Pamantasan ng Lungsod ng Maynila

(University of the City of Manila)

Intramuros, Manila

COLLEGE OF ENGINEERING & TECHNOLOGY

COMPUTER ENGINEERING DEPARTMENT

EMERGING TECHNOLOGIES

FINAL GROUP PROJECT

Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System

Submitted by:

GROUP 6 - TEAM JADE

Dalangin, John Rey G.

Davadilla, Daniela Aeryel B.

Fernando, Edmar C.

Racelis, Glorie Alynna C.

Robles, John Joe Rimuel P.

Date Submitted:

January 6, 2023

PROJECT TITLE:

Automated Merchandise Display Scheduler and
Smart Retail Store Shelves Management and Monitoring System

Project Description:

(Describe briefly the project and the motivation the group has in pursuing the project)

The Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System is a project that intends to help retail sellers organize their products, monitor the shelves in real-time, and predict which products to put on the display based on the period of time and the demand depending on the database record. The system is constructed using shelves where the retail items are placed, an Arduino Uno microcontroller interfaced to perform specified functions by taking in input from the sensors and expressing output by controlling the DC motors, IR sensors that determine when each shelf slot is empty, and linear actuators made from DC gear motors and plastic straws that pushes each item to the brim of the shelf.

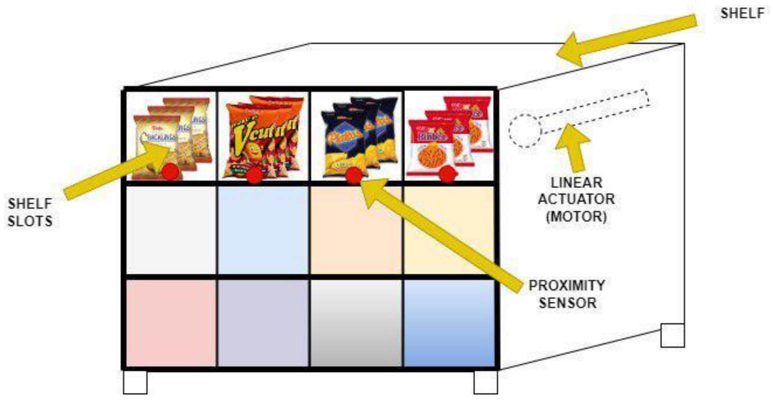


Fig. 1. Theoretical prototype of the smart retail store shelf

Figure 1 illustrates the theoretical final output of the project. Each shelf slot contains a specific retail item, an IR sensor, and a linear actuator. The sensor is placed at the edge of the shelf. Once a customer gets an item from the shelf, the IR sensor will detect that there is a missing item which will trigger the linear actuator to move until the sensor detects the remaining item again. The project will also have a mobile application useful during restocking. By utilizing this, the store admin can monitor the contents of the shelves and control the linear actuator once restocking is scheduled.

Project Background:

(Describe the environment where the project will be developed and implemented including how it will be integrated into the existing operations or process. Describe how it will be utilized or used and the intended end users of the project output)

Grocery stores usually have complete stocks of products they cater to. However, when consumers get what they need, the shelves are left with nothing to offer after restocking each shelf to the brim. Customers have to ask a salesperson to restock a certain rack. This causes disruption to the customers that are currently shopping in-store, as well as additional workload to the employee assigned to cater to customer service concerns at that specific time [1]. Furthermore, small-scale retail store owners who fail to record which products are frequently being sold out at the moment lose the opportunity to maximize their sales because they still get the same quantity of these items as those that are seldom being bought by consumers during the restocking period.

With that, the group came up with the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System. This system will automate the display production of retail products on shelves. Once the sensors detect that the front end of the shelf slot is empty, the mechanism (linear actuator) will automatically push its remaining items from the back. The number of items on the shelf is shown in the interface of the Blynk web application, the remaining and what was taken in real-time, and stores the gathered trend as it can be used as the database of the whole system. The system will predict what products to put on the shelf for a given period of time depending on the demand that the database recorded. When the count of items for the entire rack is projected empty by the application, it will notify the store clerk/owner to restock the display immediately according to the schedule. Time series analysis will also be applied in the system using the data gathered from the database. This will forecast the number of sales in the future as it reflects the trend of the products being sold.

Project Design:

(Describe the INPUT-PROCESS-OUTPUT design of the project. Include the program logic, screen design, output (softcopy or hardcopy), and the REPORT format design expected from the project. Provide a detailed description of the HUMAN-COMPUTER interaction using the output of the project.)

Input-Process-Output

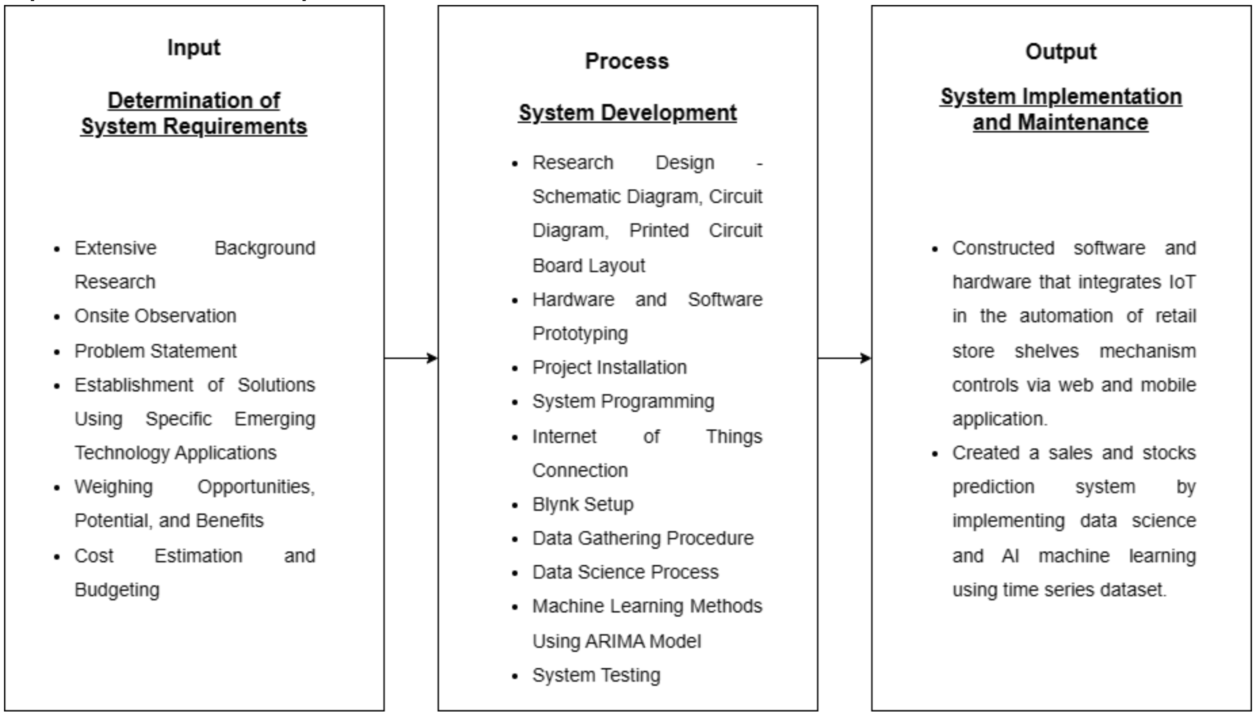
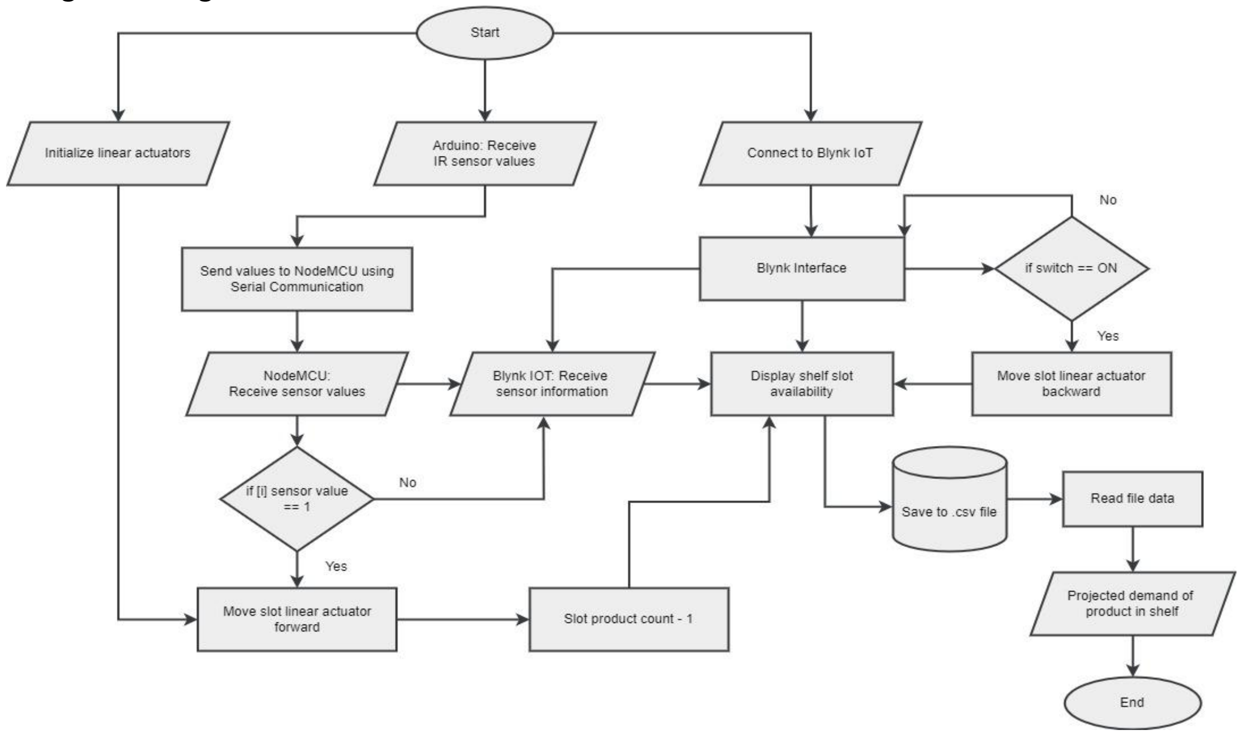

Input
Determination of System Requirements
Extensive Background Research
Onsite Observation
Problem Statement
Establishment of Solutions Using Specific Emerging Technology Applications
Weighing Opportunities, Potential, and Benefits
Cost Estimation and Budgeting
Process
System Development
Research Design - Schematic Diagram, Circuit Diagram, Printed Circuit Board Layout
Hardware and Software Prototyping
Project Installation
System Programming
Internet of Things Connection
Blynk Setup
Data Gathering Procedure
Data Science Process
Machine Learning Methods Using ARIMA Model
System Testing
Output
System Implementation and Maintenance
Constructed software and hardware that integrates IoT in the automation of retail store shelves mechanism controls via web and mobile application.
Created a sales and stocks prediction system by implementing data science and AI machine learning using time series dataset.

Fig. 2. Input-Process-Output model of the system

Figure 2 describes the overall structure of the team's implementation of the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System. As illustrated in the model, the foundational requirements were specified during the input phase. These requirements are prerequisites that must be handled throughout the whole procedure. Next is the process part where the inputs will be assigned to a certain solution in order to solve the given problems. The process part involves vital procedures such as the execution of the emerging technologies specified during the first phase. Here, the whole system is planned, designed, developed, and tested before actual implementation. Finally, the output part contains the answers to the requirements given during the initial phase. It produces a system that works as planned and is ready to be deployed in real-life administration.

Program Logic



Start
Initialize linear actuators
Arduino: Receive IR sensor values
Connect to Blynk IoT
Send values to NodeMCU using Serial Communication
Blynk Interface
if switch == ON
No
Yes
NodeMCU: Receive sensor values
Blynk IOT: Receive sensor information
Display shelf slot availability
Move slot linear actuator backward
if [i] sensor value == 1
No
Yes
Save to .csv file
Read file data
Move slot linear actuator forward
Slot product count - 1
Projected demand of product in shelf
End

Fig. 3. System flowchart

To further explain the system's inner workings and program logic, the team has provided a system flowchart illustrated in figure 3. As you can see, when the system started, we initialized the linear actuators, started the Arduino to receive IR sensor values, and connected to the Blynk IoT using the NodeMCU ESP8266 module. After the Arduino receives a sensor reading, it will send value to the NodeMCU using serial communication. When the wifi module successfully receives the data, it will initiate the Blynk application to display shelf slot availability on its interface. Then if the sensor receives data that indicates that a product was taken from the shelf, it will command the linear actuator to move forward and then indicate it to the Blynk's interface. On the admin side, if they see that the shelf stock is low, they can refill them by pressing the switch in the app. This will move the linear backward to the initial state and also allow them to refill the shelf slot without worrying that the linear actuator will trigger. Lastly, all the data gathered throughout the process will be saved to a .csv file. The file will then be used as the database for the prediction model made using Python to know if a certain slot is in demand.

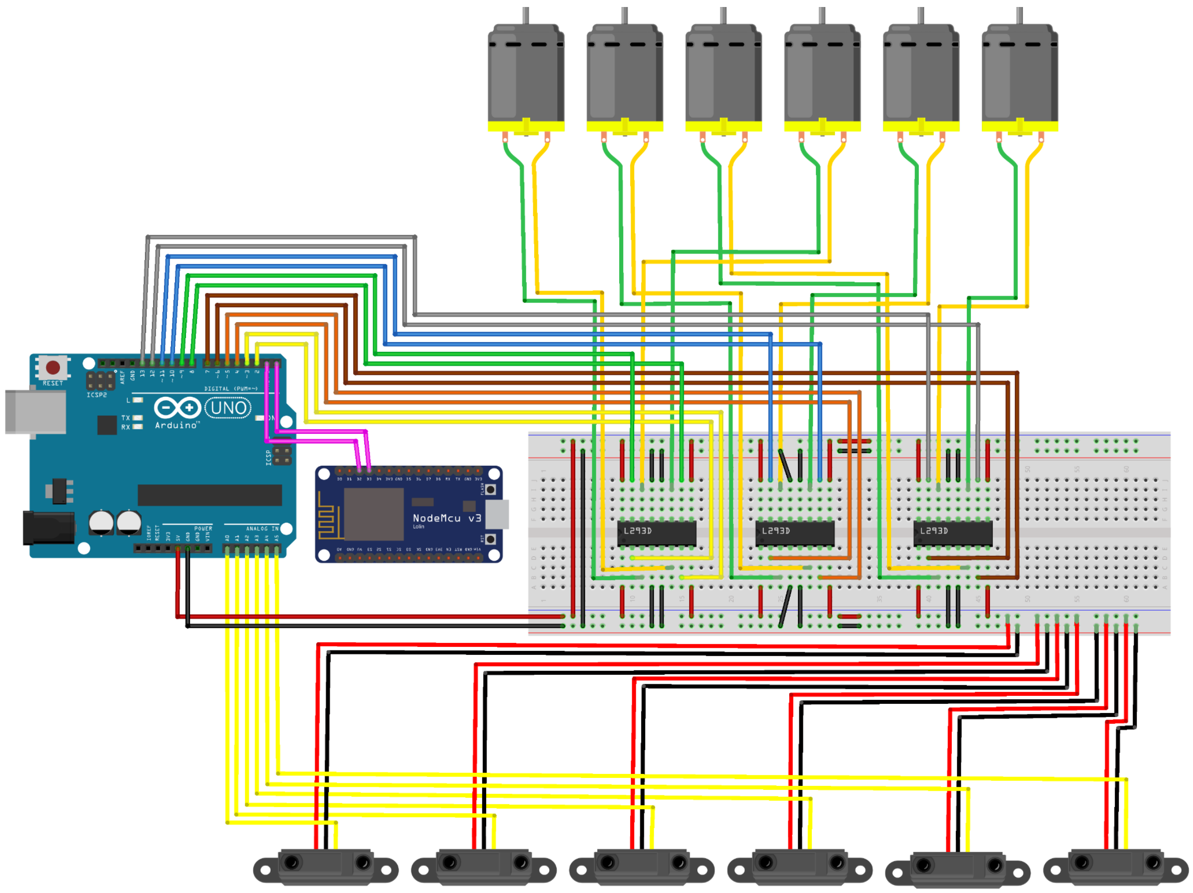


Fig. 4. Circuit diagram of the system

Figure 4 shows the actual circuit connections of the hardware part of the prototype for the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System. As you can see, it is composed of six DC motors functioning as linear actuators, six IR sensors, three L293D ICs, one Arduino Uno, and one ESP8266 NodeMCU wifi module. the inputs are generated by the IR sensors and are passed to the Arduino Uno's analog pins. On the other hand, two DC motors are connected to each motor driver ICs which receives the output from the digital pins of the Arduino Uno. Finally, two of the wifi module's digital pins are connected to the TX and RX pins of the Arduino microcontroller. These microcontrollers were programmed using Arduino IDE, a software used to alter the functions of the components connected to them.

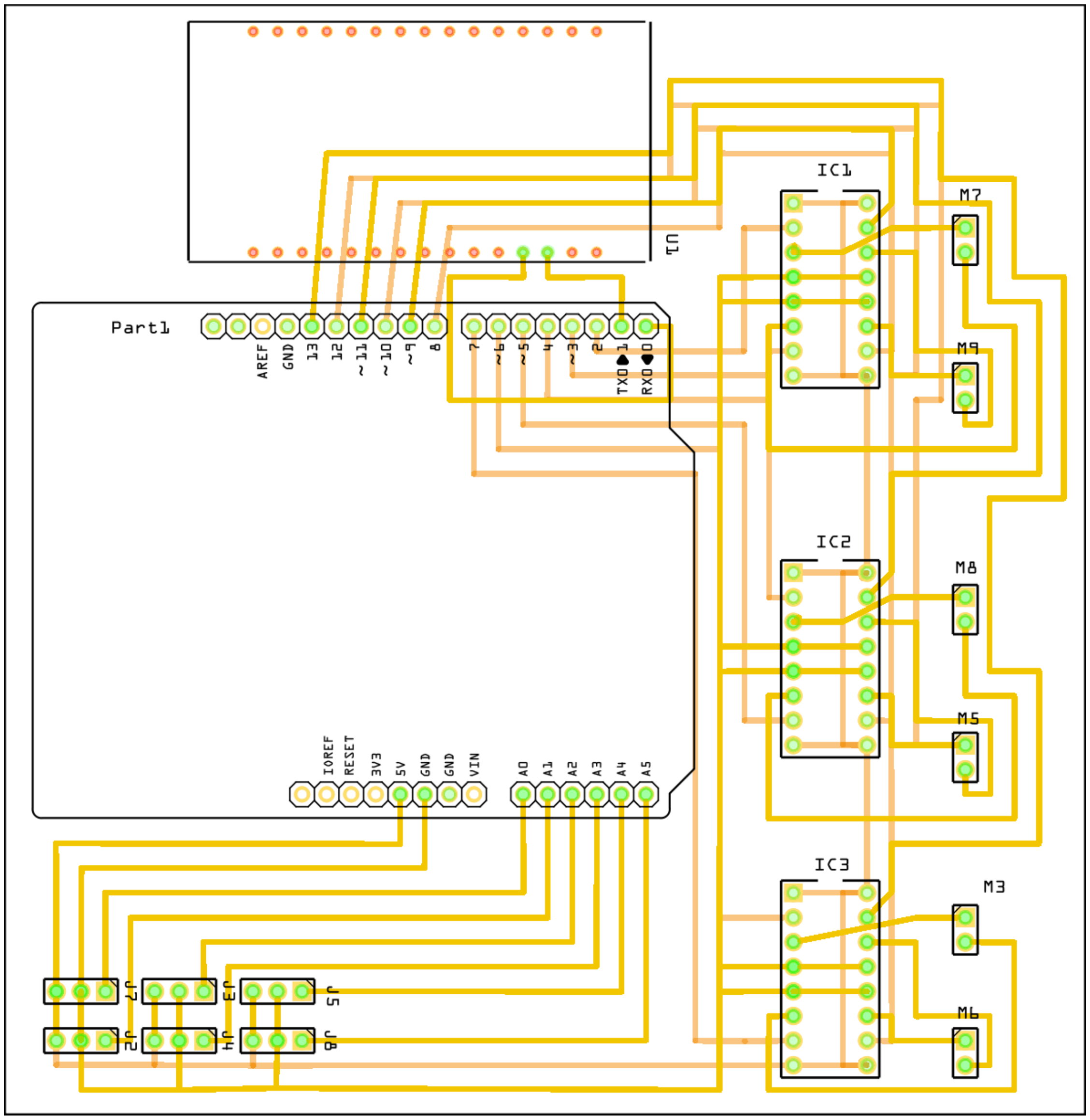


Fig. 5. PCB Design of the system

Similarly, the group also created a printed circuit board (PCB) layout for the system as illustrated in figure 5. The PCB is designed to create a more compact version of the system during mass production. The team used a double-sided PCB to imitate a more space-saving prototype where no traces of different connections intersect with one another. As seen in the figure, the necessary drills are all given a certain path to give way for their connections.

Screen Design

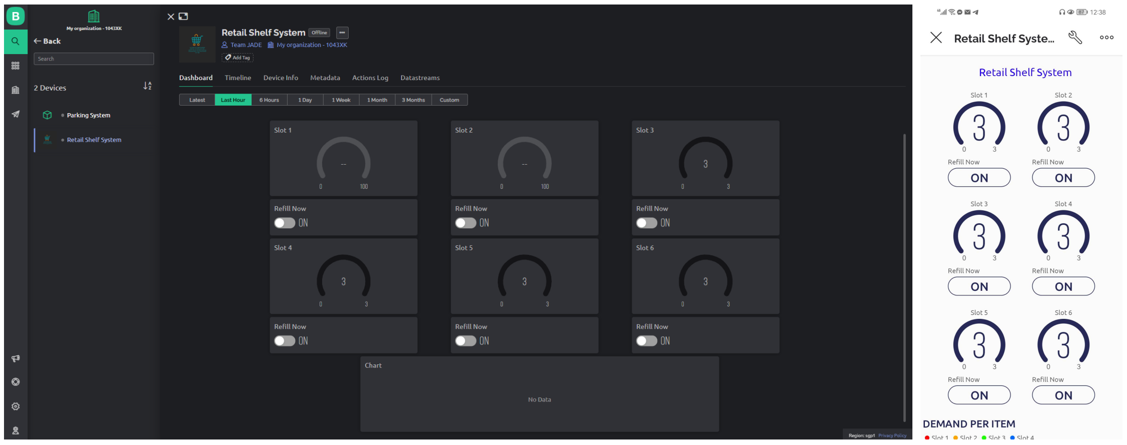


Fig. 6. Web and mobile Blynk dashboard design for PACSS

Figure 6 above shows the application of the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System, which can be accessed on different devices, including mobile phones and personal computers. As you can see, the screen design of both devices is similar. They both have six product counter gauges, which represent the six slots within the PACSS (push-and-count smart shelf). Both devices' interfaces include a separate button for the system's refilling function.

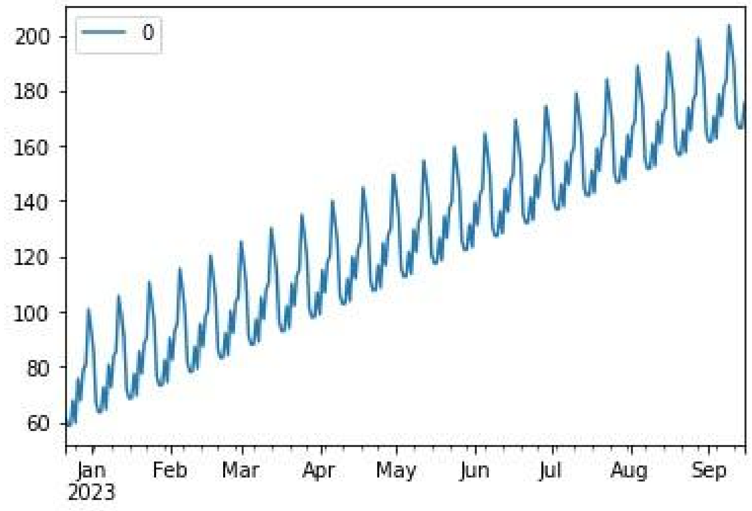


Fig. 7. Predicted sales recorded by the system

Figure 7 above shows the plot of the predicted values from the end of the dataset up to the set end date of the user. The dummy dataset is set from August 01, 2022, up to December 22, 2022. For this example, the set end date is September 23, 2023. Looking at the graph created by the predicted values of the model, the line produces an upward trend meaning that the sales of the specified slot are going upward resulting in a much greater profit for the business.

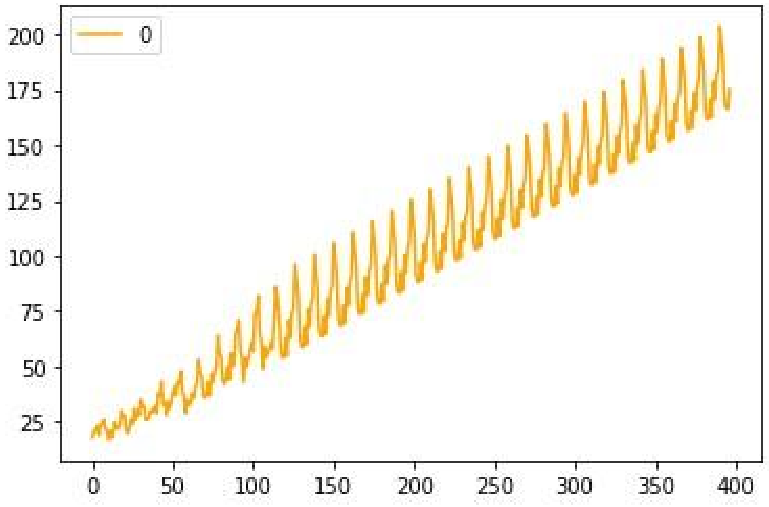


Fig. 8. Past total sales recorded by the system

The graph in figure 8 shows the overall values of the selected slot. From the first data which is August 08, 2022, up to the set end date of the user. As you can see, the graph is continuous, meaning that the predicted values of the ARIMA model are fitted to the selected dataset.

System Outputs

Hardware:

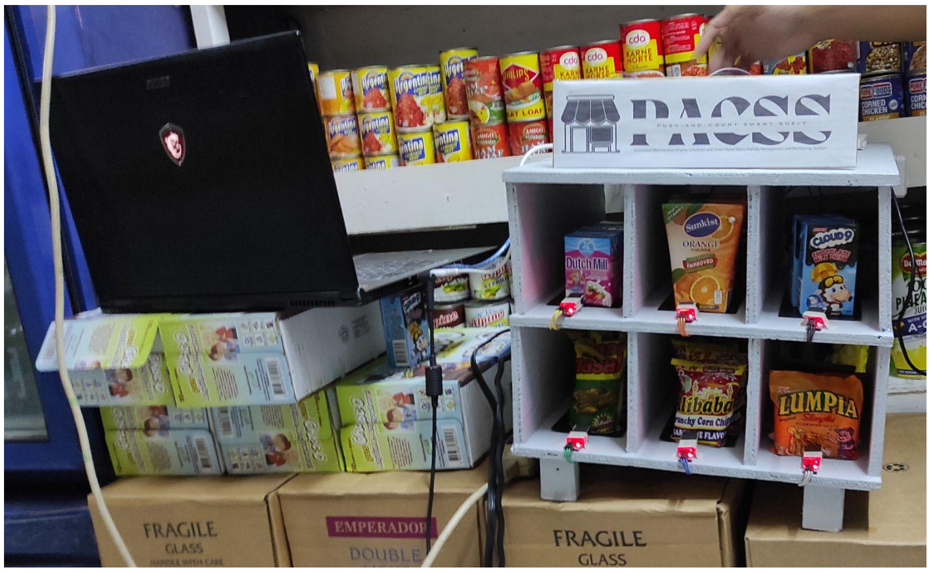


Fig. 9. Final actual PACSS prototype

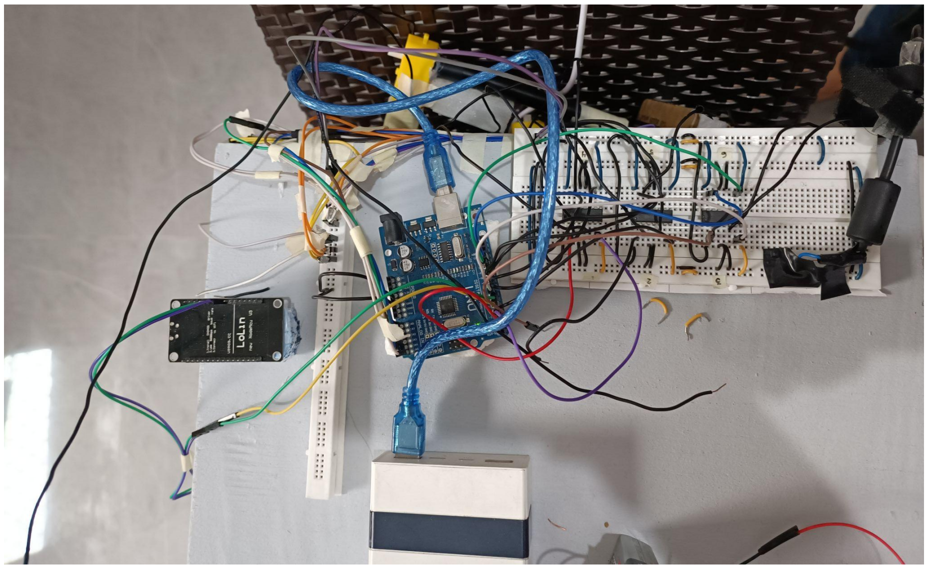


Fig. 10. Final actual PACSS circuit connections

Figures 9 and 10 show the final hardware output of the system. The components are placed on top of the shelf created by the team. The wifi module and the Arduino Uno were connected to a power bank as a power supply while the DC motors were connected to a 19V power supply. Three pieces of small sample products were placed in each shelf slot to imitate the shelves in grocery stores and supermarkets.

Software:

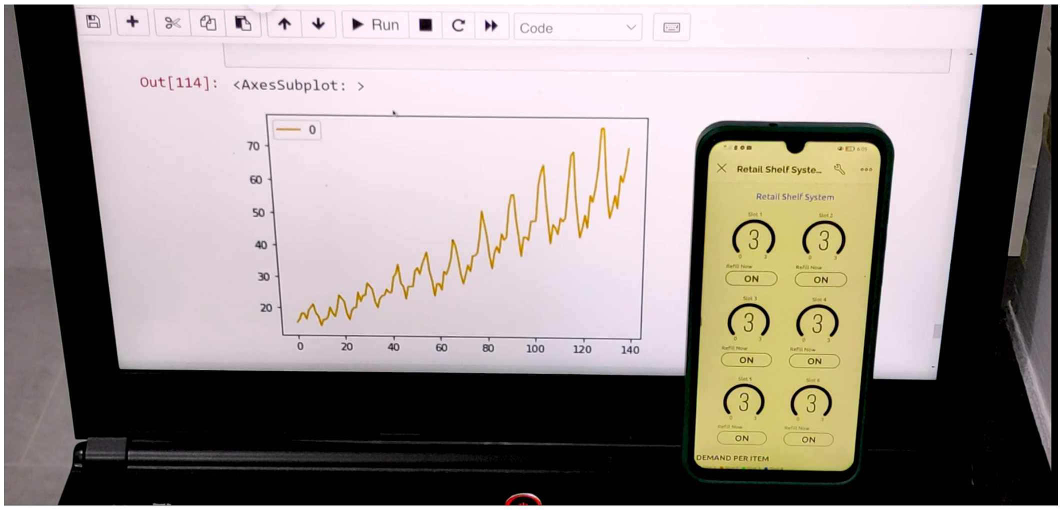
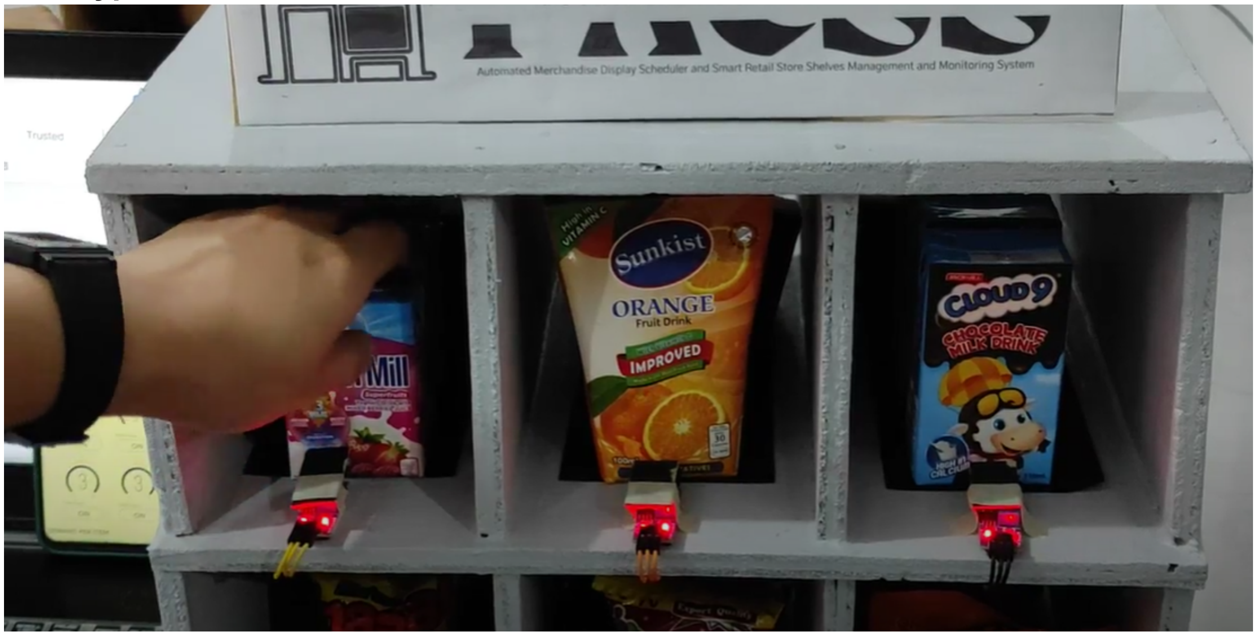


Fig. 11. Mobile Blynk dashboard and ARIMA model output for PACSS

Figure 11 shows an actual image of the output from the software section of the system. The mobile phone exemplifies the user's perspective when using Blynk's mobile application. The number of stocks left on the shelves is displayed on the screen. In addition, a button that may be pressed during refilling is also present. This may be clicked to control the linear actuators to go back to their initial position. On the other hand, the graph on the right demonstrates the output when the sales prediction program is running. The output graph shows the estimated number of items to be sold at a particular time.

Human-Computer Interaction

Prototype:



Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System
Sunkist
ORANGE Fruit Drink
IMPROVED
CLOUD 9
CHOCOLATE MILK DRINK

Fig. 12. A customer getting a product from the shelf

The image above (figure 12) shows a customer getting a product from slot 1. After the first item from the slot was taken, as sensed by the IR Sensors, the linear actuator will automatically push the remaining products from behind.

IoT Application:

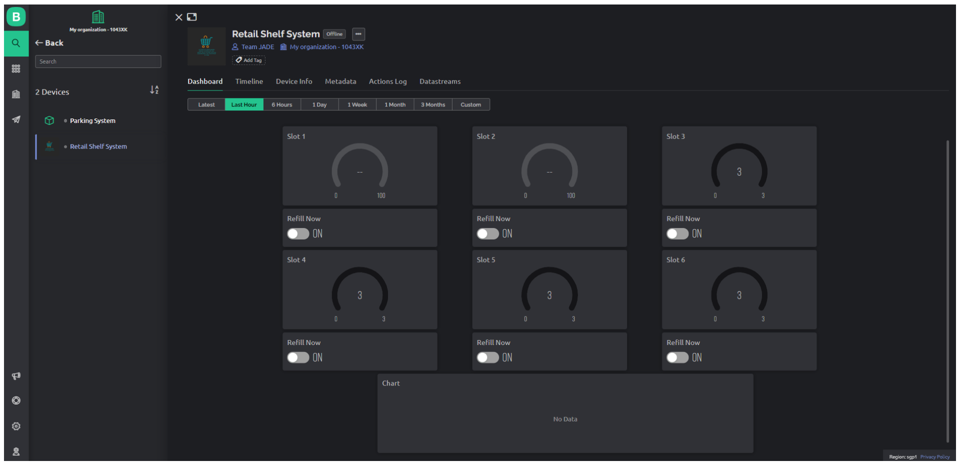


Fig. 13. The Blynk IoT application web dashboard

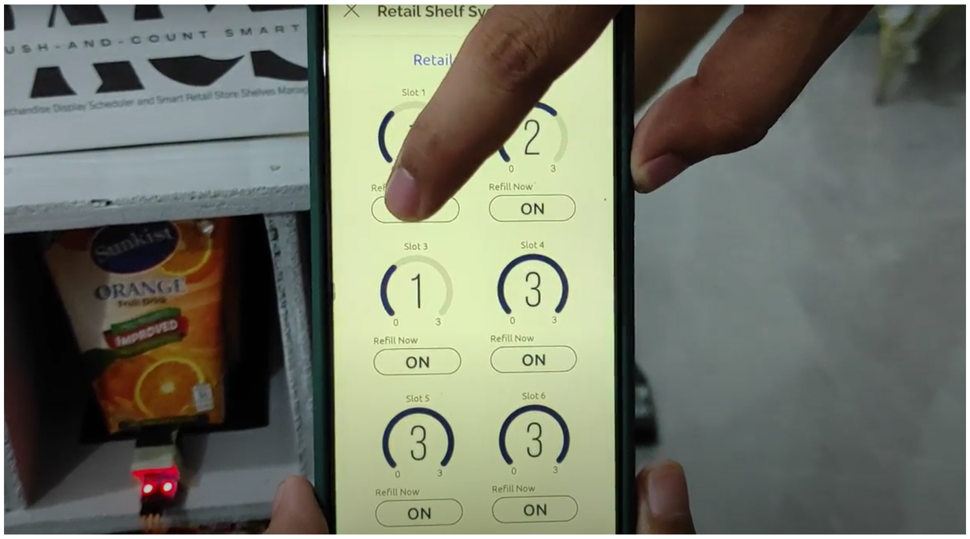
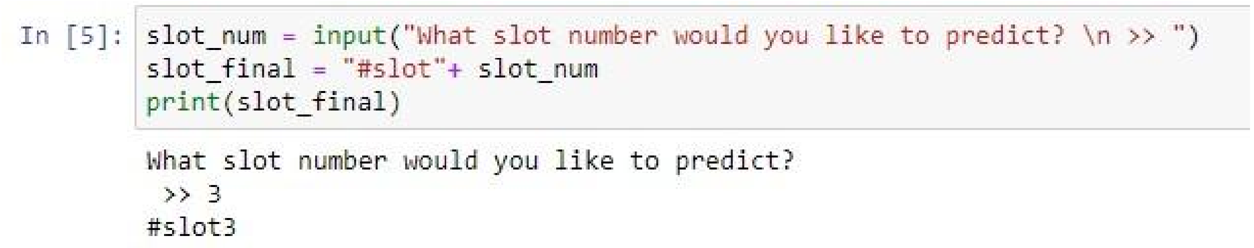


Fig. 14. The admin clicked the button just in time for restocking slot 1

Once the admin decides to refill the slot, they can click the button displayed in the Blynk web application. The linear actuator will reverse, giving space for the restock.

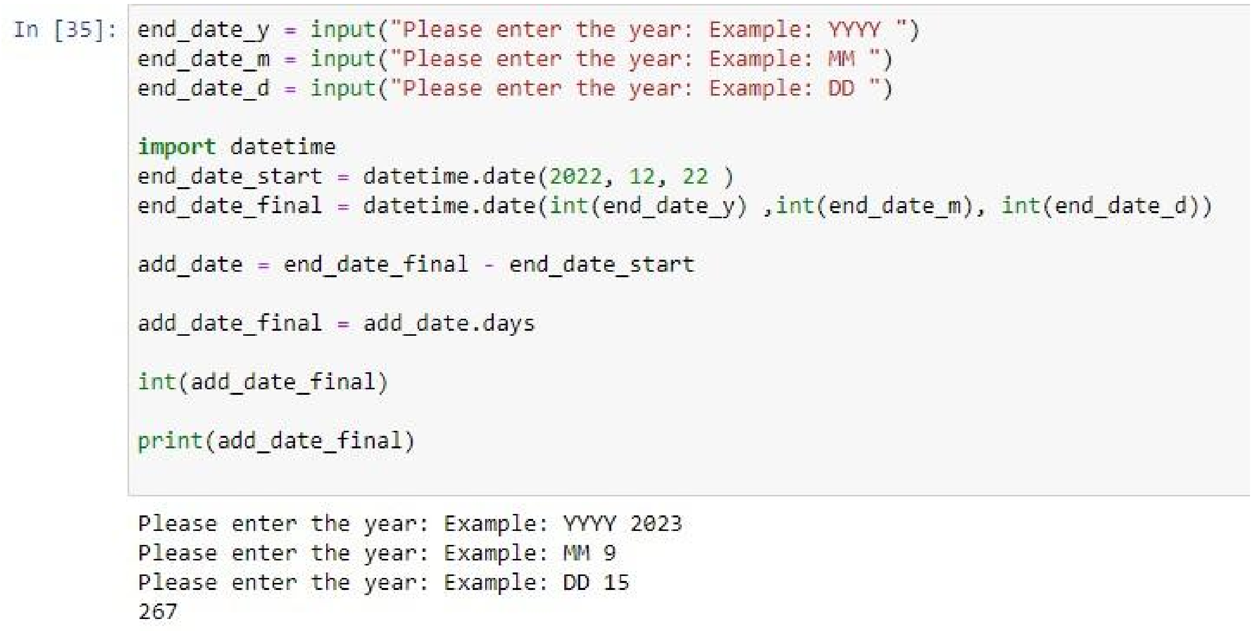
Data Science Application:



```
In [5]: slot_num = input("What slot number would you like to predict? \n >> ")
        slot_final = "#slot"+ slot_num
        print(slot_final)

        What slot number would you like to predict?
         >> 3
        #slot3
```

Fig. 15. Screenshot of simulation in Jupyter Notebook

The image above shows how the user input in the simulation of Jupyter Notebook. There, the system asked what slot number the user would like the system to predict.



```
In [35]: end_date_y = input("Please enter the year: Example: YYYY ")
         end_date_m = input("Please enter the year: Example: MM ")
         end_date_d = input("Please enter the year: Example: DD ")

         import datetime
         end_date_start = datetime.date(2022, 12, 22 )
         end_date_final = datetime.date(int(end_date_y) ,int(end_date_m), int(end_date_d))

         add_date = end_date_final - end_date_start

         add_date_final = add_date.days

         int(add_date_final)

         print(add_date_final)
```

```
Please enter the year: Example: YYYY 2023
Please enter the year: Example: MM 9
Please enter the year: Example: DD 15
267
```

Fig. 16. Screenshot of the simulation in Jupyter Notebook

Shown above is the simulation of counting of days from the date inputted by the user until the last date recorded in the database. As indicated in the code, the start date is 2022, 12, 22, or December 22, 2022, the date that the user typed in was 2023, 9, 15, or September 9, 2023, and as result, there are 267 days in between the two given dates.

Key Components:

Arduino Uno

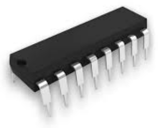
The open-source Arduino platform is used to create electronic projects. Arduino is made up of a physical programmable circuit board (also known as a microcontroller) and a piece of software called an IDE (Integrated Development Environment) that runs on your computer and is used to write and upload computer code to the physical board.

ESP8266 NodeMCU

NodeMCU is an open-source Lua-based firmware and development board designed specifically for IoT applications. It is a SOC microchip mainly used for the development of end-point IoT (Internet of things) applications. It is referred to as a standalone wireless transceiver, available at very low prices. It is used to enable the internet connection to various applications of embedded systems.

L293D Integrated Circuit (IC)

A semiconductor wafer known as an integrated circuit (IC) is used to fabricate thousands or millions of tiny resistors, capacitors, diodes, and transistors. As an amplifier, oscillator, timer, counter, logic gate, computer memory, microcontroller, or microprocessor, an IC can also perform other tasks.

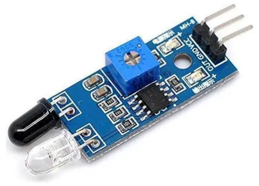


Linear Actuator using DC Motor

A DC motor, also known as a direct current motor, is a type of electrical device that uses direct current to generate a magnetic field that converts electrical energy into mechanical energy. Unlike the rotatory motion of the used electric motor, a linear actuator is an electromechanical device that can move linearly. In fact, a rotation is converted by the linear actuator into a linear movement capable of pushing, pulling, lifting, or positioning loads of up to 20 tons.

IR Sensors

An infrared sensor is a type of electronic device that emits light in order to detect certain aspects of its surroundings. An IR sensor can detect motion as well as measure the heat of an object. These sensors only measure infrared radiation rather than emitting it, which is known as a passive IR sensor. Typically, all objects in the infrared spectrum emit some form of thermal radiation.



Emerging Technologies Used:

(Describe the emerging technologies used to create, develop and implement this project. Describe the technologies used including the programming platform used to develop and implement the project)

NOTE: *Include PHOTOS and other materials to illustrate the use and functions of the technologies identified.*

To build the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System, the team integrated the following emerging technologies enumerated below in addition to the hardware prototype made from microcontrollers and integrated circuits.

Internet of Things

The Internet of Things (IoT) technology encompasses any physical object that can be connected to the internet to communicate, be controlled, or exchange information through it [2]. Since the system allows its user to pass data to an application in real-time via a WiFi connection, it is considered an implementation of IoT.

To incorporate IoT into the prototype *PACSS*, the group used a hardware component called ESP8266 NodeMCU which is a module that allows wireless connectivity to various projects.

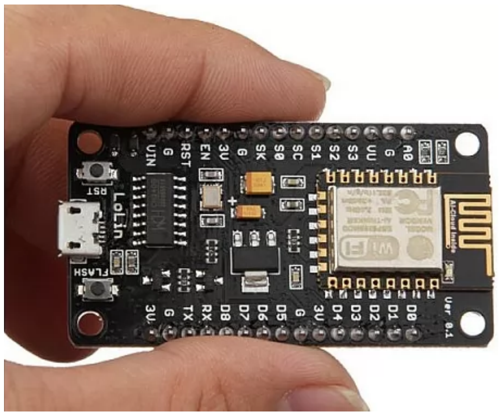


Fig. 17. The ESP8266 NodeMCU WiFi Module

Figure 17 illustrates an example of the said component which the group used to create a connection between the device and the web and mobile application. Digital data pins D2 and D3 of this module were connected to the Arduino Uno's TX and RX pins, respectively. Moreover, the module was programmed and uploaded to the system using the Arduino IDE.

The Blynk IoT platform was also utilized to create an environment where the data from the prototype can be viewed and controlled by the end users. This platform is created by Blynk, a software company that provides a no-code approach to building infrastructures suitable for IoT applications supporting multiple libraries and a wide range of devices and connectivity types [3].

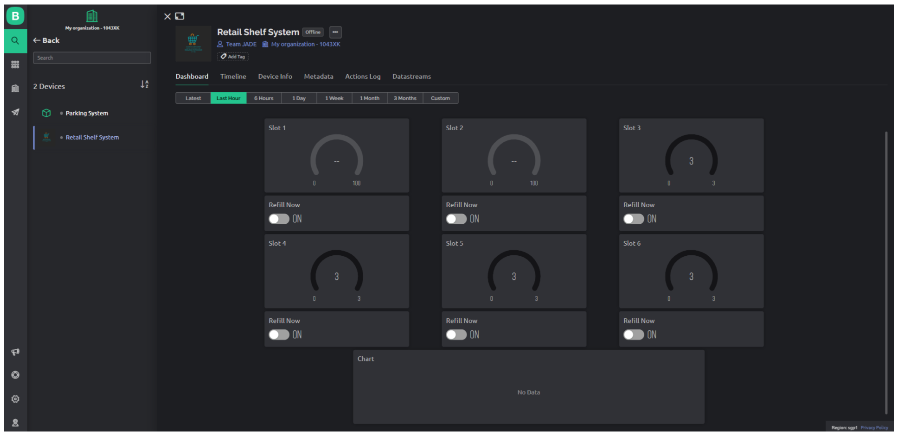
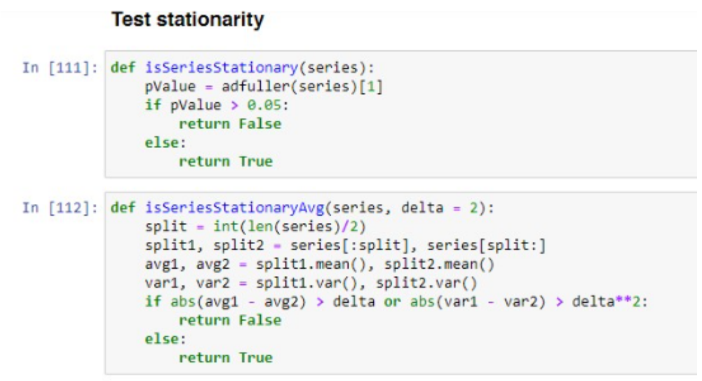


Fig. 18. The Blynk IoT application web dashboard

Figure 18 illustrates the Blynk IoT web dashboard that the team used to monitor the number of items going in and out of *PACSS*. It also includes widgets that function depending on how it was programmed. As long as the right network configuration is set on the wifi module, the Blynk application can control the linear actuators in real time even at remote locations.

Data Science

Data Science is another form of emerging technology implemented in the system. It is an area of technology combining various concepts such as mathematics, programming, advanced analytics, AI, and ML to uncover actionable insights from regular and randomly obtained datasets 4]. Data science also follows a specific functional life cycle which is being applied by data scientists and analysts to discover patterns and applications from enormous amounts of raw data.



Test stationarity

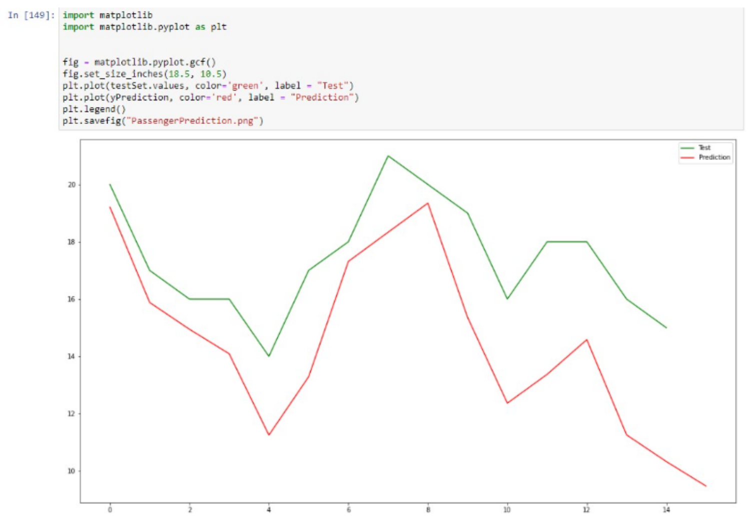
```
In [111]: def isSeriesStationary(series):
              pValue = adfuller(series)[1]
              if pValue > 0.05:
                  return False
              else:
                  return True

In [112]: def isSeriesStationaryAvg(series, delta = 2):
              split = int(len(series)/2)
              split1, split2 = series[:split], series[split:]
              avg1, avg2 = split1.mean(), split2.mean()
              var1, var2 = split1.var(), split2.var()
              if abs(avg1 - avg2) > delta or abs(var1 - var2) > delta**2:
                  return False
              else:
                  return True
```

Fig. 19. Application of Dickey-Fuller test to test and clean the dataset

In *PACSS*, data science is applied when the raw data in .csv format from the Blynk application is processed to produce relevant information from it. This is also the first step in the data science process. After this, the data obtained were scrubbed and explored; irrelevant data are discarded, significant data were linked to the Python program to assess the file, and time series graphs were formed. The given data were also tested for being stationary or not using the Dickey-Fuller test leading to further cleaning as illustrated in figure 19. Finally, the team applied the Auto-Regressive Integrated Moving Average (ARIMA) model and interpret its results for the user to understand and create better decisions for their retail store.

Artificial Intelligence

The final emerging technology applied to the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System is AI, a concept of technology that implements computer-based approaches to perform tasks that make use of intelligence [5]. More specifically, the group used Machine Learning methods, which is a branch of AI, to create a model by training and testing datasets.



```
In [149]: import matplotlib
          import matplotlib.pyplot as plt


          fig = matplotlib.pyplot.gcf()
          fig.set_size_inches(18.5, 10.5)
          plt.plot(testSet.values, color='green', label = "Test")
          plt.plot(yPrediction, color='red', label = "Prediction")
          plt.legend()
          plt.savefig("PassengerPrediction.png")
```

Fig. 20. Example of prediction using ARIMA model

To apply AI and Machine Learning to the system, the group tested the datasets using several models before ending up with Auto-Regressive Integrated Moving Average (ARIMA) model which is the most suitable model to forecast the non-seasonal datasets in time series format. A sample prediction after treating the non-stationary data from the original dataset from the system is illustrated in figure 20 using Jupyter Notebook. Using this model, the team was able to predict future sales of each item on *PACSS*.

Potential and Benefits of the Project:

(Describe the potential of your group project and the benefits it may provide. Include information to support your answers)

The food retailing industry is thriving in the Philippines, with a variety of vendors including street vendors, wet and dry markets, sari-sari shops, grocery stores, supermarkets, hypermarkets, warehouse, and discount clubs, and convenience stores. According to [6], supermarkets were the most common retail location, accounting for 40% of all retail food sales in the Philippines. In addition to being close to customers' homes and places of employment, supermarkets provide a variety of fresh and pre-packaged goods at reasonable prices.

Monitoring if the shelves are appropriately stocked and product refilling can require much effort in owning a retail franchise. With that, the group came up with the idea of an Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System. This project may benefit small-scale and large-scale grocery store owners as they can easily oversee the selling of the products they offer and keep track of the inventory.

The following are the benefits of this project:

- Automation of retail store shelves mechanism
- Comfortable and timely monitoring system with real-time updates using the Internet of Things
- Prediction of future product demands with AI and Machine Learning technology

End-User Acceptance and Approval

This is to CERTIFY that the project, *Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System* after actual testing and evaluation meets the desired and required output expected from the project based on the identified and defined project objectives presented to us.

Juner [illegible] Ruto

Signature Over Printed Name

Date: 01/04/2023

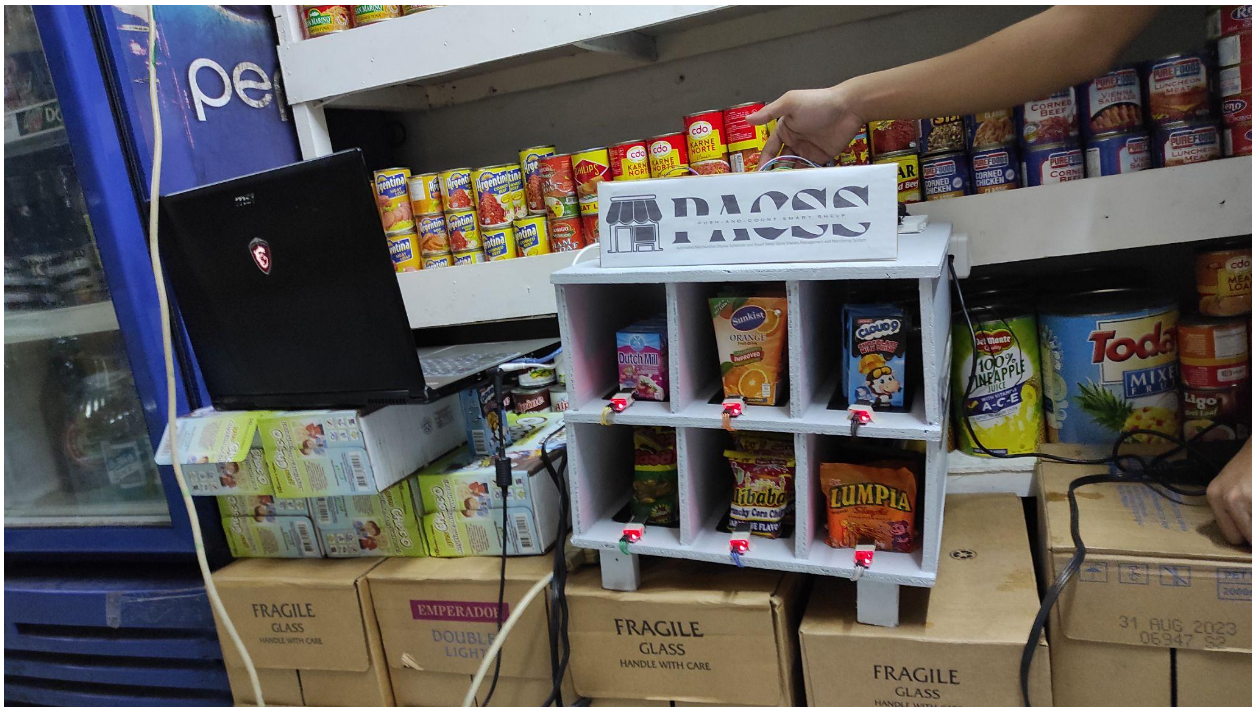


Fig. 21. Actual set-up of the prototype in the user's grocery store.

Fig. 22. Photo of the group with the store owner, Mr. Juner Rufo.

AUTOMATED MERCHANDISE DISPLAY SCHEDULER AND SMART RETAIL STORE SHELVES MANAGEMENT AND MONITORING SYSTEM: USERS MANUAL

Project Description:

(Describe briefly the project and what is it for)

The Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System is composed of a shelf with every slot housing a particular retail item, an IR sensor, and a linear actuator. The sensor is positioned at the shelf's edge to detect when an item is removed from the shelf. The linear actuator will push other items immediately after the IR sensor detects a missing item and it will continue to do so until it detects the item once more. Using a mobile application connected to the internet, the store administrator can keep track of what is on the shelf and, once restocking is planned, regulate the linear actuator to move backward to give space for the new stocks of products. This project aims to assist retail sellers in organizing their merchandise, monitoring the shelves in real time and predicting which item to put on the display depending on the database recorded from past sales.

Project Installation Requirements:

(Describe the technology requirements needed by the project to be installed. Include both the software and hardware requirements. If needed, describe the skill sets needed to make the project operational)

NOTE: *Include graphical images to describe the operation needed to be performed including specific instructions in logical order or step-by-step procedure.*

The following components are vital to properly install the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System in a local retail setup.

Hardware

- **Setting up the Shelf**
  To start the project installation, the users must ensure that there is adequate space left for the IR and linear actuators inside the shelves. Because shelf sizes differ in each retail store, they must first be measured before permanently installing the system. Moreover, the size of the linear actuators may also vary depending on the weight of each item being sold inside the store.

Fig. 23. Shelf Assembly

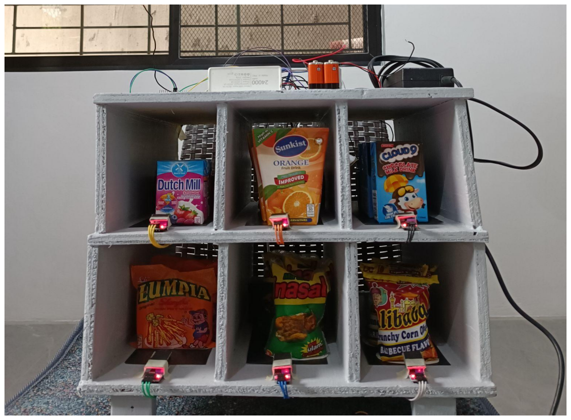


Fig. 24. Final Shelf Installation

- **Printed Circuit Board**
  Once the system is ready to be deployed, the customized PCB that emulates the original prototype will be sent to the end users for commercial use. This is produced and deployed to consumers because it eliminates the use of excessive wiring and loose connections. It also provides ease of use to the users because it is readily available and can be set up even without much knowledge of its technical aspect.

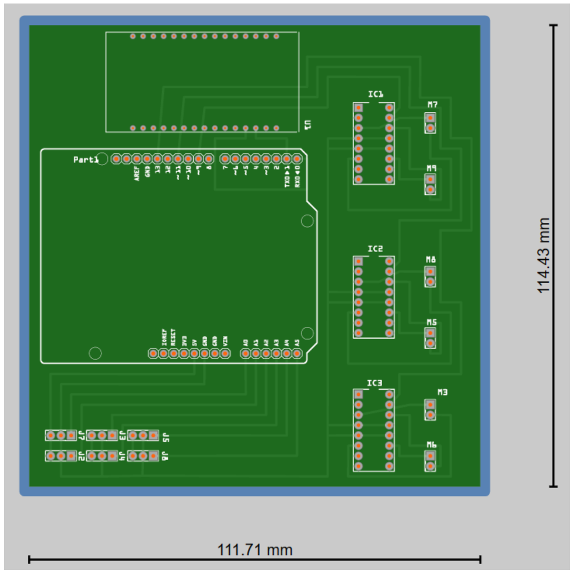


Fig. 25. The top view of the system's PCB

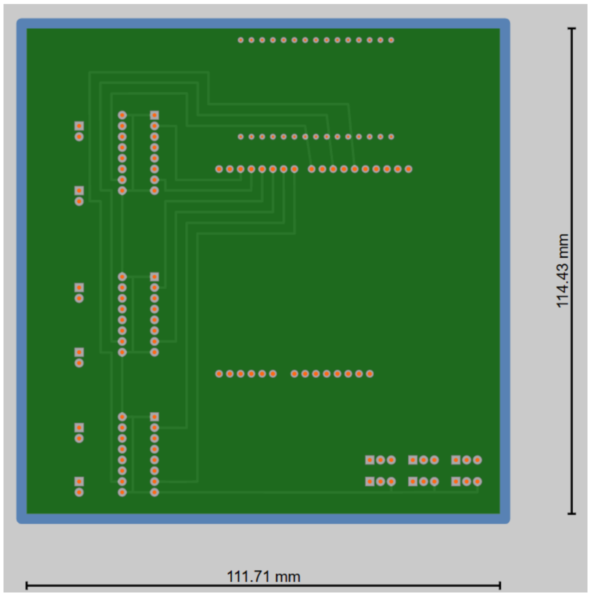


Fig. 26. The bottom view of the system's PCB

Figures 25 and 26 show the final double-sided PCB output intended for the system. In this PCB, drills are provided for the relevant pins of one Arduino Uno, one ESP8266 Wifi Module, three L293D integrated circuits, six IR sensors, and six DC motors.

- **Setting up IR Sensors**
  The system also comes with infrared sensors connected to the PCB which are supposedly placed on the brim of each shelf slot. The sensors have two red LED lights, the power, and obstacle LED light. The power LED signals the user that the system is on while the obstacle LED light signals that the shelf is full. Setting up the system must include checking that both lights are working as expected. In addition, the sensors include a distance adjustment variable resistor which can be controlled by the user to calibrate the proximity of the objects that the sensor can detect.

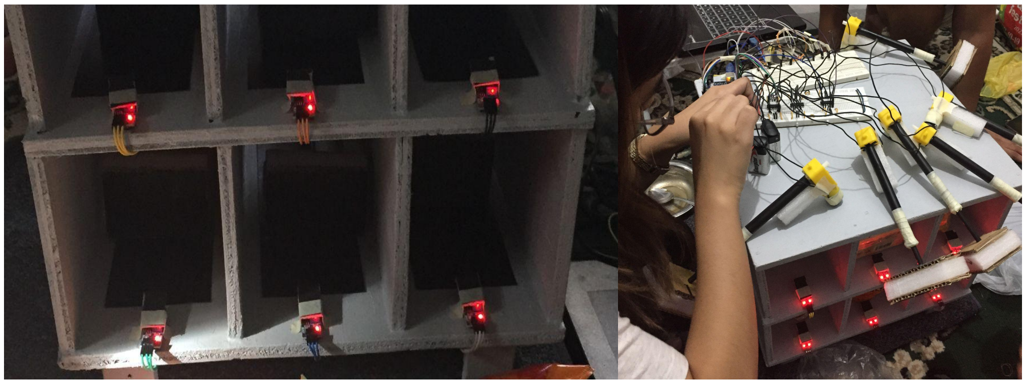


Fig. 27. Setting up the IR sensors

- **Setting up Linear Actuators**
  Linear actuators also come with the system. They are initially made using geared DC motors. However, it must be upgraded as the size of the products that must be pushed gets larger. These components must be placed at the back of each shelf slot, directly in front of the IR sensor. It is also important to keep the actuators in a stationary position during the installation process so that they will not become unstable during working hours.

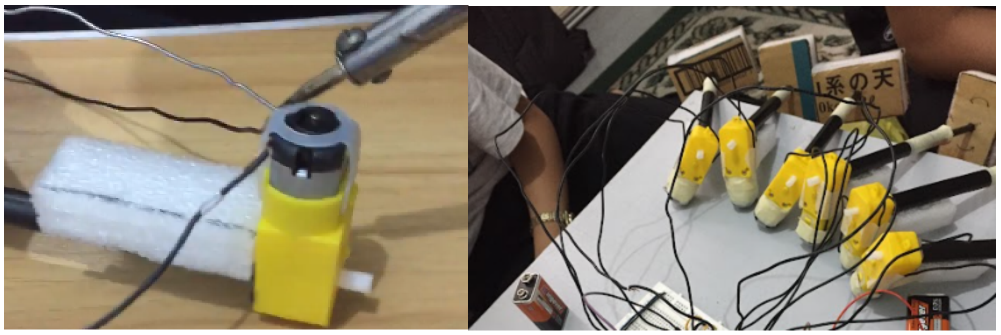


Fig. 28. Creating DIY linear actuators using DC motors

- **Power Supply**
  Because the power supply should strictly follow the standard measurement to regulate the voltage and current flowing in the circuit, it must be provided with the system during deployment. It is also important to connect the system to a power supply that matches the required measurement value to protect the components from various hazards.

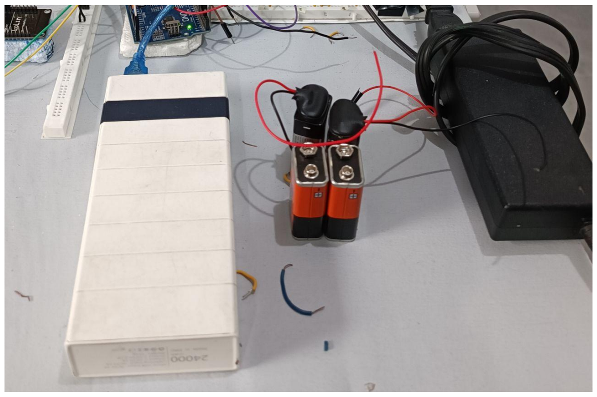
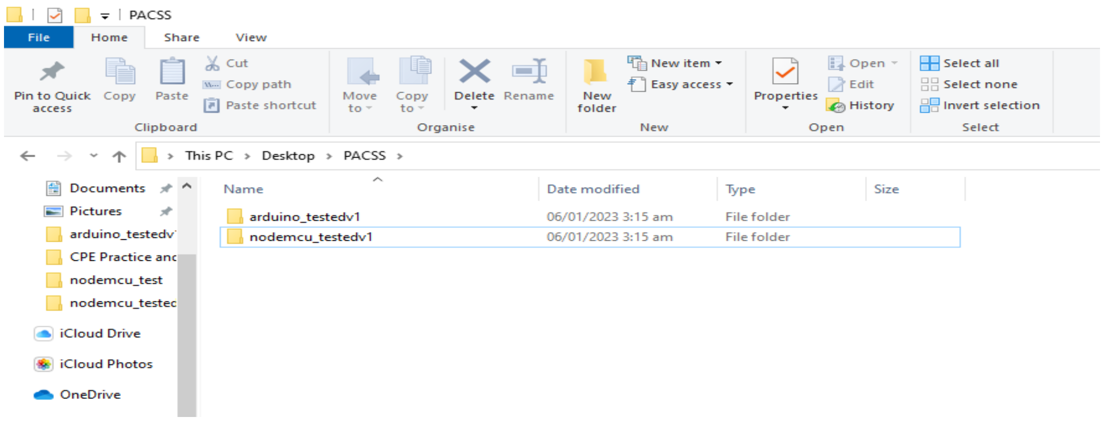


Fig. 29. Power supply for the Arduino, Wifi Module, and the motors

Software

- **.zip file containing the hardware prototype codes**
  After setting up all the hardware components of the system, the user will then be required to download the codes provided by the team into their computers in the form of a .zip file. The folder inside the .zip file will contain the hardware prototype codes including the program for the Arduino Uno microcontroller and the ESP8266 NodeMCU Wifi Module. This will be unzipped and opened in the Arduino IDE. Once opened, the user must first enter the correct details about

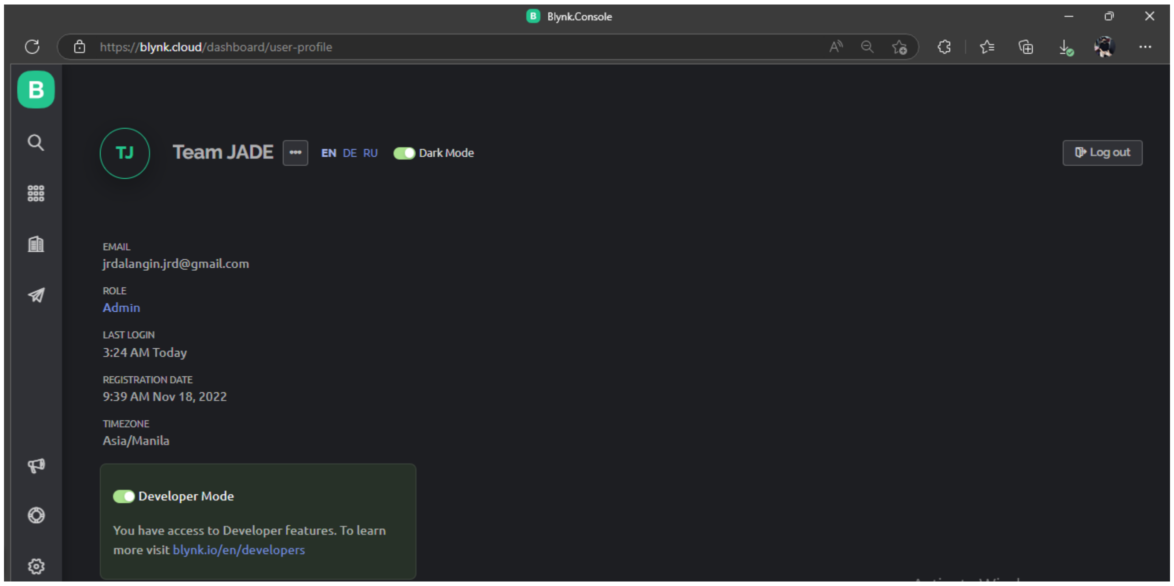
their wifi network and then plug in the necessary components on the USB ports of their computer before uploading each code to make the system run as expected.



Name	Date modified	Type	Size
arduino_testedv1	06/01/2023 3:15 am	File folder	
nodemcu_testedv1	06/01/2023 3:15 am	File folder	

Fig. 30. Folder containing the programs

- **Blynk account**
  A website called Blynk was used as a platform to create the application simulation of the system. In order to access the application intended for the smart shelf system, the end user must log into the Blynk web or mobile application using the account to be provided by the group. From there, they can access the application, view the number of items inside the shelves, control the DC motors during refilling, and see the trend of sales of the following items inside each shelf slot. From there, they can also download the .csv file which will be utilized in the data science application.

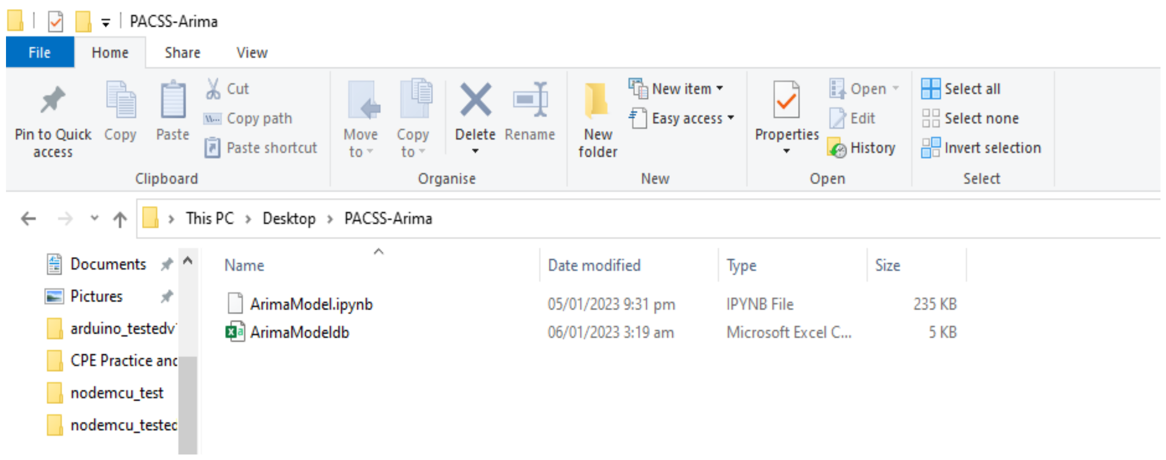


EMAIL
jrdalangin.jrd@gmail.com
ROLE
Admin
LAST LOGIN
3:24 AM Today
REGISTRATION DATE
9:39 AM Nov 18, 2022
TIMEZONE
Asia/Manila

Developer Mode
You have access to Developer features. To learn more visit blynk.io/en/developers

Fig. 31. Setting up Blynk account

- **.zip file containing the software data science codes**
  A separate .zip file containing the codes for the sales prediction and item scheduling must be downloaded by the user of the system. To access this, they must first download the .zip file on their computers provided by the team and extract it. Then, the file must be opened either in Google Colab or Jupyter Notebook. It is important to note that the .csv file produced using the Blynk application should be placed in the same folder as the files for the data science. After completing these steps, the codes will be run on the computer so that the models can process the data and turn them into valuable content which accurately represents the goals of the system.



Name	Date modified	Type	Size
ArimaModel.ipynb	05/01/2023 9:31 pm	IPYNB File	235 KB
ArimaModeldb	06/01/2023 3:19 am	Microsoft Excel C...	5 KB

Fig. 32. Folder containing the prediction codes

Project Operations:

(Describe step-by-step operations of the project. Include graphical images or diagrams to properly illustrate the processes and operations. The screen layout of the Human-Computer interactions are expected)

The detailed project operation of the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System is as follows:

Prototype Operation

1. Plug in the power source and turn the switch beside the PACSS (Push and Count Smart Shelf) on to power the automated merchandise display scheduler and smart retail store shelf management and monitoring system.

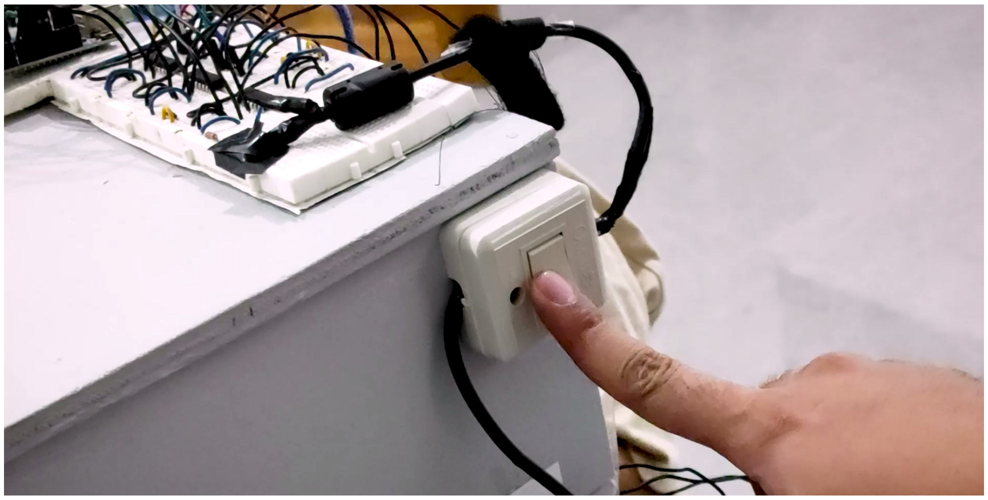


Fig. 33. Pressing the switch for system power

2. Next, press the reset button for both the ESP8266 NodeMCU and the Arduino to restart the codes uploaded for each module.

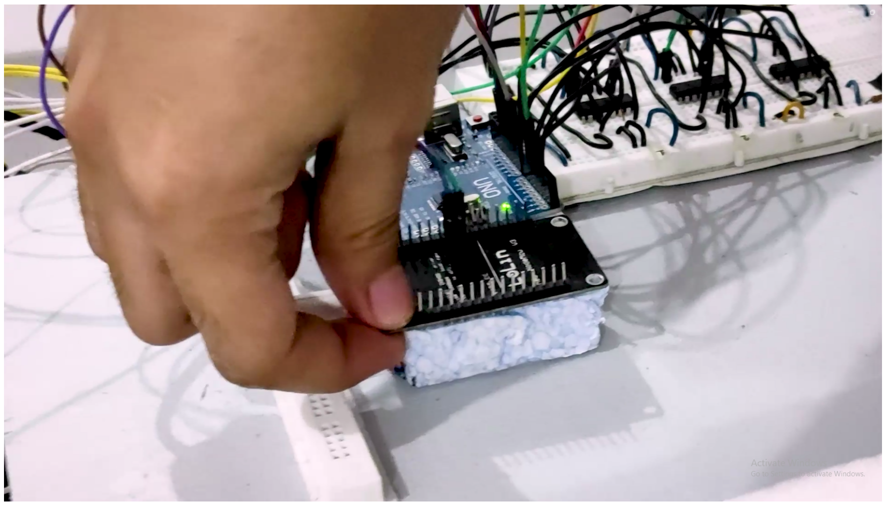


Fig. 34. Pressing the reset button of the ESP8266 NodeMCU

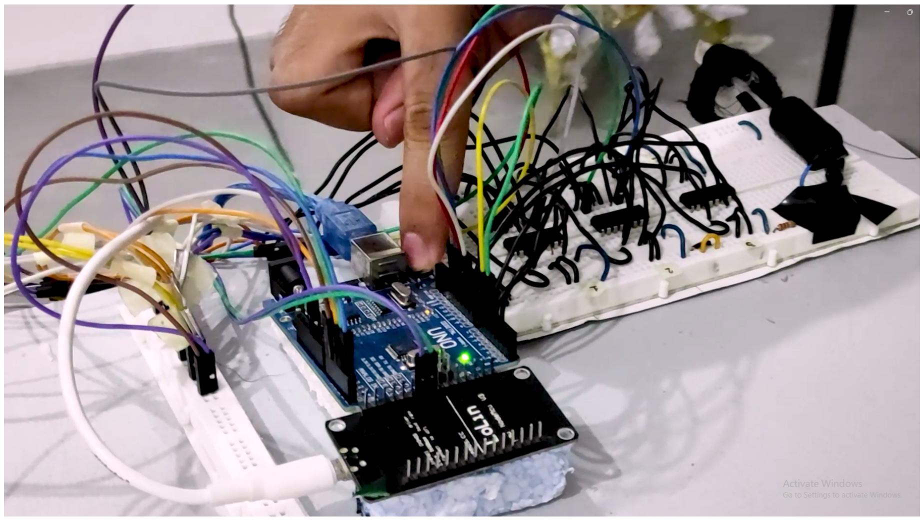


Fig. 35. Pressing the reset button of Arduino UNO

3. To see whether the code has successfully run and the wifi module has established a connection, check the web application and the mobile application of the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System.

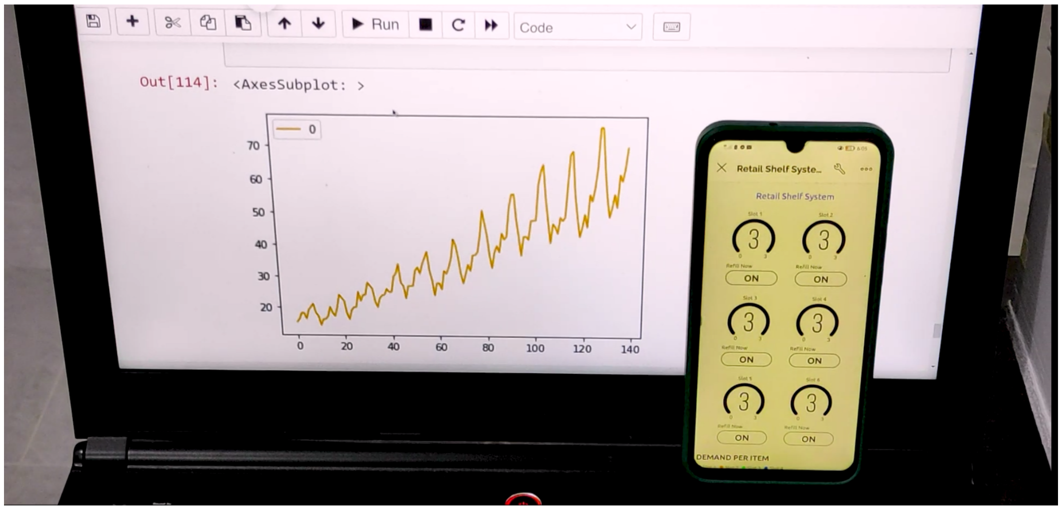


Fig. 36. Establishing Blynk connection

4. After the successful connection of the hardware of the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System to the IoT, the user can now operate the system. If the user takes one product from a slot, the linear actuator in that slot will push the next product until it reaches the front of the slot.

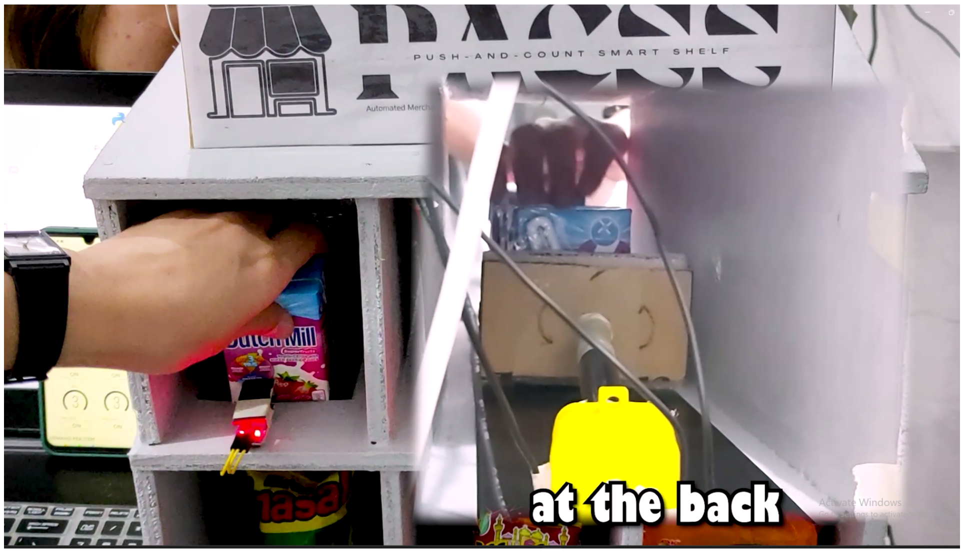


Fig. 37. Getting items from slot 1

5. On the other hand, the sensor attached to the specific slot where the product is taken will update its data, which it will send to the IoT using the WiFi module. The updated data from the sensor will be reflected in the mobile application by decreasing the product count of the slot where it was taken.

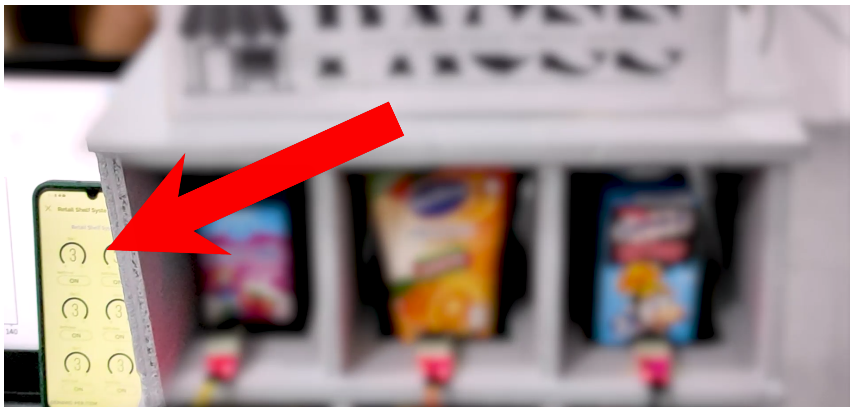


Fig. 38. Checking the Blynk app for value changes

6. When another product is taken from the same slot of the PACSS (Push and Count Smart Shelf), the same concept will apply to the system. The linear actuator of that specific slot will be activated and push the next product until it reaches the front of the slot and is sensed by the IR sensor. Upon detecting the next product, the corresponding slot in the mobile application will be updated with the remaining product within that specific slot.

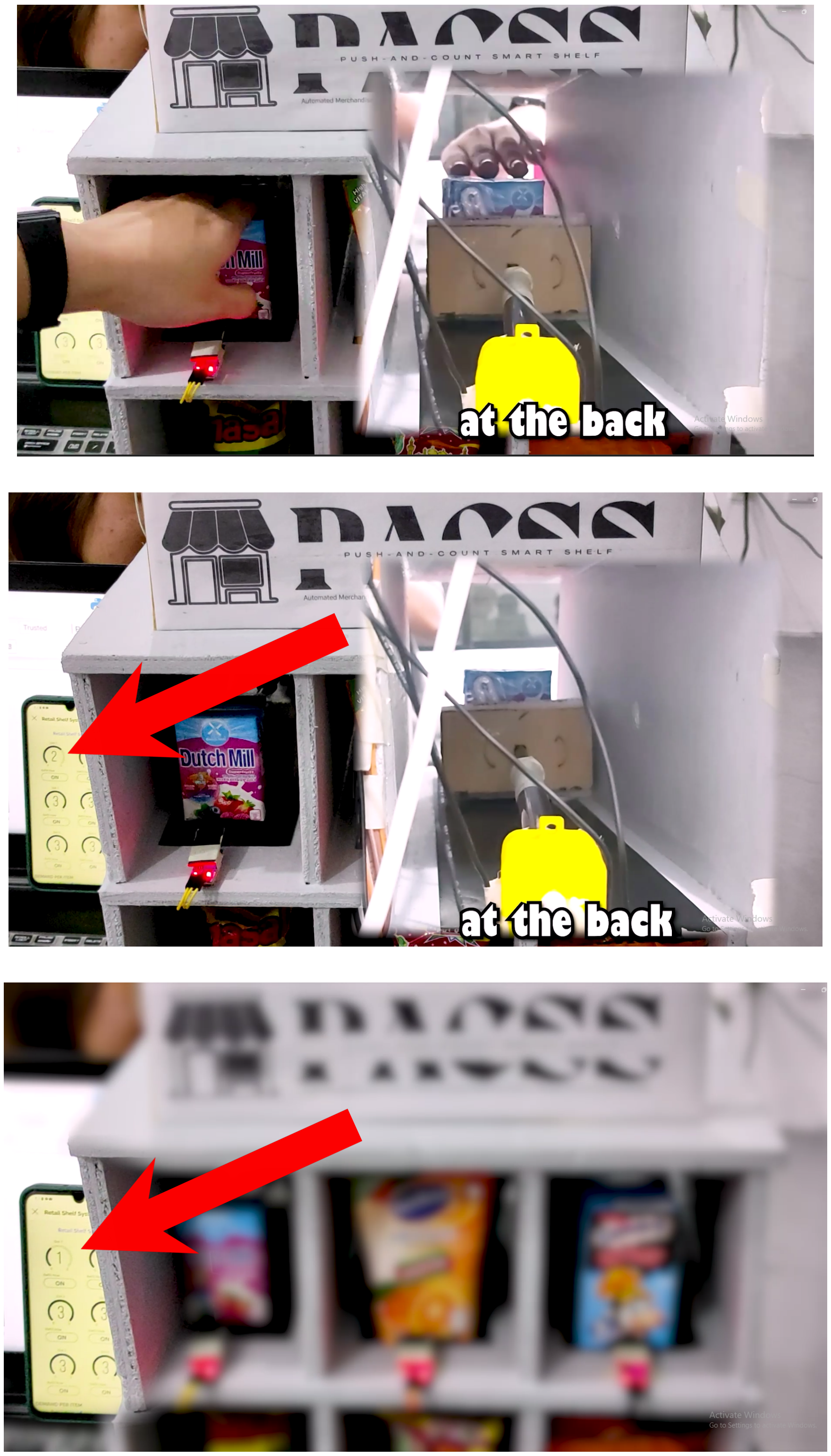


Fig. 39. Pushing mechanism of the system and Blynk IoT's fast response

7. The same logic will apply to the other slot, when the user tries to take a product from another slot, the linear actuator will activate and push the remaining product until the IR sensor at the front of the slot will detect it. Then the mobile application will update its product counter for each slot.

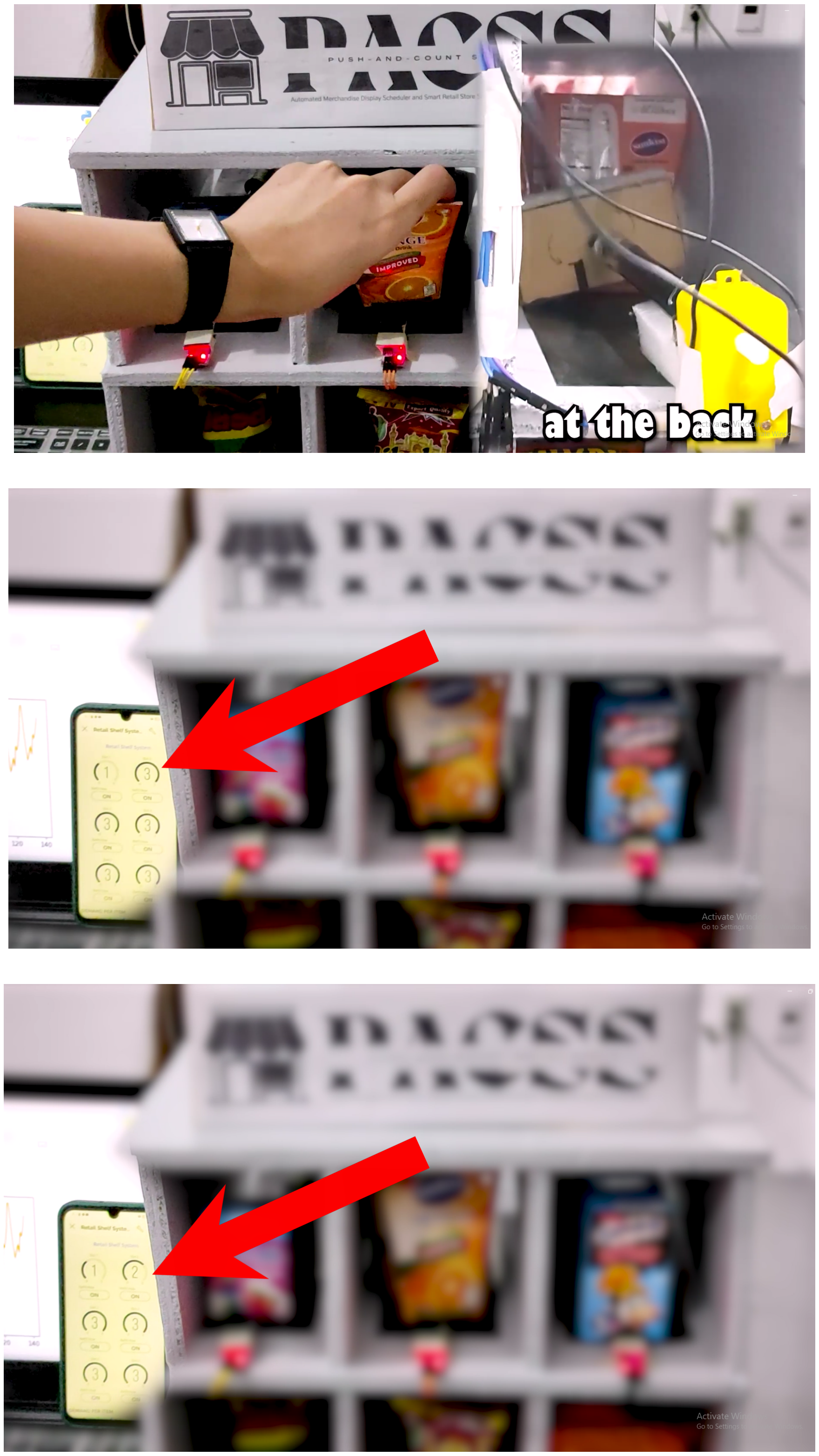


Fig. 40. Pushing mechanism of the system and quick response from Blynk IoT

8. When each product slot needs to be refilled, the user has the control to move the linear actuator back to its original position for the pushing function by using the mobile application. Upon clicking the "refill now" button within the application, the

dc motor or linear actuator will rotate counter-clockwise, which will result in the backward movement of the actuator.

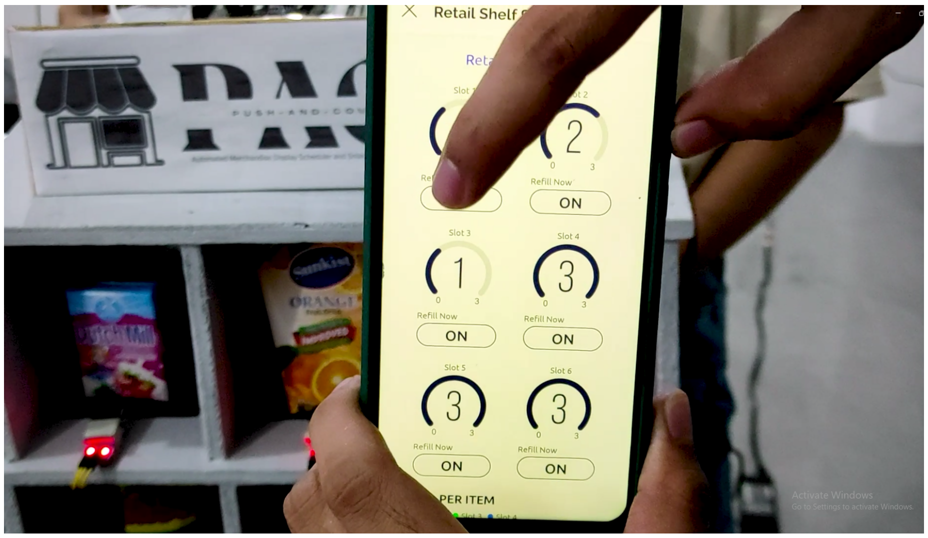


Fig. 41. Pressing the Refill button in the app to refill the slot

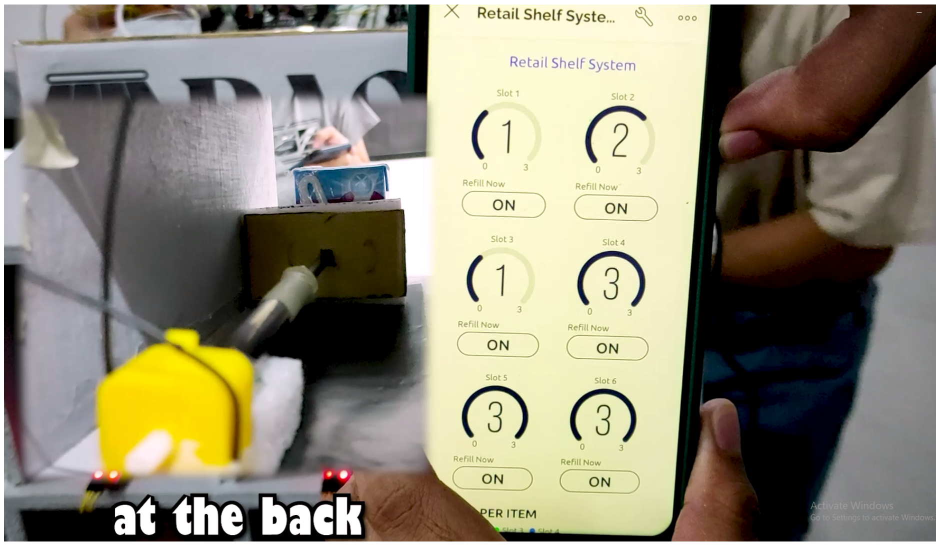
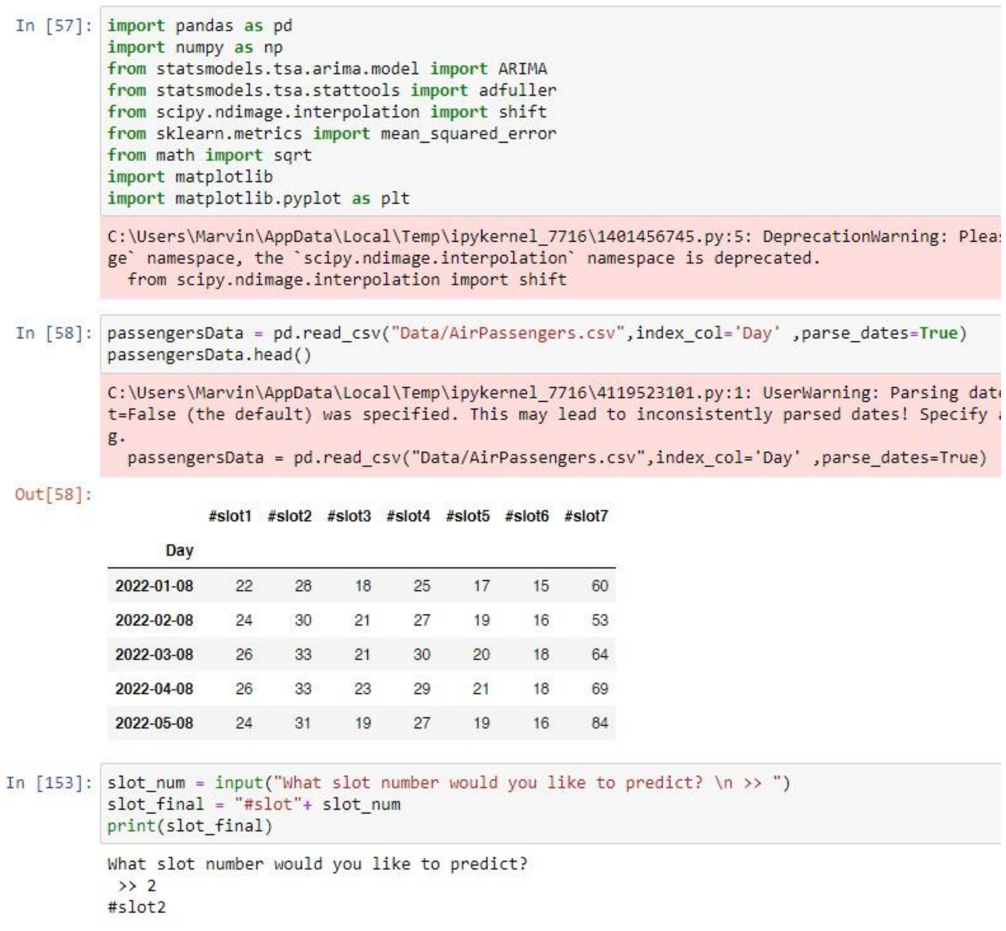


Fig. 42. Linear actuator moving back to its initial state.

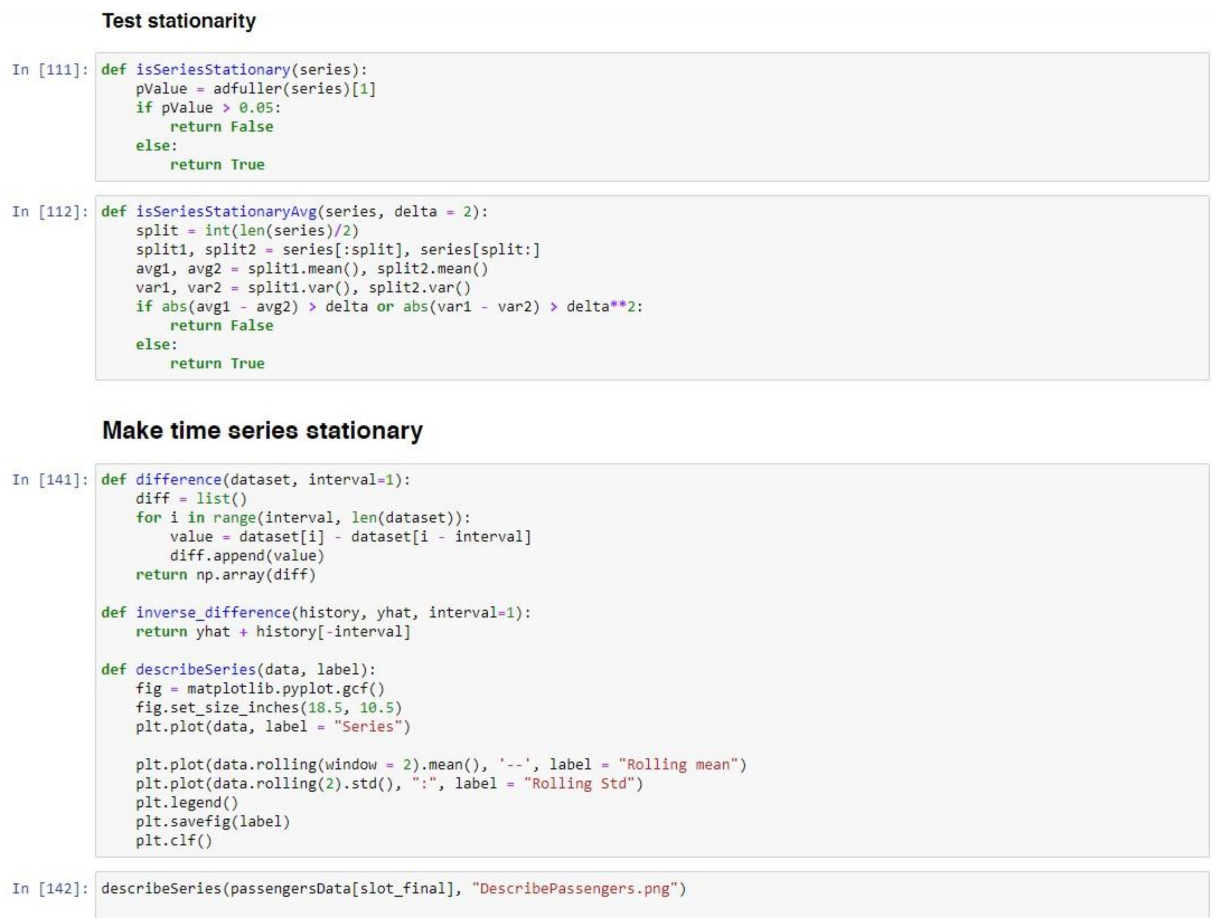


```
In [57]: import pandas as pd
         import numpy as np
         from statsmodels.tsa.arima.model import ARIMA
         from statsmodels.tsa.stattools import adfuller
         from scipy.ndimage.interpolation import shift
         from sklearn.metrics import mean_squared_error
         from math import sqrt
         import matplotlib
         import matplotlib.pyplot as plt

         C:\Users\Marvin\AppData\Local\Temp\ipykernel_7716\1401456745.py:5: DeprecationWarning: Plea
         ge` namespace, the `scipy.ndimage.interpolation` namespace is deprecated.
           from scipy.ndimage.interpolation import shift

In [58]: passengersData = pd.read_csv("Data/AirPassengers.csv",index_col='Day' ,parse_dates=True)
         passengersData.head()

         C:\Users\Marvin\AppData\Local\Temp\ipykernel_7716\4119523101.py:1: UserWarning: Parsing dat
         t=False (the default) was specified. This may lead to inconsistently parsed dates! Specify
         g.
           passengersData = pd.read_csv("Data/AirPassengers.csv",index_col='Day' ,parse_dates=True)

Out[58]:
```

Day	#slot1	#slot2	#slot3	#slot4	#slot5	#slot6	#slot7
2022-01-08	22	28	18	25	17	15	60
2022-02-08	24	30	21	27	19	16	53
2022-03-08	26	33	21	30	20	18	64
2022-04-08	26	33	23	29	21	18	69
2022-05-08	24	31	19	27	19	16	84

```
In [153]: slot_num = input("What slot number would you like to predict? \n >> ")
          slot_final = "#slot"+ slot_num
          print(slot_final)

          What slot number would you like to predict?
           >> 2
          #slot2
```

1. Once all of the data is gathered through the IoT application, let's now move on to data science and AI prediction using a time series model called ARIMA. First, let's call all of the necessary libraries for the program. Then, read the CSV file created by the prototype. After that, the program will ask the admin what slot number he/she wants to predict. For this example, I chose slot 2.



Test stationarity

```
In [111]: def isSeriesStationary(series):
              pValue = adfuller(series)[1]
              if pValue > 0.05:
                  return False
              else:
                  return True
```

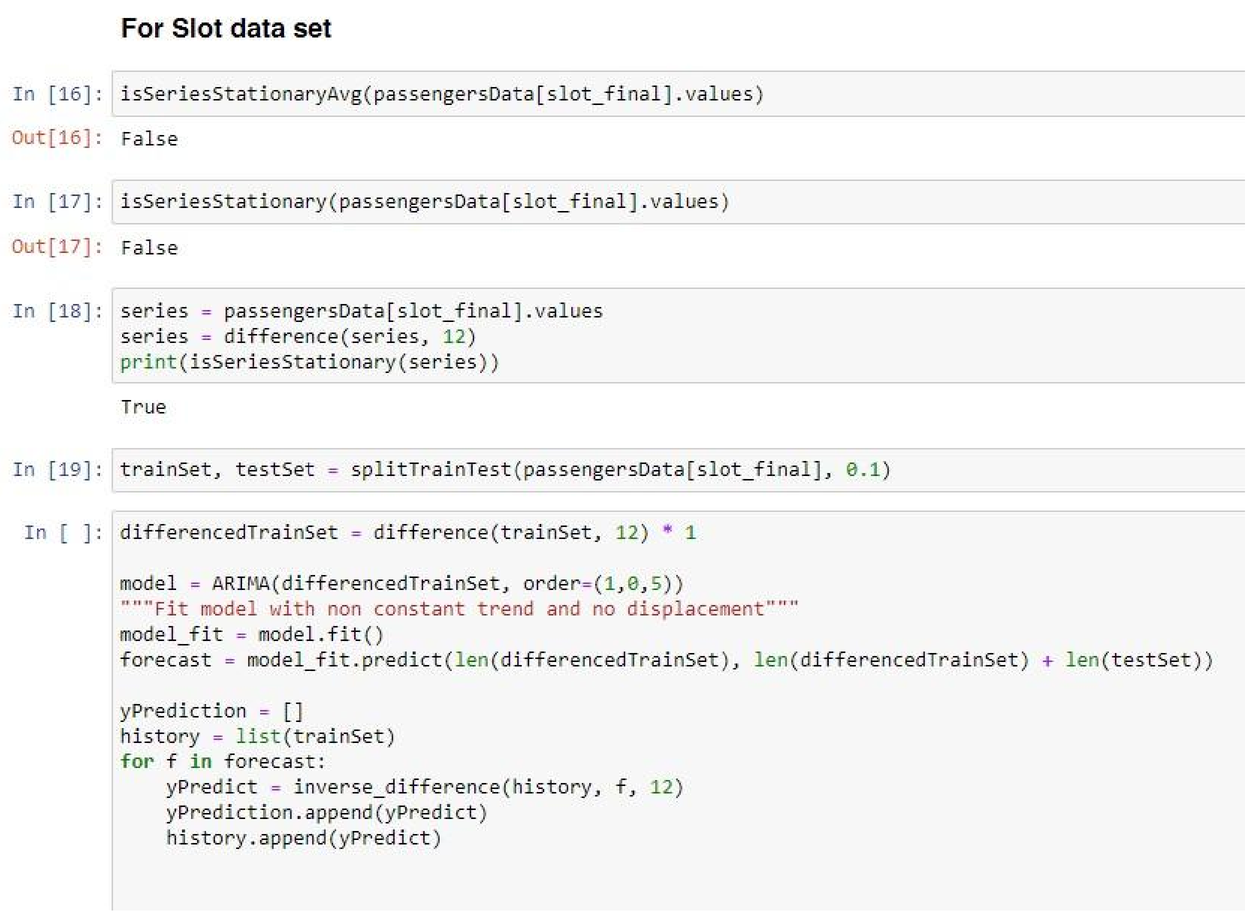
```
In [112]: def isSeriesStationaryAvg(series, delta = 2):
              split = int(len(series)/2)
              split1, split2 = series[:split], series[split:]
              avg1, avg2 = split1.mean(), split2.mean()
              var1, var2 = split1.var(), split2.var()
              if abs(avg1 - avg2) > delta or abs(var1 - var2) > delta**2:
                  return False
              else:
                  return True
```

Make time series stationary

```
In [141]: def difference(dataset, interval=1):
              diff = list()
              for i in range(interval, len(dataset)):
                  value = dataset[i] - dataset[i - interval]
                  diff.append(value)
              return np.array(diff)

          def inverse_difference(history, yhat, interval=1):
              return yhat + history[-interval]

          def describeSeries(data, label):
              fig = matplotlib.pyplot.gcf()
              fig.set_size_inches(18.5, 10.5)
              plt.plot(data, label = "Series")

              plt.plot(data.rolling(window = 2).mean(), '--', label = "Rolling mean")
              plt.plot(data.rolling(2).std(), ":", label = "Rolling Std")
              plt.legend()
              plt.savefig(label)
              plt.clf()
```

```
In [142]: describeSeries(passengersData[slot_final], "DescribePassengers.png")
```

2. In order to use the ARIMA model, there are certain criteria that the data must meet. The data must be stationary. Data can be called stationary when the mean and variance of the data are constant. Using the Dickey-Fuller test, we can have an insight if the data we have is stationary or not. In the image above, the group created different functions to test if the data is stationary or not. The group also created a function to make certain data stationary. For this, the function simply gets the difference between the data[i] to the succeeding data.

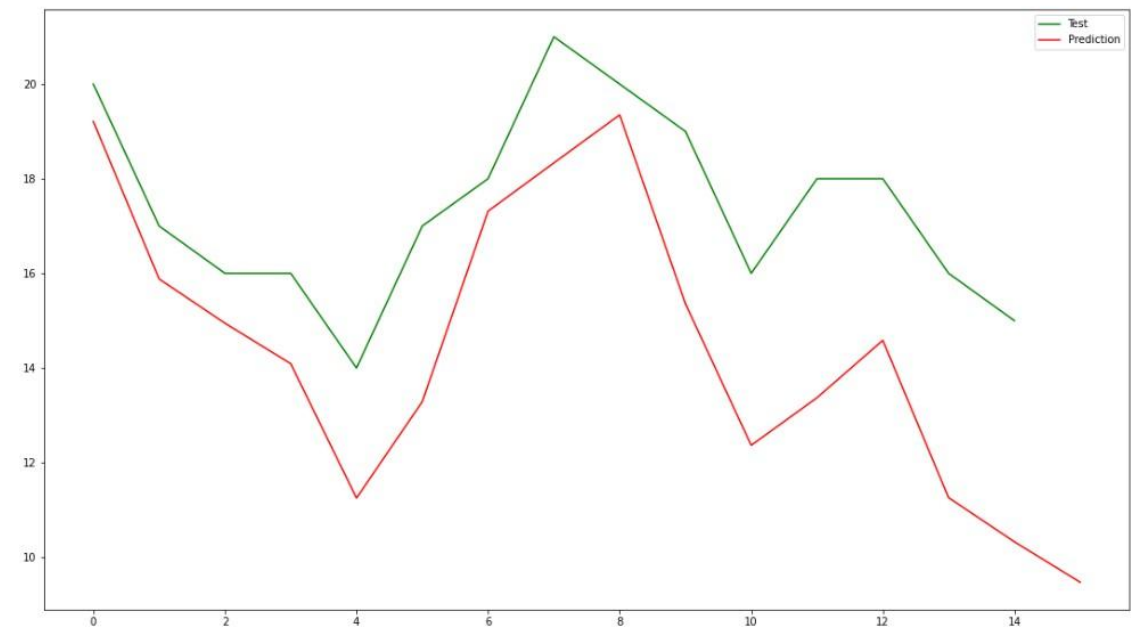


For Slot data set

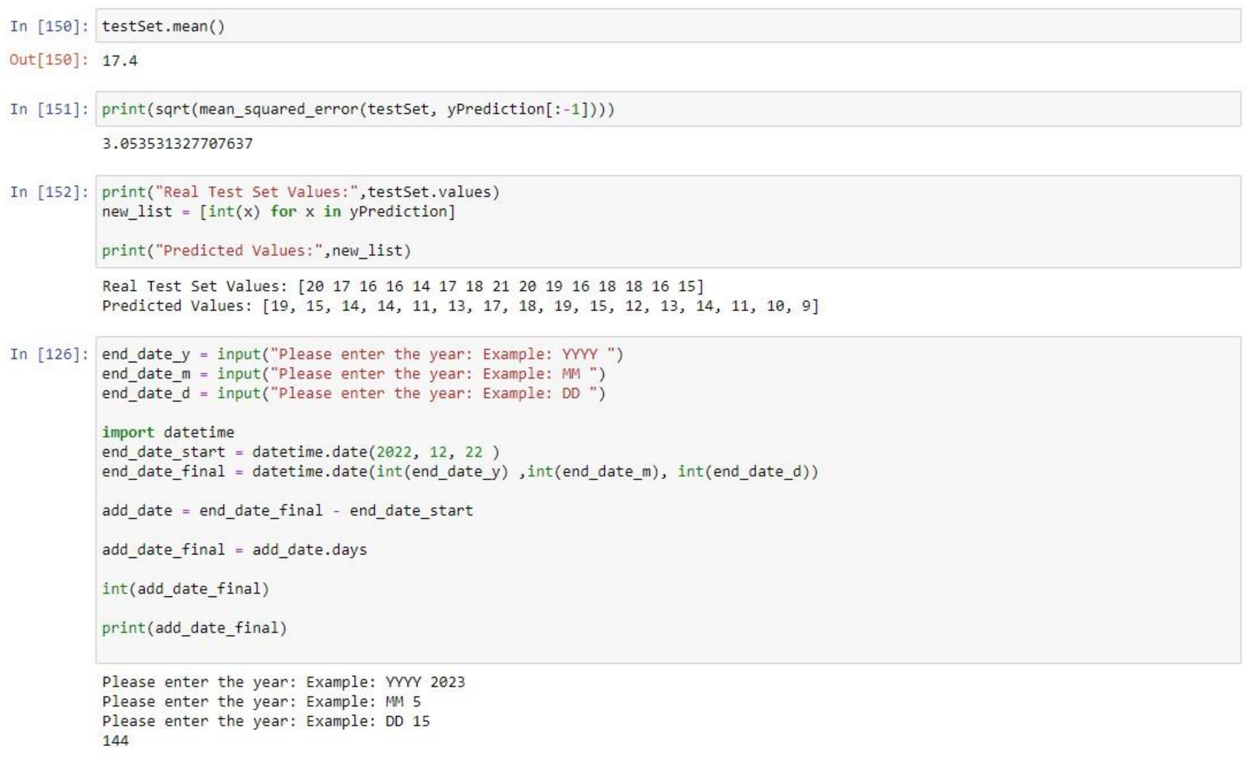
```
In [16]: isSeriesStationaryAvg(passengersData[slot_final].values)

Out[16]: False

In [17]: isSeriesStationary(passengersData[slot_final].values)

Out[17]: False

In [18]: series = passengersData[slot_final].values
         series = difference(series, 12)
         print(isSeriesStationary(series))

         True

In [19]: trainSet, testSet = splitTrainTest(passengersData[slot_final], 0.1)

In [ ]:  differencedTrainSet = difference(trainSet, 12) * 1

         model = ARIMA(differencedTrainSet, order=(1,0,5))
         """Fit model with non constant trend and no displacement"""
         model_fit = model.fit()
         forecast = model_fit.predict(len(differencedTrainSet), len(differencedTrainSet) + len(testSet))

         yPrediction = []
         history = list(trainSet)
         for f in forecast:
             yPredict = inverse_difference(history, f, 12)
             yPrediction.append(yPredict)
             history.append(yPredict)
```

3. The group created a function for splitting the data. This will help in order to test whether the created model is fitted to the dataset. Once all of the functions are ready, the slot 2 dataset is now inserted into the functions created. After transforming the data, the new data are trained using the ARIMA model. After training the model, the model is now tested to the test set in order to see if the model is fitted to slot 2.

```
In [149]: import matplotlib
          import matplotlib.pyplot as plt


          fig = matplotlib.pyplot.gcf()
          fig.set_size_inches(18.5, 10.5)
          plt.plot(testSet.values, color='green', label = "Test")
          plt.plot(yPrediction, color='red', label = "Prediction")
          plt.legend()
          plt.savefig("PassengerPrediction.png")
```

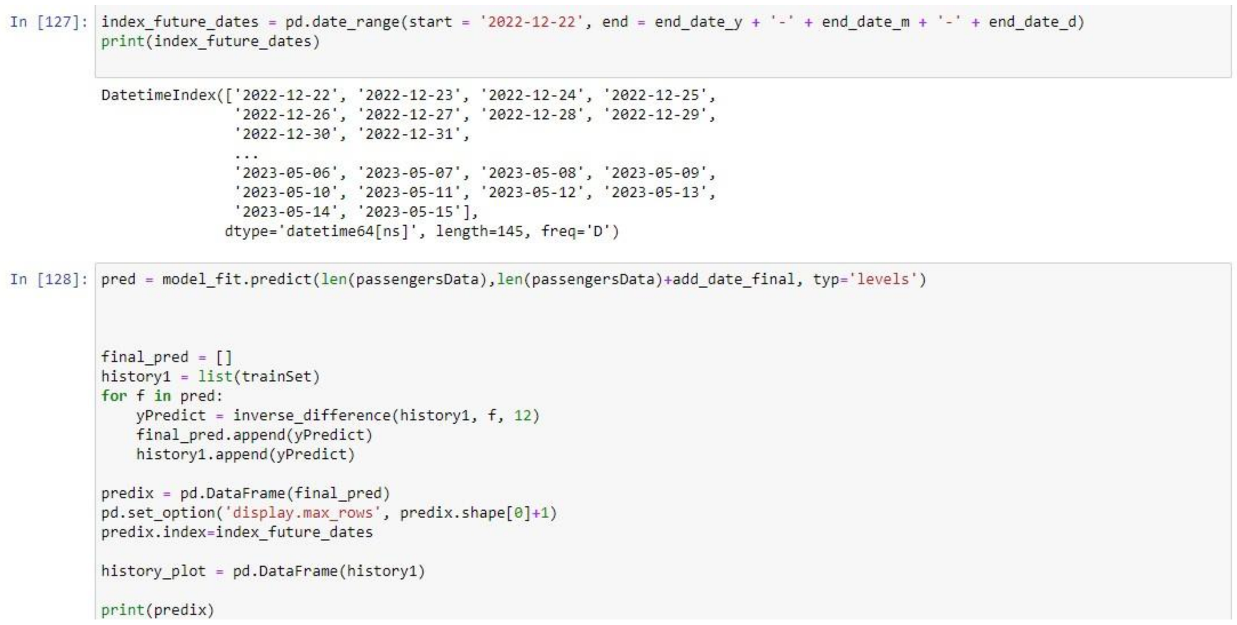


4. As shown in the image above, the model is fitted to the dataset or slot 2. The green line represents the true test set values while the red line is the predicted output of the model. Through observation, the prediction of the ARIMA model is very accurate. This model can now be used in order to predict future sales of the products.



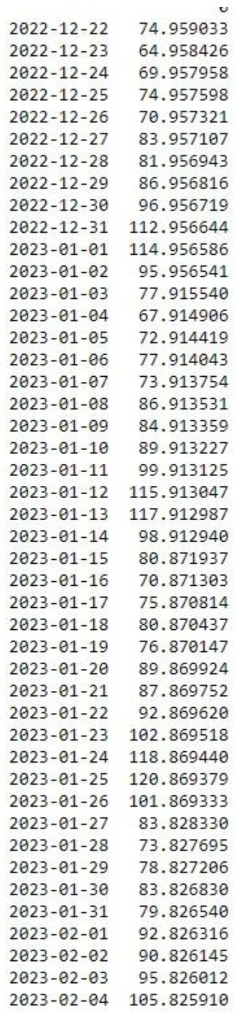
```
In [150]: testSet.mean()

Out[150]: 17.4

In [151]: print(sqrt(mean_squared_error(testSet, yPrediction[:-1])))

          3.053531327707637

In [152]: print("Real Test Set Values:",testSet.values)
          new_list = [int(x) for x in yPrediction]

          print("Predicted Values:",new_list)

          Real Test Set Values: [20 17 16 16 14 17 18 21 20 19 16 18 18 16 15]
          Predicted Values: [19, 15, 14, 14, 11, 13, 17, 18, 19, 15, 12, 13, 14, 11, 10, 9]

In [126]: end_date_y = input("Please enter the year: Example: YYYY ")
          end_date_m = input("Please enter the year: Example: MM ")
          end_date_d = input("Please enter the year: Example: DD ")

          import datetime
          end_date_start = datetime.date(2022, 12, 22 )
          end_date_final = datetime.date(int(end_date_y) ,int(end_date_m), int(end_date_d))

          add_date = end_date_final - end_date_start

          add_date_final = add_date.days

          int(add_date_final)

          print(add_date_final)

          Please enter the year: Example: YYYY 2023
          Please enter the year: Example: MM 5
          Please enter the year: Example: DD 15
          144
```

5. In box 152, you can see the real test set values and the predicted values of our ARIMA model. By looking at the data, you can see that the predicted values are very close to the real values. This means that the model we've created is fitted to the dataset given. Now that we have a good model for prediction, The admin can now predict the sales on future dates. In box 126, the program asks the user to enter the date of the sale he/she wants to predict. The output in the block of code is the number of days being inserted into the program. In this case, the user entered 05-15-2023.



```
In [127]: index_future_dates = pd.date_range(start = '2022-12-22', end = end_date_y + '-' + end_date_m + '-' + end_date_d)
          print(index_future_dates)

          DatetimeIndex(['2022-12-22', '2022-12-23', '2022-12-24', '2022-12-25',
                         '2022-12-26', '2022-12-27', '2022-12-28', '2022-12-29',
                         '2022-12-30', '2022-12-31',
                         ...
                         '2023-05-06', '2023-05-07', '2023-05-08', '2023-05-09',
                         '2023-05-10', '2023-05-11', '2023-05-12', '2023-05-13',
                         '2023-05-14', '2023-05-15'],
                        dtype='datetime64[ns]', length=145, freq='D')

In [128]: pred = model_fit.predict(len(passengersData),len(passengersData)+add_date_final, typ='levels')



          final_pred = []
          history1 = list(trainSet)
          for f in pred:
              yPredict = inverse_difference(history1, f, 12)
              final_pred.append(yPredict)
              history1.append(yPredict)

          predix = pd.DataFrame(final_pred)
          pd.set_option('display.max_rows', predix.shape[0]+1)
          predix.index=index_future_dates

          history_plot = pd.DataFrame(history1)

          print(predix)
```

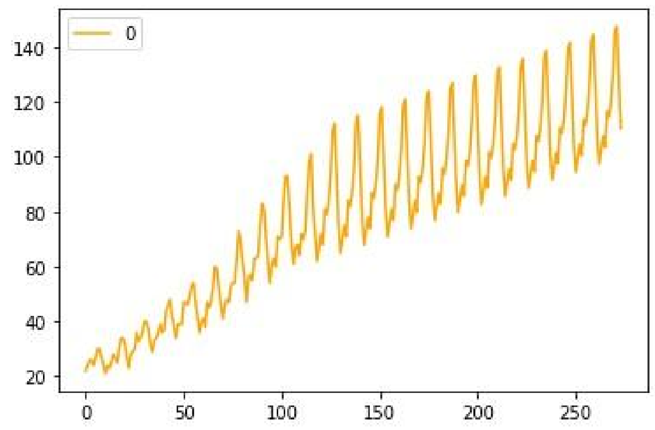
6. Once the user inputted the end date, the model will predict all of the sales starting from the end of the data set of slot 2, up to the last or end date of the user. All of the data is printed for the admin to see. To have more freedom in analyzing and observing patterns to the predicted data.



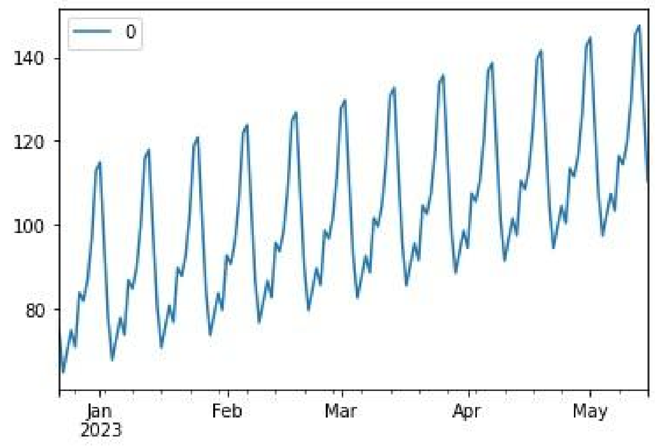
```
2022-12-22   74.959033
2022-12-23   64.958426
2022-12-24   69.957958
2022-12-25   74.957598
2022-12-26   70.957321
2022-12-27   83.957107
2022-12-28   81.956943
2022-12-29   86.956816
2022-12-30   96.956719
2022-12-31  112.956644
2023-01-01  114.956586
2023-01-02   95.956541
2023-01-03   77.915540
2023-01-04   67.914906
2023-01-05   72.914419
2023-01-06   77.914043
2023-01-07   73.913754
2023-01-08   86.913531
2023-01-09   84.913359
2023-01-10   89.913227
2023-01-11   99.913125
2023-01-12  115.913047
2023-01-13  117.912987
2023-01-14   98.912940
2023-01-15   80.871937
2023-01-16   70.871303
2023-01-17   75.870814
2023-01-18   80.870437
2023-01-19   76.870147
2023-01-20   89.869924
2023-01-21   87.869752
2023-01-22   92.869620
2023-01-23  102.869518
2023-01-24  118.869440
2023-01-25  120.869379
2023-01-26  101.869333
2023-01-27   83.828330
2023-01-28   73.827695
2023-01-29   78.827206
2023-01-30   83.826830
2023-01-31   79.826540
2023-02-01   92.826316
2023-02-02   90.826145
2023-02-03   95.826012
2023-02-04  105.825910
```

The image on the left is a portion of the predicted values obtained from the ARIMA model. The prediction starts from the end date of the data set slot 2 up to the inputted end date of the admin.

```
In [129]: history_plot.plot(color="orange")

Out[129]: <AxesSubplot: >
```



```
In [130]: predix.plot()

Out[130]: <AxesSubplot: >
```



8. To further analyze the data gathered from the prediction, here are some line graphs for the whole dataset including the predicted values of the ARIMA model. As shown in the graph, the line has an upward trend meaning the sales of slot 2 is increasing with respect to time. we can also see some patterns in the graph. Overall, the model created using ARIMA is very effective in predicting future values of sales in the project. This can benefit store owners and admins in deciding what products should they buy stocks more.

Project Expected Softcopy and Hardcopy OUTPUTS:

(Illustrate, describe and explain the OUTPUTS produced by the project. Show how this is used by the end users and how it will benefit them. Always support your answers with the proper information and data. Show the actual OUTPUTS)

Hardware:

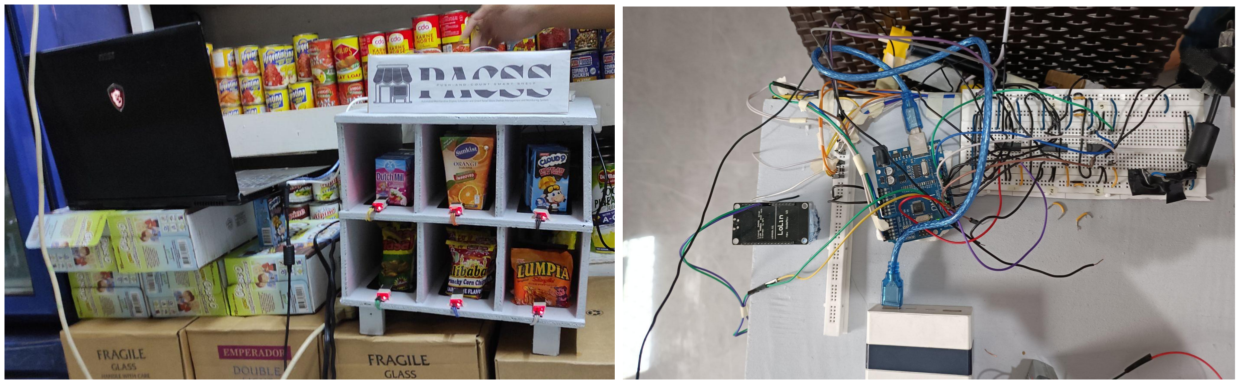


Fig. 43. Hardware components of the system

The system's final hardware output is shown in Figure 42. The team-created shelf is topped with the components. The DC motors were attached to a 19V power supply, while the wifi module and Arduino Uno were connected to a power bank. To replicate the shelves in grocery stores and supermarkets, three miniature sample product pieces were put in each shelf space.

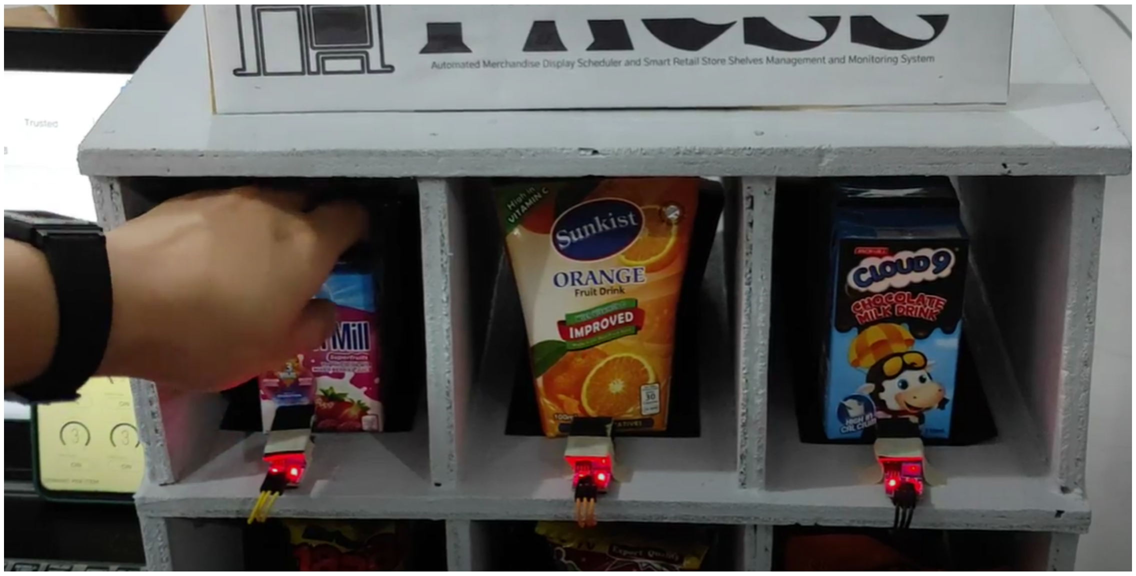

Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System

Fig. 44. A customer getting a product from the shelf

Figure 43 in the illustration above depicts a consumer taking something from slot 1. The linear actuator will automatically push the remaining items from behind once the IR Sensors have detected that the first item from the slot has been removed.

Software:

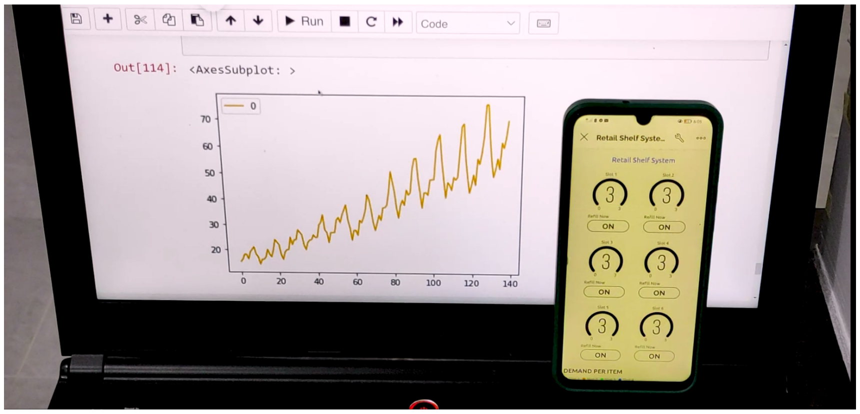


Fig. 45. Mobile Blynk dashboard and ARIMA model output for PACSS

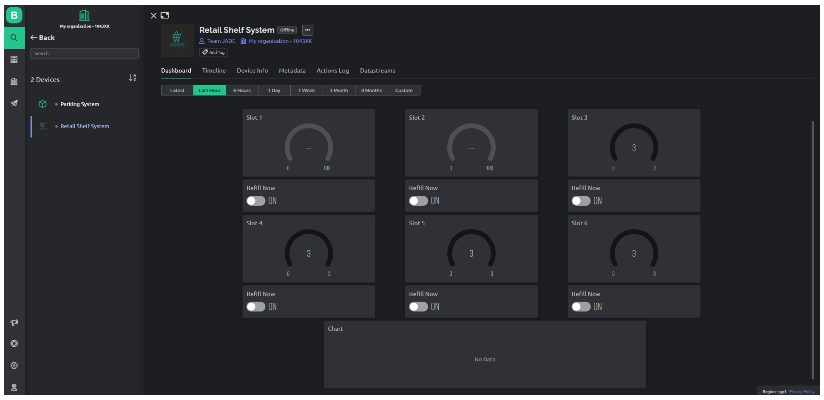


Fig. 46. The Blynk IoT application web dashboard

An actual representation of the output from the system's software is shown in Figure 44. The mobile application for Blynk serves as an example of the user's point of view. On the screen, it shows how many goods are still available on the shelves. There is also a button that can be used to initiate refills. The linear actuators can be instructed to return to their starting positions by clicking here. However, the graph on the right shows the results of running the sales prediction algorithm. The output graph displays the anticipated quantity of goods to be sold over a specific period of time.

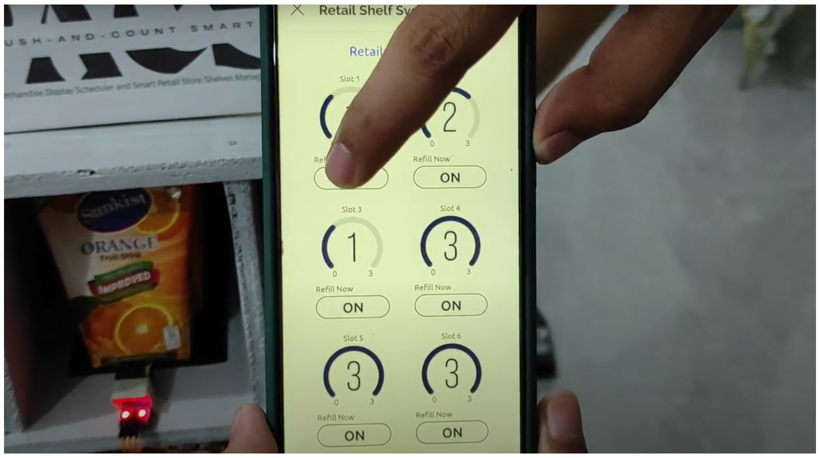
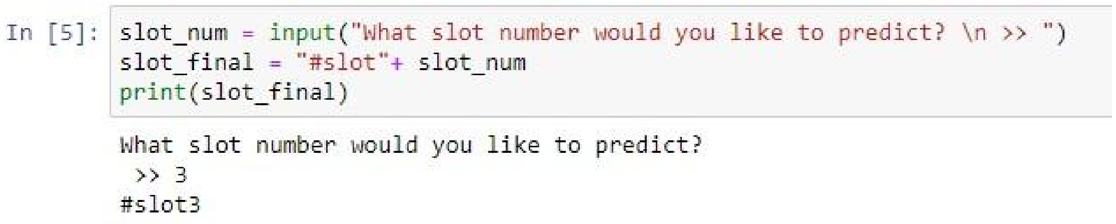


Fig. 47. The admin clicked the button just in time for restocking slot 1

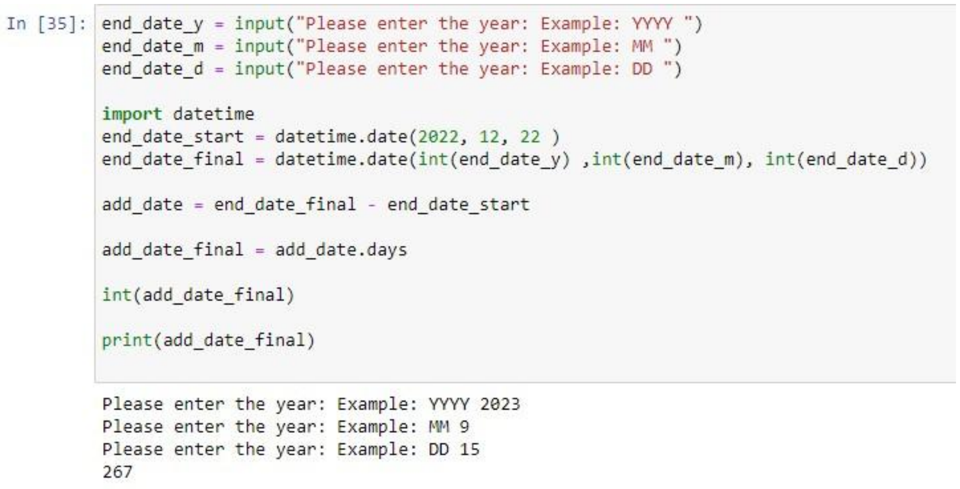
The admin can use the Blynk web application's button to refill the slot once they've made that decision. To make room for replenishment, the linear actuator will reverse.



```
In [5]: slot_num = input("What slot number would you like to predict? \n >> ")
        slot_final = "#slot"+ slot_num
        print(slot_final)

        What slot number would you like to predict?
         >> 3
        #slot3
```

Fig. 48. Screenshot of simulation in Jupyter Notebook

The aforementioned graphic depicts how user input was used in the Jupyter Notebook simulation. The system then prompted the user to enter the slot number they wanted the machine to forecast.



```
In [35]: end_date_y = input("Please enter the year: Example: YYYY ")
         end_date_m = input("Please enter the year: Example: MM ")
         end_date_d = input("Please enter the year: Example: DD ")

         import datetime
         end_date_start = datetime.date(2022, 12, 22 )
         end_date_final = datetime.date(int(end_date_y) ,int(end_date_m), int(end_date_d))

         add_date = end_date_final - end_date_start

         add_date_final = add_date.days

         int(add_date_final)

         print(add_date_final)

         Please enter the year: Example: YYYY 2023
         Please enter the year: Example: MM 9
         Please enter the year: Example: DD 15
         267
```

Fig. 49. User input of the admin to select a date

The simulation of day counting from the user-inputted date to the most recent date recorded in the database is displayed above. The user entered the date 2023, 9, 15, or September 9, 2023, which means that there are 267 days between the two given dates. As seen in the code, the start date is 2022, 12, 22, or December 22, 2022.

Overall, the end-users of the Automated Merchandise Display Scheduler and Smart Retail Store Shelves Management and Monitoring System may experience a much tolerable amount of workload because technological advancements will do most of the repetitive manual activities for them. By incorporating the currently emerging technologies, they can also improve and maximize the sales of their items and dodge possible challenges that may come their way.

References

[1] “Why Small Retailers Struggle To Manage Their Shelf Space Efficiently,” *www.dotactiv.com*. https://www.dotactiv.com/blog/small-retailers-shelf-space/

[2]J. Stoltzfus, “What is the Internet of Things (IoT)? - Definition from Techopedia,” *Techopedia.com*, Nov. 27, 2020. https://www.techopedia.com/definition/28247/internet-of-things-iot (accessed Jan. 04, 2023).

[3]“1. What is Blynk?,” *Tech Explorations*. https://techexplorations.com/guides/blynk/1-what-is-blynk/ (accessed Jan. 04, 2023).

[4]IBM, “What is Data Science? | IBM,” *www.ibm.com*. https://www.ibm.com/topics/data-science (accessed Jan. 04, 2023).

[5]Idaho National Laboratory, “Artificial Intelligence and Machine Learning,” *INL*. https://inl.gov/artificial-intelligence/ (accessed Jan. 04, 2023).

[6] Statista Research Department, “Topic: Food retail industry in the Philippines,” *Statista*, 24-Oct-2022. [Online]. Available: https://www.statista.com/topics/5798/fmcg-retail-industry-in-the-philippines/#dossierKeyfigures. [Accessed: 30-Dec-2022].