Palestine Polytechnic University
College of Engineering

GRADUATION PROJECT

Automated Drugs Dispenser Machine (ADM)

By:

Mu'min Hussein Ta'amra
Obayda Issam Khalid

Mechatronics Engineering

Supervisor:

Eng. Abdalkarim Almohtasib

December, 2017

Abstract

One of the most important pillars of community development is to improve the healthcare section. This includes facilitating the access to medications and drugs for all people and over the time.

However, many people still have many difficulties in obtaining their medications and drugs. Especially in rural places and at late night times or in Fridays.

The Automated Drugs Dispenser Machine (ADM) is a new machine which allows patients to get their medications easily 24 hours a day, 7 days a week and in all places.

The ADM has new characteristics that differ from other similar machines. This includes the working mechanisms, compatibility and the building costs. This machine consists of a combination of mechanical and electrical parts that work with each other harmonically to do the dispensing process efficiently and as required.

The design of the ADM is required to achieve a high level of safety and accuracy in dispensing drugs. So, this machine will support the barcode reading feature in order to verify each medicine type before dispensing it. This process also requires a prescription and a special card for each patient to be able to use this machine.

This machine will enhance the pharmacy operations and improve the healthcare section in the community. So that all people could get their medications easily at any time and wherever they live.

الملخص

إن من أهم ركائز قيام وتطور المجتمعات هو تطوير جانب الرعاية الصحية في كل مجتمع، هذا يتضمن تسهيل عملية وصول المرضى للأدوية والعلاجات المطلوبة على مدار الوقت وفي كل أماكن سكنهم.

على الرغم من التطور الهائل الذي نعيشه في زماننا هذا، إلا أنه لا يزال العديد من الناس يواجهون صعوبات في الحصول على أدويتهم بعد وصفها من قبل الطبيب ؛ وذلك في بعض الأماكن الريفية النائية التي تفتقر لوجود صيدليات فيها، وكذلك في ساعات الليل المتأخرة وفي أيام العطل مثل يوم الجمعة حيث تكون معظم الصيدليات مغلقة.

من هنا ظهرت الحاجة لوجود ماكنة لتزويد الأدوية بشكل آلي، حيث أن هذه الماكنة تتيح لجميع المرضى الحصول على أدويتهم على مدار الساعة وطيلة أيام الأسبوع وفي جميع مناطق تواجدهم.

هذه الماكنة تختلف عن مثيلاتها المتوافرة في السوق المحلي والعالمي. حيث أنها تدعم مواصفات جديدة مختلفة كليا على صعيد آلية عملها، ومدى توافق هذه الماكنة مع مختلف أنواع وأحجام علب الأدوية، وكذلك فإن تكلفة إنشاء هذه الماكنة أقل بكثير من سابقاتها.

إن وجود ماكنة كهذه تتعامل مع الأدوية يتطلب أن تكون على درجة عالية من الأمان أثناء صرف الدواء. حيث أن تصميم هذه الماكنة يدعم خاصية قراءة الرقم التسلسلي لكل علبة دواء قبل صرفها وذلك للتأكد من نوعها ومطابقتها للدواء المطلوب. وكذلك فإن هذه الماكنة تتطلب وجود بطاقة خاصة لكل مريض تحتوي على وصفة طبية للدواء المطلوب قبل صرفه للمستخدم.

هذه الماكنة ستدخل تحسينات ملموسة على عمل الصيدليات وستقوم بتطوير جانب الرعاية الصحية في المجتمع ؛ حيث يستطيع الجميع الوصول لأدويتهم في كل الأوقات وفي مختلف مناطق تواجدهم.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Healthcare is one of the top priorities of societies around the world. Governments strive to develop this area using all new methods and technologies.

One of the most important aspects of the healthcare is to get medications and drugs after diagnosis and prescriptions. This is the reason for the presence, expansion and development of the pharmacies worldwide.

However, many people still have many difficulties in obtaining their medications and drugs. Especially for those who live in rural places where the number of pharmacies is limited. And also, many people can't find open pharmacy at late night times or in Fridays!

Because of this, there was a need to create the Automated Drugs Dispenser Machine (ADM) which allows patients to get their urgent care and emergency prescriptions filled after clinic hours and in all places. This machine is similar to the ATM machines in Banks, but it dispenses drugs instead of money. Moreover, this machine will enhance pharmacy operations and increase productivity. Also, it could reduce working hours in pharmacies.

There are some machines similar to the ADM idea, but the ADM was designed to differ from other machines in many properties. For example, it works in a new different mechanism. And also, the ADM is compatible with all types, shapes and sizes of drugs. However, the building cost of the ADM is much less

than the other similar machines.

The design of the ADM is consist of the frame layout, Mechanical parts, electrical parts and an user interface combined to each other in order to achieve the main objectives of the machine.

The outer frame of the machine contains the user interface (touch screen), the inlet of cards reader, papers outlet and the outlet of the drugs.

The mechanical parts consist mainly from the moving frame in X-Y plane, and some mechanisms to take out the required drugs. The electrical parts like the motors, cards reader, barcode reader and other devices are all connected with the microcontroller which is considered to be the heart of the machine because it controls all the parts to do the dispensing process harmonically.

The ADM machine has been belt successfully, and there is a high correlation between its specifications and the objectives that have been set.

The first consideration that has been taken in the ADM design is to achieve a high level of safety. So, it requires a prescription for the prescribed medicines. This prescription would be written on a RFID card instead of prescription papers.

In addition, the ADM is very accurate in choosing the required medicine using the barcode reader to verify the medicine code and if it matches the required medicine or not.

The ADM was designed to be friendly to use. The user is just needed to insert his own card and password and to wait a little while his medication is ready to take.

The existence of the ADM is very important to improve the healthcare section in all societies, and especially in the Palestinian community. So that all people could get their medications easily at any time and wherever they live.

In this chapter, a brief description of the ADM project will be shown. starting with goals and objectives, recognition of the need and finally a short review

of the related works.

1.1 Goals and objectives

The Goal of This Project:

To design a machine that will provide drugs and medicines for patients automatically 24 hours a day, 7 days a week. This will make the access to drugs easier for all people in all the time.

Objectives:

in order to reach the main goal of this project , there are many objectives to achieve, and the machine must have these properties :

- Reliability: the machine should be reliable enough, and so it will deal with two kinds of medicines ... prescribed medicines (that need a doctor prescription), and over the count - non prescribed medicines (like headache, fever, cold, B.P drugs...).
- Capacity: this machine will provide the most needed medicines and general drugs about (50 type).
- Accessibility: this machine will be placed in public places like hospitals, clinics, health centers, rural places and outside pharmacies.
- Safety: because drugs are very dangerous to deal with, and any mistake in giving medications will cause many problems; this machine must be safe enough, so it will use Barcode reader to verify the type of drugs.
- Informative: the machine must do the pharmacist job and give the patient some information about the drugs that he requested.
- Friendly user interface: the machine will have a screen to deal with.

1.2 Recognition of the Need

There are many problems with the present prescription drug providing system in pharmacies. Because of this, there was a need to this machine to solve some of these problems.

In order to study the need of improving drug dispensing machine, the project team of ADM made a questionnaire (Appendices 1, 2 and 3) about this project. Here are some problems conclused from the questionnaire:

About 66% of the study sample people do not find opened and nearby pharmacy at late night time or in Fridays. The ADM machine will solve this problem by providing drugs for patient 24 hour a day and 7 days a week. It allows patients to get their urgent care and emergency prescriptions filled after clinic hours, when local pharmacies are closed. They can have those prescriptions filled at the medical center or the close street.

Moreover, about 20% of the study sample has difficulties in obtaining the medicine after doctor prescription.

ADM machine will make it easy for all people to access their medications in rural places.

In the other hand, about 87% of the pharmacists study sample need in many times to leave the pharmacy but they can't. And about 63% works alone in the same shift.

Using ADM machine can enhance pharmacy operations. For examples, dropping of one full-time technician, reallocation of pharmacists' time and an increase in productivity and operational flexibility

However, 87% of pharmacists face many difficulties in opening new pharmacies because of the government limitations. This machine may take place of opening new pharmacies in some far places.

Many times if the handwritten prescription is difficult to read, the pharmacist will need to call the doctor to confirm the prescription, this will take long

time, and cause some difficulties for patients.

This project will replace handwritten prescription by a magnetic card, and so it will be more reliable to use.

1.3 Literature Review

A vending machine is a machine that dispenses different items to customers automatically, after they pay using credit cards or coins [1].

The inventor of the earliest vending machine was the first-century AD Greek engineer and mathematician "Hero of Alexandria", and the machine was for water [2].

In the early 1880s, the first modern vending machine was built in London, England. And it dispensed postcards [1].

In the United States, the first machine was developed in 1888 by "Thomas Adams Gum Company"; it was selling gum on New York [3].

Since that time, vending machines were developed to dispense many different types of goods.

A special type of vending machines was developed to dispense drugs and medicines for patient automatically. And here are some different types of these machines:

1. 2015 new arrive modern medicine vending machine [4]:

    - Place of origin : Guangdong China (Mainland).
    - It can fit different sizes of goods, the best selling can sold in many aisles.
    - Through First-In-First-Out to ensure quality.
    - Capacity: 12 trays * 8 layers , 96 kinds of products (288 1150 psc) (As shown in figure 7.3(a)).

- CY: Helical spring drawer.
- Price: 3666 $.

Figure 1.1: 2015 new arrive modern medicine vending machine [4].

2. Medicine vending machines that dispense prescriptions 24 hours a day [5]:

   - Place of origin: UK.
   - It has been developed by the Canadian firm PharmaTrust.
   - The dispensers come in two models, a smaller one holding 330 packs and a larger one with 2,000.
   - Users insert their prescription into the machine and pick up a telephone to access a live video link to a registered pharmacist. (The machine is shown in figure 7.3(b)).
   - The pharmacist will check the prescription and ensure those who have to pay have done so before allowing the machine to dispense the drugs.
   - The prescription can be paid by inserting cash or a debit / credit card.

- An information sheet is printed out, telling out the patient how often the drugs should be taken.
- Price : costs around £50,000.

Figure 1.2: PharmaTrust medicine machine [5].

3. Automatic Prescription Drug Dispenser (InstyMeds Corporation) [6]:
   - Inventor: Ken Rosenblum, Mendota Heights, MN(US).
   - Including aremote dispenser, a prescription entry system, and a communications network.
   - The remote dispenser transmits and receives information from the communications network and dispenses prescription drugs to the patient [7].
   - This machine provide two types of drugs : prescription drug.. and over-the-counter products.
   - Payment system: debit or credit card.
   - The unit holds about 100 different drugs, 45 of those drugs represent 80% of what is usually prescribed by doctors . (the machine is shown in figure 7.3(c))

Figure 1.3: Insty Meds Prescription Drug Dispenser [6].

4. Any Time Medicine Vending Machine [8]:

   - College : Don Bosco Institute Of Technology, Bengaluru.
   - Medicines provided by the machine are only for the timely relief and in emergency case.
   - The Prototype vend out medicines that does not need doctor prescription. (Look at the prototype project – figure 7.3(d)).

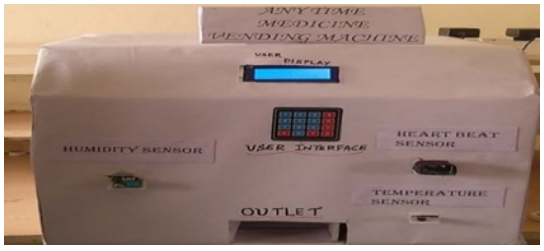


Figure 1.4: Any Time Medicine Vending Machine [8].

5. Drug dispensing system [9]: An automated drug dispensing system includes a cabinet adapted to store a variety of prepackaged pharmaceuticals in a plurality of bins for filling patient prescriptions. Each bin stores a particular variety of packaged multiple-dose pharmaceutical. Each variety of pharmaceutical is associated with a particular code. A controller

receives request signals and in response generates dispense signals. (as shown in figure 7.3(e))

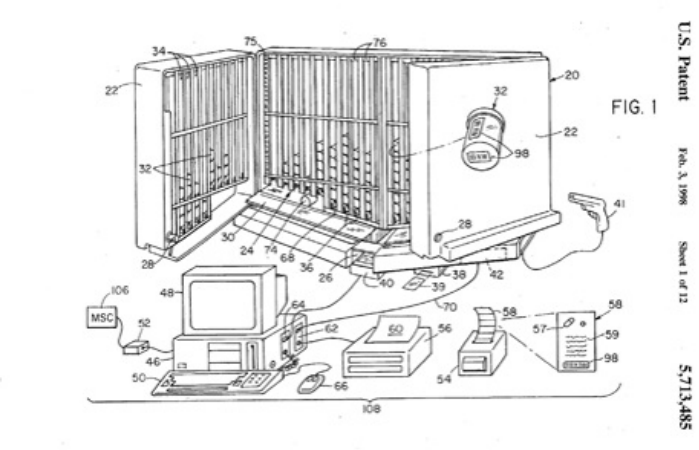


Figure 1.5: Drug Dispensing System [9].

1.3.1 What is New in the ADM?

The ADM machine has many new characteristics differ from other machines as follow:

1. It works in a new mechanism based on the gravity force to pull the packets instead of the Helical spring drawer which used in other vending machines.

2. The ADM is compatible with all types, sizes and shapes of drugs packets. whereas some other machines deal with specific sizes or shapes of drugs.

3. The cost of the ADM is much less than other available machines because it will use fewer motors instead of one motor for each type of medicine as in other machines.

Chapter 2

Conceptual Design

This chapter will review the design of the ADM , parts and components and their functions. Finally, a detailed explanation of the work method will be discussed.

2.1 Parts and Components:

This section will show all parts and components of this machine using the SolidWorks software to make the 3d design and a brief explanation about some parts in the machine.

1. Outer frame (as shown in figure 2.1).
2. Printed papers outlet: to give the user some information about the drug he requested.
3. Drugs outlet: the drugs would be taken out from this outlet.
4. Machine screen: this screen will be used to make interface between the user and the machine (enter his password and verify his order).
5. Magnetic card reader: the user should put his own card in this inlet, so the machine could read the medical prescription.

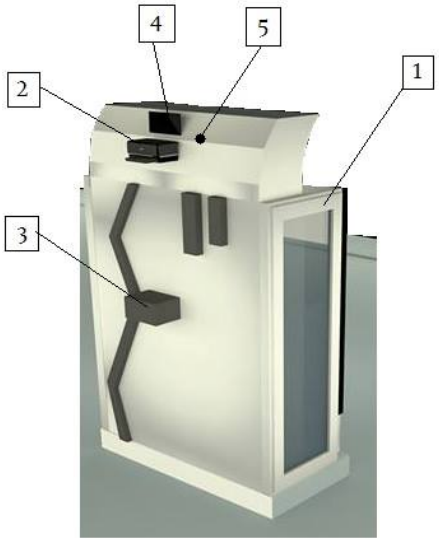


Figure 2.1: Outer Frame of The Machine.

The internal components of the machine are shown in figure 2.2:

6. Storage cells for drugs: the drugs are arranged in these cells each one in the appropriate cell due to its size.

7. X-Y coordinate frame: this is used to allow the head moving in two directions.

8. Stand frame: consists from beams to constraint whole cells inside the machine.

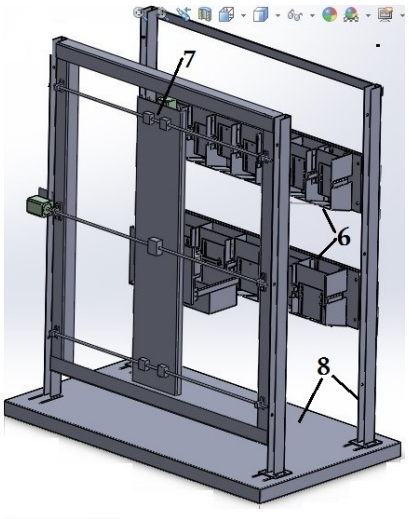


Figure 2.2: The machine from inside.

9. Stand beams are shown in figure 2.3.

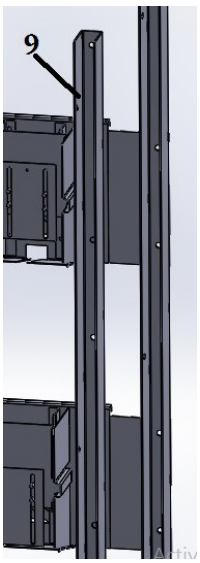


Figure 2.3: The stand frame.

10. Row of storage cells: consists mainly from 7 cells.

11. One storage cell: each cell's size could be adjustable.

12. Cell's outlet gate: from here the drug is taken out from the cell.

13. Inlet for pecking out- mechanism (as shown in figure 2.4).

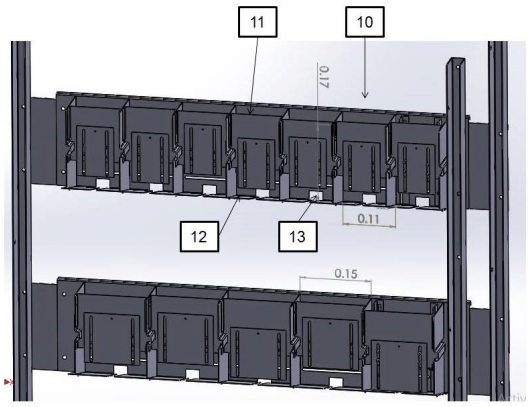


Figure 2.4: Row of storage cells.

The X-Y frame is shown in figure 2.5:

14. Connecting beams for X-Y coordinate frame.

15. Sliding head.

16. Slider rod

17. Power screw

18. Stepper Motor 1.

19. Vertical sliders.

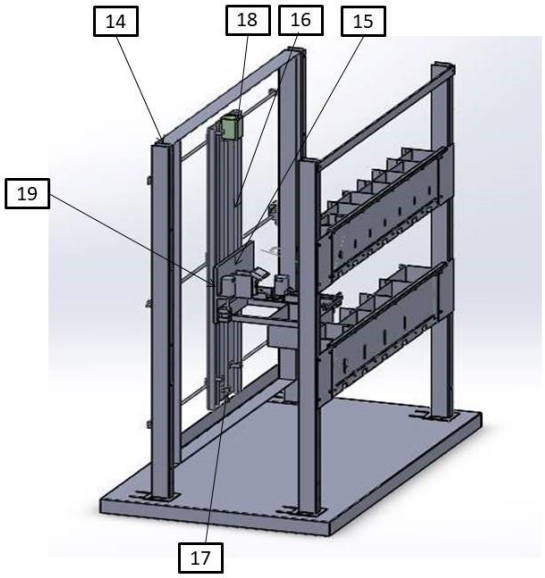


Figure 2.5: X-Y coordinate frame.

20. Empty container for carrying drugs.

21. Barcode reader.

22. Stepper Motor 2.

23. Pecking drugs out- mechanism (Rack and pinion mechanism).

24. DC Motor 1.

25. DC Motor 2.

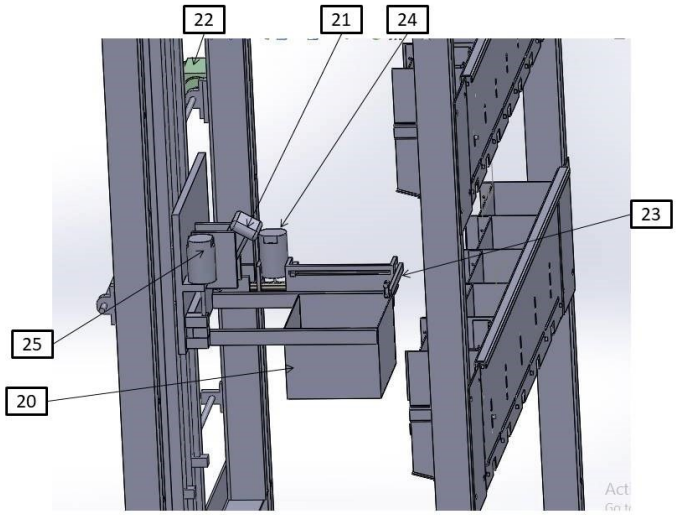


Figure 2.6: X-Y coordinate frame.

Pecking drugs out mechanism is shown in figure 2.7:

26. Roller head

27. Pulling arm

28. Rack and pinion

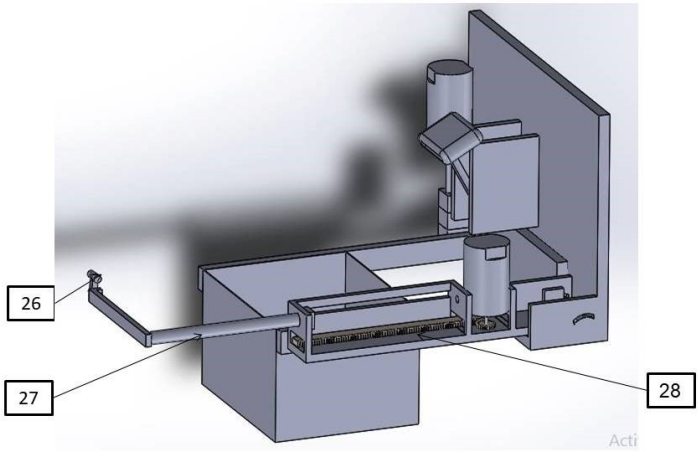


Figure 2.7: Pecking out mechanism.

When the motor rotate the Rack and pinion move the linear motiom, then the mechanism pecks out one packet of the medicine as shown below in figure 2.8:

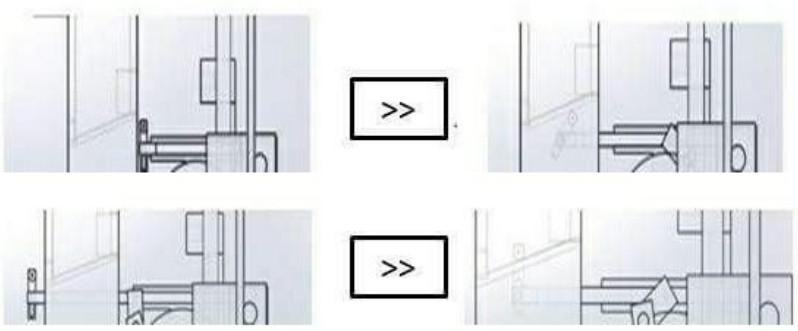


Figure 2.8: Pecking out one item.

29. Container's mechanism: this mechanism is used to rotate the container 180 degrees to deliver the drug to the user as shown in figure 2.9.

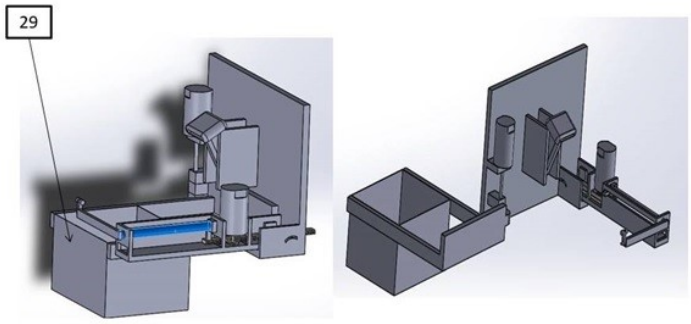


Figure 2.9: Container's gear mechanism.

2.2 How It Works:

This section will review a detailed explanation for the ADM operations; starting with the filling process , dispensing process , internal process and finally the payment system in this machine.

2.2.1 Filling Process:

When the pharmacist needs to fill the machine with some kinds of medicine, these steps should be followed:

Firstly, the pharmacist has to enter his own magnetic card, and to enter password using the screen and some input keys.

Then, for each type of medicine he will enter the cell's address that he would store in it. Then the Barcode reader should be directed to the code of the drug to store it. This process will be repeated for all kinds.

After that the pharmacist is required to enter some information about each type of drugs, that contains way of use, its price, and if it needs doctor prescription or not.

Now, the machine is ready to dispense drugs for patients.

Note: the storage cells are designed to accommodate all sizes of drugs, and the pharmacists can adjust cell's width, tall and height so it will fit the desired drug size.

Some mechanisms are used to control the size of the cells, as shown in the figure 2.10:

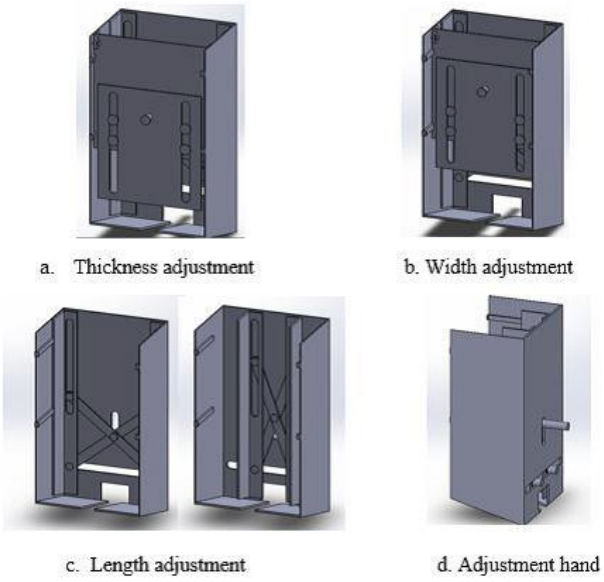


Figure 2.10: One cell adjustable size.

2.2.2 Dispensing Process:

When the patient needs to get his drug, he should do these steps: In the first, user will insert his own magnetic card in the reader's slot, and then enter his password using the screen and some input keys.

Then, the machine will read the requested drug's code from the magnetic card, and will do some internal process to dispense the drug.

This process will take few minutes, and then the user can take his drugs from the drugs outlet. After that, the machine will print a small paper containing some information about the patient, his doctor and the requested drugs.

2.2.3 Internal Process:

In order to dispense the requested drugs, the machine will do these steps internally:

When the user enters his own magnetic card in the cards reader, the reader sends signal containing needed medicine code to the control unit, and then the control unit will check the code and if the medicine is exist in the machine or not.

If the medicine is exist, the control unit sends signals to the stepper motors (1, 2) to rotate the lead screws with the appropriate number of rotations. The motors will continue to rotate until the sliding head is in the needed position (opposite to the cell that contains the required medicine).

After that, the barcode reader is activated to read the code from the drug's packet. The controller then checks if it is the right code. If it is the needed medicine then the controller will activate the motor that is connected to the pecking out mechanism to rotate. As a result, the mechanism then pecks one packet of the medicine.

The packet then falls in the empty container, and then the controller send signal to the X-Y coordinates motors (1, 2) to move the head to the drugs

outlet. At the moment that the sliding head is opposite to the outlet, the controller will send signal to the motor (3) to rotate the container 180 degrees to become near the outlet gate.

If the user takes his drug from the container, the controller will activate the printer to print a small paper about patient, doctor and medicine. And finally the magnetic cards reader/writer will erase the taken drug's code from the users card.

2.2.4 Payment system:

In this machine, the magnetic card is used also in the payment system. When the doctor writes his prescription on the magnetic card, the patient is required to pay the price of his drugs to the doctor. Then the doctor writes on the card that the requested drugs are paid. To do this, the doctor should have a financial treatment with the machine's owner (pharmacy or any supporter to this machine).

After that, when the patient put his card on the machine, the machine checks if the requested drugs are paid or not. If yes, then the machine completes dispensing process. If not, it rejects his order.

However, this card can be charged from the owner's pharmacy or any place that deals with this pharmacy. Then the magnetic card will record the patient's balance of money.

Now, the patient can buy his medications easily from this machine just if he has an enough balance in his card. If he requests a drug that does not need a prescription, then the machine checks his balance and dispenses his request. If the drug is prescribed only, then the machine dispenses it just if there is a prescription and enough balance.

Chapter 3

Mathematical Modeling and Mechanical Design

This chapter will discuss the dynamic analysis of the moving parts in the machine, the structural design of the guides and finally mathematical modeling of the system.

3.1 Dynamic Analysis:

The mobility of this system is 3, because it has three degree of freedom (motion in X, motion in Y, motion in Z). So, three motors are needed to do the pulling process. Another motor is needed to perform the rotation motor about Y-axis. Therefore, the total amount of motors is 4 motors (2 Stepper motors and 2 Dc motors).

This section will talk about the moving parts in the machine, and make some calculations for the torque and velocity for the motion in X-Y plane.

General mechanical specifications and requirements:

Friction coefficient of the guides: μ=0.05

M_{screw} =0.5 Kg

$M_{barcode}$ = 0.5 Kg

M_{pocket} = 0.5 Kg

$M_{mechanism}$ = 0.5 Kg

M_{slider}= 1 Kg

M_{guide}= 0.5 Kg

M_{motor}= 0.5 Kg

3.1.1 Motion in X-axis:

Power screw mechanism used for x - axis motion as shown in figure 3.1. When sizing a motor, the load Torque must be calculated.

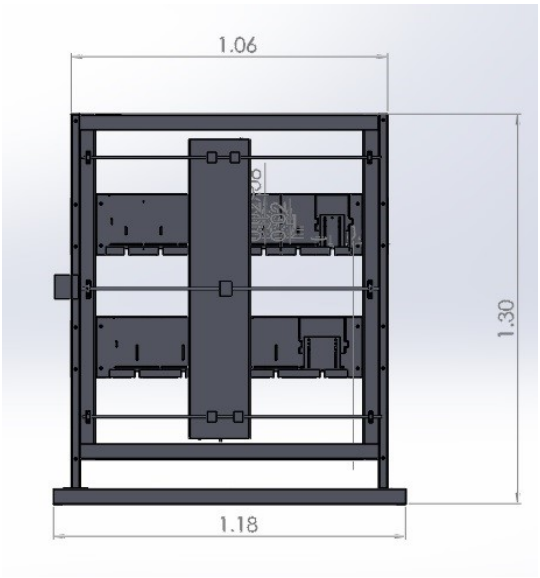


Figure 3.1: Motion in X-axis.

Total mass (M_{tx}):

$$M_{tx} = M_{ty} + M_{screw} + 2 \times M_{guide} + M_{motor} + M_{slider} \tag{3.1}$$

$$M_{tx} = 1.5 + 0.5 + 1 + 0.5 + 1 = 4.5Kg \tag{3.2}$$

While M_{tx} is the total mass moving in X direction.

To find the forces acting on the moving head as shown in figure 3.2:

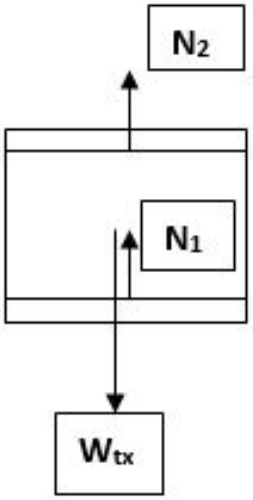


Figure 3.2: Moving head in X direction.

$$W_{tx} = N_1 + N_2 \tag{3.3}$$

$$N_1 = N_2 = \frac{W_{tx}}{2}$$

$$N_1 = N_2 = \frac{M_{tx} \times g}{2}$$

$$N_1 = N_2 = \frac{4.5 \times 10}{2}$$

$$N_1 = N_2 = 22.5N \tag{3.4}$$

Now, look at the free body diagram in figure 3.3:

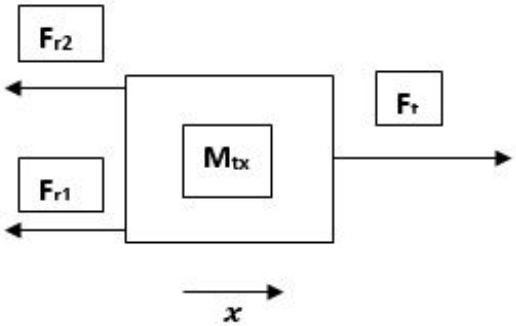


Figure 3.3: Free body diagram for moving head.

Fr_1 , Fr_2: friction forces between the slider and the guides.

$$F_{r1} = \mu_{slider} \times N_1 \tag{3.5}$$

While μ_{slider} is the static coefficient of friction between sliding surfaces.

$$\mu_{slider} = 0.05 \tag{3.6}$$

$$F_{r1} = 0.05 \times 22.5$$

$$F_{r1} = 1.125N$$

$$F_{r2} = F_{r1} = 1.125N$$

1. **Specifications and operating conditions of the drive mechanism** are shown in figure 3.4

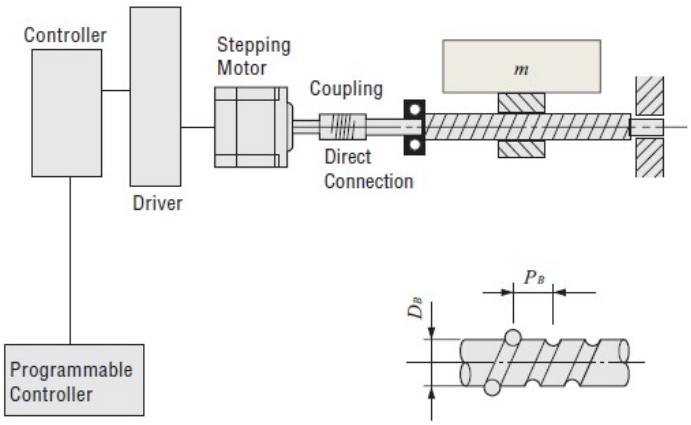


Figure 3.4: Specifications and operating conditions of the drive mechanism.

Total mass of the table and load	$M_{tx} = 4.5$ kg
Friction coefficient of sliding surface	$\mu = 0.05$
Ball screw efficiency	$\eta = 0.9$
Internal friction coefficient of preload nut	$\mu_0 = 0.3$
Ball screw shaft diameter	$D_B = 8$ mm
Total length of ball screw	$L_B = 1000$ mm
Ball screw material	Iron (density $\rho = 7.9 \times 10^3$ kg/ m^3)
Ball screw lead	$P_B = 8$ mm
Desired resolution	$\Delta l = 0.04$ mm/step
Feed	l= 650 mm
Positioning time	t_0 = within 10 sec.
Tilt angle	$\theta = 0$ deg

2. **Calculate the Required Resolution θ_s**

$$\begin{aligned} \theta_s &= \frac{(360^\circ \times \Delta_l)}{P_B} \\ &= \frac{(360^\circ \times 0.04)}{8} \\ &= 1.8^\circ \end{aligned} \tag{3.7}$$

AR Series can be connected directly to the application.

3. **Determine the Operating Pattern**

(a) Calculate the number of operating pulses A [Pulse].

$$\begin{aligned} A &= \frac{l}{P_B} \times \frac{360^\circ}{\theta_s} \\ &= \frac{650}{8} \times \frac{360^\circ}{1.8^\circ} \\ &= 16250 pulse \end{aligned} \tag{3.8}$$

(b) Determine the acceleration (deceleration) time $t_1[s]$ An acceleration

(deceleration) time of 5% of the positioning time is appropriate.

$$t_1 = 0.5s \tag{3.9}$$

(c) Calculate the operating pulse speed f_2 [Hz]

$$\begin{aligned} f_2 &= \frac{(A - f_1 \times t_1)}{(t_0 - t_1)} \\ &= \frac{(16250 - 0)}{(10 - 0.5)} \\ &= 1710Hz \end{aligned} \tag{3.10}$$

(d) Calculate the operating speed N_M[r/min]

$$\begin{aligned} N_M &= f_2 \times \frac{\theta_s}{360} \times 60 \\ &= 1710 \times \frac{1.8^\circ}{360} \times 60 \\ &= 513r/min \end{aligned} \tag{3.11}$$

4. **Calculate the Required Torque** T_M **[N·m]** The free body diagram of the horizontal motion is shown in figure 8.2(a).

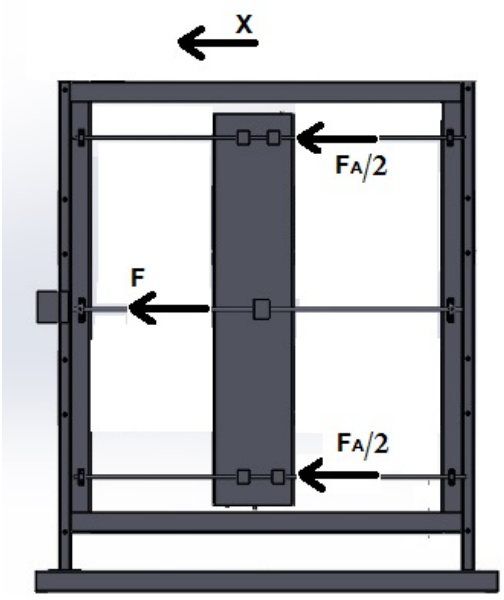


Figure 3.5: The free body diagram of the horizontal motion.

(a) Calculate the load torque T_L [N·m]

Force of moving direction:

$$\begin{aligned} F &= F_A + M_x g(\sin\theta + \mu\cos\theta) \\ &= 2.25 + 4.5 \times 9.807(\sin 0^\circ + 0.05\cos 0^\circ) \\ &= 4.45N \end{aligned} \tag{3.12}$$

Preload

$$\begin{aligned} F_0 &= \frac{F}{3} \\ &= \frac{4.45}{3} \\ &= 1.48N \end{aligned} \tag{3.13}$$

Load torque

$$
\begin{aligned}
T_L &= \frac{F.PB}{2\pi\eta} + \frac{\mu_0.F_0.P_B}{2\pi} \\
&= \frac{4.45 \times 8 \times 10^{-3}}{2\pi \times 0.9} + \frac{0.3 \times 1.48 \times 8 \times 10^{-3}}{2\pi} \\
&= 0.006860N.m
\end{aligned} \tag{3.14}
$$

(b) Calculate the acceleration torque T_a [N·m]

i. Calculate the moment of load inertia $J_L[kg\Delta m^2]$. Inertia of ball screw

$$
\begin{aligned}
J_B &= \frac{\pi}{32} \times L_B \times {D_B}^4 \times \rho \\
&= \frac{\pi}{32} \times 7.9 \times 10^3 \times 1000 \times 10^{-3} \times 8 \times (10^{-3})^4 \\
&= 3.176 \times 10^4 Kg.m^2
\end{aligned} \tag{3.15}
$$

Inertia of table and load

$$
\begin{aligned}
J_T &= M_X(P_B/2\pi)^2 \\
&= 4.5 \times (\frac{8 \times 10^{-3}}{2\pi})^2 \\
&= 7.295 \times 10^{-6} kg.m^2
\end{aligned} \tag{3.16}
$$

Load inertia

$$
\begin{aligned}
J_L &= J_B + J_T \\
&= 3.176 \times 10^{-4} + 7.295 \times 10^{-6} \\
&= 3.24 \times 10^{-4} kg.m^2
\end{aligned} \tag{3.17}
$$

ii. Calculate the acceleration torque T_a[N·m]

$$
\begin{aligned}
Ta &= J_L \times (\theta_S.\pi)/180^\circ \times (f_2 - f_1)/t_1 \\
&= 3.24 \times 10^{-4} \times (\pi \times 1.8^\circ)/180^\circ \times (1710 - 0)/0.5 \\
&= 0.03481 N.m
\end{aligned} \tag{3.18}
$$

iii. Calculate the required torque T_M [N·m]

Safety factor $S_f = 2$

$$
\begin{aligned}
T_M &= (T_L + T_a) \times s_f \\
&= (0.006860 + 0.03481) \times 2 \\
&= 0.08334 N.m
\end{aligned} \tag{3.19}
$$

5. **Select a Motor**

(a) Tentative motor selection T_M=0.08334[N.m] N_M=513 [r/min]

(b) Determine the motor from the speed – torque characteristics shown in figure 3.6.

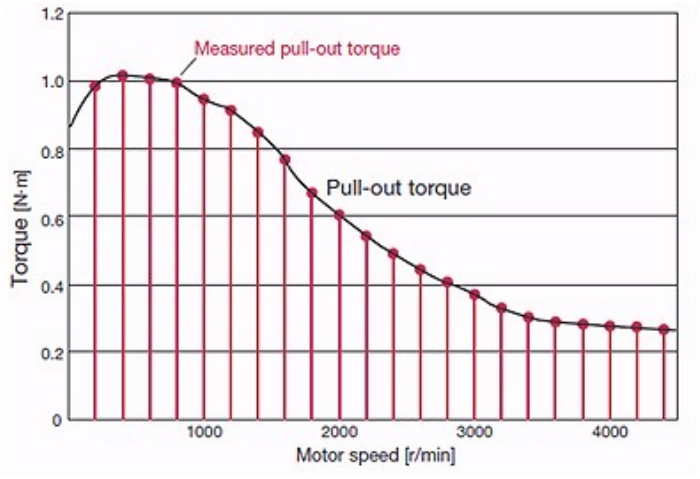


Figure 3.6: Determine the motor from the speed – torque characteristics.

This curve is the characteristic curve of NEMA 23 stepper motor.

Because the working area is under the curve, so it is suitable for the

X-axis motion.

3.1.2 Motion in Y-axis

Power screw mechanism used for y-axis motion as shown in figure 3.7. The mass (M_{ty}) is the total mass of the moving frame in Y direction.

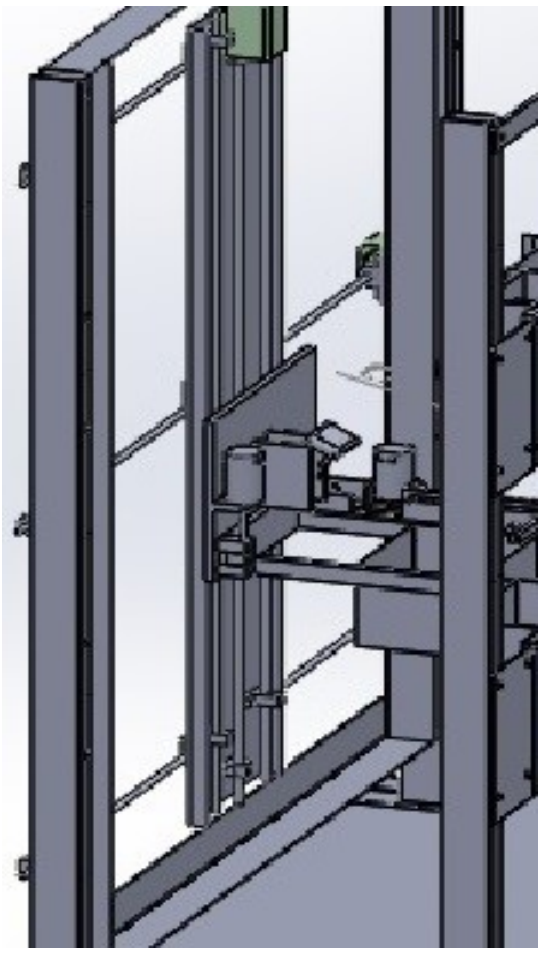


Figure 3.7: Power screw mechanism used for y-axis motion.

Total mass (M_{ty}):

$$\begin{aligned} M_{ty} &= M_{barcode} + M_{mechanism} + M_{pocket} \\ &= 0.5 + 0.5 + 0.5 \\ &= 1.5kg \end{aligned} \tag{3.20}$$

1. **Specifications and Operating Conditions of the Drive Mechanism**as shown in figure 3.8:

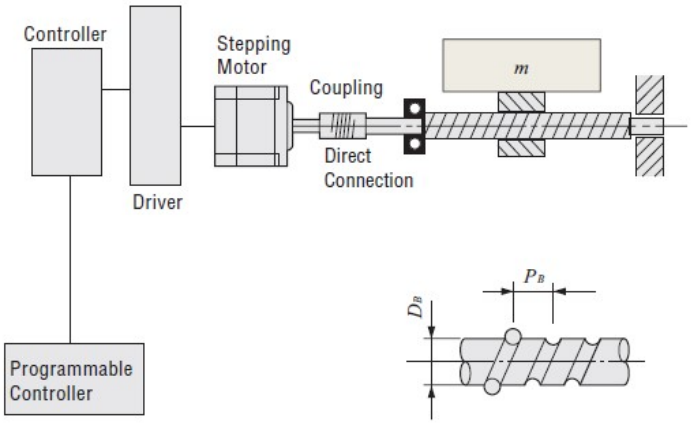


Figure 3.8: Specifications and Operating Conditions of the Drive Mechanism.

Total mass of the table and load	$M_t y$= 1.5 [kg]
Friction coefficient of sliding surface	$\mu = 0.05$
Ball screw efficiency	$\eta = 0.9$
Internal friction coefficient of preload nut	μ_0= 0.3
Ball screw shaft diameter	$D_B = 8$ [mm]
Total length of ball screw	$L_B = 1000$ [mm]
Ball screw material	Iron (density $\rho = 7.9 \times 10^3$ [kg/ m^3])
Ball screw lead	P_B= 8 [mm]
Desired resolution	$\Delta l = 0.04$ [mm/step]
Feed	l= 650 [mm]
Positioning time	t_0 = within 10 sec.
Tilt angle	θ= 90 [deg]

2. **Calculate the Required Resolution θ_s**

$$\begin{aligned} \theta_s &= (360^\circ \times \Delta_l)/P_B \\ &= (360^\circ \times 0.04)/8 \\ &= 1.8^\circ \end{aligned} \tag{3.21}$$

AR Series can be connected directly to the application.

3. **Determine the Operating Pattern**

   (a) Calculate the number of operating pulses A [Pulse]

$$\begin{aligned} A &= l/P_B \times 360^\circ/\theta_s \\ &= 650/8 \times 360^\circ/1.8^\circ \\ &= 16250 pulse \end{aligned} \tag{3.22}$$

   (b) Determine the acceleration (deceleration) time t_1[s] An acceleration (deceleration) time of 5% of the positioning time is appropriate.

$$t_1 = 0.5s \tag{3.23}$$

   (c) Calculate the operating pulse speed f_2 [Hz]

$$\begin{aligned} f_2 &= (A - f_1 \times t_1)/(t_0 - t_1) \\ &= (16250 - 0)/(10 - 0.5) \\ &= 1710 Hz \end{aligned} \tag{3.24}$$

   (d) Calculate the operating speed N_M[r/min]

$$\begin{aligned} N_M = f_2 \times \theta_s/360 \times 60 & \\ &= 1710 \times 1.8^\circ/360 \times 60 \\ &= 513 r/min \end{aligned} \tag{3.25}$$

4. **Calculate the Required Torque T_M [N·m]**

   (a) Calculate the load torque T_L [N·m]

Force of moving direction

$$F = F_A + M_y g(\sin\theta + \mu cos\theta) = 0 + 1.5 \times 9.807(\sin 90° + 0.05\cos 90°) = 14.71N \tag{3.26}$$

Preload

$$\begin{aligned} F_0 &= F/3 \\ &= 14.71/3 \\ &= 4.90N \end{aligned} \tag{3.27}$$

Load torque

$$\begin{aligned} T_L &= (F.PB)/2\pi\eta + \mu_{0.F_0.P_B}/2\pi \\ &= (14.71 \times 8 \times 10^{-3})/(2\pi \times 0.9) + (0.3 \times 4.90 \times 8 \times 10^{-3})/2\pi \\ &= 0.02268N.m \end{aligned} \tag{3.28}$$

Calculate the load torque T_L [N·m]

Force of moving direction

$$\begin{aligned} F &= F_A + M_y g(\sin\theta + \mu\cos\theta \\ &= 0 + 1.5 \times 9.807(\sin 90° + 0.05\cos 90° \\ &= 14.71N \end{aligned} \tag{3.29}$$

Preload

$$
\begin{aligned}
F_0 &= F/3 \\
&= 14.71/3 \\
&= 4.90N
\end{aligned}
\tag{3.30}
$$

Load torque

$$
\begin{aligned}
T_L = (F.PB)/2\mu\eta + \mu_{0.F_0.PB)}/2\pi \\
&= (14.71 \times 8 \times 10^{-3})/(2\pi \times 0.9) \\
&+ (0.3 \times 4.90 \times 8 \times 10^{-3})/2\pi \\
&= 0.02268N.m
\end{aligned}
\tag{3.31}
$$

5. Calculate the acceleration torque T_a[N·m]

   (a) Calculate the moment of load inertia J_L[kg·m^2] .

   Inertia of ball screw

$$
\begin{aligned}
J_B &= \pi/32 \times L_B \times {D_B}^4 \times \rho \\
&= \pi/32 \times 7.9 \times 10^3 \times 1000 \times 10^{-3} \times (8 \times 10^{-3})^4 \\
&= 3.176 \times 10^{-4} Kg.m^2
\end{aligned}
\tag{3.32}
$$

   Inertia of table and load

$$
\begin{aligned}
J_T &= M_y(P_B/2\pi)^2 \\
&= 1.5 \times ((8 \times 10^{-3})/2\pi)^2 \\
&= 2.431 \times 10^{-6} kg.m^2
\end{aligned}
\tag{3.33}
$$

Load inertia

$$
\begin{aligned}
J_L &= J_B + J_T \\
&= 3.176 \times 10^{-4} + 2.431 \times 10^{-6} \\
&= 3.20 \times 10^{-4} kg.m^2
\end{aligned}
\tag{3.34}
$$

(b) Calculate the acceleration torque Ta [N·m]

$$
\begin{aligned}
Ta &= J_L \times (\theta_S.\pi)/180^\circ \times (f_2 - f_1)/t_1 \\
&= 3.20 \times 10^{-4} \times (\pi \times 1.8^\circ)/180^\circ \times (1710 - 0)/0.5 \\
&= 0.03438 N.m
\end{aligned}
\tag{3.35}
$$

(c) Calculate the required torque T_M [N·m]

Safety factor $S_f = 2$

$$
\begin{aligned}
T_M &= (T_L + T_a) \times s_f \\
&= (0.02268 + 0.03438) \times 2 \\
&= 0.11412 N.m
\end{aligned}
\tag{3.36}
$$

6. Select a Motor

(a) Tentative motor selection

T_M=0.11412 [N.m]

N_M=513 [r/min]

(b) Determine the motor from the speed – torque characteristics shown in figure 3.9.

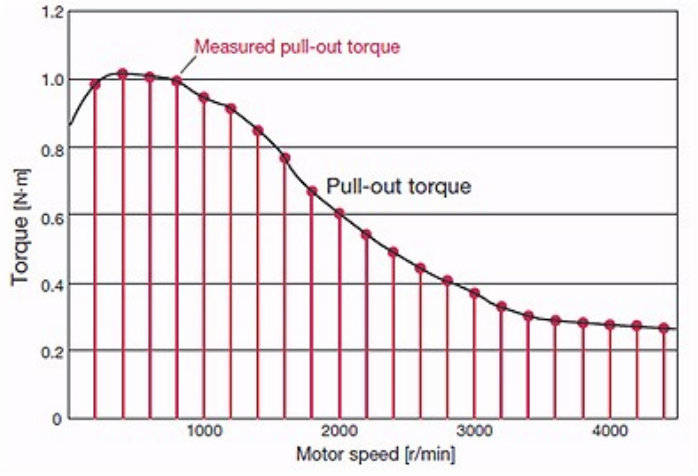


Figure 3.9: Determine the motor from the speed – torque characteristics.

This curve is the characteristic curve of NEMA 23 stepper motor. Because the working area is under the curve, so it is suitable for the y-axis motion.

3.2 Mechanical Design

This section will discuss the structural design of the machine, and make some calculations related to stress and stiffness of the guides. And also studying the mechanism and gear ratios in the machine.

3.2.1 Guides Design

The guides that will be used are rod shaped guides. In order to select the suitable diameter for these rods, stress and stifiness calculations must be used.

- Stress study: to calculate the reactions acting on the guides, the free body diagram should be drawn as in figure 3.10.

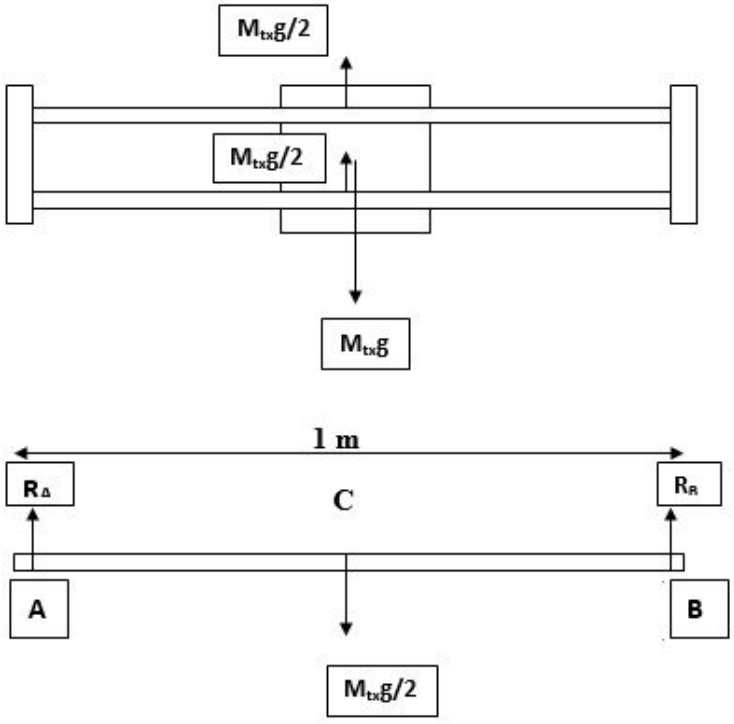


Figure 3.10: Rod guides free body diagram.

The guides are in static equilibrium, so:

$$\sum F_y = 0 \tag{3.37}$$

$$R_A + R_B - (\frac{(M_{tx}g)}{2}) = 0$$

$$R_A + R_B - 22 = 0$$

$$R_A + R_B = 22 \tag{3.38}$$

$$\sum M_A = 0 \tag{3.39}$$

$$(22 \times 0.5) - (R_B \times 1) = 0$$

$$R_B = 11N \tag{3.40}$$

$$R_A = 11N \tag{3.41}$$

$$M_A = R_A \times 0.5 \tag{3.42}$$

$$M_A = 11 \times 0.5$$

$$M_A = 5.5N.m \tag{3.43}$$

The normal stress in bending case acting on the guides is [10]:

$$\sigma_x = \frac{(M_A D)}{2I} \tag{3.44}$$

While I is the second moment of area for the circular section [10]:

$$I = \frac{\pi D^4}{64} \tag{3.45}$$

$$\sigma_x = \frac{5.5D}{2\frac{\pi D^4}{64}}$$

$$\sigma_x = \frac{56}{D^3} \tag{3.46}$$

Assume the rod is made from hot rolled-carbon steel (ASTM 1050) [10]: Yield strength = 620 MPa Using factor of safety = 1.5

$$S_{allawable} = \frac{620}{1.5} = 413MPa \tag{3.47}$$

$$\sigma_x < S_{allawable} \tag{3.48}$$

$$\frac{56}{D^3} < 413MPa$$

$$\frac{56}{413 \times 10^6} < D^3$$

$$D > 5 \times 10^{-3}m$$

$$D > 5mm \tag{3.49}$$

- Deflection study: To calculate the maximum deflection in the guides caused by the forces as shown in figure 3.11:

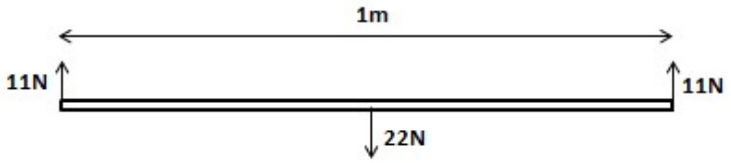


Figure 3.11: Rod guides maximum deflection.

The maximum defection for the simple support - center load [10]:

$$Y_{max} = \frac{(M_{tx} g l^3)}{48EI} \tag{3.50}$$

While E is the Modulus of Elasticity [10]:

$$E_{carbon_{steel}} = 207GPa \tag{3.51}$$

$$I = \frac{\pi D^4}{64}$$

$$Y_{max} = \frac{22 \times 1^3}{48 \times 207 \times 10^9 \times \frac{3.14 \times D^4}{64}}$$

$$Y_{max} = \frac{4.5 \times 10^{-11}}{D^4} \tag{3.52}$$

Assume the allowable deflection is 2 mm:

$$Y_{max} < 2 \times 10^{-3} \tag{3.53}$$

$$\frac{4.5 \times 10^{-11}}{D^4} < 2 \times 10^{-3}$$

$$\frac{4.5 \times 10^{-11}}{2 \times 10^{-3}} < D^4$$

$$D^4 > 2.25 \times 10^{-8}$$

$$D > 12.5mm \tag{3.54}$$

As a result, the guides diameter must be grater than 12.5 mm to achieve acceptable strength and stiffness, with deflection 2 mm.

3.2.2 12V DC Motor:

In order to select the suitable motor that will be used to rotate the container that contains the required drugs as shown in figure 3.12:

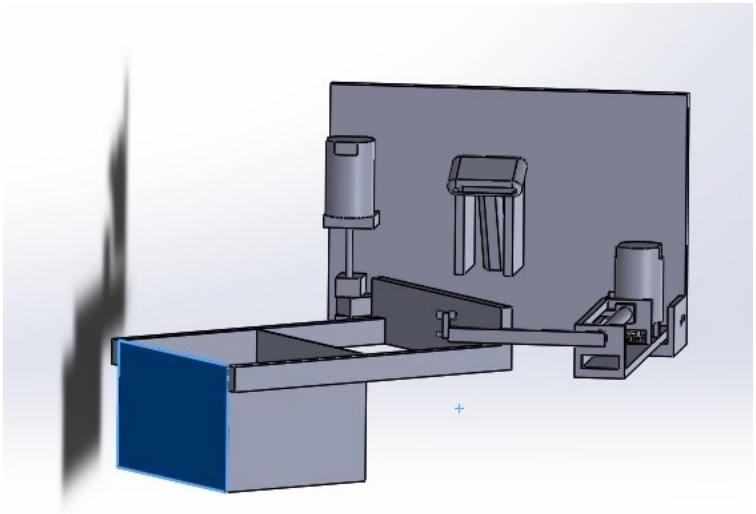


Figure 3.12: Container's rotation mechanism.

M_{pocket}= 0.5 Kg

M_{arm}= 0.2 Kg

L_{arm}=15cm

The time required to rotate the motor t=1sec, and the acceleration $\ddot{\theta} = \pi$ rad/sec^2

$$\sum M_y = J\ddot{\theta} \tag{3.55}$$

$$\tau - C_t\dot{\theta} = M_{pocket}{L_{arm}}^2 + (M_{arm}{L_{arm}}^2)/3 \tag{3.56}$$

$C_t \approx 0$

$$\begin{aligned} \tau &= [M_{pocket}{L_{arm}}^2 + (M_{arm}{L_{arm}}^2)/3] \\ &= [0.5 \times 0.15^2 + (0.2 \times 0.15^2)/3] \\ &= 0.01275 N.m \end{aligned} \tag{3.57}$$

The torque of the DC Motor is 0.01275 N.m

Select 12v DC motor with gear and the torque 0.03 N.m.

To pick drugs out of the storage cells, the quick return mechanism is used as shown in figure 3.13. In order to calculate the suitable torque must be applied on the disk of the mechanism, these calculations are required:

The inclination of the cell is designed to be 14. Assume the maximum mass of one packet of drugs to be 300 gram (0.3 Kg).

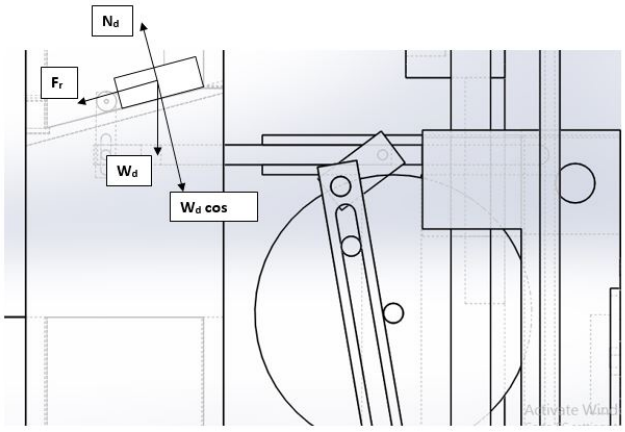


Figure 3.13: Analysis of quick return mechanism.

$$W_d = M_d g$$

$$W_d = 0.3 \times 10$$

$$W_d = 3N \tag{3.58}$$

$$N_d = W_d cos 14 \tag{3.59}$$

$$N_d = 3 \times 0.97$$

$$N_d = 2.9N$$

Now, the friction force equals:

$$F_r = \mu N_d \tag{3.60}$$

While $\mu$is the coefficient of friction between the sliding surfaces, assumed to be 0.5:

$$F_r = 0.5 \times 2.9$$

$$F_r = 1.45N \tag{3.61}$$

$$\sum F_x = F_r + W_d sin14 \tag{3.62}$$

$$\sum F_x = 1.45 + 3 \times 0.24$$

$$\sum F_x = 2.18N \tag{3.63}$$

Now the applied force on the packet must be greater than $\sum F_x$ as shown in figure 3.14:

$$F_{app} = 2.5N \tag{3.64}$$

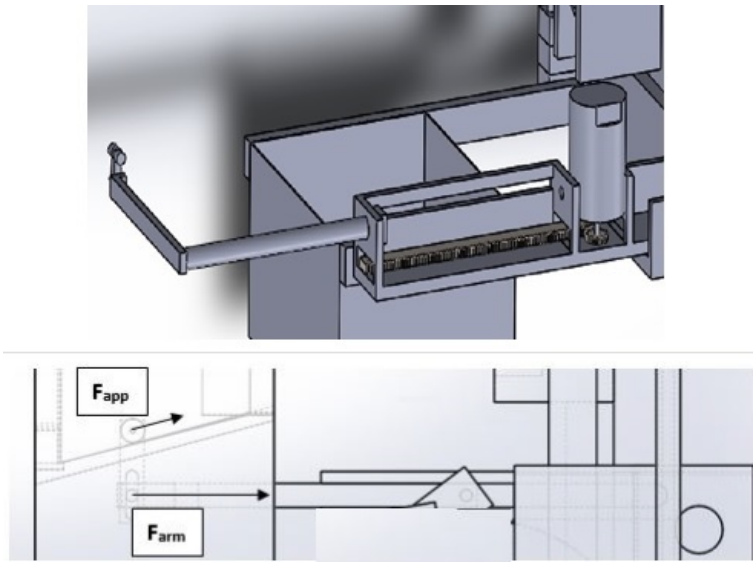


Figure 3.14: The applied force and arm force in the mechanism.

$$F_{arm} = F_{app} cos 14 \tag{3.65}$$

$$F_{arm} = 2.5 \times 0.97$$

$$F_{arm} = 2.4N \tag{3.66}$$

$$\tau_{app} = F_{arm} \times l_{crank} \tag{3.67}$$

$$\tau_{app} = 2.4 \times 0.01$$

$$\tau_{app} = 0.024N.m \tag{3.68}$$

Because of the small values of inertias of the disk and links, the motor torque assumed to be little larger than τ_{app}:

$$\tau_m = 0.03N.m \tag{3.69}$$

Chapter 4

Control Architecture and Software Design

This chapter will talk about the control design of the moving parts in the ADM machine in order to obtain the desired output in high accuracy. In the other hand, the software design will be clarified using flowchart and an explanation of the ADM procedure operation.

4.1 Control Architecture

Due to the motion study in chapter 3, this machine has four independent motions; motion in X- axis, and motion in Y- axis, pulling motion, rotational motion.

In order to achieve high accuracy in positioning the moving head, two stepper motors were used (open loop control) with high accuracy.

Because of the high accuracy of the stepper motors, there is no need to design a controller for this motion.

4.1.1 Stepper Motors Control:

This section will discuss the method that is used to move the stepper motors to the desired position.

Due to the motors selection in chapter 3, NEMA23 stepper motor is used in the ADM machine. This motor moves 200 step/revolution.

The used lead screw pitch is 8 mm.

This means that to move the head 8 mm ,the stepper must move 200 step (one revolution).

To convert the distance to steps, equation 4.1 is used:

$$Steps = (Distance(mm)/8) \times stepsPerRevolution \tag{4.1}$$

The large row of cells consist of 5 cells, each cell is 155 mm width. So, to move the head to the required cell:

$$Distance = ((cell - 1) \times 155) \tag{4.2}$$

Then find the equivalent number of steps for this distance using the previous equation. To move the head in the small row of cells, the same method is used. However, the width of each cell is 115 mm.

4.2 Software Design

In order to write the procedure code correctly, the operation should be well studied through drawing the flowchart and understand its details.

4.2.1 Flowchart

The following flowchart (shown in Figure refm5) describes the sequence of the operation of the ADM machine:

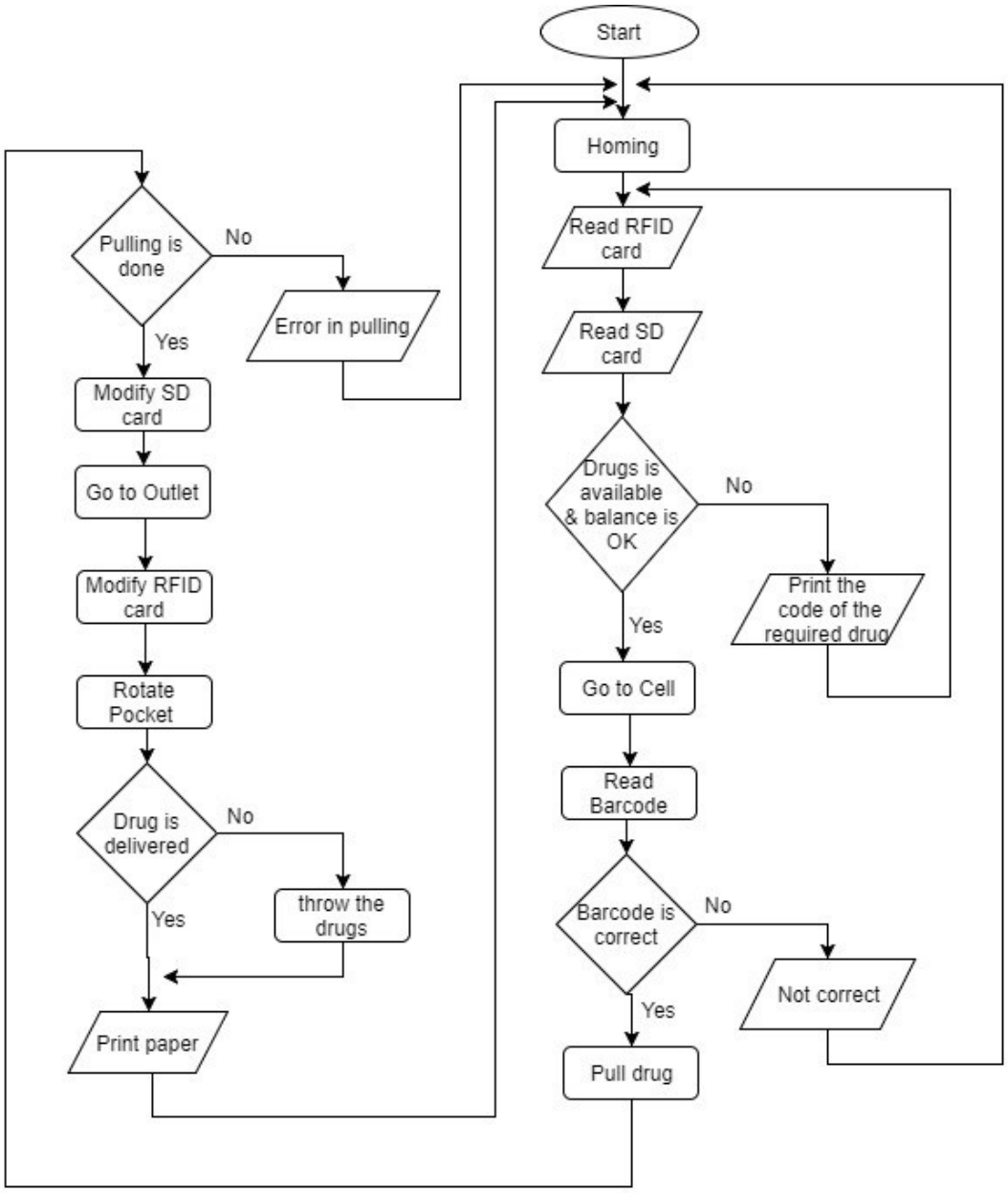


Figure 4.1: Flowchart for ADM.

4.2.2 ADM Procedure Operation

The operation of the ADM machine follows these steps:

1. The RFID reader reads the patient number and the required drug's number from the magnetic card, and sends these numbers to the micro- controller.

2. The microcontroller compares the drug number with the list of available

drugs in the machine(stored in SD card) and decides if it is available or not.

3. If the required number is not available, then the microcontroller activates the printer to print a small paper that contains the name of the required medicine to buy it from the pharmacy.

4. If the required number is available, the microcontroller finds the address of the cell that contains the required drug.

5. Then the microcontroller activates the X-Y Steppers motors to move the head to the required cell.

6. The motors continue running until the current position equal required position reading shows that the head in the required position.

7. After that, the microcontroller breaks the motors, and activates the motor of the mechanism to rotate the DC Motor to pick out the required packet.

8. Then, some sensors give a signal to make sure that the packet was fallen in the container.

9. Then, the microcontroller activates the X-Y motors again until the head reaches the outlet gate.

10. After that, motor 3 is activated to rotate the container one half cycle to the outlet side.

11. Then the microcontroller activates the outlet gate and the printer to print a small paper contains information about the medicine, doctor and the patient.

12. The machine go to home position until the next dispensing process .

Chapter 5

Components Selection

In this chapter, the required electrical components will be selected during to their functions by comparing the required characteristics with the available devices in the market.

5.1 Microcontroller Selection

The microcontroller will be used to control the dispensing operation:

- Receive data from card's reader and barcode reader and compare it with available data.
- Send signals to control the motion of the motors.
- Send data to the screen.
- Store data about the drugs inside the machine.

Arduino microcontroller could do these jobs easily. Arduino has many different types, like Arduino Mini, Arduino UNO, and Arduino MEGA.

The suitable microcontroller for this project is Arduino MEGA 2560 as shown in figure 5.1. It has many features that qualify it for this selection as follow [11]:

1. It has 54 digital input/output pins (of which 15 can be used as PWM outputs).

2. It has 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection.

Figure 5.1: Arduino MEGA 2560.

In order to connect the barcode reader and cards reader to the Arduino, Arduino USB Host Shield must be used. Because these readers are connected using USB cables.

The Arduino USB Host Shield is selected for this project as shown in figure 5.2.

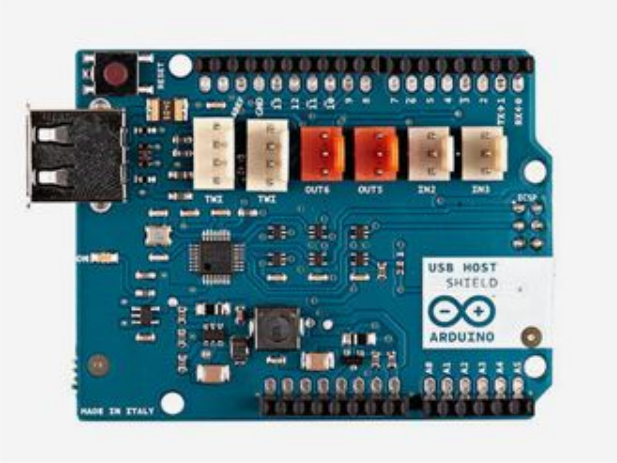


Figure 5.2: Arduino USB Host Shield.

5.2 Motors Selection

5.2.1 Stepper Motors Selection

The stepper motor is suitable selection for the motion in X-Y plane in this machine for many reasons [12]:

- Stepper motors are DC motors that move in discrete steps. They have multiple coils that are organized in groups called "phases". By energizing each phase in sequence, the motor will rotate, one step at a time.
- Since steppers move in precise repeatable steps, they excel in applications requiring precise positioning such as 3D printers, CNC, Camera platforms and X,Y Plotters. Some disk drives also use stepper motors to position the read/write head.

In this chapter, the required electrical components will be selected during to their functions by comparing the required characteristics with the available devices in the market.

Figure 5.3: NEMA 23-1.5NM Stepper Motor.

5.2.2 DC Motors Selection

Due to the torque calculations in chapter 3,The resulting torques for the mechanism and container motors were much less than the torques needed for X-Y motion.

So, The 12V-DC Motor as shown in figure 5.4 was used for the mechanism and for rotating the container because it provides less torque than the first motor [14].

Figure 5.4: 12V-DC Motor with gear.

5.3 Drivers Selection

This driver used to control the position, direction and the speed of stepper motor by sending pulses from host controller such that the number of pulses controls the position and the frequency control the speed of the stepper motor.

The L 298N (as shown in figure 5.5) is the suitable driver for NEMA23 stepper motors. The driver L298N need current 2A, for each stepper need 2 parallel driver (L298N), because stepper current is 3A.

This driver is a new generation Microstepping controller running smoother and cooler suitable for a wide range of stepping motors with its wide range of

settings for both current and micro steps [15]. And also this driver is suitable for NEMA 17-0.6A steppers.

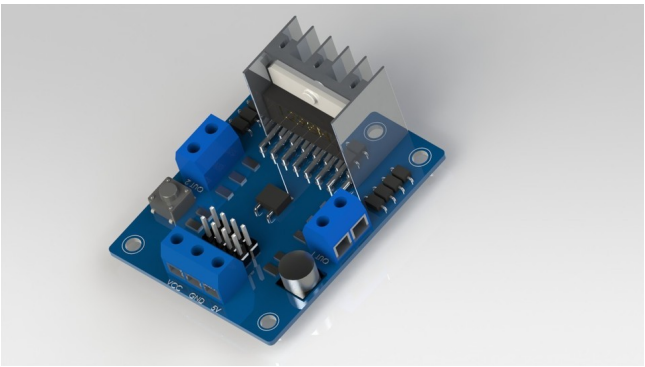


Figure 5.5: CW-5045 stepper motor Driver.

5.4 RFID Reader Writer Module

RFID frequency identification is a non-contact object recognition technology, it can automatically recognize target and get relevant data through rf signal, so there are so many interactive projects which need to use RFID to achieve object recognition. Generally speaking, one basic kit of RFID general system mainly consists of three parts:

- Tag : composed by coupling components and chip, every tag have only electronic coding, adhere to the target object identifier.
- Reader : equipment used to read (sometimes can write) tag information, can be designed for handheld or stationary.
- Antenna : transmit rf signal between the tag and reader.

This RFID module is desgined based on MFRC522. It is a highly integrated reader/writer for contactless communication at 13.56MHz. It supports ISO 14443A/MIFARE mode and MIFARE Classic (e.g. MIFARE Standard) products. Contactless communication using MIFARE higher transfer speeds up to 848kbit/s in both directions as show in figure 5.6.

This great RFID kit contains a combination reader/writer module powered by the popular MFRC522 chip, and comes with a RFID card and a RFID keyfob and straight and right angle headers.

Powered from a 3.3V supply, it is ideal for use with Arduino, AVR or ARM processors.

Figure 5.6: RFID Reader and Writer Module.

5.5 Barcode Reader Selection

The barcode reader is needed to read the code of each type of drug. It is necessary to make sure that the packets in the cell are the required drugs and have the same code number.

In this machine the required barcode reader must has a USB cable to connect it with the Arduino through the USB shield.

The (USB Port Handheld Barcode Scanner UPC EAN) as shown in figure 5.7 is a suitable selection for this machine. This device supports a very high scanning speed and bi-directional scanning type [17]:

Figure 5.7: USB Port Handheld Barcode Scanner and Writer.

5.6 Screen selection:

The (2.8" TFT Touch Shield for Arduino) as shown in figure 5.8 is suitable for this machine because it has these characteristics:

- 240x320 resolution, 18-bit (262,000) color.
- Works with any Arduino '328 or Mega (Leonardo not supported yet).
- 5V compatible! Use with 3.3V or 5V logic.

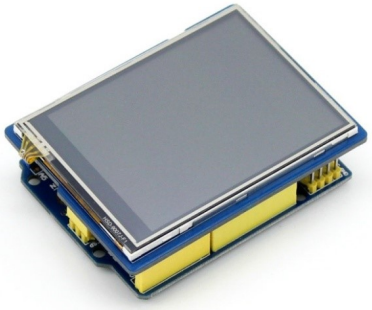


Figure 5.8: 2.8" TFT Touch Shield for Arduino.

Chapter 6

Electrical Design

This chapter will talk about the electrical components in the ADM machine, and the electrical block diagram to show the overall design. And finally how the components in the machine are connected to each other.

6.1 Electrical Components

The ADM machine consists of several electrical components connected to each other to achieve the main objective of the machine. These components are:

1. Microcontroller (Arduino): to control all the machine and connect all components to each other.

2. Stepper motors: to move the head, picking out the drugs and rotate the container to the outlet gate side.

3. Drivers for stepper motors: to control the speed and direction of each stepper motor.

4. Barcode reader: to read the code of each required drug before taking it out the cell.

5. RFID Reader Writer Module: to read the medicine code that prescribed by the doctor.

6. Screen: to enable the user dealing with the machine easily.

7. Power supply)12v): to feed all components with the appropriate electrical power.

8. Printer: to print small informative papers after the dispensing process.

9. Limit switches: to detect that the moving head reaches the home position, and to start each cycle from this position.

6.2 Electrical Block Diagram

The electrical block diagram for the ADM machine is shown in figure 6.1.

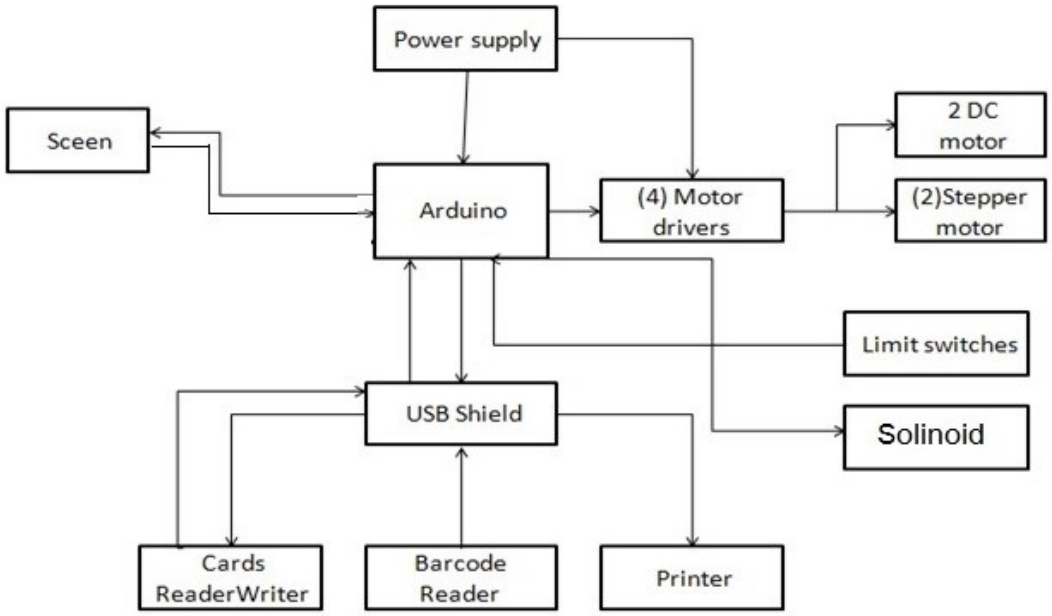


Figure 6.1: Electrical Block Diagram for ADM.

6.3 Wiring Diagram of ADM

The wiring diagram of this machine is shown in figure 6.2

- the connection between parts is as follow:
- Barcode reader is connected to the Arduino using USB Host Shield.
- RFID is connected using SPI serial connection.
- SD card shield is connected using SPI serial connection.
- Touch screen is connected directly to UNO Arduino, and the two Arduinos are connected using (Tx/Rx) serial.
- Printer is connected using (Tx/Rx) serial.

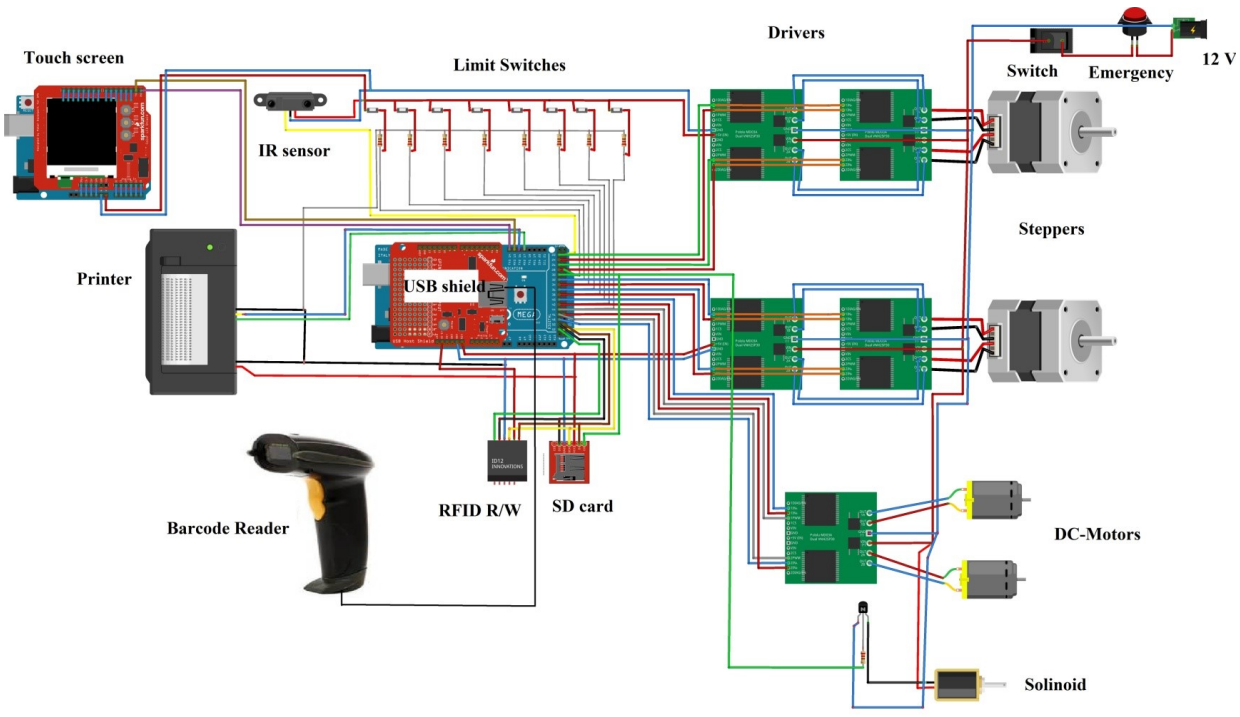


Figure 6.2: The wiring diagram of ADM machine.

Chapter 7

Implementation and Assembly

This Chapter discuss fabrication and installation of all components and parts.

7.1 Internal Cells Manufacturing and Assembly

The best way to form the drug cells is to use sheet metal manufacturing process. This was done using laser CNC machine. The suitable thickness of the steel sheet is 1.25 mm. this make it easy to form the cells.

The laser CNC requires 2D drawings for the project design, this was done by exporting "DXF" drawing types from the sheet metal design as shown in figure 7.1

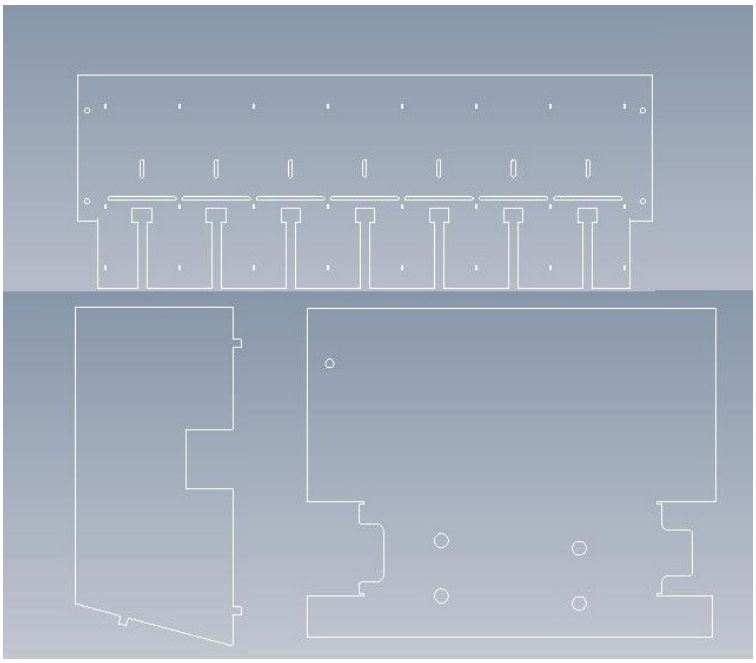


Figure 7.1: Exporting “DXF” drawing types from the sheet metal design.

Figure 7.2 below shows some parts of the sheet after the laser cut:

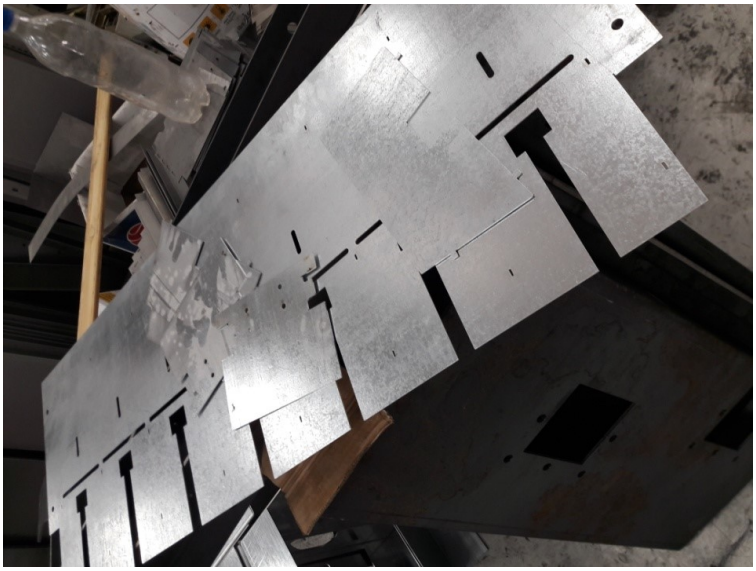


Figure 7.2: Some parts of the sheet after the laser cut.

The next step is to bend the previous sheet using CNC bending machine to have their final shape.

These steps were done for all parts of the cells and internal size limitation mechanisms.

After that the whole row of cells was assembled using pin joints as shown in the figure 7.3 below.

The row has a length of 100 cm , height of 20 cm and its width is 11 cm.

(a) (b)

(c) (d)

(e) (f)

Figure 7.3

This process was repeated to make the second row of cells.

7.2 Motion Frame Assembly

7.2.1 Frame Structure

The frame of X-Y motion has been made by aluminum plates, because it easy to machine and it has a light weight. The figure 7.4 shows the assembly of the X-axis motion frame.

This frame dimensions are:

1. Length: 100 cm.

2. Height: 120 cm.

- 

Figure 7.4: The assembly of the X-axis motion frame.

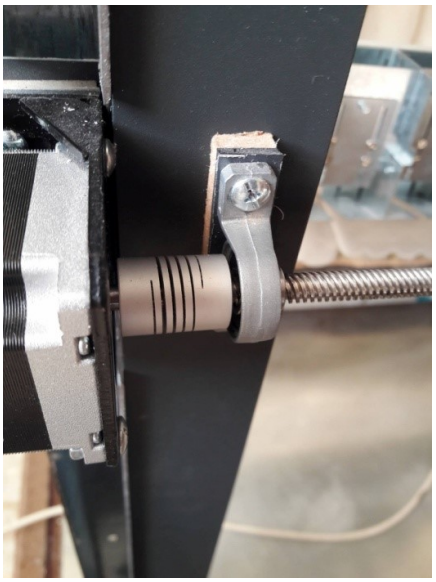
The vertical motion frame is shown in figure 7.5.

Figure 7.5: The vertical motion frame.

7.2.2 Lead Screw Installation

The lead screw motion require many other parts in addition to the lead screw; this includes guides, linear bearings, rotational bearings, nut, nut holder , coupler. These all parts are shown in figure 7.6.

(a) (b)



(c)

Figure 7.6: Lead screw installation

7.2.3 Stepper Motors Installation

After that, the stepper motors were fitted in their prickets and connected with the power screws using suitable couplers as shown in figure 7.7.

(a) (b)

Figure 7.7: Stepper motors installation.

7.3 Stand Assembly

The stand of the machine which carry the rows of cells and also the motion frame is consist mainly from vertical beams, horizontal beams and the base of the machine. The beams were made from sheet metal with 2 mm thickness. The beams take U shape as shown in figure 7.8.

(a) (b)

Figure 7.8: The stand assembly of the machine.

The base was made from 0.9 mm sheet and it takes the box shape as shown in figure 7.9.

Figure 7.9: The base.

The complete stand is shown in figure 7.10.

Figure 7.10: The complete stand.

7.4 Moving Head Assembly

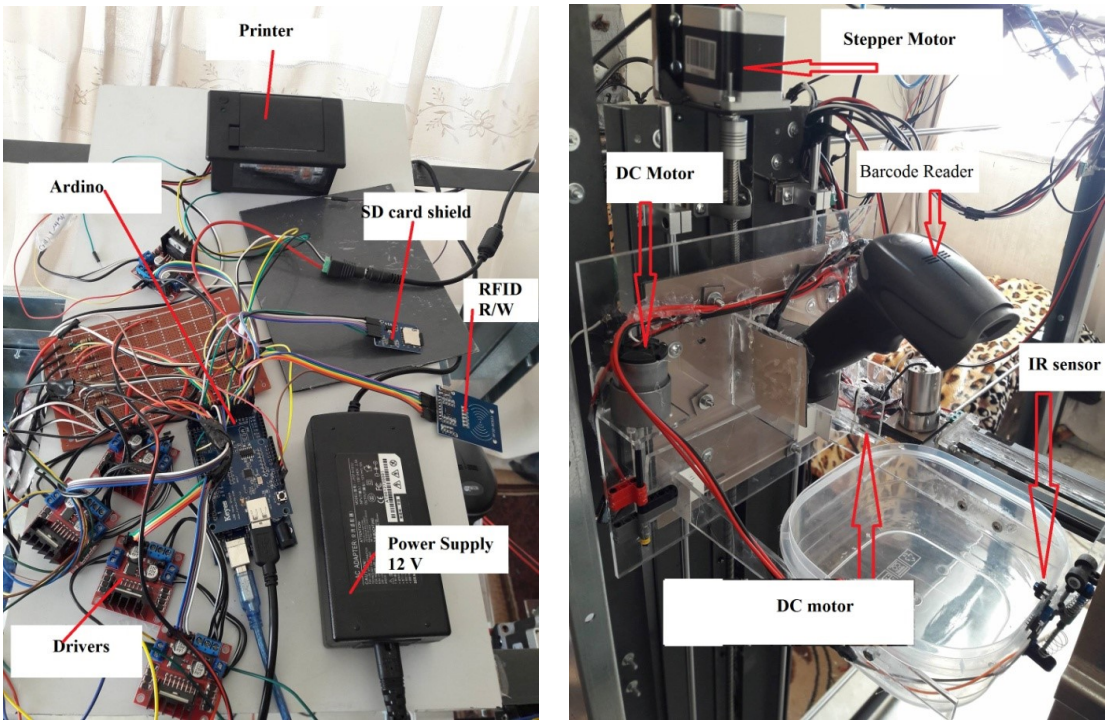
The moving head was designed to carry the barcode reader, pulling mechanism and the pocket to carry the drugs to the user. The suitable material for this head is plastic. It was designed to be cut using laser cutting machine. After cutting the head parts, they were gathered using very strong super glue as shown in figure 7.11.

Figure 7.11: Moving head assembly.

7.5 Electrical Parts Installation

After all, the electrical parts were fitted in their places. This includes the Barcode reader, Motors, switches, sensors, RFID reader, screen, printer and

microcontroller. The figure 7.12 below shows all components fitted in the machine.



(a) (b)

Figure 7.12: Electrical parts installation.

7.6 Arduino code

After understanding the operation of the ADM, and install all the mechanical and electrical components, it is easy to write the code. This section will describe the main functions in the Arduino code of this machine. However, Appendix (4) contains the complete code.

The Arduino code starts by defining the libraries used in the code, and definition of stepper motor driver pins, the following code shows the definitions.

```
#include <AccelStepper.h>
#include <MultiStepper.h>
#include <SPI.h>
#include <MFRC522.h>
#include <SdFat.h>
#include <CSVFile.h>
#include <SoftwareSerial.h>

#include <avr/pgmspace.h>
#include <hiduniversal.h>
#include <Usb.h>
#include <usbhub.h>
#include <avr/pgmspace.h>
#include <hidboot.h>
```

Then, it should define the output and input pins for the system, as well as other variables that are used in the code.

The following code shows some pins definitions.

```
#define PIN_SPI_CLK 52
#define PIN_SPI_MOSI 51
#define PIN_SPI_MISO 50
#define PIN_SD_CS 29
#define PIN_SD_ON 27
#define RFID_SS_PIN 53
#define RFID_RST_PIN 49
#define USB_SS_PIN 10
#define SD_CARD_SPEED SPI_FULL_SPEED

#define FILENAME "CSV.csv"
int H=0;
int FD = 40;
int BD = 42;
int enM1 = 44;
```

The next step in the code, defines the pins setup. This operation is done

for one time at the start of the code. It can set system homing in void setup, because it is also done for one time, the following code shows these options:

```
void setup() {

  pinMode(2, INPUT);
  pinMode(3, INPUT);
  pinMode(FD, OUTPUT);
  pinMode(BD, OUTPUT);
  pinMode(enM1, OUTPUT);
  pinMode(enM2, OUTPUT);
  pinMode(Op, OUTPUT);
  pinMode(Cl, OUTPUT);

 stepper1.setMaxSpeed(800);
 stepper2.setMaxSpeed(800);

  steppers.addStepper(stepper1);
  steppers.addStepper(stepper2);

 digitalWrite(BarcodeOn,HIGH);
 digitalWrite(PIN_SD_CS, HIGH);
  digitalWrite(PIN_SD_ON, LOW);
   digitalWrite(RFID_SS_PIN, HIGH);
   digitalWrite(USB_SS_PIN, HIGH);
```

After that, the main loop function is defined. This will be executed repeatedly every cycle.

Firstly, it checks if the machine in the home position or not through the variable “H”.

If not, it starts the “Homing”function, this function is defined outside the mail loop (refer to Appendix (4)).

After that, the machine waits for a new card reading by calling the function “readFromRFID”. It also read all the data from the patient card. Then, “read SD card” function will start to read the desired data from the SD card. Then,

a new loop will begin to check the amount and price for the required drugs.

```
void loop() {

if(H==0)
Homing();
delay(2000);

if(!readFromRFID()){
return;
}

   Serial.println("\n");

if(!readSDcard()){
  return;
}

newBalance_int=Balance_int;

fst=0;
snd=0;
thd=0;

for(D=0 ; D<=2 ; D++){
  cell=DRUG[D].cell;

    if(DRUG[D].amount==0){
    Serial.print(DRUG[D].drugName);
    Serial.print("  Not available !");
    }

    if(newBalance_int < DRUG[D].price){
    Serial.print("  No enough balance !");
    }


  if((cell!=0) && (DRUG[D].amount!=0) && (newBalance_int >= DRUG[D].price)){


  GotoCell();
  delay(5000);

  readBarcode();
```

If the amount and balance is OK, then “GotoCell” function start working to move the head toward the required cells.

When the head is at the front of the required cell, then “readBarcode” function is activated to read the serial barcode from the drug’s pocket. if the Barcode is correct, then “PullDrug” function begin to pull one pocket from the cell.

If the pull is done (checked by IR sensor), then “modifySDcard” function is called to modify the amount of the taken drug.

If not, then the code returns to the first of this loop to check the next required drug.

```
 if(RdBarcode==DRUG[D].sdBarcode){
    Serial.println("correct");
delay(2000);

PullDrug();
if(Pull==0){
   Serial.println("Pull Failed!");
   return;
}
else
if(Pull==1){
Serial.println("Pull Done");
delay(2000);

newBalance_int=newBalance_int - DRUG[D].price;
modifySDcard();
```

After pulling all required drugs, then " GotoOutlet" function starts to move the head to the outlet position.

```
  GotoOutlet();
  delay(2000);

   modifyRFID();

  RotatePocket();
  if(Del==1)
    State="Drug is delivered";

  else
  State="Drug not delivered";

  delay(2000);

  Serial.println(State);


  Print();
  delay(2000);

  }
  }
```

Then, “modifyRFID” function is called to erase the pulled drugs from the patient card and to discount their total price from the main balance.

After that, “RotatePocket” function starts to rotate the pocket 180 degrees towards the outlet gate.

Finally, the “Print” function activates the printer to print a short report about this dispensing process. Then the main loop starts again by the “Homing” function to be ready for a new dispensing process.

Chapter 8

Validation Result, Recommendations and Manual of the System

8.1 Validation Result of the System

In order to systematically demonstrate that the specifications and needs for system have been met, by conducting the following empirical tests after finalizing the design of the system.

8.1.1 Testing the Dispensing Process

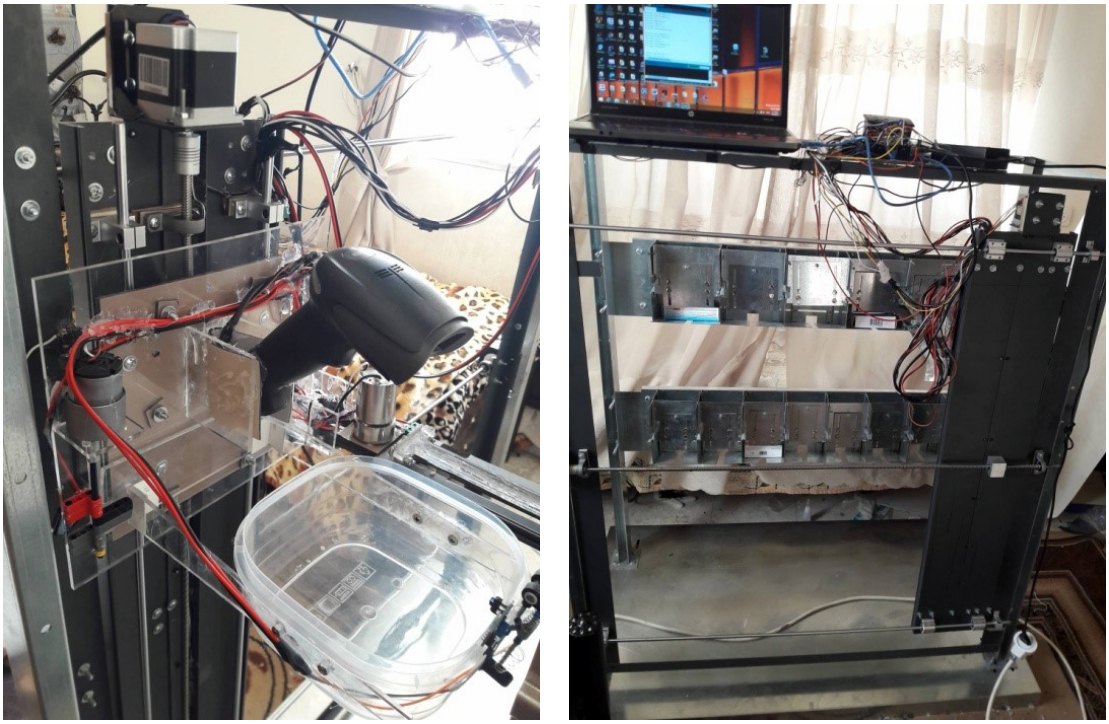
In order to make sure that this machine achieves its main goal (dispensing drugs), by conducting the following test . Note that the patient card contains one prescribed drug, and enough balance.

The results of this test were as follows:

1. **Homing process**:

   When the machine is run for the first time, then the moving head is moved to its home position (which is detected by limit switches). This is

done firstly by close the rotational pocket, pulling mechanism. And then moving the head to the rightmost and to the upper position as shown in the figure 8.1.



(a) (b)

Figure 8.1: Homing process.

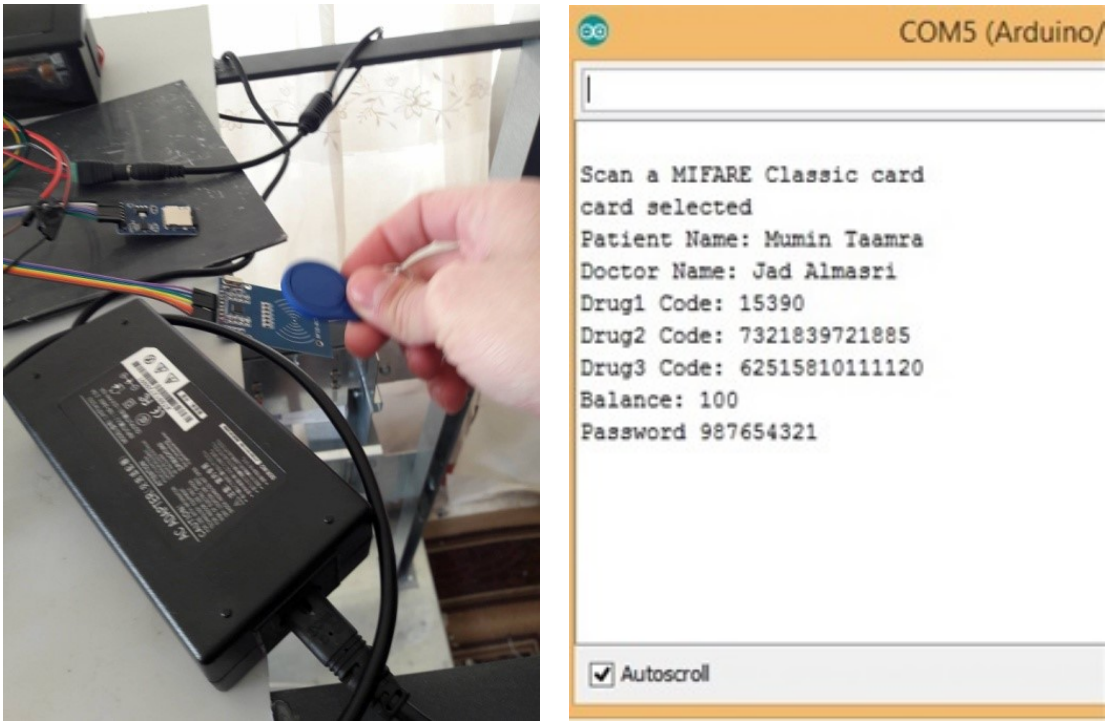
Note that the homing process is occurred also after each dispensing process.

2. **Reading the patient card**:

   The first step in the dispensing process is to read the RFID card. The machine read the card successfully, and all the information is stored in matrix in the microcontroller.

   The figure 8.2 below shows the data that was read from the card, and printed on the serial monitor in the PC.

   Note that the system was designed to read up to 3 types of drugs for each user.



(a) (b)

Figure 8.2: Reading the patient card.

3. **Reading the SD card**:

   After that, the Arduino begin a serial connection to the SD card and find out the required data from the file.

   All required data about the needed drugs is read from the file and stored in specific variables.

   The serial monitor in figure 8.3, shows this process.

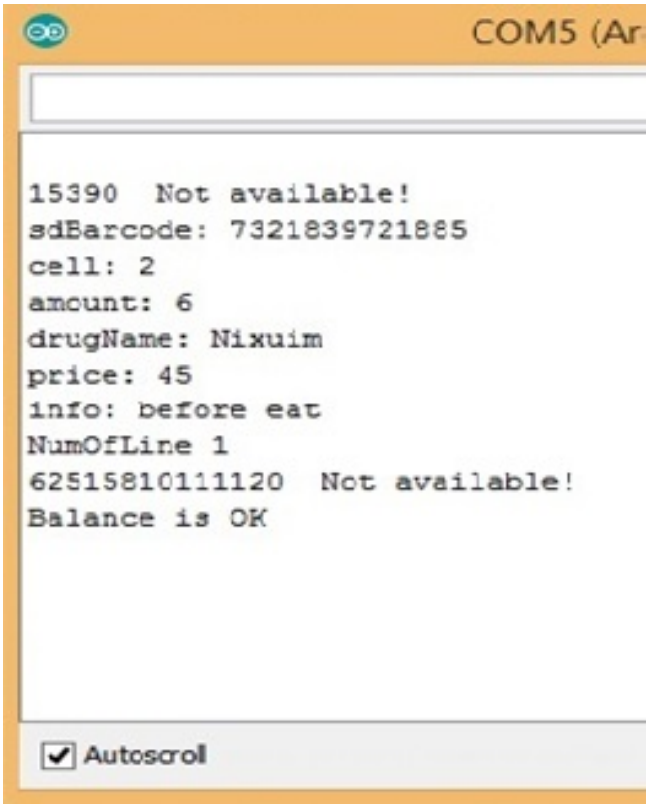


Figure 8.3: Reading the SD card.

4. **Moving the head**:

   If the required drugs are available, and the balance is enough to buy them, then the head will be moved by stepper motors toward the cell that contains the required drug(as read from the SD card).

   This process is shown in the figure 8.4.

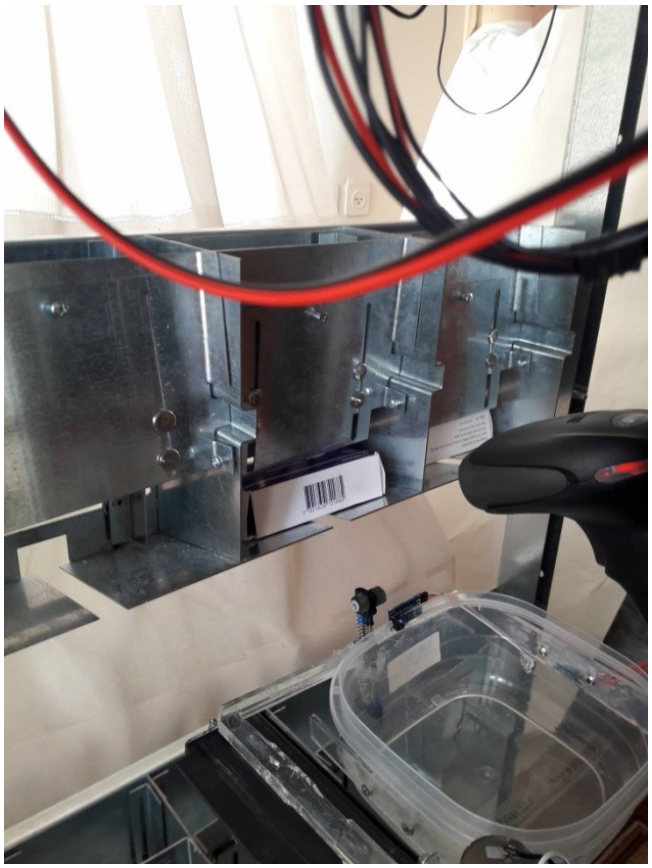


Figure 8.4: Moving the head.

5. **Reading the Barcode Serial**:

   after that, the barcode reader is activated to read the barcode serial of the drug to verify that is the required drug.

   In the same time, the head is moved vertically to cover the region that may contain the barcode serial.

   The figure 8.5 shows how this process happens, and the barcode is printed on the serial monitor as shown.

(a) (b)

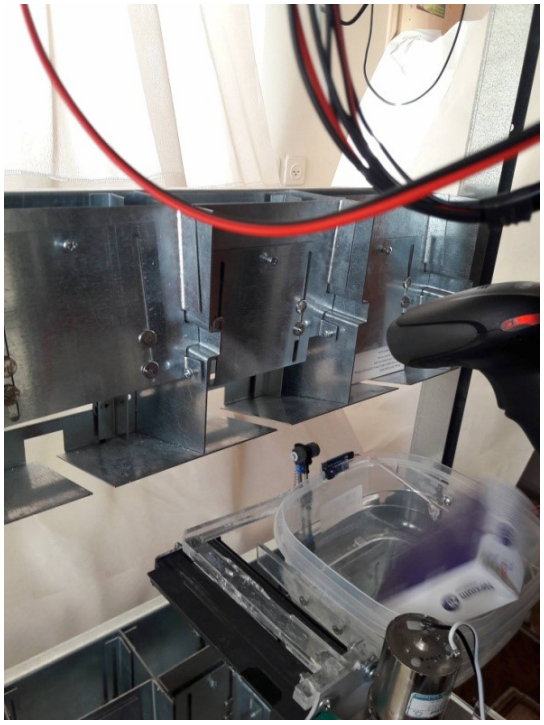
Figure 8.5: Reading the Barcode Serial.

6. **Pulling drug process**:

   If the barcode was correct as required, then the pulling mechanism is activated to pull the drug pocket as shown in the figure 8.6.

   Note that the IR sensor detects if the pulling process was done or not.

(a)



(b)

(c)

Figure 8.6: Pulling drug process.

If there is more than one required drug, then these processes are repeated for each available drug.

7. **Move the head to the outlet**:

   If the pulling process was done, then the head is moved to the outlet position as shown in figure 8.7.

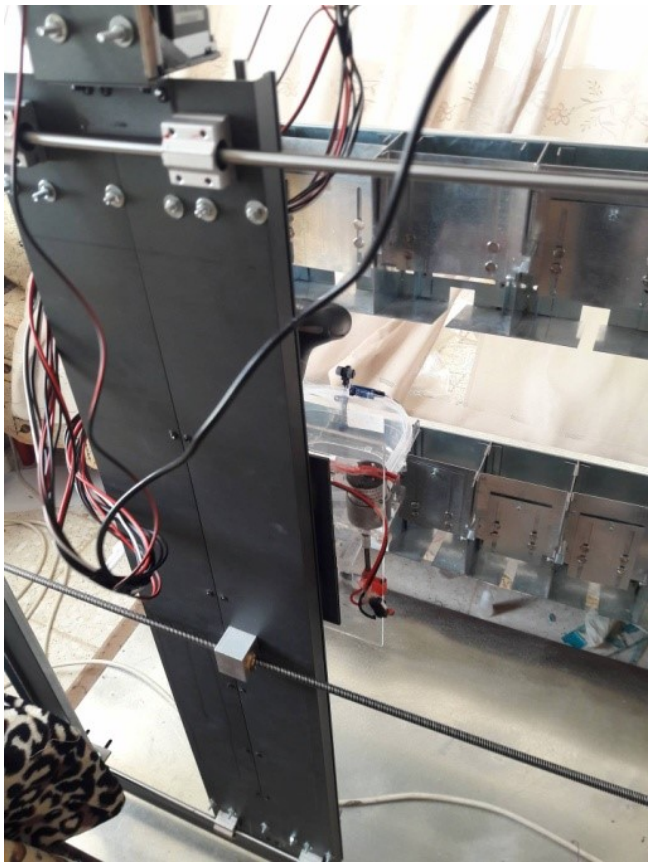
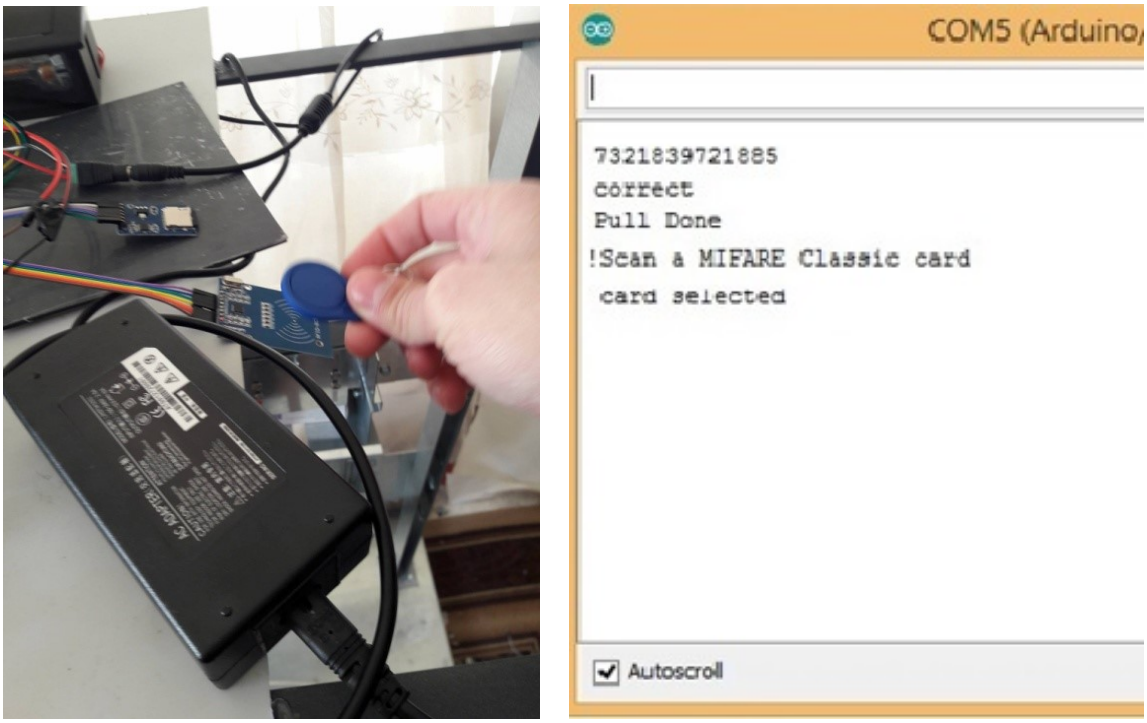


Figure 8.7: Move the head to the outlet.

8. **Modify patient card and SD card**:

   After that the file in the SD card is modified to reduce the amount of the available drugs in the machine. And also, the patient is required to insert his card again in order to erase the pulled drugs and to store the new balance.

   Figure 8.8 shows this process.

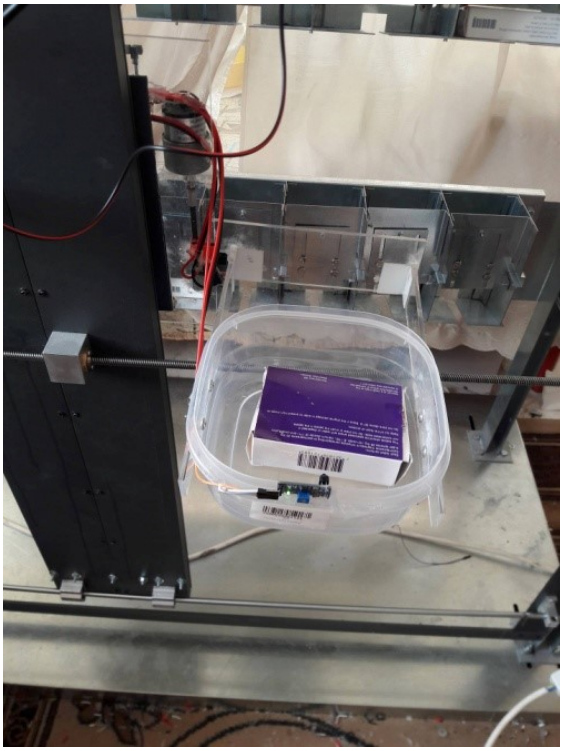


(a) (b)

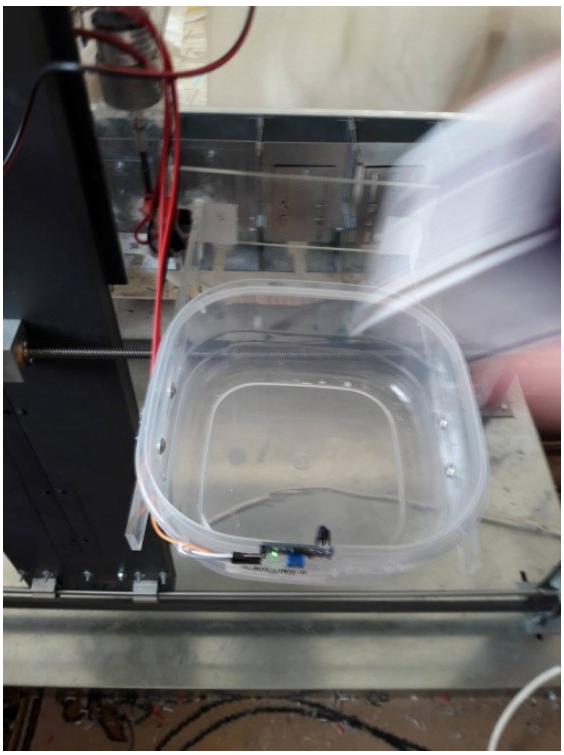
Figure 8.8: Modify patient card and SD card.

9. **Rotate the pocket**: After that, the pocket is rotated toward the outlet, wait until the patient takes his drug. This is detected using the IR sensor as shown in figure 8.9.

(a)



(b)



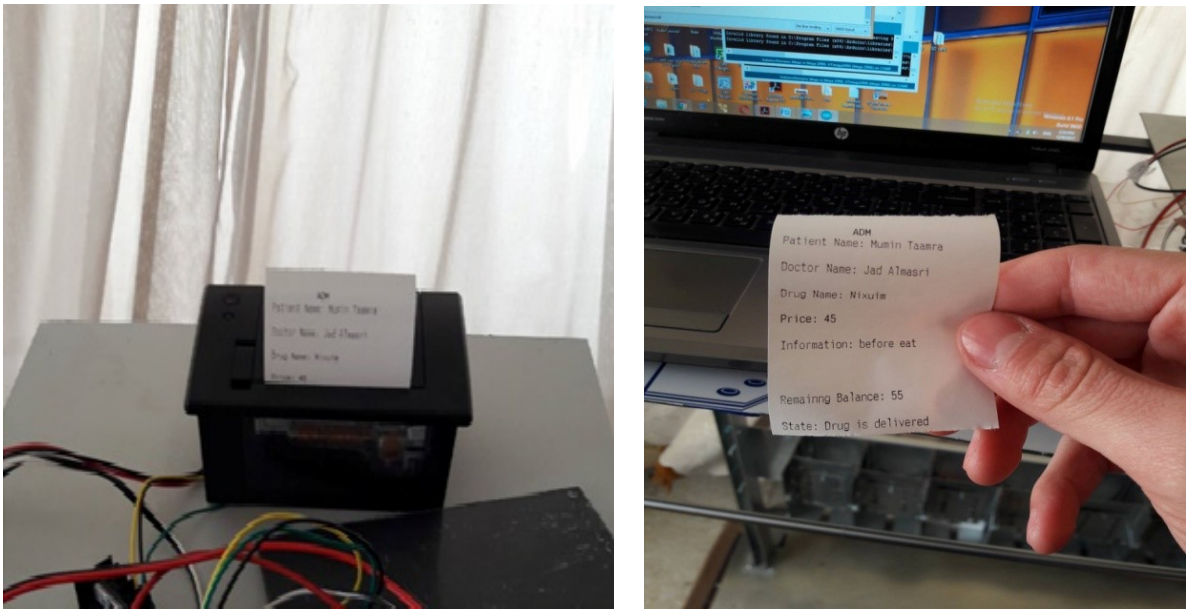
(c)

Figure 8.9: Rotate the pocket.

10. **Printing process**:

    After the success of the dispensing process, then the thermal printer is activated to print a small paper contains information about the dispensing process.

    The paper is shown in the figure 8.10.



(a) (b)

Figure 8.10: Printing process.

After that the machine moves the head to its home position to be ready for a new dispensing process.

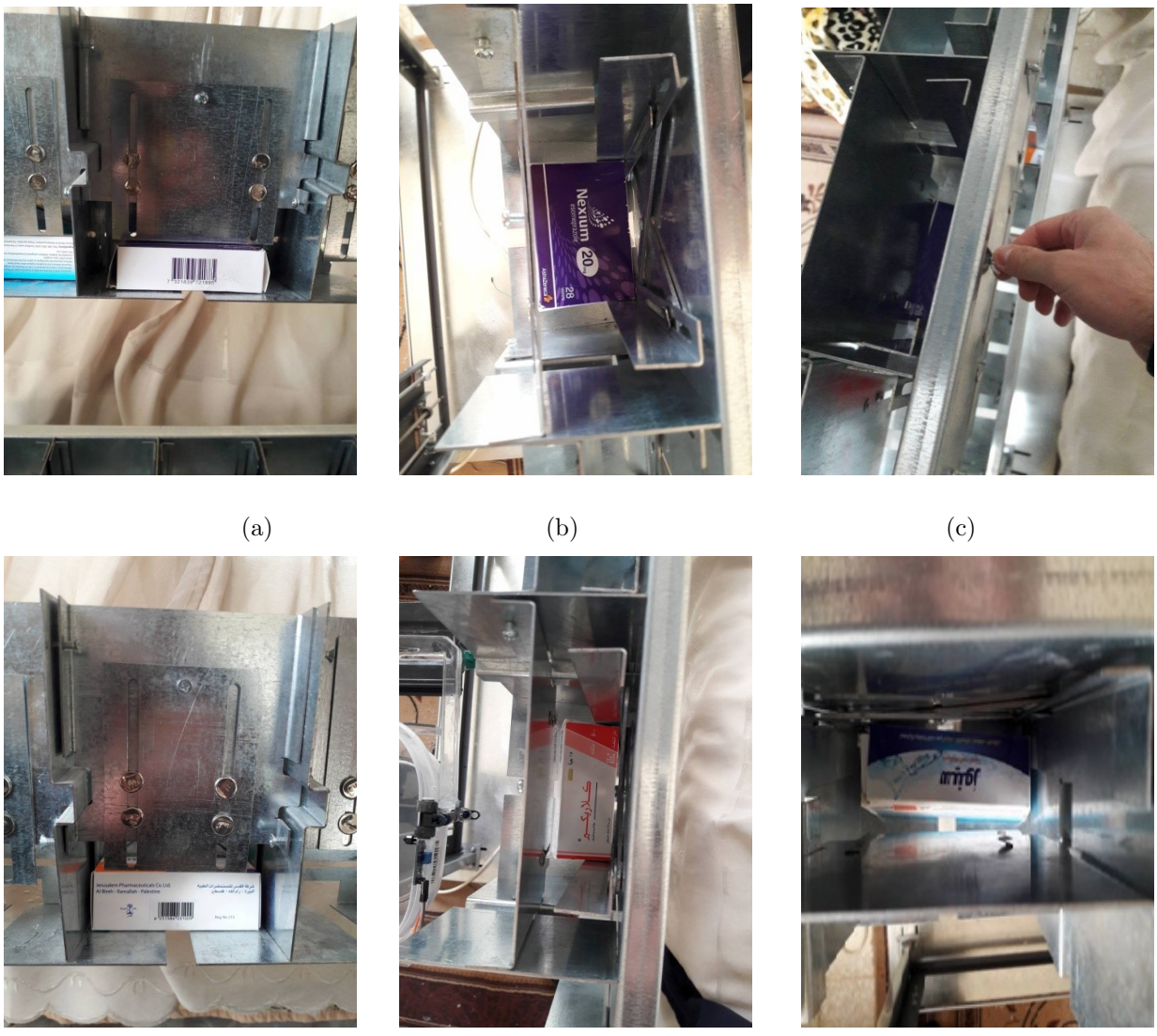
8.1.2 Reliability

This objective was achieved through dispensing two kinds of medicines ... prescribed medicines (that need a doctor prescription), and over the count - non prescribed medicines as shown in figure

And also it gives the patient some information about the drugs that he requested.

8.1.3 Compatibility

The compatibility was successfully achieved, so that this machine is compatible with all kinds and sizes as shown in the figure 8.11.



(a) (b) (c)

(d) (e) (f)

Figure 8.11: The compatibility of the machine.

8.1.4 Capacity

This machine was designed to accommodate 50 types of drugs. But in the implementation, just two rows of cells (12 types) are contained in this machine because of the high cost.

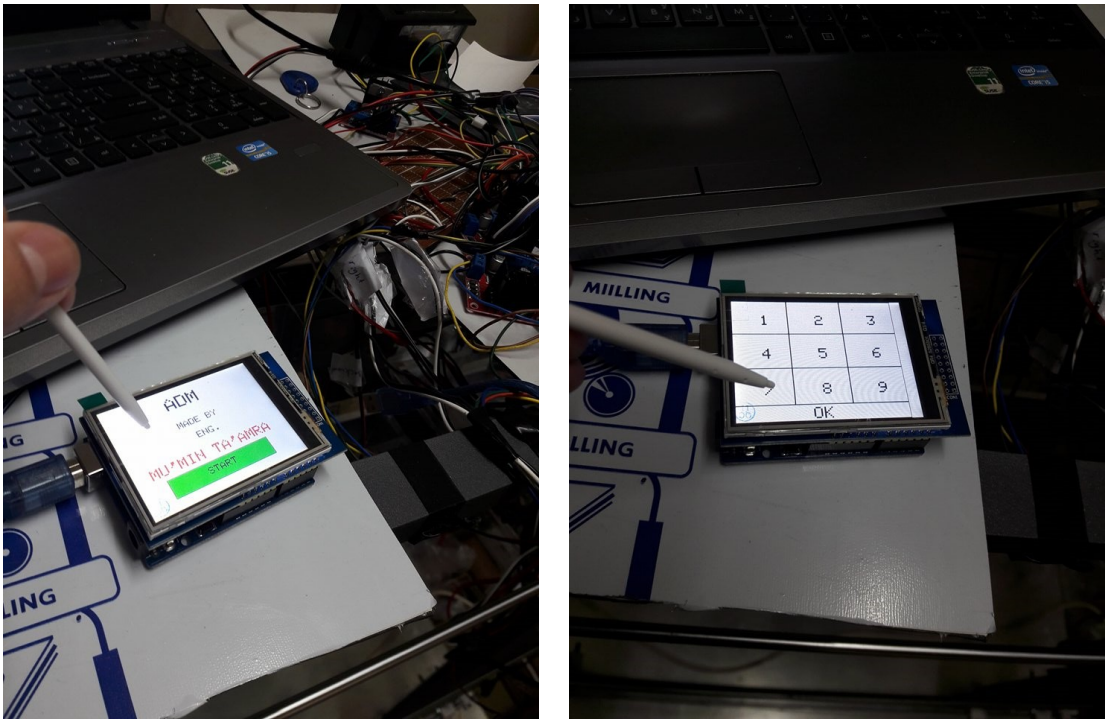
However, the machine now in its current dimensions can contain another 2 rows without any change in the machine size.

8.1.5 Safety

This machine achieves high level of safety, so it is using Barcode reader to verify the type of drugs.

8.1.6 Friendly User Interface

This point was achieved by using a touch screen that gives the user instructions while using the machine, and also the user can enter his password easily using the same screen as shown in the figure 8.12.



(a) (b)

Figure 8.12: Friendly user interface.

8.2 Difficulties and Challenges

Many challenges appeared while implementation this machine. The most important of them and their solutions were as follow:

1. Lack of financial support, this made it difficult to purchase the necessary materials.

2. The high cost of machining the cells of drugs, so just two rows were implemented in this machine. Whereas the design contain five rows.

3. The power screws and guides were not available in the required dimensions, so the used guides and screws are smaller in diameter and length than those in the design.

4. Some electrical components like RFID reader, Barcode reader and SD card reader are using the same serial port to interface with the Arduino (SPI). The solution for this problem is to use select slave pin for each device so that just one of them is connected to the master(Arduino) in the same time.

8.3 Recommendation for the Machine

After completing this machine design, there are many points of recommendations to improve the design going forward in the future.

1. To increase the capacity of the machine, so it could contain more types of drugs.

2. To be developed as online system that will link all health care system with this machine.

3. To increase the speed of dispensing by using special Kit module for stepper motors.

4. To use this machine to dispense another products in addition to drugs, like milk for babies.

5. To use an additional lead screw in the horizontal motion to achieve stability.

6. To use temperature control system in this machine.

8.4 Manual for the System Operation

Users of this system are three types of people who are pharmacist, doctor and patient. Each type of these users needs to follow different steps while using this system. Here are the different manuals of this machine.

8.4.1 Pharmacist's Manual

1. The pharmacist need to full the machine with the appropriate drugs from the back side of the machine, and adjust the cells sizes to fit the dimensions of each drug . Keep with consideration to put the barcode side in the outer side of the cell as shown in the figure 8.13.

Figure 8.13: The barcode side in the outer side of the cell.

2. Insert the SD card of the machine in his PC, then open the “CSV” file in the card as shown in figure 8.14

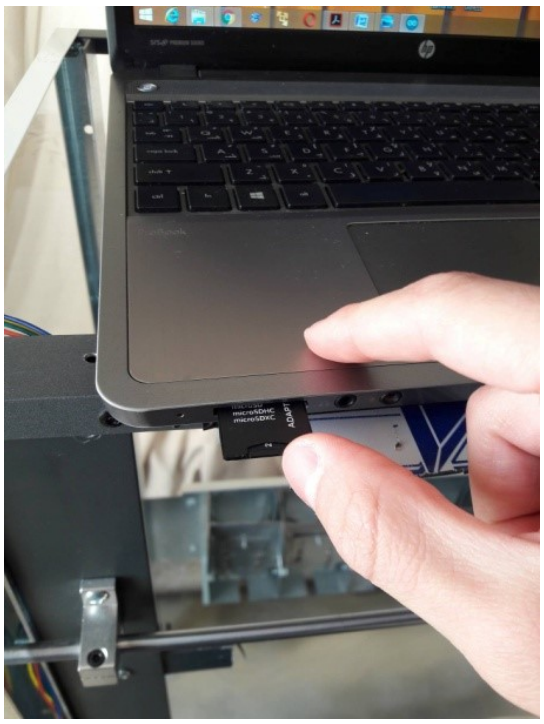


Figure 8.14: Insert the SD card of the machine in the PC.

3. Fill out the drugs data in the columns assigned to it. This includes drug serial barcode, cell number, amount, price, some using instructions about each drug. Look at figure 8.15.

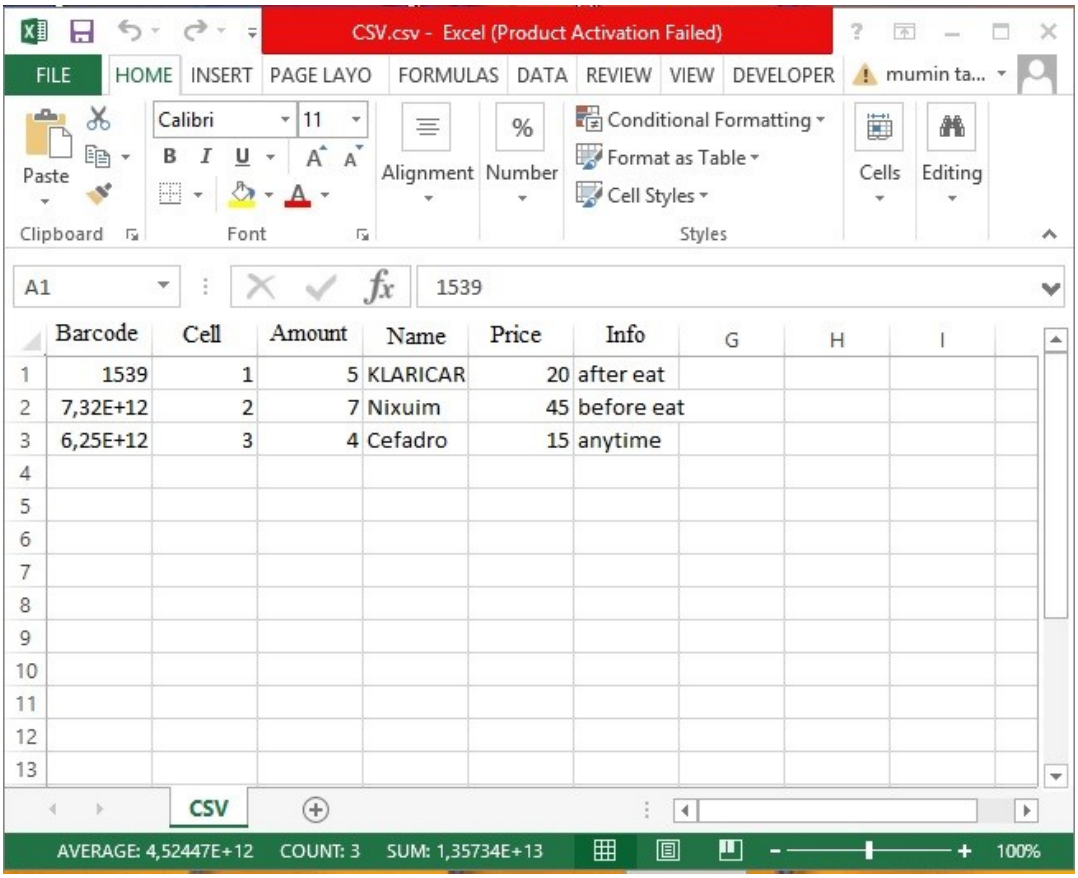


Figure 8.15: Fill out the drugs data in the columns assigned to it.

4. Return back the SD card to its slot in the machine, close the machine, and then turn on the machine. Look at figure 8.16.

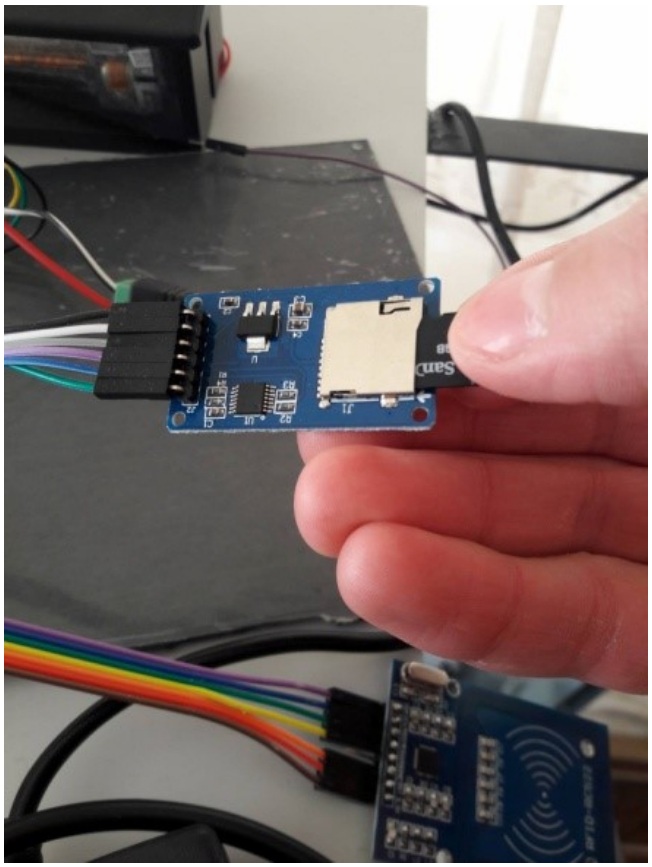


Figure 8.16: Return back the SD card to its slot in the machine.

The machine now is ready to use.

5. The pharmacist is required to set the password and charge balance for the users in their own cards using a special device or an Arduino Uno with RFID shield by using the "$Writing_on_RFID_card$" code (Appendix (4)) as shown in the figure 8.17.

```
Writing_on_Rfid_card    functions
int Patientblock=2;//this is the block
int Doctorblock=4;
int BarCode1block= 5;
int BarCode2block= 6;
int BarCode3block= 8;
int balanceblock=9;
int Passwordblock=10;


byte balance[16]={"100"};
byte Password[16]={"987654321"};
```

(a)

(b)

Figure 8.17: Writing on RFID card.

8.4.2 Doctor's Manual

The doctor should fill his prescription in the patient's card as follow:

1. Insert the doctor name.
2. Choose the serial barcode for the required drugs (maximum 3 types).
3. Load these data to the RFID writer.
4. Put the card near the writer for 2 seconds until all data is stored in the card.

This is done using a special device or an Arduino Uno with RFID shield by using the $\backslash Writing_o n_R FID_c ard$" code (Appendix (4)). Look at the figure **??**.

Writing_on_Rfid_card | functions

```
byte patient[16] = {"Mumin Taamra"};//an array
//byte blockcontent[16] = {0,0,0,0,0,0,0,0,0,0,
byte doctor[16]={"Jad Almasri"};

byte Barcode1[16]={"15390"};
byte Barcode2[16]={"7321839721885"};
byte Barcode3[16]={"62515810111120"};
```

(a)

(b)

Figure 8.18: Writing on RFID card.

8.4.3 Patient's Manual

To take the required medications form this machine, follow these steps:

1. Each patient must have a RFID card special for this machine (Bought from the owner pharmacy) as shown in the figure 8.1(a)(97).

Figure 8.19: RFID card for ADM machine.

2. The card must be charged with enough credit (from the owner pharmacy).

3. put your card close to the assigned position for 2 seconds to read the data.

4. Follow the instructions that appear on the screen (write password .. select if you want to by OTC drugs or prescribed), look at the figure8.20.

Figure 8.20: The instructions on the screen.

5. Wait until the machine pull your drug.

6. insert your card again to erase the pulled drugs and discount their price.

7. Take your drug from the outlet gate.

8. Wait a while until the printer print out a small detailed paper about this process.

Chapter 9

Project Management

This chapter will discuss the management and planning of the project. Firstly, it will talk about the division of the main system into subsystems and small components. Then, it will talk about the main tasks and time table. The final section will be about the Budget and table of costs.

9.1 Work Breakdown Structure

In order to organize the team work in this project, the main system (ADM) was divided into subsystems and smaller components.

The Work Breakdown Structure (WBS) is a tree structure which shows a subdivision of effort required to achieve an objective. It is a useful tool that provides the necessary framework for detailed cost estimating and guidance for schedule development.

The WBS of the ADM machine is shown in figure 9.1, the system is divided into four levels as shown in the figure. Each level has the same block color.

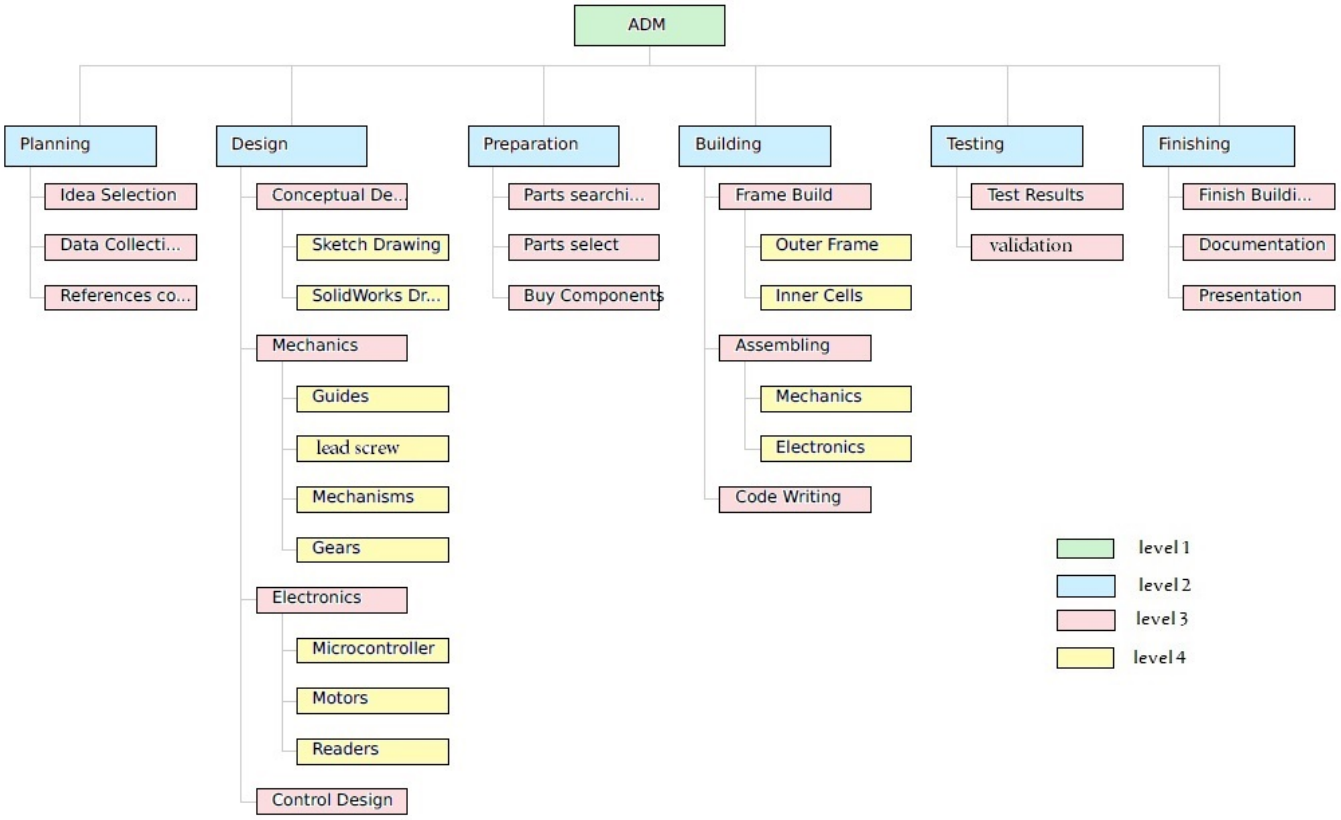


Figure 9.1: Work Breakdown Structure for ADM.

9.2 Schedule and Gantt chart

In order to achieve the main objective of this project, the work was divided into several tasks as shown in table 9.1.

Table 9.1: Tasks description for ADM.

	Task description
T1	Selection of Idea
T2	Collecting the Data
T3	Collecting References
T4	Make the Questionnaire
T5	Select an initial design
T6	Draw the selected design
T7	Selection of the Mechanics
T8	Selection of the Electronics
T9	Documentation
T10	Prepare the 1st presentation
T11	Searching about the requested components
T12	Buy the mechanical and electronic parts
T13	Build the project
T14	Assembling the mechanical and electronic parts of the system
T15	Write the Arduino code
T16	Test the result
T17	Correction the errors
T18	Documentation
T19	Make a final adjustments on the text
T20	Prepare for the final presentation

The Gantt chart is shown in table 9.2; it shows the distribution of the tasks along the second semester 2016/2017.

Table 9.2: Gantt chart for the second semester.

Task/Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14
T1														
T2														
T3														
T4														
T5														
T6														
T7														
T8														
T9														
T10														

The schedule of the first semester 2017/2018 tasks is shown in table 9.3.

Table 9.3: Gantt chart for the first semester.

Task/Week	1	2	3	4	5	6	7	8	9	10	11	12	13	14
T11														
T12														
T13														
T14														
T15														
T16														
T17														
T18														
T19														
T20														

9.3 Budget and Costs

The actual cost of this machine components and the number of each component is shown in Table 9.4. The total cost of the ADM project is calculated below:

Table 9.4: Total cost of the ADM.

Part name	No. of required parts	Cost of each part (JD)	Total (JD)
Outer Frame	1	200	200
Inner Frame	1	200	200
Guides and Sliders	4	50	200
Lead Screws and bearings	2	40	80
DC Motors	2	15	30
Stepper Motors	2	40	80
Barcode Reader	1	80	80
RFID reader	1	40	40
Printer	1	100	100
Screen	1	40	40
Arduino	2	25	50
Other electrical components	1	100	100
Total Cost			1200

References

[1] Wikipedia website. Vending machine. https://en.wikipedia.org/wiki/Vending_machine. Accessed: 2017-03-10.

[2] Eric Jaffe. Old world, high tech. *SMITHSONIAN MAGAZINE*, 2006.

[3] NAMA. History of vending and coffee services. *National Automatic Merchandising Association*, 2016.

[4] Alibaba. 2015 new arrive modern medicine vending machine. https://wholesaler.alibaba.com/product-detail/2015-new-arrive-modern-medicine-vending_1946573848.html. Accessed: 2017-03-12.

[5] Daily Mail. Medicine vending machines that dispense prescriptions 24 hours a day go on trial. http://www.dailymail.co.uk/health/article-1288434/Medicine-vending-machine-dispenses-prescriptions-pharmacist-launched.html. Accessed: 2017-03-12.

[6] InstyMeds. Automated medication dispensing system. http://www.instymeds.com/index.php?page=services#1. Accessed: 2017-03-13.

[7] Ken Rosenblum. Automatic prescription drug dispenser, October 28 2008. US Patent 7,444,203.

[8] SUJAY R PATIL RAJKIRAN K, SOLOMON SUJITH V B. Any time medicine vending machine. *ELECTRONICS AND COMMUNICATION ENGINEERING*, 2014.

[9] Harold J Liff, Brian T Hart, Robert L Wallace, and Arthur A Berube. Drug dispensing system, February 3 1998. US Patent 5,713,485.

[10] Richard G. Budynas and J. Keith Nisbett. *Shigley's Mechanical Engineering Design*. 9 edition.

[11] Arduino. Arduino. https://www.arduino.cc/en/Main/arduinoBoardMega2560. Accessed: 2017-04-29.

[12] Bill Earl. adafruit. https://learn.adafruit.com/all-about-stepper-motors/what-is-a-stepper-motor. Accessed: 2017-04-29.

[13] CNC4YOU. Cnc4you. https://www.cnc4you.co.uk/Stepper-Motor/Nema23-1.5Nm/Stepper-Motor-1.5Nm-57HS64-3008-Nema23. Accessed: 2017-04-29.

[14] Technolab. Technolab. http://www.technolab.ps/#/products/page/1/view/505. Accessed: 2017-04-29.

[15] CNC4YOU. Cnc4you. https://www.cnc4you.co.uk/Microstepping-Driver/Stepper-Motor-Driver-4.5A,-50V-CNC-Microstepping-CW5045. Accessed: 2017-04-29.

[16] ebay. ebay. http://www.ebay.com/itm/New-Encoder-600-P-R-5V-24V-Incremental-Rotary-AB-2-Phase-6mm-Shaft-/162190810777. Accessed: 2017-04-27.

[17] made in china.com. made in china.com. http://szocomtech.en.made-in-china.com/product/OKIxTlmCnZWz/China-Msr605-RS-232-USB-Msr-Magnetic-Card-Reader-Writer.html#. Accessed: 2017-04-25.

[18] AliExpress. Aliexpress. https://www.aliexpress.com/item/Long-Laser-USB-Port-Handheld-Barcode-Scanner-Bar-Code-Reader-UPC-EAN-Scann32804385996.html?spm=2114.40010308.4.11.6YlBs5. Accessed: 2017-04-30.

Appendices

Appendix 1 (Doctors' Questionnaire)

استبيان للأطباء حول مشروع تخرج بعنوان: ماكنة لتزويد الأدوية بشكل آلي

نحن الطالبان : مؤمن حسين تعامرة وعبيدة خالد أبو جاموس ندرس الهندسة الميكانيكية تخصص هندسة الميكاترونيكس في جامعة بوليتكنك فلسطين ، نقوم بعمل استبيان حول فكرة تصميم ماكنة لتزويد الأدوية الطبية للمرضى بشكل آلي ،وهو مشروع يهدف لخدمة المرضى بالوصول للدواء باي وقت على مدار ٢٤ ساعة تماما كخدمة الصراف الالي الخاص بالبنوك وذلك لتسهيل الحصول على الأدوية في الأوقات المتأخرة وفي المناطق النائية ..

نرجو من حضرتكم الإجابة على هذا الاستبيان لمساعدتنا في دراسة أهمية المشروع وإمكانية تطبيقه ..

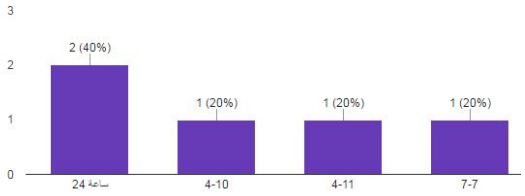
١. اسم العيادة الطبية الصحي

عيادة بيت لحم
الحكمة
حوسان الطبية
ترقوميا الطبي
الدوحة الطبي

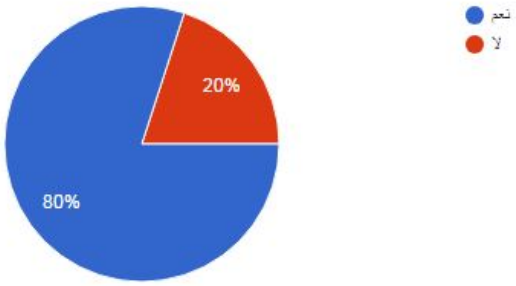
٢. عنوان العيادة الطبية الصحي

بيت لحم
نفوع
حوسان بيت لحم
ترقوميا ، الخليل
دوحة بيت لحم

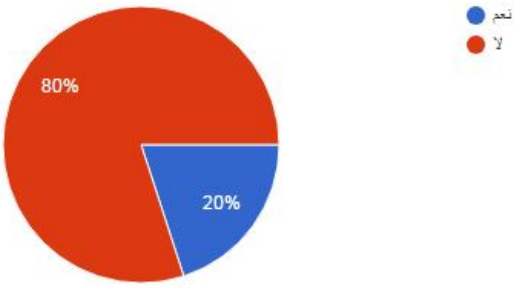
٣. ساعات الدوام اليومي



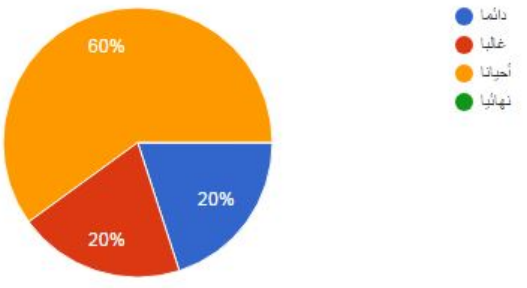
٤. هل يوجد صيدلية قريبة من مكان المستوصف او العيادة طبية



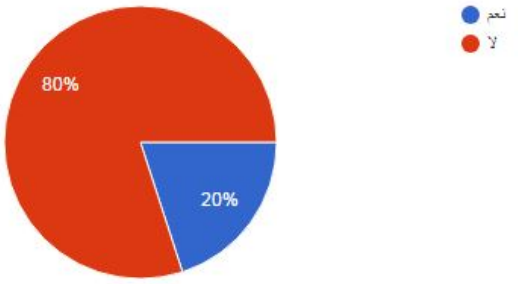
٥. هل الصيدلية تبقى مفتوحة في ساعات الليل المتأخر



٦. هل الصيدلية تبقى مناوبة في أيام الجمعة



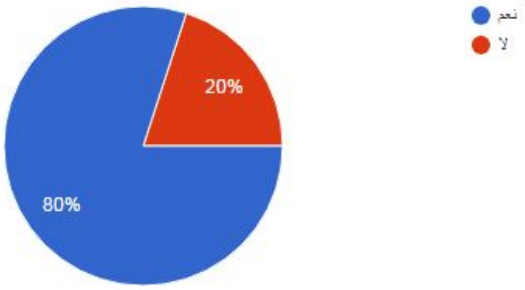
٧. هل تسد الصيدليات الموجودة حاجة المواطنين وفي جميع الأوقات



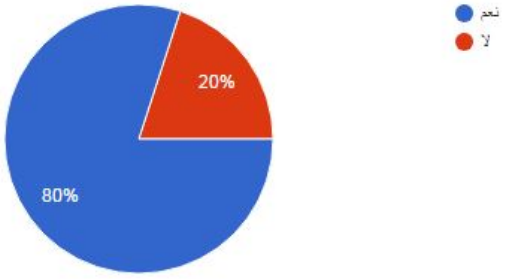
٨. كم عدد المرضى الذين يأتون للمركز ليلا

5
10-15
8
10-20
الى 2 5

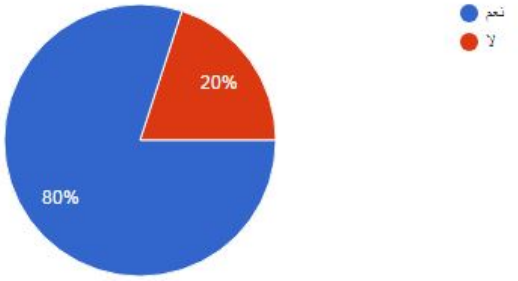
٩. هل يجد المرضى صعوبة في الحصول على الدواء بعد وصفه



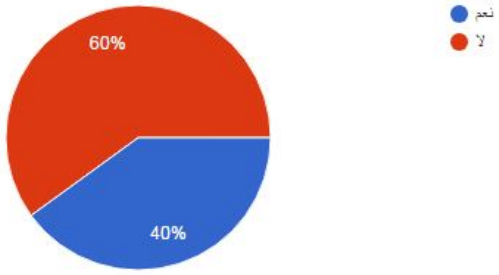
١٠. هل تؤيد فكرة عمل ماكنة آلية لتزويد الأدوية



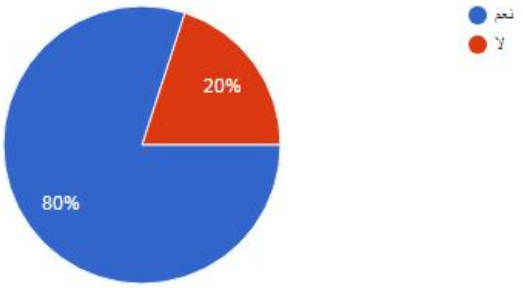
١١. هل ترى أن السوق بحاجة إلى ماكنة كهذه لسد احتياجاتهم الصحية



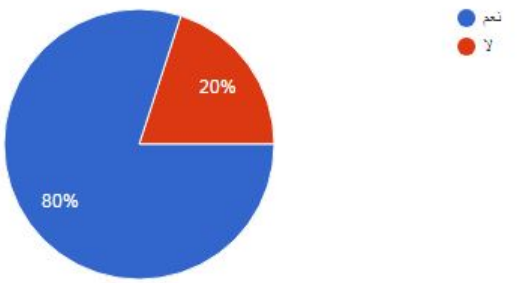
١٢ هل تفضل أن تكون هذه الماكنة بالقرب من الصيدليات أم في مناطق بعيدة



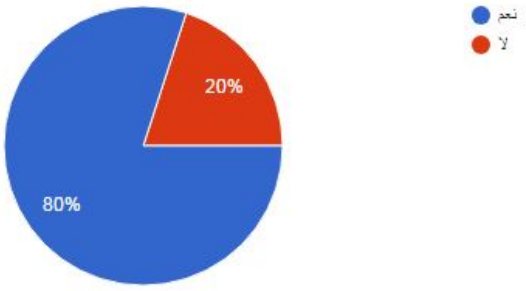
١٣. هل تؤيد فكرة استعمال بطاقة الكترونية بدلا من الرشيتة الورقية للوصفة الطبية



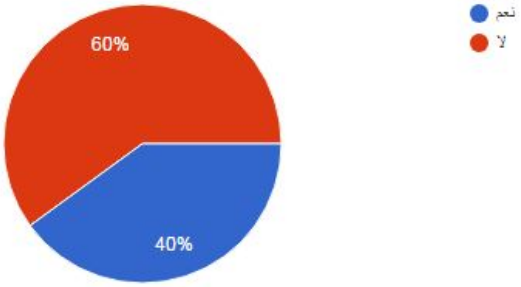
١٤. هل تؤيد فكرة استعمال بطاقة الكترونية بدلا من الرشيتة الورقية للوصفة الطبية



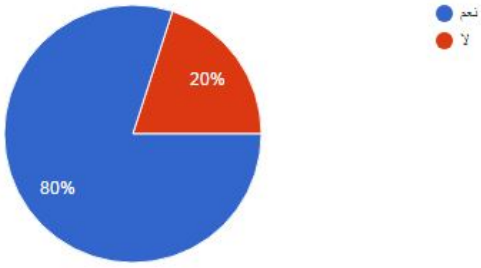
١٥. هل أنت على استعداد للتعاون مع الصيدليات لوضع ماكنة تتبع للصيدلية في عيادتك



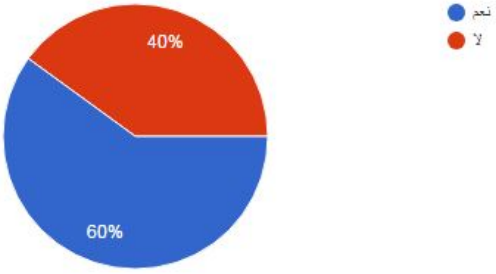
١٦. هل أنت على استعداد لإعلام المريض بطريقة استعمال الدواء وأعراضه الجانبية بدلا من الصيدلي



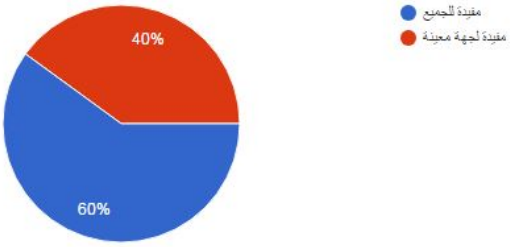
١٧. هل ترى أن الماكنة ستكون فعالة لتزويد الأدوية الاكثر طلبا



١٨. في حال كانت الماكنة تغطي ٥٠ نوعا من الأدوية ، هل تتوقع أن تسد حاجتها



١٩. هل ترى أن الفكرة مفيدة لجميع المواطنين .أم فقط لجهة معينة (صيدلية أو مركز صحي)



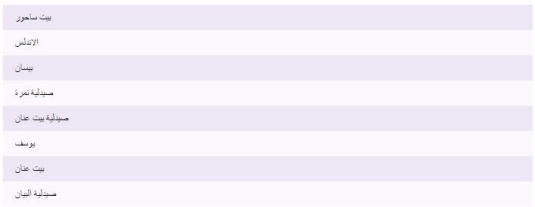
Appendix 2 (Pharmacists' Questionnaire)

استبيان للصيادلة حول مشروع تخرج بعنوان: ماكنة لتزويد الأدوية بشكل آلي

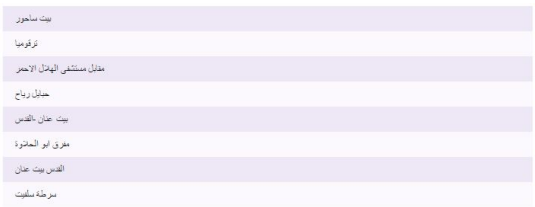
نحن الطالبان : مؤمن حسين تعامرة وعبيدة خالد أبو جاموس ندرس الهندسة الميكانيكية تخصص هندسة الميكاترونيكس في جامعة بوليتكنك فلسطين ، نقوم بعمل استبيان حول فكرة تصميم ماكنة لتزويد الأدوية الطبية للمرضى بشكل آلي ،وهو مشروع يهدف لخدمة المرضى بالوصول للدواء باي وقت على مدار ٢٤ ساعة تماما كخدمة الصراف الالي الخاص بالبنوك وذلك لتسهيل الحصول على الأدوية في الأوقات المتأخرة وفي المناطق النائية ..

نرجو من حضرتكم الإجابة على هذا الاستبيان لمساعدتنا في دراسة أهمية المشروع وإمكانية تطبيقه ..

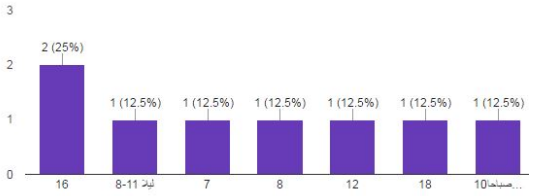
١. اسم الصيدلية



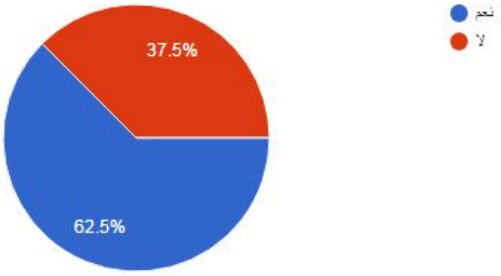
٢. عنوان الصيدلية



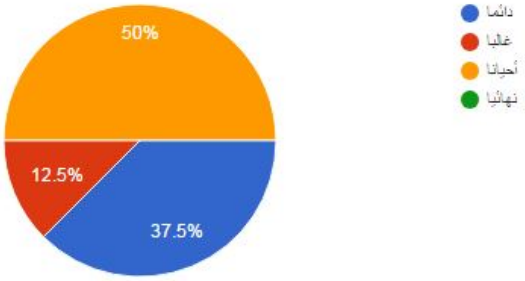
٣. ساعات الدوام اليومي



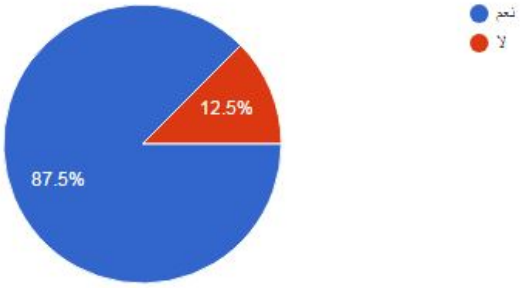
٤. هل الصيدلية تبقى مفتوحة في ساعات الليل المتأخر



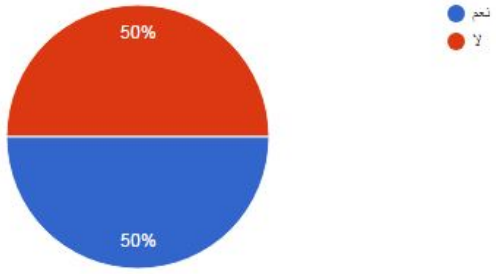
٥. هل الصيدلية تبقى مناوبة في أيام الجمعة



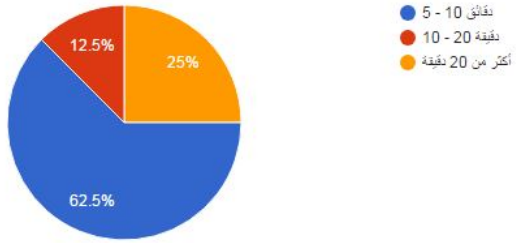
٦. هل الصيدلية قريبة من مكان علاج (مستوصف او عيادة طبية)



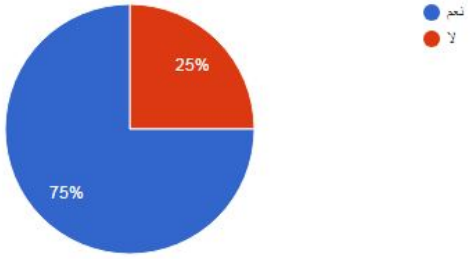
٧. هل تسد الصيدليات الموجودة حاجة المواطنين وفي جميع الأوقات



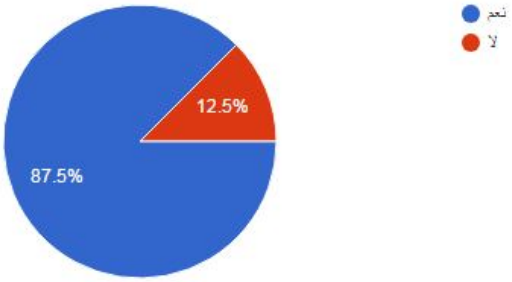
٨. كم من الوقت يستغرق المرضى الانتظار في الصيدلية لحين تسليمهم الدواء المطلوب



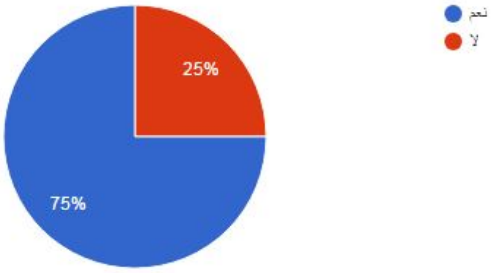
٩. هل يجد المرضى صعوبة في الحصول على الدواء بعد وصفه



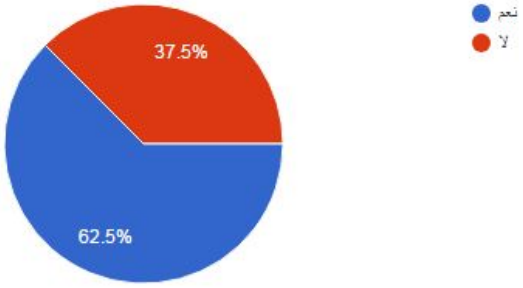
١٠. هل تحتاج في كثير من الاحيان للخروج من الصيدلية ولا تتمكن من اغلاقها في اي وقت



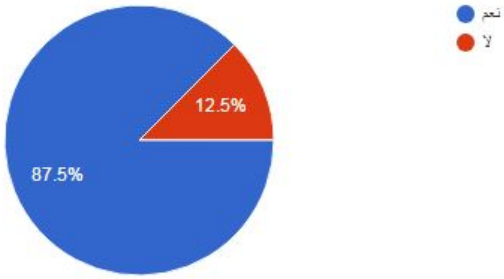
١١. هل تعاني من ضغط الزبائن في نفس الوقت



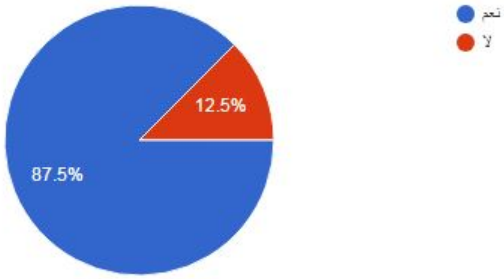
١٢ هل تعمل وحدك في الصيدلية في الفترة الواحدة



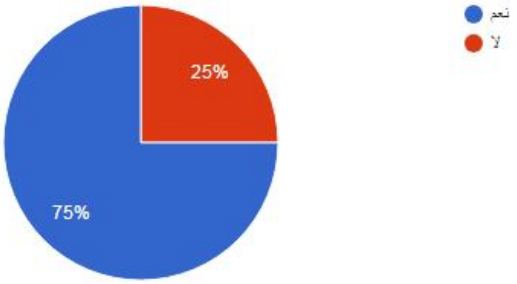
١٣. هل يجد الصيدلاني صعوبة في فتح صيدلية جديدة أو فتح فروع أخرى لصيدليته لتغطية حاجة المواطنين



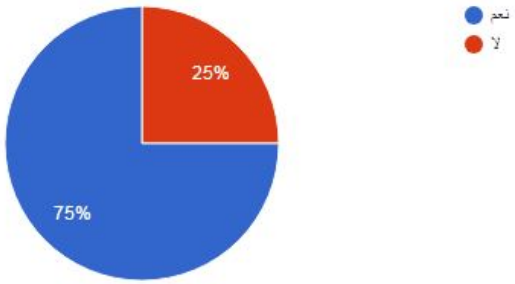
١٤. هل تؤيد فكرة عمل ماكنة آلية لتزويد الأدوية



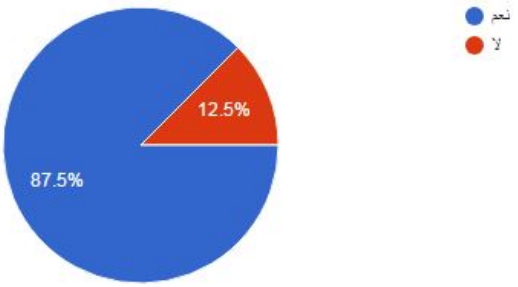
١٥. هل ستقوم بشراء هذه الماكنة في حال وجدت



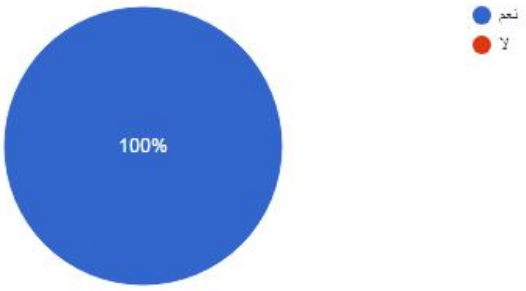
١٦. هل ترى أن السوق بحاجة إلى ماكنة كهذه لسد احتياجاتهم الصحية



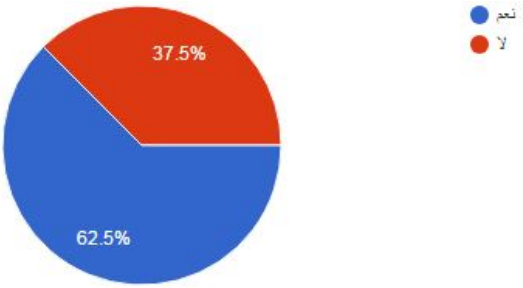
١٧. هل تفضل أن تكون هذه الماكنة بالقرب من الصيدليات أم في مناطق بعيدة



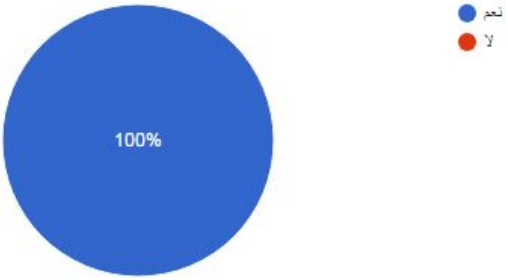
١٨. هل تؤيد فكرة استعمال بطاقة الكترونية بدلا من الرشيتة الورقية للوصفة الطبية



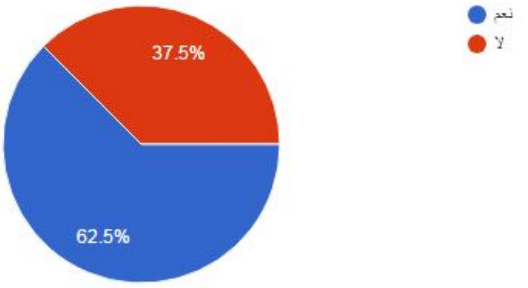
١٩. هل أنت على استعداد للتعاون مع المراكز الصحية البعيدة والعيادات النائية لوضع ماكنة تتبع لصيدليتك هناك



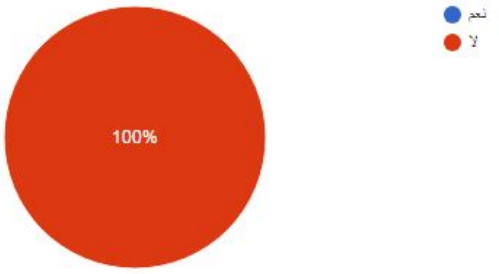
٢٠. هل باستطاعتك حصر الأدوية الأكثر طلبا عند المواطنين



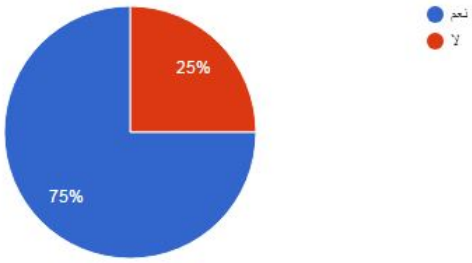
٢١. في حال كانت الماكنة تغطي ٥٠ نوعا من الأدوية ، هل تتوقع أن تسد حاجتها



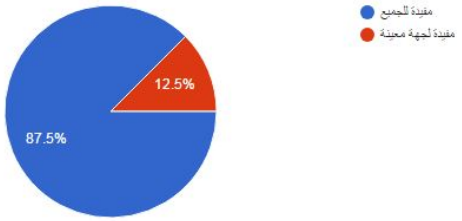
٢٢. هل جميع الأدوية تحتاج إلى وصفة طبية لشراءها



٢٣. هل تؤيد وجود أدوية في الماكنة يستطيع المواطن شراءها بدون وصفة طبية



٢٤. هل ترى أن الفكرة مفيدة لجميع المواطنين .أم فقط لجهة معينة (صيدلية أو مركز صحي)



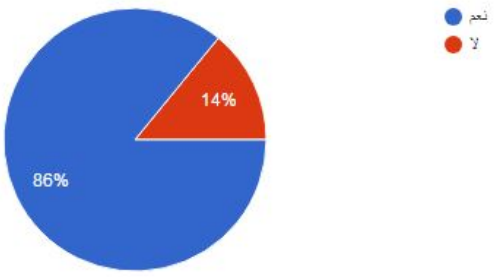
Appendix 3 (Public Questionnaire)

استبيان حول مشروع تخرج بعنوان: ماكنة لتزويد الأدوية بشكل آلي

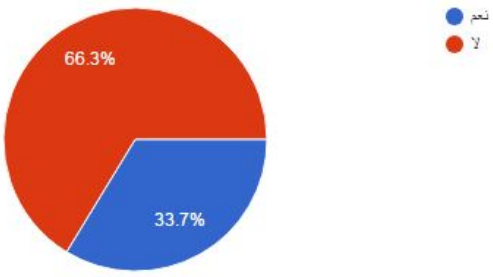
نحن الطالبان : مؤمن حسين تعامرة وعبيدة خالد أبو جاموس ندرس الهندسة الميكانيكية تخصص هندسة الميكاترونيكس في جامعة بوليتكنك فلسطين ، نقوم بعمل استبيان حول فكرة تصميم ماكنة لتزويد الأدوية الطبية للمرضى بشكل آلي ،وهو مشروع يهدف لخدمة المرضى بالوصول للدواء باي وقت على مدار ٢٤ ساعة تماما كخدمة الصراف الالي الخاص بالبنوك وذلك لتسهيل الحصول على الأدوية في الأوقات المتأخرة وفي المناطق النائية ..

نرجو من حضرتكم الإجابة على هذا الاستبيان لمساعدتنا في دراسة أهمية المشروع وإمكانية تطبيقه ..

١. هل يوجد صيدلية قريبة من منطقتك

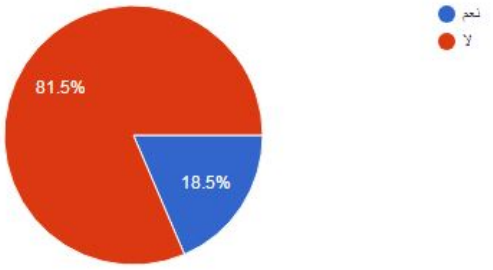


٢. هل الصيدلية القريبة تبقى مفتوحة في ساعات الليل المتأخر

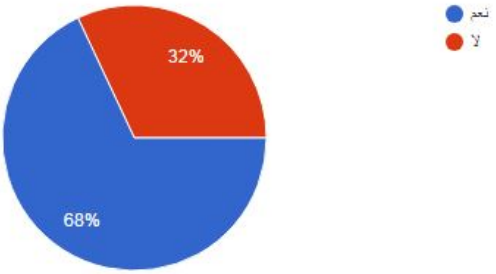


٣. هل يوجد صيدلية قريبة من مكان العلاج (مستوصف او عيادة طبية) القريب من منطقتك

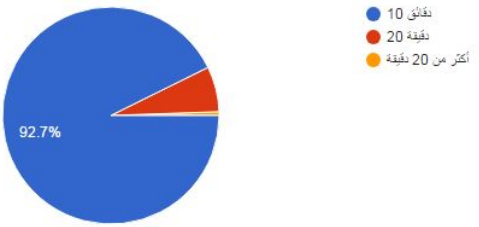
٤. هل تجد صعوبة في الحصول على الدواء بعد وصفه من قبل الطبيب



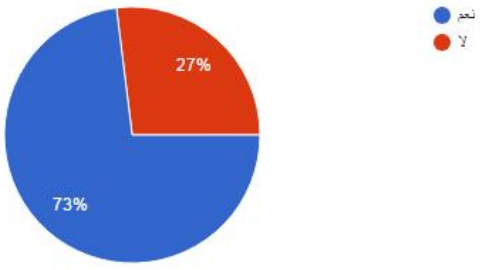
٥. هل حدث أن بحثت طويلا عن صيدلية مناوبة ليلا أو أيام الجمعة



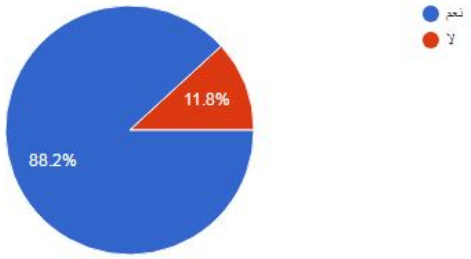
٦. كم من الوقت تستغرق الانتظار في الصيدلية لحين تسليمك الدواء المطلوب



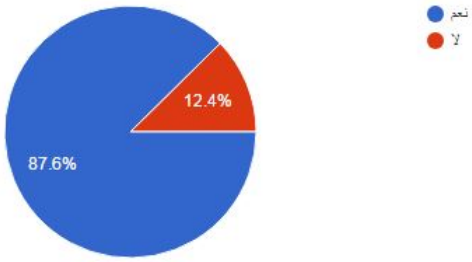
٧. هل انت راض عن خدمات الرعاية الصيدلية في المنطقة



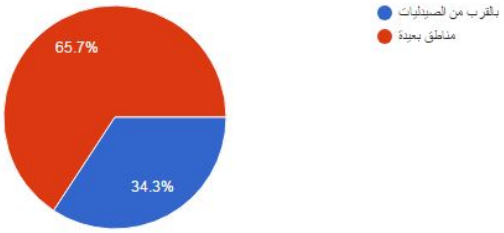
٨. هل تؤيد فكرة عمل ماكنة آلية لتزويد الأدوية



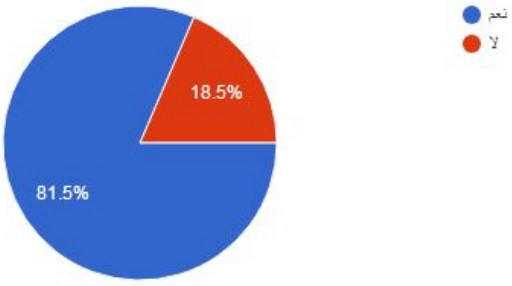
٩. هل ستقوم باستعمال هذه الماكنة في حال وجدت



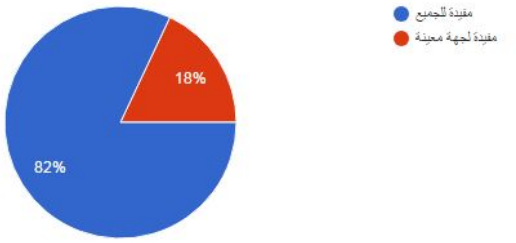
١٠. هل تفضل أن تكون هذه الماكنة بالقرب من الصيدليات أم في مناطق بعيدة



١١. هل تؤيد فكرة استعمال بطاقة الكترونية بدلا من الرشيتة الورقية للوصفة الطبية



١٢. هل ترى أن الفكرة مفيدة لجميع المواطنين .أم فقط لجهة معينة (صيدلية أو مستوصف)


مفيدة للجميع
مفيدة لجهة معينة
18%
82%

Appendix 4 (The ADM Arduino Code)

1. Arduino code

```
#include <AccelStepper.h>
#include <MultiStepper.h>
#include <SPI.h>
#include <MFRC522.h>
#include <SdFat.h>
#include <CSVFile.h>
#include <SoftwareSerial.h>

#include <avr/pgmspace.h>
#include <hiduniversal.h>
#include <Usb.h>
#include <usbhub.h>
#include <avr/pgmspace.h>
#include <hidboot.h>


#define PIN_SPI_CLK 52
#define PIN_SPI_MOSI 51
#define PIN_SPI_MISO 50
#define PIN_SD_CS 29
#define PIN_SD_ON 27
#define RFID_SS_PIN 53
#define RFID_RST_PIN 49
#define USB_SS_PIN 10
#define SD_CARD_SPEED SPI_FULL_SPEED

#define FILENAME "CSV.csv"
int H=0;
int FD = 40;
int BD = 42;
int enM1 = 44;
int enM2 = 46;
int Op = 48;
int Cl = 47;
int Throw=30;
int limitR = 31;
```

```
int limitL = 33;
int limitUp = 35;
int limitDn = 37;
int limitFD = 39;
int limitBD = 41;
int limitOp = 43;
int limitCl = 45;
int x ,y ,d , xo,yo;
int limR , limL , limUp , limDn , limFD , limBD , limOp ,
    limCl,IR;
int man , man2;
const int stepsPerRevolution = 200;
char z;
int cont = 0 , l=0;
String State;

int one=0, two=0, three=0;
int IRsensor=23;
int BarcodeOn=25;

int heatTime = 100;
int heatInterval = 255;
char printDensity = 15;
char printBreakTime = 15;


int Balance_int;
String newBalance;
int newBalance_int;
char NB [16];
String barcode="2013";
int Patientblock=2;//this is the block number we will write
    into and then read. Do not write into 'sector trailer'
    block, since this can make the block unusable.
int Doctorblock=4;
int BarCode1block= 5;
int BarCode2block= 6;
int BarCode3block= 8;
int balanceblock=9;
int Passwordblock=10;
String PatientName;
String DoctorName;
String RfBarcode;
String RfBarcode1;
String RfBarcode2;
String RfBarcode3;
String Balance_str;
String Password;
String RdBarcode;
String readString;
```

```
String type="";
String Reqtype="";
String MSG="";

byte readbackpatient[18];//This array is used for reading out a
    block. The MIFARE_Read method requires a buffer that is at
    least 18 bytes to hold the 16 bytes of a block.
byte readbackdoctor[18];
byte readbackBarcode1[18];
byte readbackBarcode2[18];
byte readbackBarcode3[18];
byte readbackbalance[18];
byte readbackPassword[18];
byte newBarcode[16]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};

int RL , UD , Pull, Del;
int Outlet=0;
long positions[2];
int cell;
int fst,snd,thd;
struct MED {
String sdBarcode;
int cell;
int amount;
String drugName;
int price;
String info;
int numOfline;
};

MED DRUG[3];
MED OTC[10];

int D=0;
int Totalprice;
SdFat sd;
CSVFile csv;

MFRC522 mfrc522(RFID_SS_PIN, RFID_RST_PIN);
MFRC522::MIFARE_Key key;

AccelStepper stepper1(AccelStepper::FULL4WIRE, 22, 24, 26, 28);
AccelStepper stepper2(AccelStepper::FULL4WIRE, 32, 34, 36, 38);

MultiStepper steppers;

uint8_t keyBD;
USB    Usb;
USBHub    Hub(&Usb);
HIDUniversal  Hid(&Usb);
```

```
SoftwareSerial Thermal(15,14);
//SoftwareSerial uno(17,16);

int readFromRFID()
{
digitalWrite(PIN_SD_ON, LOW);
digitalWrite(PIN_SD_CS, HIGH);
digitalWrite(USB_SS_PIN, HIGH);
digitalWrite( RFID_SS_PIN, LOW);

SPI.begin();            // Init SPI bus
mfrc522.PCD_Init();     // Init MFRC522 card (in case you
    wonder what PCD means: proximity coupling device)
Serial.println("Scan a MIFARE Classic card");
for (byte i = 0; i < 6; i++) {
key.keyByte[i] = 0xFF;//keyByte is defined in the "MIFARE_Key"
    'struct' definition in the .h file of the library

}

/*****************************************establishing contact
    with a
    tag/card*******************************************************************
mfrc522.PCD_Init(); // Init MFRC522

delay(100);
// Look for new cards (in case you wonder what PICC means:
    proximity integrated circuit card)
while ( ! mfrc522.PICC_IsNewCardPresent()) {//if
    PICC_IsNewCardPresent returns 1, a new card has been found
    and we continue
//if it did not find a new card is returns a '0' and we return
    to the start of the loop
}
delay (300);
// Select one of the cards
if ( ! mfrc522.PICC_ReadCardSerial()) {//if PICC_ReadCardSerial
    returns 1, the "uid" struct (see MFRC522.h lines 238-45))
    contains the ID of the read card.
Serial.println("Error");
return;//if it returns a '0' something went wrong and we return
    to the start of the loop
}
Serial.println("card selected");

readBlock(Patientblock, readbackpatient);//read the block back

readBlock(Doctorblock, readbackdoctor);//read the block back
```

```
readBlock(BarCode1block, readbackBarcode1);//read the block back

readBlock(BarCode2block, readbackBarcode2);//read the block back

readBlock(BarCode3block, readbackBarcode3);//read the block back

readBlock(balanceblock, readbackbalance);//read the block back

readBlock(Passwordblock, readbackPassword);//read the block back


PatientName= String ((char*)readbackpatient);
DoctorName = String((char*)readbackdoctor);
RfBarcode1 =String((char*)readbackBarcode1);
RfBarcode2 = String((char*) readbackBarcode2);
RfBarcode3 = String((char*)readbackBarcode3);
Balance_str = String((char*)readbackbalance);
Password = String((char*)readbackPassword);

Balance_int=Balance_str.toInt();

Serial.print("Patient Name: ");Serial.println(PatientName);
Serial.print("Doctor Name: ");Serial.println(DoctorName);
Serial.print("Drug1 Code: ");Serial.println( RfBarcode1);
Serial.print("Drug2 Code: ");Serial.println( RfBarcode2);
Serial.print("Drug3 Code: ");Serial.println(RfBarcode3);
Serial.print("Balance: ");Serial.println(Balance_str);
Serial.print("Password ");Serial.println(Password);

digitalWrite( RFID_SS_PIN, HIGH);

if((RfBarcode1=="")&&(RfBarcode2=="")&&(RfBarcode3=="")){
Serial.println("No descriped drug!");
delay(2000);

return 1;
}
else
return 1;
}

void modifyRFID()
{

digitalWrite(PIN_SD_ON, LOW);
digitalWrite(PIN_SD_CS, HIGH);
digitalWrite(USB_SS_PIN, HIGH);
digitalWrite( RFID_SS_PIN, LOW);
```

```
SPI.begin();            // Init SPI bus
mfrc522.PCD_Init();     // Init MFRC522 card (in case you
    wonder what PCD means: proximity coupling device)
Serial.println("Scan a MIFARE Classic card");
for (byte i = 0; i < 6; i++) {
key.keyByte[i] = 0xFF;//keyByte is defined in the "MIFARE_Key"
    'struct' definition in the .h file of the library

}

/*********************************************establishing contact
    with a
    tag/card*********************************************************************
mfrc522.PCD_Init(); // Init MFRC522

delay(100);
// Look for new cards (in case you wonder what PICC means:
    proximity integrated circuit card)
while ( ! mfrc522.PICC_IsNewCardPresent()) {//if
    PICC_IsNewCardPresent returns 1, a new card has been found
    and we continue
//if it did not find a new card is returns a '0' and we return
    to the start of the loop
}
delay(300);
// Select one of the cards
if ( ! mfrc522.PICC_ReadCardSerial()) {//if PICC_ReadCardSerial
    returns 1, the "uid" struct (see MFRC522.h lines 238-45))
    contains the ID of the read card.
Serial.println("Error");
return;//if it returns a '0' something went wrong and we return
    to the start of the loop
}
Serial.println("card selected");

newBalance=String(newBalance_int);
newBalance.getBytes(NB, 16);

if(Reqtype=="Pre"){
if(fst==1)
writeBlock(BarCode1block, newBarcode);

if(snd==1)
writeBlock(BarCode2block, newBarcode);

if(thd==1)
writeBlock(BarCode3block, newBarcode);
}
writeBlock(balanceblock, NB);
```

```
digitalWrite( RFID_SS_PIN, HIGH);

}

int readSDcard(){

int av=0;

digitalWrite( RFID_SS_PIN, HIGH);
digitalWrite(USB_SS_PIN, HIGH);
digitalWrite(PIN_SD_CS,LOW);
digitalWrite(PIN_SD_ON, HIGH);

if (!sd.begin(PIN_SD_CS, SD_CARD_SPEED))
{
Serial.println("SD card begin error");
return;
}
for(D=0; D<=2 ; D++){
DRUG[D].sdBarcode="";
DRUG[D].drugName="";
DRUG[D].info="";
DRUG[D].cell=0;
DRUG[D].price=0;
DRUG[D].amount=0;
DRUG[D].numOfline=0;
}

const byte NUM_SIZE =2;
const byte BUFFER_SIZE =200;
char buffer[BUFFER_SIZE + 1];
char NUMbuffer [NUM_SIZE+1];
buffer[BUFFER_SIZE] = '\0';

Totalprice=0;

csv.open(FILENAME, O_RDWR | O_CREAT);

for(D=0 ; D<=2 ; D++){

if(Reqtype=="Pre"){
if(D==0)
RfBarcode=RfBarcode1;
else
if(D==1)
RfBarcode=RfBarcode2;
else
if(D==2)
RfBarcode=RfBarcode3;
}
```

```
else if(Reqtype=="OTC"){
if((D==0)&&(one!=0))
RfBarcode=OTC[one-1].sdBarcode;
else
if((D==1)&&(two!=0))
RfBarcode=OTC[two-1].sdBarcode;
else
if((D==2)&&(three!=0))
RfBarcode=OTC[three-1].sdBarcode;

else
RfBarcode="";
}

csv.gotoBeginOfFile();

csv.readField(buffer, BUFFER_SIZE);
DRUG[D].sdBarcode=buffer;
while((DRUG[D].sdBarcode!=RfBarcode)&&!(csv.isEndOfLine())){
csv.nextLine();
csv.readField(buffer, BUFFER_SIZE);
DRUG[D].sdBarcode=buffer;
}

if(DRUG[D].sdBarcode==RfBarcode)
{
DRUG[D].numOfline= csv.getNumberOfLine();
csv.nextField();
csv.readField(DRUG[D].cell, NUMbuffer, NUM_SIZE);

csv.nextField();
csv.readField(DRUG[D].amount, NUMbuffer ,NUM_SIZE);

csv.nextField();
csv.readField(buffer, BUFFER_SIZE);
DRUG[D].drugName=buffer;

csv.nextField();
csv.readField(DRUG[D].price, NUMbuffer ,NUM_SIZE+1);

csv.nextField();
csv.nextField();
csv.readField(buffer, BUFFER_SIZE);
DRUG[D].info=buffer;

Serial.print("sdBarcode: "); Serial.println(DRUG[D].sdBarcode);
Serial.print("cell: "); Serial.println(DRUG[D].cell);
Serial.print("amount: "); Serial.println(DRUG[D].amount);
Serial.print("drugName: "); Serial.println(DRUG[D].drugName);
```

```
Serial.print("price: "); Serial.println(DRUG[D].price);
Serial.print("info: "); Serial.println(DRUG[D].info);
Serial.print("NumOfLine "); Serial.println(DRUG[D].numOfline);

Totalprice= Totalprice + DRUG[D].price;
}

else{
Serial.print(RfBarcode);
Serial.println(" Not available!");
delay(2000);
av++;
}

}
if(Balance_int>=Totalprice)
Serial.println("Balance is OK");
else
Serial.println("Balance is not enough for all Drugs!");
delay(2000);
csv.close();
delay(500);
digitalWrite(PIN_SD_CS,HIGH);
digitalWrite(PIN_SD_ON, LOW);
if(av==3)
return 0;
else
return 1;
}
void modifySDcard()
{
digitalWrite( RFID_SS_PIN, HIGH);
digitalWrite(USB_SS_PIN, HIGH);
digitalWrite(PIN_SD_CS,LOW);
digitalWrite(PIN_SD_ON, HIGH);

if (!sd.begin(PIN_SD_CS, SD_CARD_SPEED))
{
Serial.println("SD card begin error");
return;
}
csv.open(FILENAME, O_RDWR | O_CREAT);
csv.gotoBeginOfFile();
csv.gotoLine(DRUG[D].numOfline);
csv.nextField();
csv.nextField();
csv.editField(DRUG[D].amount-1);
csv.close();

delay(1000);//time for save changes
```

```
digitalWrite(PIN_SD_CS,HIGH);
digitalWrite(PIN_SD_ON, LOW);
}
void Homing()
{

//pocket Homing

limCl=digitalRead(limitCl);
while(limCl==HIGH){
digitalWrite(Op, LOW);
digitalWrite(Cl, HIGH);
analogWrite(enM2, 140);
limCl=digitalRead(limitCl);
}
digitalWrite(enM2, LOW);

limBD=digitalRead(limitBD);
while(limBD==HIGH){
digitalWrite(FD, LOW);
digitalWrite(BD, HIGH);
analogWrite(enM1, 140);
limBD=digitalRead(limitBD);
}
digitalWrite(enM1, LOW);

delay(1000);

//X-Y Homing

H=0;
RL=0;
UD=0;

stepper1.setCurrentPosition(0);
stepper2.setCurrentPosition(0);

x=-650;
y=600;
x=  (x/8)*stepsPerRevolution;
y=  (y/8)*stepsPerRevolution;

if(abs(x)>= abs(y)){

if(y!=0)
d= abs(x/y);
else
d=0;
positions[0] = x;
positions[1] = d*y;
```

```
}

if(abs(x)< abs(y)){

if(x!=0)
d= abs(y/x);
else
d=0;
positions[0] =d*x;
positions[1] =y;
}

steppers.moveTo(positions);
while(H==0)
{
limR=digitalRead(limitR);
limL=digitalRead(limitL);
limUp=digitalRead(limitUp);
limDn=digitalRead(limitDn);
limFD=digitalRead(limitFD);
limBD=digitalRead(limitBD);
limOp=digitalRead(limitOp);
limCl=digitalRead(limitCl);

if(((limR==LOW)&&(x<0))||((limL==LOW)&&(x>0)))
RL=1;
if(((limUp==LOW)&&(y>0))||((limDn==LOW)&&(y<0)))
UD=1;

if((limUp==LOW)&&(limR==LOW)){
H=1;

}

if((limBD==HIGH)||(limCl==HIGH)){
stepper1.disableOutputs();
stepper2.disableOutputs();
}


if(RL==0)
stepper1.runSpeed();
else
stepper1.disableOutputs();

if(UD==0)
stepper2.runSpeed();
else
stepper2.disableOutputs();
```

```
}
stepper1.disableOutputs();
stepper2.disableOutputs();

stepper1.setCurrentPosition(0);
stepper2.setCurrentPosition(0);
xo=0;
yo=0;

}

void GotoCell()
{


stepper1.enableOutputs();
stepper2.enableOutputs();
stepper1.setMaxSpeed(800);
stepper2.setMaxSpeed(800);
positions[0] =0;
positions[1] =0;

RL=0;
UD=0;

if (cell==0){
x=0;
y=0;
}

else
if(cell<=5){
x=((cell-1)*155)+20+cell;
y=-180;
}

else
if((cell>5)&&(cell<=12)){
x=((cell-6)*115)-(cell-5);
y=-585;
}
else
if(cell==40){
x=485;
y=-155;
}
x=  (x/8)*stepsPerRevolution;
y=  (y/8)*stepsPerRevolution;

if(abs(abs(x)-abs(xo))>= abs(abs(y)- abs(yo))){
```

```
if(y!=0)
d= (abs(y)- abs(yo))/abs(abs(y)- abs(yo));
else
d=0;

positions[0] = x;
positions[1] = yo - d*abs(abs(x)- abs(xo));
}

if(abs(abs(x)-abs(xo))< abs(abs(y)- abs(yo))){

if(x!=0)
d= (x-xo)/abs(x-xo);
else
d=0;

positions[0] =xo+ d*abs(abs(y)- abs(yo));
positions[1] =y;
}

steppers.moveTo(positions);

while(!(RL&&UD))
{
limR=digitalRead(limitR);
limL=digitalRead(limitL);
limUp=digitalRead(limitUp);
limDn=digitalRead(limitDn);
limFD=digitalRead(limitFD);
limBD=digitalRead(limitBD);
limOp=digitalRead(limitOp);
limCl=digitalRead(limitCl);

if(((limR==LOW)&&(x<0))||((limL==LOW)&&(x>0)))
RL=1;

if(((limUp==LOW)&&(y>0))||((limDn==LOW)&&(y<0)))
UD=1;

if((limBD==HIGH)||(limCl==HIGH)){
RL=1;
UD=1;
}


if(abs(stepper1.currentPosition())==abs(int(x)))
RL=1;

if(abs(stepper2.currentPosition())== abs(int(y)))
```

```
UD=1;

if(RL==0)
stepper1.runSpeed();
else
stepper1.disableOutputs();


if(UD==0)
stepper2.runSpeed();
else
stepper2.disableOutputs();
}

stepper1.disableOutputs();
stepper2.disableOutputs();
H=0;

xo=x;
yo=y;
}



void PullDrug()
{

Pull=0;
limFD=digitalRead(limitFD);

while(limFD==HIGH){
digitalWrite(FD, HIGH);
digitalWrite(BD, LOW);
analogWrite(enM1, 140);
limFD=digitalRead(limitFD);
}
digitalWrite(enM1, LOW);
delay(2000);
limBD=digitalRead(limitBD);

while(limBD==HIGH){
digitalWrite(FD, LOW);
digitalWrite(BD, HIGH);
analogWrite(enM1, 170);

IR=digitalRead(IRsensor);
limBD=digitalRead(limitBD);

if(IR==LOW)
Pull=1;
```

```
}
digitalWrite(enM1, LOW);

}

void RotatePocket()
{
double t=0;
Del=0;
limFD=digitalRead(limitFD);

while(limFD==HIGH){
digitalWrite(FD, HIGH);
digitalWrite(BD, LOW);
analogWrite(enM1, 140);
limFD=digitalRead(limitFD);
}
digitalWrite(enM1, LOW);
delay(2000);

limOp=digitalRead(limitOp);
while(limOp==HIGH){
digitalWrite(Op, HIGH);
digitalWrite(Cl, LOW);
analogWrite(enM2, 230);
limOp=digitalRead(limitOp);
}
digitalWrite(enM2, LOW);

t=millis();
IR=digitalRead(IRsensor);
while(millis()<=(t+7000)){
IR=digitalRead(IRsensor);
if(IR==LOW)
Del=1;
}


limCl=digitalRead(limitCl);
while(limCl==HIGH){
digitalWrite(Op, LOW);
digitalWrite(Cl, HIGH);
analogWrite(enM2, 140);
limCl=digitalRead(limitCl);
}
digitalWrite(enM2, LOW);

limBD=digitalRead(limitBD);
while(limBD==HIGH){
digitalWrite(FD, LOW);
```

```
digitalWrite(BD, HIGH);
analogWrite(enM1, 140);
limBD=digitalRead(limitBD);
}
digitalWrite(enM1, LOW);

delay(1000);

t=0;

if(Del==0){
t=millis();
while(millis()<=(t+3000)){
digitalWrite(Throw, HIGH);
}
digitalWrite(Throw, LOW);
}

}
class KbdRptParser : public KeyboardReportParser
{

void PrintKey(uint8_t mod, uint8_t keyBD);
protected:
virtual void OnKeyDown (uint8_t mod, uint8_t keyBD);
virtual void OnKeyPressed(uint8_t keyBD);
virtual void OnKeyUp (uint8_t mod, uint8_t keyBD);
};

void KbdRptParser::OnKeyDown(uint8_t mod, uint8_t keyBD)
{

uint8_t c = OemToAscii(mod, keyBD);

if (c)
OnKeyPressed(c);

}

/* what to do when symbol arrives */
void KbdRptParser::OnKeyPressed(uint8_t keyBD)
{
//static uint32_t next_time = 0;  //watchdog
//static uint8_t current_cursor = 0; //tracks current cursor
   position

z=((char)keyBD); // Read characters that arrive from serial port
RdBarcode += z;       //each character builds in a string
cont = cont+1;
```

```
if (char(keyBD)== 19) { //verify the las digit of the scanner
RdBarcode.remove(cont-1); //Remove the last digit
l=RdBarcode.length();
Serial.println(l);

cont=0;                    // Reset the counter
l=0;

}
}
void KbdRptParser::OnKeyUp(uint8_t mod, uint8_t keyBD)
{
}

KbdRptParser Prs;
void readBarcode()
{
z="";
RdBarcode="";

RL=0;
UD=0;
digitalWrite(PIN_SD_ON, LOW);
digitalWrite(PIN_SD_CS, HIGH);
digitalWrite( RFID_SS_PIN, HIGH);
digitalWrite(USB_SS_PIN, LOW);

Serial.println("Start");

digitalWrite(BarcodeOn,LOW);

if(cell<5)
y=-120;
else
if(cell==5)
y=-122;
else
y=-530;

y=  (y/8)*stepsPerRevolution;

positions[0] =x;
positions[1] =y;

steppers.moveTo(positions);
while(!(RL&&UD))
{

Usb.Task();//activate Reader
```

```
limR=digitalRead(limitR);
limL=digitalRead(limitL);
limUp=digitalRead(limitUp);
limDn=digitalRead(limitDn);
limFD=digitalRead(limitFD);
limBD=digitalRead(limitBD);
limOp=digitalRead(limitOp);
limCl=digitalRead(limitCl);

if(((limR==LOW)&&(x<0))||((limL==LOW)&&(x>0)))
RL=1;

if(((limUp==LOW)&&(y>0))||((limDn==LOW)&&(y<0)))
UD=1;

if((limBD==HIGH)||(limCl==HIGH)){
RL=1;
UD=1;
}

if(abs(stepper1.currentPosition())==abs(x))
RL=1;

if(abs(stepper2.currentPosition())== abs(y))
UD=1;

if(RL==0)
stepper1.runSpeed();
else
stepper1.disableOutputs();


if(UD==0)
stepper2.runSpeed();
else
stepper2.disableOutputs();
}

stepper1.disableOutputs();
stepper2.disableOutputs();
delay(1000);
digitalWrite(BarcodeOn,HIGH);
Serial.println(RdBarcode);

digitalWrite(USB_SS_PIN, HIGH);

xo=x;
yo=y;
}
void GotoOutlet()
```

```
{
cell=40;

GotoCell();
}

void initPrinter()
{
//Modify the print speed and heat
Thermal.write(27);
Thermal.write(55);
Thermal.write(3); //Default 64 dots = 8*('7'+1)
Thermal.write(heatTime); //Default 80 or 800us
Thermal.write(heatInterval); //Default 2 or 20us
//Modify the print density and timeout
Thermal.write(18);
Thermal.write(35);
int printSetting = (printDensity<<4) | printBreakTime;
Thermal.write(printSetting); //Combination of printDensity and
    printBreakTime
Serial.println();
Serial.println("Printer ready");
delay(2000);
}




int FindOTC(){

int ot=0;

digitalWrite( RFID_SS_PIN, HIGH);
digitalWrite(USB_SS_PIN, HIGH);
digitalWrite(PIN_SD_CS,LOW);
digitalWrite(PIN_SD_ON, HIGH);

if (!sd.begin(PIN_SD_CS, SD_CARD_SPEED))
{
Serial.println("SD card begin error");
return;
}


for(D=0; D<=9 ; D++){
OTC[D].sdBarcode="";
OTC[D].drugName="";
OTC[D].info="";
OTC[D].cell=0;
OTC[D].price=0;
```

```
OTC[D].amount=0;
OTC[D].numOfline=0;
}

const byte NUM_SIZE =2;
const byte BUFFER_SIZE =200;
char buffer[BUFFER_SIZE + 1];
char NUMbuffer [NUM_SIZE+1];
buffer[BUFFER_SIZE] = '\0';

csv.open(FILENAME, O_RDWR | O_CREAT);

csv.gotoBeginOfFile();

for(D=0 ; D<=9 ; D++){
type="";
csv.readField(buffer, BUFFER_SIZE);

type=buffer;
while((type!="OTC")&&!(csv.isEndOfFile())){
csv.nextField();
csv.readField(buffer, BUFFER_SIZE);
type=buffer;
if(csv.isEndOfLine())
csv.nextLine();
}

if(type=="OTC")
{

OTC[D].numOfline= csv.getNumberOfLine();
csv.gotoBeginOfLine();

csv.readField(buffer, BUFFER_SIZE);
OTC[D].sdBarcode=buffer;

csv.nextField();
csv.readField(OTC[D].cell, NUMbuffer, NUM_SIZE);

csv.nextField();
csv.readField(OTC[D].amount, NUMbuffer ,NUM_SIZE);

csv.nextField();
csv.readField(buffer, BUFFER_SIZE);
OTC[D].drugName=buffer;

csv.nextField();
csv.readField(OTC[D].price, NUMbuffer ,NUM_SIZE+1);

csv.nextField();
```

```
csv.nextField();
csv.readField(buffer, BUFFER_SIZE);
OTC[D].info=buffer;

Serial.print(" OTC sdBarcode: ");
    Serial.println(OTC[D].sdBarcode);
Serial.print("cell: "); Serial.println(OTC[D].cell);
Serial.print("amount: "); Serial.println(OTC[D].amount);
Serial.print("drugName: "); Serial.println(OTC[D].drugName);
Serial.print("price: "); Serial.println(OTC[D].price);
Serial.print("info: "); Serial.println(OTC[D].info);
Serial.print("NumOfLine "); Serial.println(OTC[D].numOfline);
ot++;
}
}
csv.close();
delay(500);
digitalWrite(PIN_SD_CS,HIGH);
digitalWrite(PIN_SD_ON, LOW);
if(ot==0)
return 0;
else
return 1;
}

void Print(){
initPrinter();
Thermal.write(29);
Thermal.write(33);
Thermal.write(255);
Thermal.println("         ADM ");
Thermal.print("Patient Name: ");
Thermal.println(PatientName);
Thermal.write(10);
Thermal.print("Doctor Name: ");
Thermal.println(DoctorName);
Thermal.write(10);

if(fst==1){
Thermal.print("Drug Name: ");
Thermal.println(DRUG[0].drugName);
Thermal.write(10);
Thermal.print("Price: ");
Thermal.println(DRUG[0].price);
Thermal.write(10);
Thermal.print("Information: ");
Thermal.println(DRUG[0].info);
Thermal.write(10);
Thermal.write(10);
}
```

```
if(snd==1){
Thermal.print("Drug Name: ");
Thermal.println(DRUG[1].drugName);
Thermal.write(10);
Thermal.print("Price: ");
Thermal.println(DRUG[1].price);
Thermal.write(10);
Thermal.print("Information: ");
Thermal.println(DRUG[1].info);
Thermal.write(10);
Thermal.write(10);
}

if(thd==1){
Thermal.print("Drug Name: ");
Thermal.println(DRUG[2].drugName);
Thermal.write(10);
Thermal.print("Price: ");
Thermal.println(DRUG[2].price);
Thermal.write(10);
Thermal.print("Information: ");
Thermal.println(DRUG[2].info);
Thermal.write(10);
Thermal.write(10);
}

Thermal.print("Remainng Balance: ");
Thermal.println(newBalance);
Thermal.write(10);

Thermal.print("State: ");
Thermal.println(State);
Thermal.write(10);
Thermal.write(10);
delay(2000);
}

void setup() {

pinMode(2, INPUT);
pinMode(3, INPUT);
pinMode(FD, OUTPUT);
pinMode(BD, OUTPUT);
pinMode(enM1, OUTPUT);
pinMode(enM2, OUTPUT);
pinMode(Op, OUTPUT);
pinMode(Cl, OUTPUT);

pinMode(limitR,INPUT);
```

```
pinMode(limitL,INPUT);
pinMode(limitUp,INPUT);
pinMode(limitDn,INPUT);
pinMode(limitFD,INPUT);
pinMode(limitBD,INPUT);
pinMode(limitOp,INPUT);
pinMode(limitCl,INPUT);
pinMode(Throw,OUTPUT);
pinMode(PIN_SPI_MOSI, OUTPUT);
pinMode(PIN_SPI_MISO, INPUT);
pinMode(PIN_SPI_CLK, OUTPUT);
pinMode(PIN_SD_CS, OUTPUT);
pinMode(PIN_SD_ON, OUTPUT);
pinMode(RFID_SS_PIN, OUTPUT);
pinMode(USB_SS_PIN, OUTPUT);
pinMode(BarcodeOn,OUTPUT);

stepper1.setMaxSpeed(800);
stepper2.setMaxSpeed(800);

steppers.addStepper(stepper1);
steppers.addStepper(stepper2);

digitalWrite(BarcodeOn,HIGH);
digitalWrite(PIN_SD_CS, HIGH);
digitalWrite(PIN_SD_ON, LOW);
digitalWrite(RFID_SS_PIN, HIGH);
digitalWrite(USB_SS_PIN, HIGH);


if (Usb.Init() == -1) {
Serial.println("OSC did not start.");
}

delay( 200 );

Hid.SetReportParser(0, (HIDReportParser*)&Prs);


Serial2.begin(9600);
Thermal.begin(19200); // to write to our new printer
initPrinter();
// Setup serial
Serial.begin(9600);
while (!Serial2) { /* wait for Leonardo */ }
}
void loop() {
MSG="";
one=0;
two=0;
```

```
three=0;
if(H==0){
Homing();
}

delay(500);

while(Serial2.available()==0){
}
while(Serial2.available()>0){
MSG+= char(Serial2.read());
delay(50);
}

if(MSG=="St"){

if(!readFromRFID()){
return;
}
Serial.println("Enter Your Password ");
delay(500);
Serial2.write("Nm");
delay(1000);
Serial2.println(PatientName);
delay(2000);

MSG="";

while(Serial2.available()==0){

}

while(Serial2.available()>0){
MSG+= char(Serial2.read());
delay(50);
}

Serial.println("MSG");
Serial.println(MSG);
if(MSG!=Password){

Serial.println("Incorrect Password");
Serial2.println("IN");
return;
}

Serial2.println("CO");
MSG="";
delay(500);
Serial.println("Select Process Type");
```

```
while(Serial2.available()==0){

}

Reqtype="";
while((Serial2.available()>0)){
Reqtype+= char(Serial2.read());
delay(250);
}
Serial.print(Reqtype);
if(Reqtype=="OTC"){
FindOTC();
delay(1000);

for(int i=0 ; i<=9 ; i++)
{
if((OTC[i].drugName!="")){
Serial2.print("  ");
Serial2.print(OTC[i].drugName);
Serial2.print("    ");
Serial2.println(OTC[i].price);
Serial2.print("\n");
}
}
Serial.println("Select the OTC Drugs");

while(Serial2.available()==0){
}

if(Serial2.available()>0){
one= Serial2.parseInt();
delay(250);
}
if(Serial2.available()>0){
two= Serial2.parseInt();
delay(250);
}
if(Serial2.available()>0){
three= Serial2.parseInt();
delay(250);
}
Serial.println(one);
Serial.println(two);
Serial.println(three);
delay(2000);

}
Serial.println("\n");

if(!readSDcard()){
```

```
return;
}
newBalance_int=Balance_int;
fst=0;
snd=0;
thd=0;

for(D=0 ; D<=2 ; D++){
cell=DRUG[D].cell;
if(DRUG[D].amount==0){
Serial.print(DRUG[D].drugName);
Serial.print(" Not available !");
}

if(newBalance_int < DRUG[D].price){
Serial.print(" No enough balance !");
}


if((cell!=0) && (DRUG[D].amount!=0) && (newBalance_int >=
    DRUG[D].price)){

Serial2.println("Running..");
Serial2.println("Wait a minite");
GotoCell();
delay(2000);

Serial2.println("Reading Barcode..");
readBarcode();

if(RdBarcode==DRUG[D].sdBarcode){
Serial.println("correct");
Serial2.println("correct");
delay(2000);

PullDrug();
if(Pull==0){
Serial.println("Pull Failed!");
Serial2.println("Pull Failed!");
// return;
}
else
if(Pull==1){
Serial.println("Pull Done");
delay(2000);

newBalance_int=newBalance_int - DRUG[D].price;
modifySDcard();

if(D==0)
```

```
fst=1;
if(D==1)
snd=1;
if(D==2)
thd=1;

}

}
else
{
Serial.println("incorrect");
Serial2.println("incorrect");
delay(2000);
}
}
}

if(fst || snd || thd){
GotoOutlet();
delay(2000);
Serial2.println("Insert Your ");
Serial2.println("    Card again");
modifyRFID();

RotatePocket();
Serial2.println("Take your drug");

if(Del==1)
State="Drug is delivered";

else
State="Drug not delivered";

delay(2000);

Serial.println(State);


Print();
Serial2.println("Thank you");
delay(2000);
}
}
}
```

```
#include <SPI.h>//include the SPI bus library
#include <MFRC522.h>//include the RFID reader library

#define PIN_SD_ON 3
#define SS_PIN 10 //slave select pin
#define RST_PIN 9 //reset pin
#define USB_SS_PIN 4
MFRC522 mfrc522(SS_PIN, RST_PIN);    // instatiate a MFRC522
    reader object.
MFRC522::MIFARE_Key key;//create a MIFARE_Key struct named
    'key', which will hold the card information
void setup() {

pinMode(USB_SS_PIN, OUTPUT);
pinMode(PIN_SD_ON, OUTPUT);
digitalWrite(USB_SS_PIN, HIGH);

Serial.begin(9600);      // Initialize serial communications
    with the PC
SPI.begin();             // Init SPI bus
mfrc522.PCD_Init();      // Init MFRC522 card (in case you
    wonder what PCD means: proximity coupling device)
Serial.println("Scan a MIFARE Classic card");
digitalWrite(PIN_SD_ON, LOW);
// Prepare the security key for the read and write functions -
    all six key bytes are set to 0xFF at chip delivery from the
    factory.
// Since the cards in the kit are new and the keys were never
    defined, they are 0xFF
// if we had a card that was programmed by someone else, we
    would need to know the key to be able to access it. This key
    would then need to be stored in 'key' instead.

for (byte i = 0; i < 6; i++) {
key.keyByte[i] = 0xFF;//keyByte is defined in the "MIFARE_Key"
    'struct' definition in the .h file of the library
}

}

int Patientblock=2;//this is the block number we will write
    into and then read. Do not write into 'sector trailer'
    block, since this can make the block unusable.
int Doctorblock=4;
int BarCode1block= 5;
int BarCode2block= 6;
int BarCode3block= 8;
```

```
int balanceblock=9;
int Passwordblock=10;
byte patient[16] = {"Mumin Taamra"};//an array with 16 bytes to
    be written into one of the 64 card blocks is defined
//byte blockcontent[16] =
    {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};//all zeros. This can be
    used to delete a block.
byte doctor[16]={"Husam Taamra"};
byte Barcode1[16]={"1539"};
byte Barcode2[16]={"7321839721885"};
byte Barcode3[16]={"6251581011112"};

byte balance[16]={"100"};
byte Password[16]={"12345"};
void loop()
{

/*****************************************establishing contact
    with a
    tag/card*********************************************************************
mfrc522.PCD_Init(); // Init MFRC522
delay(100);

// Look for new cards (in case you wonder what PICC means:
    proximity integrated circuit card)
if ( ! mfrc522.PICC_IsNewCardPresent()) {//if
    PICC_IsNewCardPresent returns 1, a new card has been found
    and we continue

return;//if it did not find a new card is returns a '0' and we
    return to the start of the loop
}

// Select one of the cards
if ( ! mfrc522.PICC_ReadCardSerial()) {//if PICC_ReadCardSerial
    returns 1, the "uid" struct (see MFRC522.h lines 238-45))
    contains the ID of the read card.
Serial.println("Error");
return;//if it returns a '0' something went wrong and we return
    to the start of the loop
}
Serial.println("card selected");

writeBlock(Patientblock, patient);//the blockcontent array is
    written into the card block
writeBlock(Doctorblock, doctor);
writeBlock(BarCode1block, Barcode1);
writeBlock(BarCode2block, Barcode2);
writeBlock(BarCode3block, Barcode3);
writeBlock(balanceblock, balance);
```

```
writeBlock(Passwordblock, Password);

while (1);

}
```

```
int writeBlock(int blockNumber, byte arrayAddress[])
{
//this makes sure that we only write into data blocks. Every
    4th block is a trailer block for the access/security info.
int largestModulo4Number=blockNumber/4*4;
int trailerBlock=largestModulo4Number+3;//determine trailer
    block for the sector
if (blockNumber > 2 && (blockNumber+1)%4 ==
    0){Serial.print(blockNumber);Serial.println(" is a trailer
    block:");return 2;}//block number is a trailer block (modulo
    4); quit and send error code 2
Serial.print(blockNumber);
Serial.println(" is a data block:");

/****************authentication of the desired block for
    access**********************/
byte status =
    mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A,
    trailerBlock, &key, &(mfrc522.uid));
//byte PCD_Authenticate(byte command, byte blockAddr,
    MIFARE_Key *key, Uid *uid);
//this method is used to authenticate a certain block for
    writing or reading
//command: See enumerations above -> PICC_CMD_MF_AUTH_KEY_A =
    0x60 (=1100000), // this command performs authentication
    with Key A
//blockAddr is the number of the block from 0 to 15.
//MIFARE_Key *key is a pointer to the MIFARE_Key struct defined
    above, this struct needs to be defined for each block. New
    cards have all A/B= FF FF FF FF FF FF
//Uid *uid is a pointer to the UID struct that contains the
    user ID of the card.
if (status != MFRC522::STATUS_OK) {
Serial.print("PCD_Authenticate() failed: ");
Serial.println(mfrc522.GetStatusCodeName(status));
return 3;//return "3" as error message
}
//it appears the authentication needs to be made before every
    block read/write within a specific sector.
//If a different sector is being authenticated access to the
    previous one is lost.


/*********************************writing the
    block*******************************/
```

```
status = mfrc522.MIFARE_Write(blockNumber, arrayAddress,
    16);//valueBlockA is the block number, MIFARE_Write(block
    number (0-15), byte array containing 16 values, number of
    bytes in block (=16))
//status = mfrc522.MIFARE_Write(9, value1Block, 16);
if (status != MFRC522::STATUS_OK) {
Serial.print("MIFARE_Write() failed: ");
Serial.println(mfrc522.GetStatusCodeName(status));
return 4;//return "4" as error message
}
Serial.println("block was written");
}


int readBlock(int blockNumber, byte arrayAddress[])
{
int largestModulo4Number=blockNumber/4*4;
int trailerBlock=largestModulo4Number+3;//determine trailer
    block for the sector

/*****************************************authentication of the
    desired block for
    access************************************************************/
byte status =
    mfrc522.PCD_Authenticate(MFRC522::PICC_CMD_MF_AUTH_KEY_A,
    trailerBlock, &key, &(mfrc522.uid));
//byte PCD_Authenticate(byte command, byte blockAddr,
    MIFARE_Key *key, Uid *uid);
//this method is used to authenticate a certain block for
    writing or reading
//command: See enumerations above -> PICC_CMD_MF_AUTH_KEY_A =
    0x60 (=1100000), // this command performs authentication
    with Key A
//blockAddr is the number of the block from 0 to 15.
//MIFARE_Key *key is a pointer to the MIFARE_Key struct defined
    above, this struct needs to be defined for each block. New
    cards have all A/B= FF FF FF FF FF FF
//Uid *uid is a pointer to the UID struct that contains the
    user ID of the card.
if (status != MFRC522::STATUS_OK) {
Serial.print("PCD_Authenticate() failed (read): ");
Serial.println(mfrc522.GetStatusCodeName(status));
return 3;//return "3" as error message
}
//it appears the authentication needs to be made before every
    block read/write within a specific sector.
//If a different sector is being authenticated access to the
    previous one is lost.
```

```
/*************************reading a
    block*****************************************/

byte buffersize = 18;//we need to define a variable with the
    read buffer size, since the MIFARE_Read method below needs a
    pointer to the variable that contains the size...
status = mfrc522.MIFARE_Read(blockNumber, arrayAddress,
    &buffersize);//&buffersize is a pointer to the buffersize
    variable; MIFARE_Read requires a pointer instead of just a
    number
if (status != MFRC522::STATUS_OK) {
Serial.print("MIFARE_read() failed: ");
Serial.println(mfrc522.GetStatusCodeName(status));
return 4;//return "4" as error message
}
Serial.println("block was read");
}
```

4. Touch Screen (Uno code).

```
#include <Adafruit_TFTLCD.h>
#include <Adafruit_GFX.h>
#include <TouchScreen.h>
#include <SoftwareSerial.h>
#define LCD_CS A3
#define LCD_CD A2
#define LCD_WR A1
#define LCD_RD A0
#define LCD_RESET A4

#define TS_MINX 130
#define TS_MINY 130
#define TS_MAXX 950
#define TS_MAXY 900

#define YP A3
#define XM A2
#define YM 9
#define XP 8

#define BLACK  0x0000
#define BLUE   0x001F
#define RED    0xF800
#define GREEN  0x07E0
#define CYAN   0x07FF
#define MAGENTA 0xF81F
#define YELLOW 0xFFE0
#define WHITE  0xFFFF
#define FRN    0x669999
#define BNF    0xe6b3cc
Adafruit_TFTLCD tft(LCD_CS, LCD_CD, LCD_WR, LCD_RD, LCD_RESET);

TouchScreen ts = TouchScreen(XP, YP, XM, YM, 300);

boolean buttonEnabled = true;

String c="";
String password="";
int ok=1;
String msg="";
String Nm="";
String Type="";
String Drugs[10];
int Fst=0 , Snd=0 ,Thd=0;
int Lp=0;

SoftwareSerial mega(0,1);
void setup(void) {
```

```
mega.begin(9600);
Serial.begin(9600);
tft.reset();

uint16_t identifier = tft.readID();
identifier=0x9341;
tft.begin(identifier);
tft.setRotation(1);

}

void HomeScreen(){
tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);

tft.setCursor(120,20);
tft.setTextColor(BLACK);
tft.setTextSize(4);
tft.print("ADM");

tft.setCursor(120,80);
tft.setTextColor(BLACK);
tft.setTextSize(2);
tft.print("MADE BY \n\n         ENG.");



tft.setCursor(5,150);
tft.setTextColor(RED);
tft.setTextSize(2);
tft.print("MU'MIN TA'AMRA & Obayda");

tft.fillRect(50,180, 210, 40, GREEN);
tft.drawRect(50,180,210,40,BLACK);
tft.setCursor(125,190);
tft.setTextColor(BLACK);
tft.setTextSize(2);
tft.print("START");


buttonEnabled = true;
pinMode(XM, INPUT);
pinMode(YP, INPUT);

while(buttonEnabled){
TSPoint p = ts.getPoint();
p.x = map(p.x, TS_MAXX, TS_MINX, 0, 240);
p.y = map(p.y, TS_MINY, TS_MAXY, 320, 0);

if (p.z > 300) {
```

```
if(p.y>50 && p.y<260 && p.x>180 && p.x<270){

buttonEnabled = false;
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);
}
}
}
}
void Keypad(){

tft.fillScreen(WHITE);
tft.drawRect(0,0,100,70,BLACK);
tft.setCursor(50,30);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("1");

tft.drawRect(100,0,100,70,BLACK);
tft.setCursor(150,30);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("2");

tft.drawRect(200,0,100,70,BLACK);
tft.setCursor(250,30);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("3");

tft.drawRect(0,70,100,70,BLACK);
tft.setCursor(50,100);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("4");

tft.drawRect(100,70,100,70,BLACK);
tft.setCursor(150,100);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("5");

tft.drawRect(200,70,100,70,BLACK);
tft.setCursor(250,100);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("6");
```

```
tft.drawRect(0,140,100,70,BLACK);
tft.setCursor(50,170);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("7");

tft.drawRect(100,140,100,70,BLACK);
tft.setCursor(150,170);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("8");

tft.drawRect(200,140,100,70,BLACK);
tft.setCursor(250,170);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("9");

tft.drawRect(0,210,300,30,BLACK);
tft.setCursor(130,215);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.print("OK");
}

void Enterpass(){
c="";
password="";
ok=0;

while(ok!=1)
{

TSPoint p = ts.getPoint();
p.x = map(p.x, TS_MAXX, TS_MINX, 0, 240);
p.y = map(p.y, TS_MINY, TS_MAXY, 320, 0);

if (p.z > 300){

if(p.x>0 && p.x<70 && p.y>0 && p.y<100)
c="1";

if(p.x>0 && p.x<70 && p.y>100 && p.y<200)
c="2";


if(p.x>0 && p.x<70 && p.y>200 && p.y<300)
c="3";

if(p.x>70 && p.x<140 && p.y>0 && p.y<100)
```

```
c="4";

if(p.x>70 && p.x<140 && p.y>100 && p.y<200)
c="5";

if(p.x>70 && p.x<140 && p.y>200 && p.y<300)
c="6";

if(p.x>140 && p.x<210 && p.y>0 && p.y<100)
c="7";

if(p.x>140 && p.x<210 && p.y>100 && p.y<200)
c="8";

if(p.x>140 && p.x<210 && p.y>200 && p.y<300)
c="9";
if(p.x>210 && p.x<250 && p.y>10 && p.y<300){
ok=1;
c="";
}
delay(250);
password=password+c;

}

}
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
// tft.fillScreen(WHITE);
//  tft.setCursor(120,120);
//   tft.setTextColor(BLACK);
//   tft.setTextSize(3);
//   tft.print("P= " );
//    tft.println(password);
//     tft.println(password.length());
//Serial.print("Password = ");
//   Serial.println(password);
//Serial.println(password);

Serial.print(password);
delay(500);
}
void welcome(){

tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);

tft.setCursor(70,30);
tft.setTextColor(RED);
tft.setTextSize(4);
```

```
tft.print("Welcome");

tft.setCursor(70,90);
tft.setTextColor(FRN);
tft.setTextSize(3);
tft.print(Nm);

tft.setCursor(60,150);
tft.setTextColor(BNF);
tft.setTextSize(3);
tft.println("Enter Your\n");
tft.print("   Password");
delay(2500);
}

void InsertCard(){

tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);

tft.setCursor(25,80);
tft.setTextColor(RED);
tft.setTextSize(4);
tft.println("Insert Your\n");

tft.println("  Card!");

}

void Wait(){

tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);

tft.setCursor(50,100);
tft.setTextColor(BLUE);
tft.setTextSize(4);
tft.println("Wait...");
}
void SelectType(){

Type="";

tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);

tft.setCursor(25,30);
tft.setTextColor(RED);
tft.setTextSize(3);
tft.println("Select Process\n");
```

```
tft.print("     Type");

tft.fillRect(50,120, 210, 40, RED);
tft.drawRect(50,120,210,40,BLACK);
tft.setCursor(75,130);
tft.setTextColor(WHITE);
tft.setTextSize(2);
tft.print("Prescription");

tft.fillRect(50,180, 210, 40, RED);
tft.drawRect(50,180,210,40,BLACK);
tft.setCursor(65,190);
tft.setTextColor(WHITE);
tft.setTextSize(2);
tft.print("Over The Count");

while(Type==""){
TSPoint p = ts.getPoint();
p.x = map(p.x, TS_MAXX, TS_MINX, 0, 240);
p.y = map(p.y, TS_MINY, TS_MAXY, 320, 0);

if (p.z > 300) {

if(p.y>50 && p.y<260 && p.x>120 && p.x<160)
Type= "Pre";

if(p.y>50 && p.y<260 && p.x>180 && p.x<220)
Type= "OTC";
}
}
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);

Serial.print(Type);
delay(3000);
}
void SelectDrugs(){

int OK=0;
int Del=0;
int Chs=0;
Fst=0;
Snd=0;
Thd=0;
Lp=0;

tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);
```

```
tft.setCursor(30,10);
tft.setTextColor(BLACK);
tft.setTextSize(3);
tft.println("Select Drugs");

tft.drawRect(10,50,280,30,BLACK);
tft.drawRect(10,80,280,30,BLACK);
tft.drawRect(10,110,280,30,BLACK);
tft.drawRect(10,140,280,30,BLACK);
tft.drawRect(10,170,280,30,BLACK);
tft.drawCircle(20,65,7,BLACK);
tft.drawCircle(20,95,7,BLACK);
tft.drawCircle(20,125,7,BLACK);
tft.drawCircle(20,155,7,BLACK);
tft.drawCircle(20,185,7,BLACK);

tft.drawRect(50,210,50,30,BLACK);
tft.drawRect(240,210,50,30,BLACK);

tft.fillRect(50,210,50,30,RED);
tft.fillRect(240,210,50,30,GREEN);

tft.setCursor(60,220);
tft.setTextColor(BLACK);
tft.setTextSize(2.8);
tft.println("Del");

tft.setCursor(250,220);
tft.println("OK");

tft.setCursor(0,60);
tft.setTextColor(RED);
tft.setTextSize(2.8);
tft.println(Drugs[0]);

while(OK!=1){

TSPoint p = ts.getPoint();
p.x = map(p.x, TS_MAXX, TS_MINX, 0, 240);
p.y = map(p.y, TS_MINY, TS_MAXY, 320, 0);

if (p.z > 300){

if(Lp<3){

if(p.x>50 && p.x<80 && p.y>10 && p.y<290){
Chs=1;
Lp++;
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
```

```
delay(200);
tft.fillCircle(20,65,7,GREEN);
}

if(p.x>80 && p.x<110 && p.y>10 && p.y<290){
Chs=2;
Lp++;
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);
tft.fillCircle(20,95,7,GREEN);
}

if(p.x>110 && p.x<140 && p.y>10 && p.y<290){
Chs=3;
Lp++;
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);
tft.fillCircle(20,125,7,GREEN);
}

if(p.x>140 && p.x<170 && p.y>10 && p.y<290){
Chs=4;
Lp++;
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);
tft.fillCircle(20,155,7,GREEN);
}

if(p.x>170 && p.x<200 && p.y>10 && p.y<290){
Chs=5;
Lp++;
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);
tft.fillCircle(20,185,7,GREEN);
}
}

if(p.x>210 && p.x<240 && p.y>240 && p.y<290){
OK=1;
}

if(p.x>210 && p.x<240 && p.y>50 && p.y<100){
Lp=0;
Fst=0;
Snd=0;
Thd=0;
```

```
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);
tft.fillCircle(20,65,7,WHITE);
tft.fillCircle(20,95,7,WHITE);
tft.fillCircle(20,125,7,WHITE);
tft.fillCircle(20,155,7,WHITE);
tft.fillCircle(20,185,7,WHITE);
}
delay(250);

if(Lp==1)
Fst=Chs;

if(Lp==2)
Snd=Chs;

if(Lp==3)
Thd=Chs;

}
}
pinMode(XM, OUTPUT);
pinMode(YP, OUTPUT);
delay(200);
delay(500);
}



void Blnk(){


tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);

tft.setCursor(50,70);
tft.setTextColor(BLUE);
tft.setTextSize(3);
tft.print("Fst : ");
tft.println(Fst);

tft.print("Snd : ");
tft.println(Snd);
tft.print("Thd : ");
tft.println(Thd);
}
void Rot(){
tft.fillScreen(WHITE);
```

```
tft.drawRect(0,0,319,240,YELLOW);

tft.fillCircle(160,70,13,RED);
delay(150);

tft.fillCircle(195,85,11,BLUE);
delay(150);

tft.fillCircle(210,120,9,GREEN);
delay(150);
tft.fillCircle(195,155,8,FRN);
delay(150);

tft.fillCircle(160,170,7,RED);
delay(150);

tft.fillCircle(125,155,6,BLUE);
delay(150);

tft.fillCircle(110,120,5,GREEN);
delay(150);

tft.fillCircle(125,85,4,FRN);
delay(150);
}

void BLN(){
tft.fillScreen(WHITE);
tft.drawRect(0,0,319,240,YELLOW);

tft.setCursor(50,70);
tft.setTextColor(RED);
tft.setTextSize(3);

}


void loop() {
msg="";

HomeScreen();
Serial.print("St");
delay(1000);
InsertCard();

Nm="";
msg="";

while(mega.available()==0){
```

```
}

Wait();

while(mega.available()>0){
msg+= char(mega.read());
delay(250);
}
delay(500);

if(msg=="Nm")
msg="";

while(mega.available()==0){
}

while(mega.available()>0){
msg+= char(mega.read());
delay(50);
}
delay(500);
Nm=msg;
msg="";


welcome();

Keypad();
Enterpass();

while(mega.available()==0){
}



while(mega.available()>0){
msg+=char(mega.read());
delay(50);
}
if(msg=="IN"){

BLN();
tft.println("Incorrect Password");
tft.println("Try again later");
delay(500);
}

if(msg=="CO"){

BLN();
```

```
tft.println("Correct");
delay(500);
}

msg="";



SelectType();

if(Type=="OTC"){
for(int i=0 ; i<=9 ; i++){
Drugs[i]="";
}

char k="";
int i=0;
while(mega.available()==0){
}

Wait();

while(mega.available()>0){
Drugs[0]+=char(mega.read());

delay(50);
}

delay(100);

SelectDrugs();
Serial.print(Fst);
Serial.print("m");
Serial.print(Snd);
Serial.print("m");
Serial.print(Thd);
}

Wait();
delay(1000);

msg="";

Wait();
while(msg!="Thank you"){
while(mega.available()==0){
}

while(mega.available()>0){
msg+=char(mega.read());
```

```
delay(50);
}
BLN();
tft.println(msg);
delay(1000);
msg="";
}

}
```