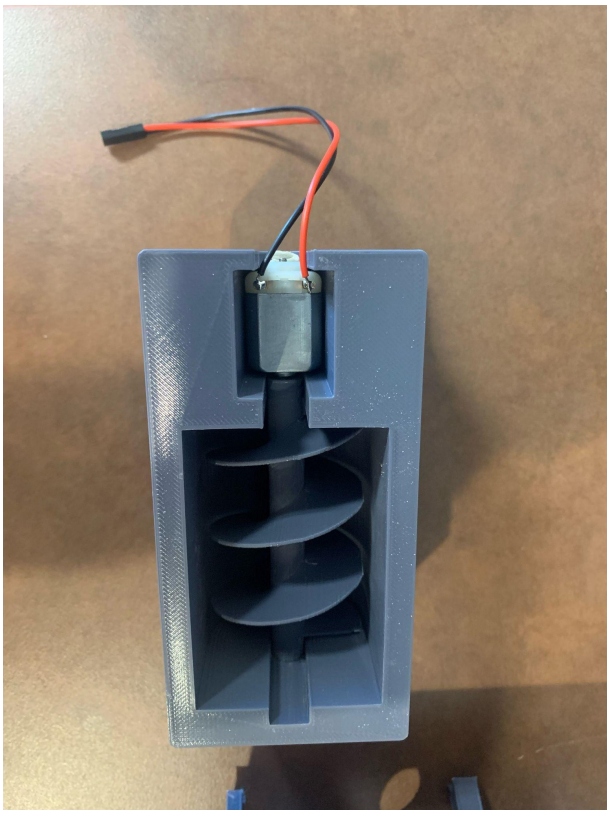



Welcome to S.P.I.C.E.
Select an option below to proceed.
Spices
Select spices of your choice.
Recipes
Select spices based off a recipe.
Mappings
View mappings for each spice.







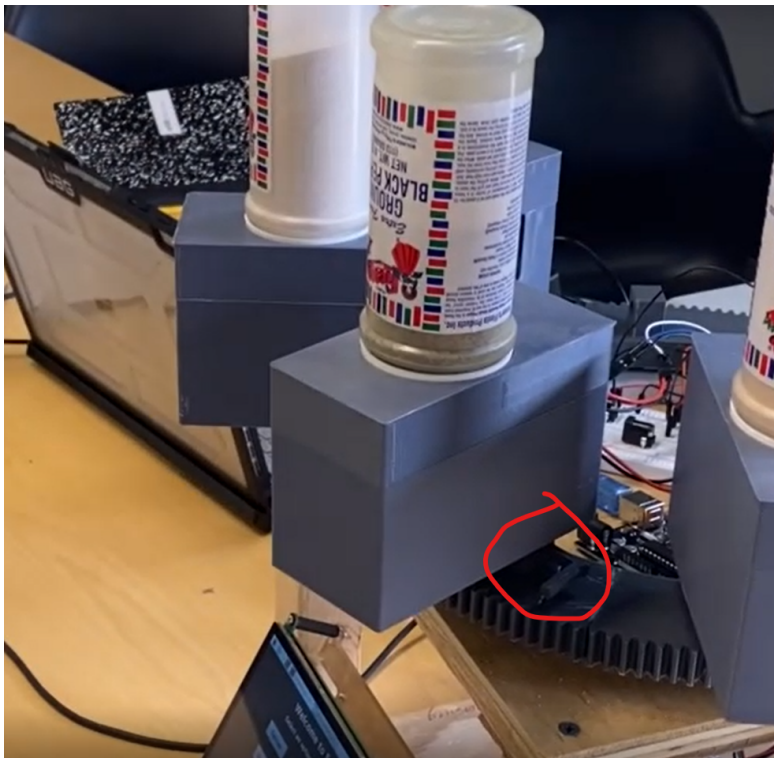


6 User's Manuals

- Modularity (Taking parts off and on)
    - The general spice housings are composed of only a few parts: The housing, the lid, and the connector.
    - To attach a housing to main S.P.I.C.E. platform, first gather each of the parts shown below.

- Set the housing in place with any open slot on the S.P.I.C.E. platform

- Slide the F-shaped connector into the slot, and the housing is secure!



    - Housings
        - Lids to cover each housing; spirals inside each housing as well to dispense spices, with each being connected to a DC motor
        - F-shaped connectors to install each housing to the system; insert through circular hole at bottom of housings when securing them to the main gear
    - Base gear
        - Attached to bearing table; can be taken off or put back on; easy to install (match the legs of the bearing table with the holes carved in the wood base)
    - Stepper motor gear
        - Can just slide the gear off and slide it back on
        - Stepper motor fastened with zip-ties
- Hardware
    - Power supply for Raspberry Pi
    - Required amount of voltage for stepper motor to spin the main gear (?)
        - A separate power supply was used in this case
- Operation Instructions
    - Make sure Pi is plugged into the wall with power supply
    - Check that the Arduino is connected to whatever is going to be running the UI (Raspberry Pi, laptop, etc.)
    - Instructions on how to navigate through UI, what each page and button does
    - Configuring housings, populating them with spices

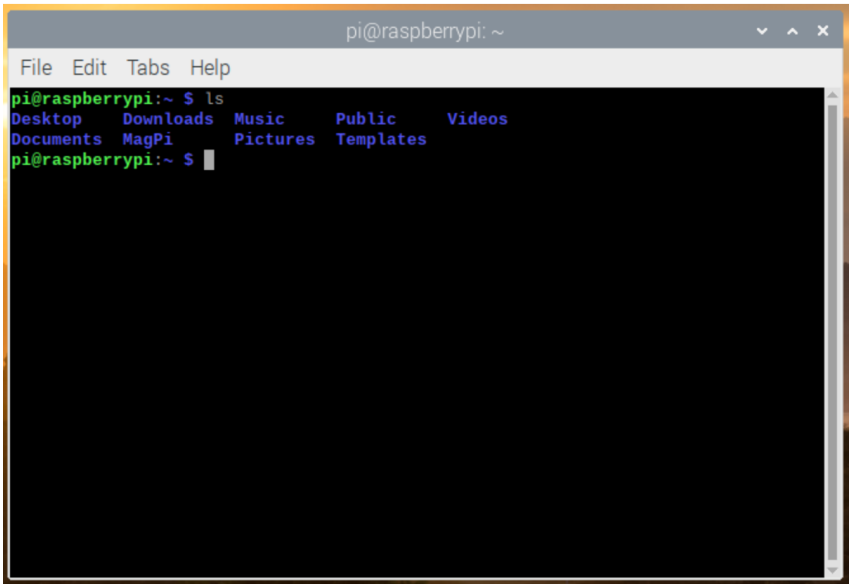
Hardware Installation

Hardware wise, S.P.I.C.E requires the use of a Raspberry Pi, an Arduino, and a series of motors along with motor controllers. A complete hardware schematic was previously shown in section 3.2 figure 8 of this report. Essentially, the Pi is only connected to the Arduino through power (5V) and ground pins. To control the motors, motor controllers are hooked up to the motors themselves. The motor controllers are powered by a 12V supply and grounded to the same ground rail as the Pi and Arduino. Specifically, for the stepper motor, all four input and output pins are used. The inputs are connected to the PWM pins on the Arduino which allows the Arduino to send signals to the motor controller which in turn controls the motor. The actual motor pins are connected to the output pins of the motor controller. Similarly, for the DC motors a 12V supply is provided to another motor controller. The controller is grounded to the same rail as everything else. Another four input pins are connected between the Arduino and motor controller where each connection corresponds to one DC motor. The output pins of the motor controller are connected to the power side of the motors. The motors are then grounded to the same rail as everything else. For prototyping purposes only four motors were hooked up. However, our device is capable of holding up to 8 spices. To account for more spices a third motor controller will have to be used and be hooked up in the same way the current four are. Due to our initial power source (9V) being too little, an adjustable power supply was used instead in the final iteration of the project. However, for the average user we would have liked to have the ability to simply use a wall outlet. Unfortunately, due to time constraints this issue wasn't looked much into.

Software Installation

S.P.I.C.E. requires certain software to be installed for running the user interface on the Raspberry Pi touch screen. The Liberica Java Development Kit (JDK) should be installed to the Pi so that the UI is able to be interacted with and run. Detailed instructions for this installation are the following:

1. Open up a command prompt on the Raspberry Pi. This can be found on the main toolbar and should look like this:



2. Navigate to the main pi directory by typing the following command: cd /home/pi
3. Run the following command next to download the appropriate packages: wget https://download.bell-sw.com/java/13/bellsoft-jdk13-linux-arm32-vfp-hflt.deb
4. Next, install the JDK to the system with this command: sudo apt-get install ./bellsoft-jdk13-linux-arm32-vfp-hflt.deb
5. After the installation is finished, run the following commands next:
    a. sudo update-alternatives --config javac
    b. sudo update-alternatives --config java
6. Check the version of java with this command: java -version

After Liberica JDK is installed, the .jar files can be compiled with the command *javac <all .jar files with commas separating each> *.java*. The source code can also be found at https://github.com/jbartsch23/S.P.I.C.E. The repository can be cloned on the Raspberry Pi with *git clone https://github.com/jbartsch23/S.P.I.C.E.git*. After the repository has been cloned, navigate to the src folder which contains all the source code for this project. Once inside, run the compile command mentioned above and then run the landing page file with *java LandingPage*. The UI should be ready to run afterwards.

The jSerialComm library is also an essential library for this project as it requires necessary communications between the system running the UI and the Arduino, which processes signals for the motors. The library can be found here: https://fazecast.github.io/jSerialComm/. Similarly to compiling the FX .jar files, the .jar file for this library can be compiled alongside the FX files.

If the Raspberry Pi is not working properly, the UI and jSerialComm library can also be run on a laptop with a USB port. Java FX can be found here: https://openjfx.io/. NOTE: Make sure the version of Java FX is compatible with the version being used in the source code, which is 19.0.2.1. Follow the directions below to set up Java FX and jSerialComm on a laptop with VS Code:

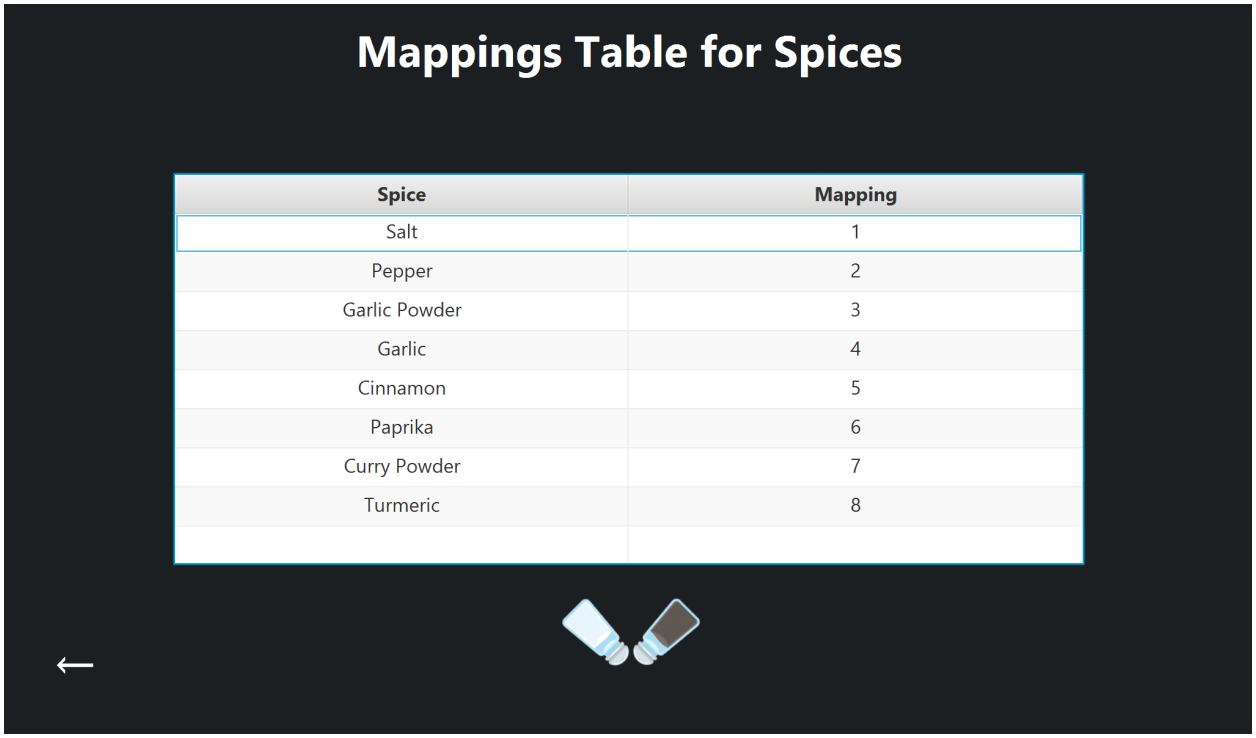
1. Install version 19.0.2.1 of Java FX and create a Java directory under the local :C drive in the Program Files folder. That directory is where the files should be installed.
2. Open VS Code and clone the Github repository with the same directions mentioned above.

3. After cloning the repository, under the "Java Projects" tab in the Explorer toolbar, add the FX and jSerialComm .jar files to the referenced libraries section.
4. The UI should be able to run afterwards with the serial communication library.

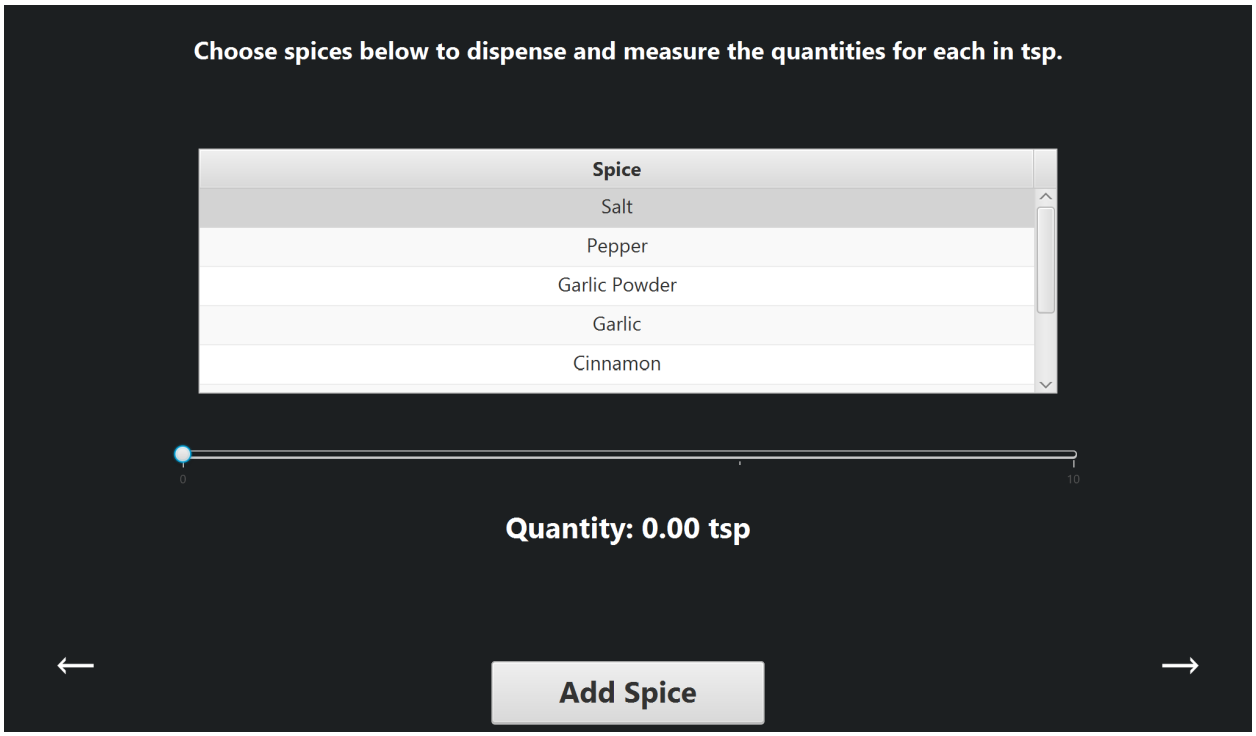
Operation Instructions

Assuming that the entire system is intact (hardware is connected properly, software is installed), operating the system is fairly straightforward. The human interactions with the touch screen are completely based on the operation of S.P.I.C.E., as each page of the UI has different functionalities as shown below for the user's reference:

This page is what the user should initially start with. There are three different buttons that the user can press to navigate to different pages. The "Spices" button will take the user to the spices page, where they can select custom spices and assign quantities to them. The "Recipes" button will take the user to the recipes page, where the user can select a recipe that contains a predetermined list of spices and quantities. Finally, the "Mappings" button will show the user which spice should be assigned to which housing. The "X" at the bottom right of the screen closes the application.



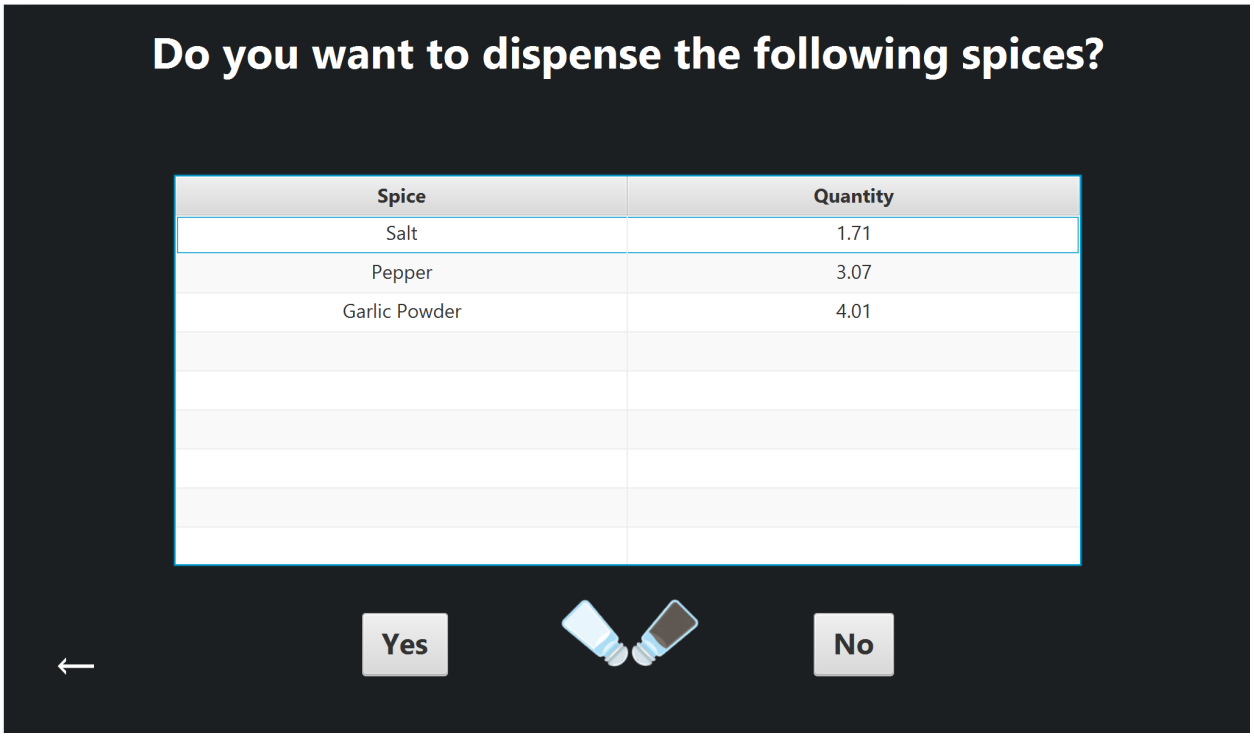
The mappings page is a simple page that lets the user view the container assignments for spices. If the user wants to go back, they can press the left-arrow button at the bottom left of the screen.



The spices page lets a user select their own spices and quantities to dispense. Firstly, the user would select a spice from the table. Once that spice is selected, the user can use the slider to measure the quantity they want to dispense of that spice. Afterwards, the user can press the "Add Spice" button, which will keep track of what the user has selected. The user can click on the right-arrow button to view what they have

selected once they are finished with the selection process on this page. This list will be on the confirmation page shown below.

The recipes page just lets a user select a recipe, which contains a list of spices and quantities. This list will be shown on the confirmation page below, depending on what the user has selected.



The confirmation page lets the user see what spices they will be dispensing, along with their quantities. If the user is satisfied with their selection, they can press the “Yes” button to start dispensing their spices. Otherwise, they can press “No”, which will clear the table of its entries.

7 Course debriefing

One management issue that arose during the implementation of our project was our individual programming environments were not compatible with each other. Initially when the team assigned specialized jobs to everyone, we all isolated our parts of the project without much thought about environment compatibility. Initially this didn’t pose a problem, but when it came to combining all of our individual components together problems started to arise. The Raspberry Pi was unable to download the necessary packages to make the serial communication work, for testing purposes, Kile was unable to get the UI working on his personal computer, and initially JP was unable to run the Arduino code on his laptop, for testing purposes, but this problem was later solved. Fortunately, we were able to get all of the components working together on JP’s personal computer, but the Pi touch screen was only partially working. If we were to do the project again we would probably put more effort into ensuring that all of us working on the software side had similar coding environments to avoid these kinds of issues.

There are some glaring safety issues with our final iteration of the project. The one that stands out the most is all of our circuitry is exposed at the top of our wooden frame. Throughout production this made it easy to access whenever it came to testing and debugging. However, we ran out of time to come up with a solution for covering it all. One proposed idea was to simply put a dome shaped object on the top of our frame to cover it all. Another issue was our power source. For the sake of proof of concept, 9V batteries were used, however, if this product were to ever be commercially available, we would need a way to allow our product to go into wall outlets which can provide the necessary power for our system. For demonstration purposes, we had to use an adjustable power supply and through trial and error found that to reliably power our system we needed about 2A and 12V. Another safety concern was the plastic we used in all of our 3D modeled components. Even though PLA has been deemed safe for food contact, one can never be too careful. The team did a lot of sanding and stripping of particular components which could have exposed them to microplastics. To address this issue, the team would need a bigger budget to select a different material to use for our components that would be safer for food contact.

In terms of testing and validation, the team originally planned to have ample time towards the end of the project's life to do thorough testing to make sure all of our features and functionality are working properly. However, due to all the setbacks and hurdles throughout the design and implementation, the team had a lot less time for this stage of the project. We were able to do some light testing for various functions and features. The backend code provided a “debug” mode which allowed the team to manually send messages through the serial communication through a terminal instead of interacting with the UI. This helped isolate problems related to serial communication. Unit tests consisted of running the stepper motor left or right to rotate our bearing table, running a single DC motor to dispense a single spice, and dispensing multiple spices by running multiple DC motors. In order to make these tests possible, a better power source was required because our initial power source didn’t provide sufficient power. The team was successful in spinning the bearing table left and right. However, one issue we saw was if the table spun one way for too many iterations the wires got twisted and tangled because of the way the hardware was laid out. When it came to testing the dispensing functionality, we had basic tests pass where we can dispense an unmeasured amount of a single spice and an unmeasured amount of multiple spices. However, this functionality wasn’t complete since we weren’t monitoring how much spice was actually being dispensed. When it came to testing the housing motors, one of the housing was unable to run if the lid was on it. The lid was most likely blocking a part of the motor shaft which prevented it from spinning.

With the little time we had left, the team feels like we tested as much as possible and are glad that some basic tests ended up working. If we were to do the project again we would like to do more rigorous testing but in order for that to happen we would have to have a more complete project.

8 Budgets

Item	Quantity	Cost Per Item	Total Cost
Raspberry Pi Touch Screen	1	$63.99	$63.99
Brown PLA 1.75mm (1kg)	2	$19.99	$39.98
4 Pack L298N Motor Controller	1	$11.59	$11.59
Bearing Table	1	$20.82	$20.82
20 kg Servo Motor	1	$12.99	$12.99
6 Pack DC Motor	1	$8.99	$8.99
Stepper Motor	1	$12.99	$12.99
3D Printing Services ($/g)	982	$0.03	$29.47
Total	$200.82		

The most expensive component for our project was the Touch Screen Display which unfortunately in the end didn't completely work as we were hoping. Initially we also intended to 3D print our own models which is why the team chose to buy brown PLA. However, as the semester progressed we found out that 3D printing by ourselves would take too long and required printing knowledge that none of us had. The team found an alternative solution which was to utilize the FEDC in Zachry. Fortunately, the costs were low and the school provided us with an initial budget. The FEDC was able to produce our models very efficiently and accurately. The budget also called for three different types of motors: DC, stepper, and servo. The final iteration of the project didn't use the 20kg servo motor because the component it would have been used in was scrapped. The rest of the items in the budget were used in the project, with some extras laying around.

The finalized budget contains slightly less spending than the proposed budget because we didn't spend nearly as much as what we were given to print at the FEDC. Either way, this didn't affect the limit on budget since the FEDC provided their own student budget before we had to pay out of pocket. Some purchased items could have been omitted from the budget because they weren't used in the project. Unfortunately, we couldn't have known these things until after we began implementing our ideas. Our estimate for the cost in mass production was $220, which is slightly more than what the final budget came out to be. The old budget is shown below as reference in comparison to the updated budget:

Item	Quantity	Cost Per Item	Total Cost
Raspberry Pi Touch Screen	1	$63.99	$63.99
Brown PLA 1.75mm (1kg)	2	$19.99	$39.98
High Torque DC Motor	0	$30.72	$0.00
HDMI to Micro HDMI Adapter	0	$8.99	$0.00
4 Pack L298N Motor Controller	1	$11.59	$11.59
Bearing Table	1	$15.99	$15.99
20 kg Servo Motor	1	$12.99	$12.99
3D Printing Services ($/g)	2500	$0.03	$75.00
Total	$219.54		

9 Appendices

10 References

1. J. Ausubel, “Older people are more likely to live alone in the U.S. than elsewhere in the world,” *Pew Research Center*, 24-Sep-2020. [Online]. Available: https://www.pewresearch.org/fact-tank/2020/03/10/older-people-are-more-likely-to-live-alone-in-the-u-s-than-elsewhere-in-the-world/. [Accessed: 13-Feb-2023].
2. C. Martinez, “Disability statistics in the US: Looking beyond figures,” Inclusive City Maker, 21-Nov-2022. [Online]. Available: https://www.inclusivecitymaker.com/disability-statistics-in-the-us/. [Accessed: 13-Feb-2023].
3. C. Martinez, “Disability statistics in the US: Looking beyond figures,” *Inclusive City Maker*, 8-Apr-2022. [Online]. Available: https://www.inclusivecitymaker.com/disability-statistics-in-the-us/. [Accessed: 13-Feb-2023].
4. T. Wilkinson, “Tastetro,” *TasteTro*. [Online]. Available: https://www.tastetro.com/. [Accessed: 08-Feb-2023].
5. J. Wang, “Automatic spice dispenser,” *Jeffrey Wang - Automatic Spice Dispenser*. [Online]. Available: https://jeffreywangdesign.com/SeniorDesign.html. [Accessed: 08-Feb-2023].
6. A. Garcia, J. Wood, M. Barros, and N. Campbell, “‘Spicer’ Automated Spice Dispenser,” *“Spicer” Automated Spice Dispenser*. [Online]. Available:

https://www.ece.ucf.edu/seniordesign/sp2020su2020/g10/assets/files/8PagePaperFinal.pdf. [Accessed: 08-Feb-2023].

7. Doc_10, “R/3dprinting - I've seen some recent interest in spice racks, so I wanted to share my senior design project, MeasureMINT, the automatic spice dispenser!,” *reddit*, 01-May-2021. [Online]. Available: https://www.reddit.com/r/3Dprinting/comments/n2nb85/ive_seen_some_recent_interest_in_spice_racks_so_i/. [Accessed: 08-Feb-2023].
8. M. Fahiz K.P, “Final project,” *Final Project - Muhammed Fahiz K.P - Fab Academy*, 14-Jul-2022. [Online]. Available: https://fabacademy.org/2022/labs/kochi/students/muhammed-fahiz/projects/final-project/. [Accessed: 08-Feb-2023].