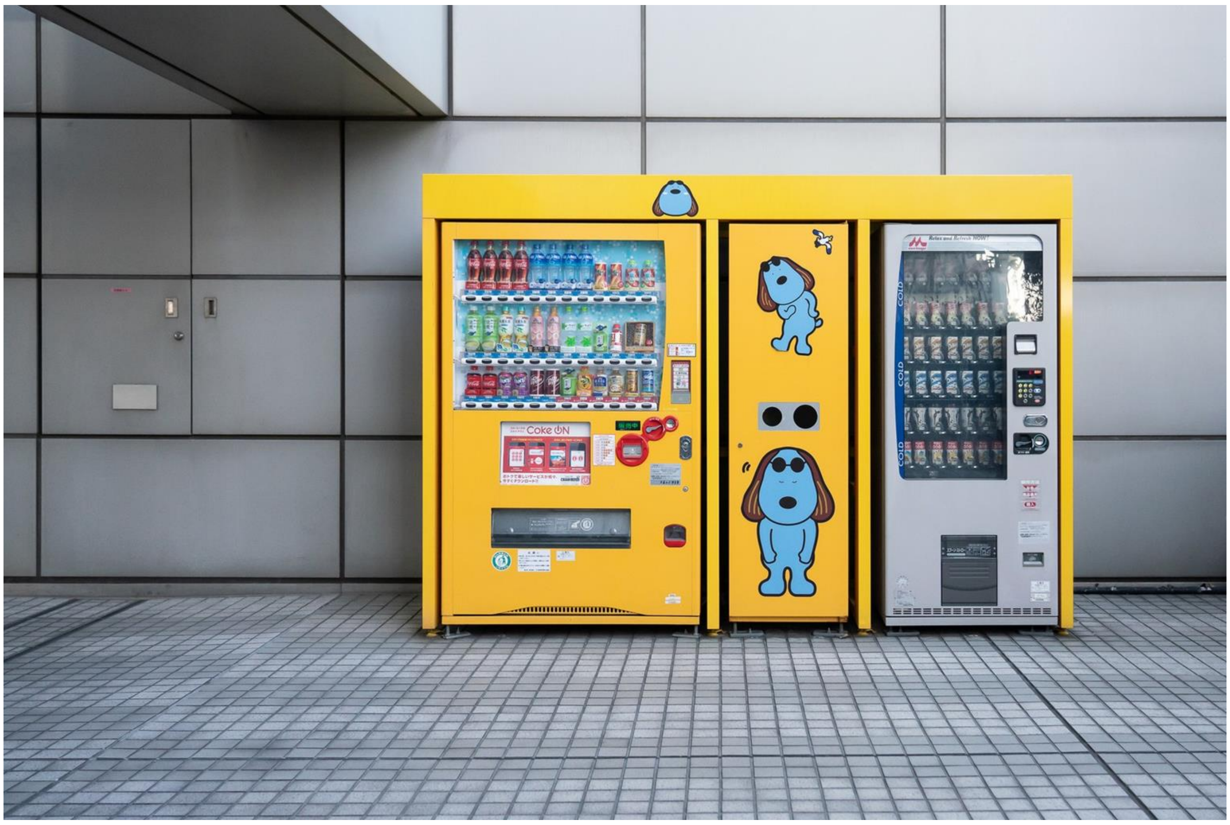
Report on Project:

ECE Vending Machine Model



Bangladesh University of Engineering and Technology

Course No.: EEE 212

Name of the Course: Numerical Technique Laboratory

Project Title:

ECE Vending Machine Model

Submitted To:

Mr. Hamidur Rahman

Associate Professor, Department of Electrical and Electronic Engineering,

Bangladesh University of Engineering and Technology

Shahed Ahmed

Lecturer, Department of Electrical and Electronic Engineering,

Bangladesh University of Engineering and Technology

Shaimur Salehin Akash

Lecturer (PT), Department of Electrical and Electronic Engineering,

Bangladesh University of Engineering and Technology

Submitted By:

1906067: Kazi Abid Hasan

1906079: Md. Faiyaz Abid

Table of Contents

List of Illustrations

1. Introduction to Our Project

The "ECE Vending Machine Model" is our first MATLAB project. Our project was inspired by the vending machine situated at the ground floor of ECE Building, BUET. Our program will do the following tasks:

- It will display some sample products and their prices
- Customer will select the product and enter money (note image)
- Taking the input, the program will identify the value of the note and calculate if more money needs to be paid or not
- If yes, the program will again take input
- If no, the program will deliver the product (display its image)
- If any money needs to be returned, the program will return it (display note images)

2. Theory

Image matching is the essential part of this project. Here, we gave input image of currency note. Our program matches the input image with the images in database. For that, we at first resized the images to same dimension so that the processing gets easier. Then we converted the images to grayscale as it is easier and efficient to work with grayscale images in comparison with RGB images. An RGB image has three color channels or layers. On the other hand, a grayscale image only has a single-color channel or layer, making the processing and analysis easier and efficient.

After that, a 2D median filtering was performed to all the images in two dimensions. Each output pixel contains the median value in a 3-by-3 neighborhood around the corresponding pixel in the input image. Then sequentially, the 2D correlation coefficient was calculated between the input image and database images. We set the threshold correlation coefficient to 90% or above. So, if the correlation between the input image and a database image is 90% or above, the program will identify the corresponding note value of the input image comparing with the database images.

Then, the program checks whether the input image (note) was able to pay the price of the product. If so, the product will be displayed and if any money needs to be returned, the amount will be displayed by notes which will be mathematically determined by another function written. Otherwise, the program will again take input until the product price is paid.

3. Algorithm

Step 1: Start

Step 2: A GUI window will display the products and **'buy'** option to purchase the product

Step 3: Customer will choose a product and a new window will be opened for that product

Step 4: The window will display the price of the product as well as the **'proceed to pay'** option, the paid total and money needed to pay more

Step 5: When the customer clicks 'proceed to pay' button, a new pop-up window will appear to pay money (image) from the **'Input'** folder

Step 6: Customer will select a note (pay money) which will be taken input by GUI

Step 7: The GUI will pass the image to **'note_identification'** function

Step 8: In the **'note_identification'** function, the database note images and input image will be read and stored to arrays.

Step 9: All the images will be resized to same dimensions

Step 10: The images will be converted to grayscale

Step 10: The resized, grayscaled database images will be stored in a 3D array

Step 11: 2D median filtering will be performed on the elements of the 3D array in a loop

Step 12: 2D median filtering will be performed on the resized, grayscaled input image

Step 13: 2D correlation coefficient will be calculated between the processed input image and the processed database images

Step 14: If the correlation coefficient, R >= 0.9 (90%), go to step 15

Step 15: Based on the index (k), the note value (5, 10, 20, 50) of the input image will be stored to variable **'note'**

Step 16: The **'note_identification'** function will return the value of the note to variable **'note'**

Step 17: In GUI of the product, inp_money (input money) = note (value of the note)

Step 18: paid_money = inp_money

Step 19: If paid_money >= prod_price (product price), return amount will be calculated (ret_money = paid_money - prod_price)

Step 20: The program will display the image of the product (purchase completed) and run the **'out_display'** function which will display the return amount by images of notes

Step 21: Else, total_paid = 0

Step 22: total_paid = total_paid + inp_money

Step 23: The program will again take input calling **'note_identification'** function until total_paid >= prod_price and repay amount will be showed (repay = prod_price – total_paid)

Step 24: If total_paid >= prod_price, return amount will be calculated (ret_money = paid_money - prod_price)

Step 25: The program will display the image of the product (purchase completed) and run the **'out_display'** function which will display the return amount by images of notes

Step 26: End

4. Code:

4.1. Function: note_identification.m

```
function note = note_identification(Img_file)


    % database dataset
    I5 = imread("db 5 taka.jpg");
    I10 = imread("db 10 taka.jpg");
    I20 = imread("db 20 taka.jpg");
    I50 = imread("db 50 taka.jpg");

    % input image
    Iin = imread(Img_file);
```

This is the beginning of the function. The function takes an image file as input parameter (**Img_file**) and returns the value of the note image as the variable **note**. Then the program reads the database images of 5, 10, 20, 50 taka note images in the variables **I5, I10, I20, I50** respectively. Also, the input image file is read in the variable **Iin**.

```
    % dataset image resizing
    I5_rsz = imresize(I5, [63 146]);
    I10_rsz = imresize(I10, [63 146]);
    I20_rsz = imresize(I20, [63 146]);
    I50_rsz = imresize(I50, [63 146]);

    % input image resizing
    Iin_rsz = imresize(Iin, [63 146]);
```

This part of the function resizes all the database images and input image to a same dimension [63 146]. This dimension was taken as it was the minimum of all the images used in the project.

```
% rgb to grayscale conversion of dataset images
I5_gray = im2gray(I5_rsz);
I10_gray = im2gray(I10_rsz);
I20_gray = im2gray(I20_rsz);
I50_gray = im2gray(I50_rsz);

% rgb to gray conversion of input image
Iin_gray = im2gray(Iin_rsz);
```

In this part of the function, we converted the rgb images to grayscale as grayscale images are easier and efficient to work with.

```
% grayscaled dataset
Idb_gray = zeros(4,63,146);
Idb_gray(1,:,:) = I5_gray;
Idb_gray(2,:,:) = I10_gray;
Idb_gray(3,:,:) = I20_gray;
Idb_gray(4,:,:) = I50_gray;
```

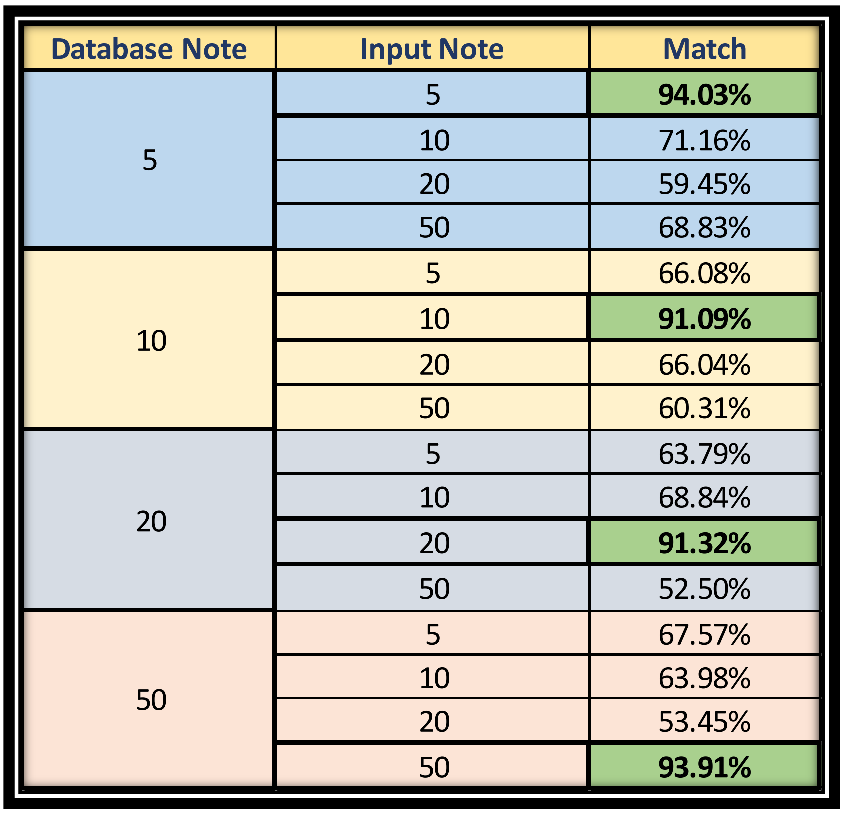
Now the grayscaled, resized database images are stored in a three-dimensional array **Idb_gray**.

```
% 2D median filtering of grayscaled dataset
J = zeros(4,63,146);
for i = 1:4
    A = shiftdim(Idb_gray(i,:,:));
    B = medfilt2(A);
    J(i,:,:) = B;
end

% 2D median filtering of grayscaled input
Jin = medfilt2(Iin_gray);
```

This part of the function performs 2D median filtering on the processed images using the **medfilt2** function. The filtered processed database images are stored in a 3D array (**J**) and the filtered processed input image is stored in **Jin**. After using medfilt2, each output pixel contains the median value in a 3-by-3 neighborhood around the corresponding pixel in the input image.

Finally, this part of the function identifies the value of the input note image. Using a loop, the 2D correlation coefficient (**R**) between the filtered input image (**Jin**) and the filtered database images (**J**) stored in **C2** respectively is calculated using the built-in function **corr2**. Here, the threshold is 90%. So, if the correlation coefficient is 90% or more, that particular note value is determined and set in the variable **note**. Finally, the function returns the value of the note.

```matlab
    % identification of note
    for k = 1:4
        C3 = J(k,:,:);
        C2 = shiftdim(C3);
        % 2D correlation coefficient calculation
        R = corr2(C2,Jin);
        if R >= 0.9  % Threshold = 90%
            if k == 1
                note = 5;
                break
            elseif k == 2
                note = 10;
                break
            elseif k == 3
                note = 20;
                break
            else
                note = 50;
                break
            end
        end
    end

end
```



Database Note	Input Note	Match
5	5	**94.03%**
	10	71.16%
	20	59.45%
	50	68.83%
10	5	66.08%
	10	**91.09%**
	20	66.04%
	50	60.31%
20	5	63.79%
	10	68.84%
	20	**91.32%**
	50	52.50%
50	5	67.57%
	10	63.98%
	20	53.45%
	50	**93.91%**

Figure 1: Results of the function

We have used different images for database notes and input notes. However, our program was able to uphold the 90% threshold as we can see in the figure above.

4.2. Function: out_display.m

```
function out_display(ret_money, value)

    taka50 = 0;
    taka20 = 0;
    taka10 = 0;
    taka5 = 0;
    tp=num2str(value);
```

This function displays the returned money by images of the database notes. It takes two parameters as input – **ret_money** (the amount to be returned) and **value** (the total paid amount). Initially, number of all the notes are set to zero and the total paid amount is stored to variable **tp** as a string.

```
    if ret_money/50 >= 1
        while ret_money/50 >= 1
            taka50 = taka50 + 1;
            ret_money = ret_money - 50;
        end
    end
    if ret_money/20 >= 1
        while ret_money/20 >= 1
            taka20 = taka20 + 1;
            ret_money = ret_money - 20;
        end
    end
    if ret_money/10 >= 1
        while ret_money/10 >= 1
            taka10 = taka10 + 1;
            ret_money = ret_money - 10;
        end
    end
    if ret_money/5 >= 1
        while ret_money/5 >= 1
            taka5 = taka5 + 1;
            ret_money = ret_money - 5;
        end
    end
```

This part of the function mathematically determines the notes and the number of notes to be displayed to return the amount to be returned after payment. Here, the program will try to return the money with the highest possible note value. For example, if the amount to be return is 25, the program will display a 20 taka note and a 5 taka note.

```
    i = 1;
    if taka50 ~= 0
        for j = i:taka50
            Note_disp(1,j) = "db 50 taka.jpg";
            i = i + 1;
        end
    end
    if taka20 ~= 0
        for j = i:(taka50+taka20)
            Note_disp(1,j) = "db 20 taka.jpg";
            i = i + 1;
        end
    end
    if taka10 ~= 0
        for j = i:(taka50+taka20+taka10)
            Note_disp(1,j) = "db 10 taka.jpg";
            i = i + 1;
        end
    end
    if taka5 ~= 0
        for j = i:(taka50+taka20+taka10+taka5)
            Note_disp(1,j) = "db 5 taka.jpg";
            i = i + 1;
        end
    end
```

Then, in the array **Note_disp**, the file name of the note images are stored corresponding to the number of notes for a particular note value.

```
    figure
    montage(Note_disp)
    title(sprintf('Total Paid: %s . Returned Money:', tp))
end
```

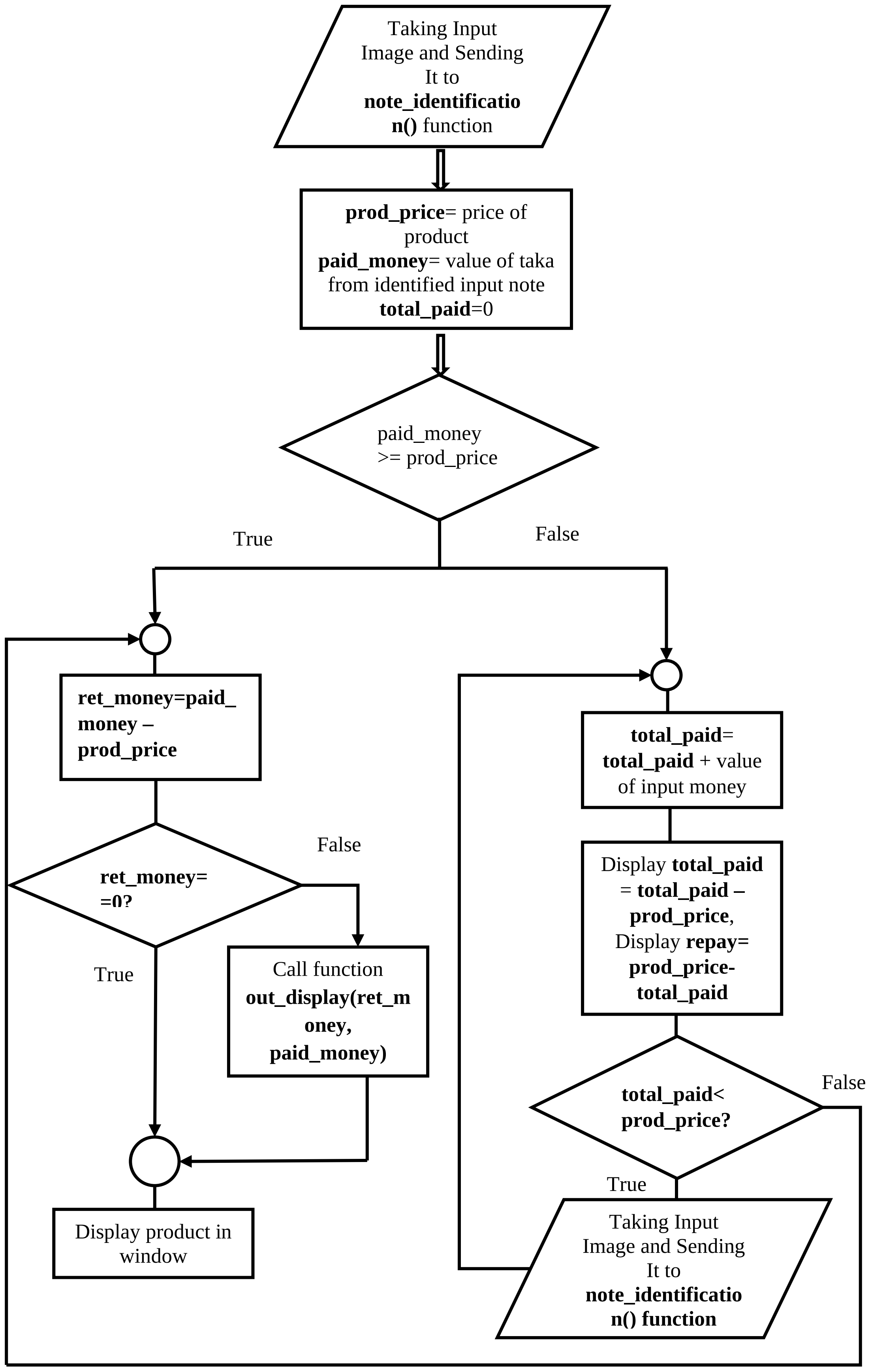
Finally, the function displays all the notes to be returned in montage view in a figure and total paid amount is also displayed in the title of the figure.

4.3. Code written in GUI:

Code written for the GUI window's "Proceed to pay (Insert Money)" button:

```
function ProceedtopayInsertMoneyButtonPushed(app, event)
            Img_file = uigetfile('*.jpg*','Select input note');
            inp_money = note_identification(Img_file);
            prod_price = 5;
            paid_money = inp_money;
            if paid_money >= prod_price
                ret_money = paid_money - prod_price;
                if ret_money == 0
                    figure
                    imshow("candy.png");
                    title("Thank you for purchasing. Here is your 
Product:", "FontName","Cambria Math")
                    delete(app)
                else
                    out_display(ret_money, paid_money);
                    figure
                    imshow("candy.png");
                    title("Thank you for purchasing. Here is your 
Product:", "FontName","Cambria Math")
                    delete(app)
                end
            else
                total_paid=0;
                app.NeedtopaymoreEditField.Value=50;
                while(total_paid<prod_price)
                    total_paid=total_paid+inp_money;
                    repay = prod_price - total_paid;
                    app.NeedtopaymoreEditField.Value=repay;
                    app.TotalPaidEditField.Value=total_paid;
                    if total_paid>= prod_price
                        ret_money = total_paid - prod_price;
                         if ret_money == 0
                            figure
                            Ip = imread("candy.png");
                            imshow(Ip)
                            title("Thank you for purchasing. Here is your 
Product:", "FontName","Cambria Math")
                            delete(app)
                        else
                            out_display(ret_money, total_paid);
                            figure
                            Ip = imread("candy.png");
                            imshow(Ip)
                            title("Thank you for purchasing. Here is your 
Product:", "FontName","Cambria Math")
                            delete(app)
                         end
                         break
                    end
                    Img_file = uigetfile('*.jpg*','Select input note');
                    inp_money = note_identification(Img_file);
                end
            end
        end
    end
end
```

4.4. Flow chart of this code:


Taking Input Image and Sending It to **note_identification()** function
prod_price= price of product
paid_money= value of taka from identified input note
total_paid=0
paid_money >= prod_price
True
False
ret_money=paid_money – prod_price
ret_money==0?
False
True
Call function **out_display(ret_money, paid_money)**
Display product in window
total_paid= total_paid + value of input money
Display **total_paid = total_paid – prod_price**, Display **repay= prod_price- total_paid**
total_paid< prod_price?
False
True
Taking Input Image and Sending It to **note_identification() function**

5. Test Cases:

5.1. Case 1: If input money is equal to price of the product:

1. Running “vendingmachine.mlapp”

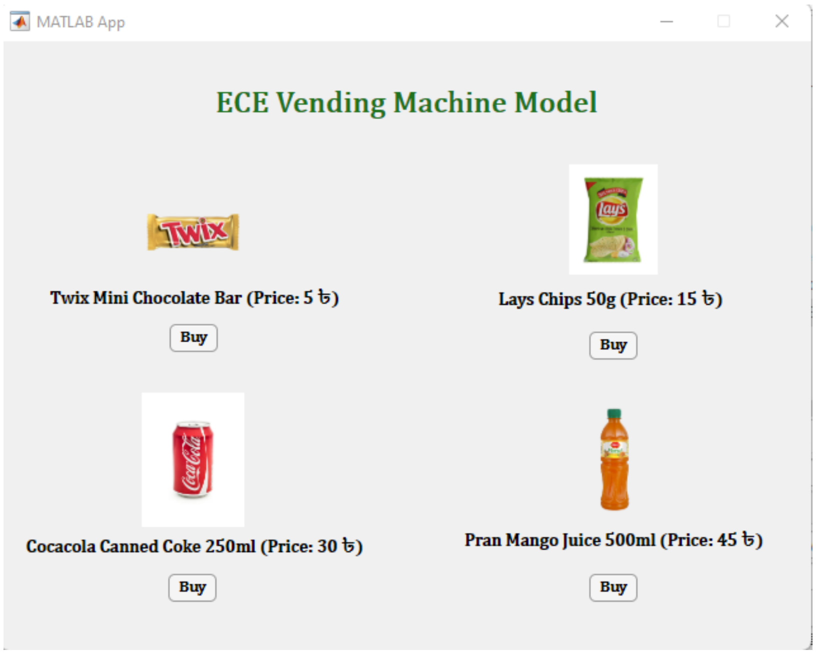
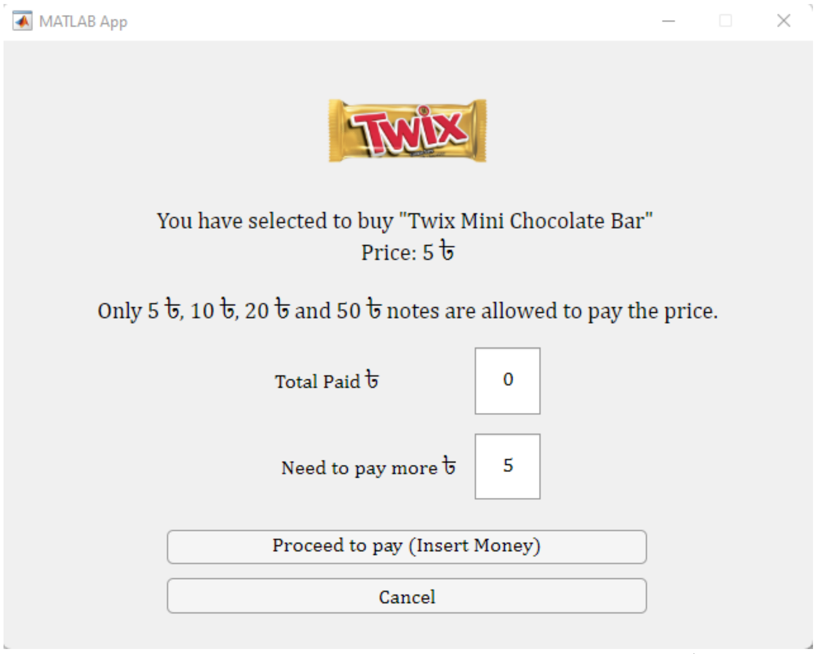

MATLAB App
ECE Vending Machine Model
Twix Mini Chocolate Bar (Price: 5 ৳)
Buy
Lays Chips 50g (Price: 15 ৳)
Buy
Cocacola Canned Coke 250ml (Price: 30 ৳)
Buy
Pran Mango Juice 500ml (Price: 45 ৳)
Buy

Figure 2: GUI of "ECE Vending Machine Model"

2. Selecting to buy “Twix Mini Chocolate Bar (Price: 5 ৳)”



You have selected to buy "Twix Mini Chocolate Bar"
Price: 5 ৳

Only 5 ৳, 10 ৳, 20 ৳ and 50 ৳ notes are allowed to pay the price.

Total Paid ৳ 0

Need to pay more ৳ 5

Figure 2: GUI window of “Twix Mini Chocolate Bar (Price: 5 ৳)”

3. Pressing “Proceed to pay (Insert Money)” and inserting 5 taka as input

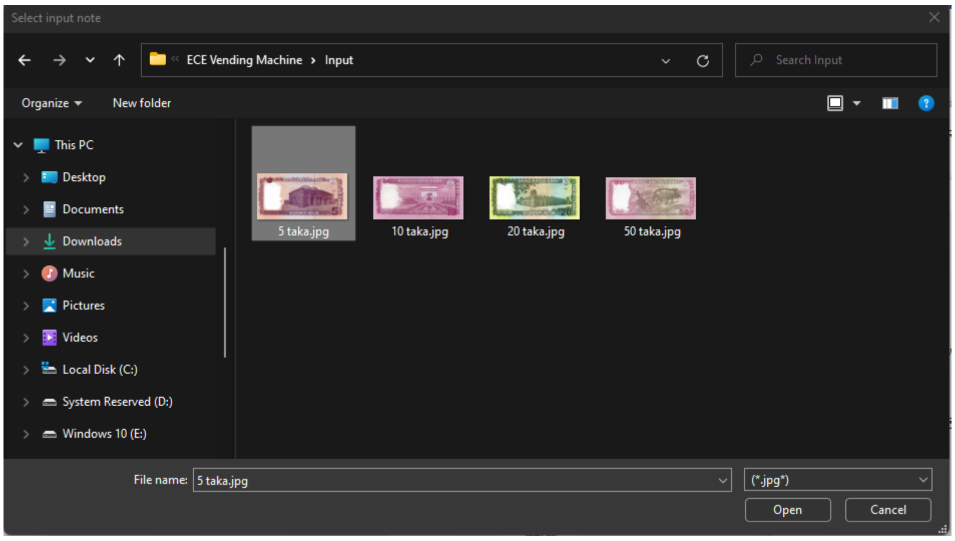


Figure 4: "Select Input Note" window is shown

4. Product picture is shown in a figure window as output

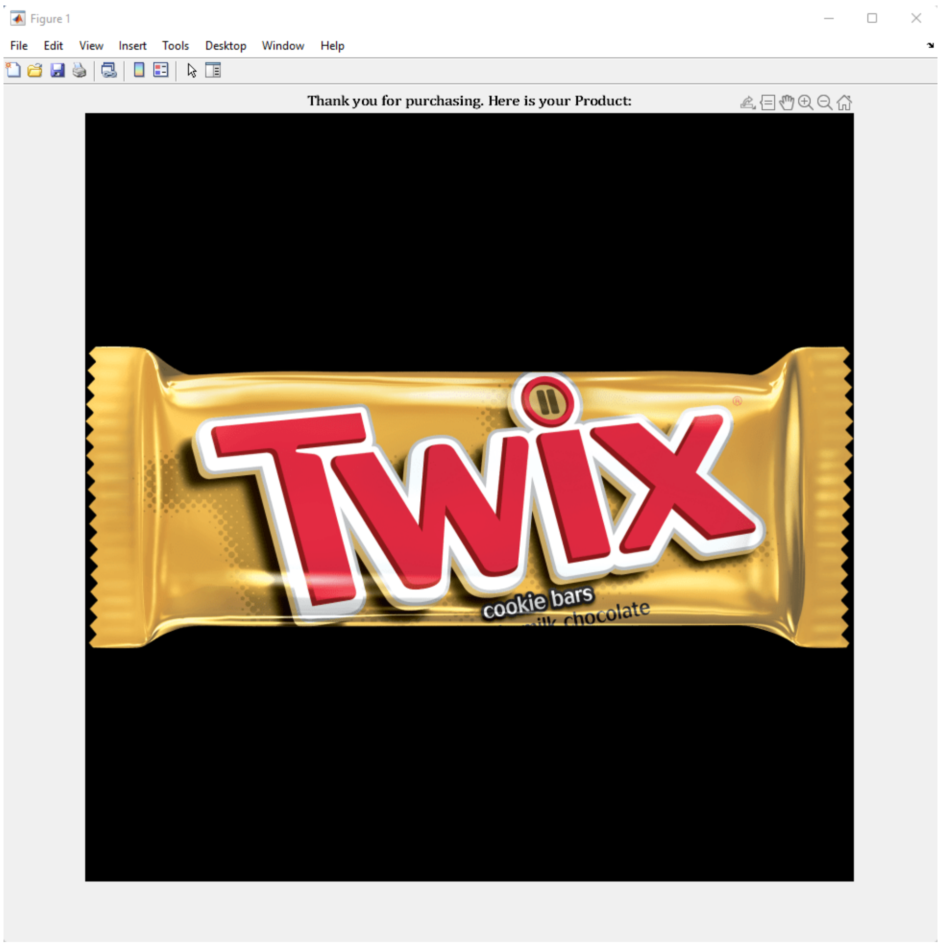
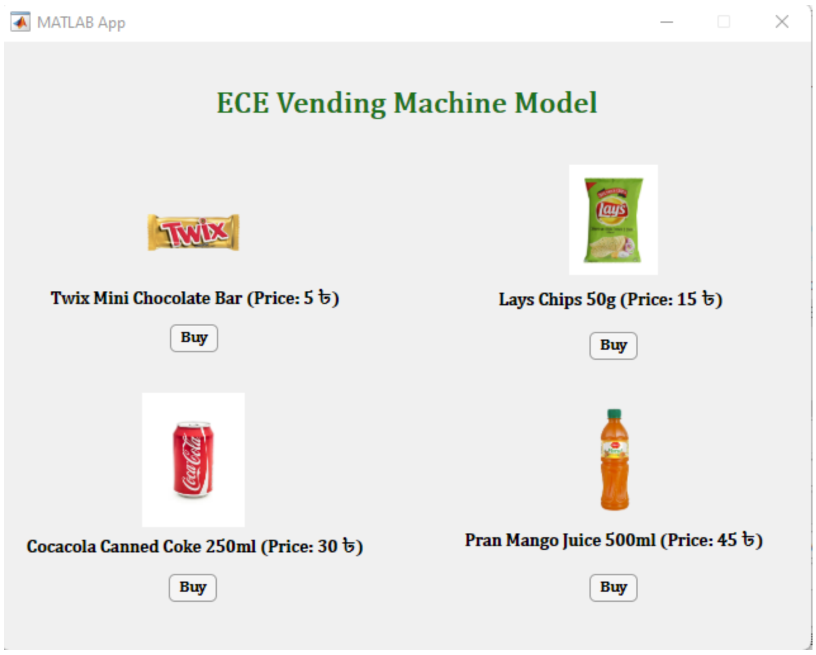


Figure 1
File Edit View Insert Tools Desktop Window Help
Thank you for purchasing. Here is your Product:
Twix
cookie bars
milk chocolate

Figure 5: Picture of "Twix Mini Chocolate Bar"

5.2. Case 2: If input money is less than price of the product:

1. Running “vendingmachine.mlapp”



ECE Vending Machine Model

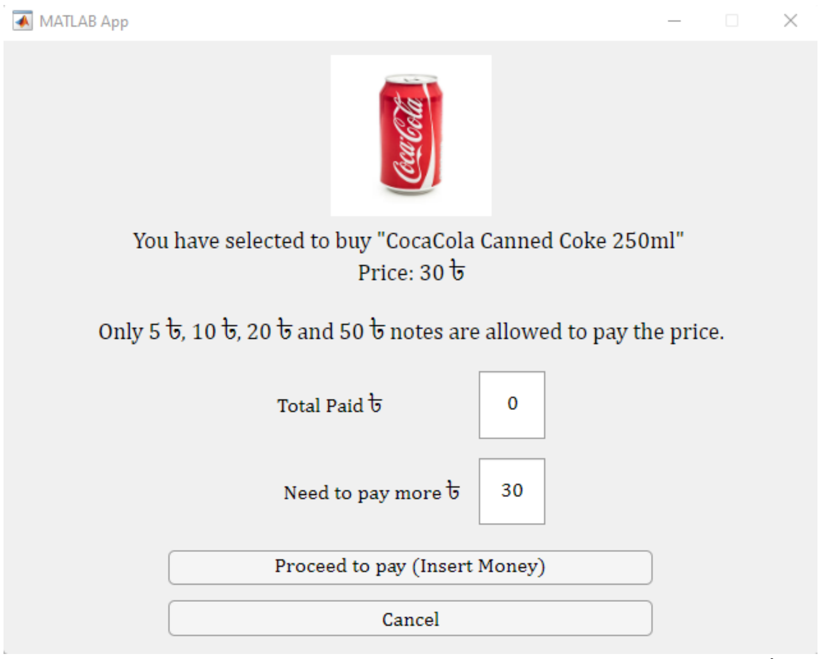
Twix Mini Chocolate Bar (Price: 5 ৳)

Lays Chips 50g (Price: 15 ৳)

Cocacola Canned Coke 250ml (Price: 30 ৳)

Pran Mango Juice 500ml (Price: 45 ৳)

2. Selecting to buy “CocaCola Canned Coke 250ml (Price: 30 ৳)”



You have selected to buy "CocaCola Canned Coke 250ml"
Price: 30 ৳

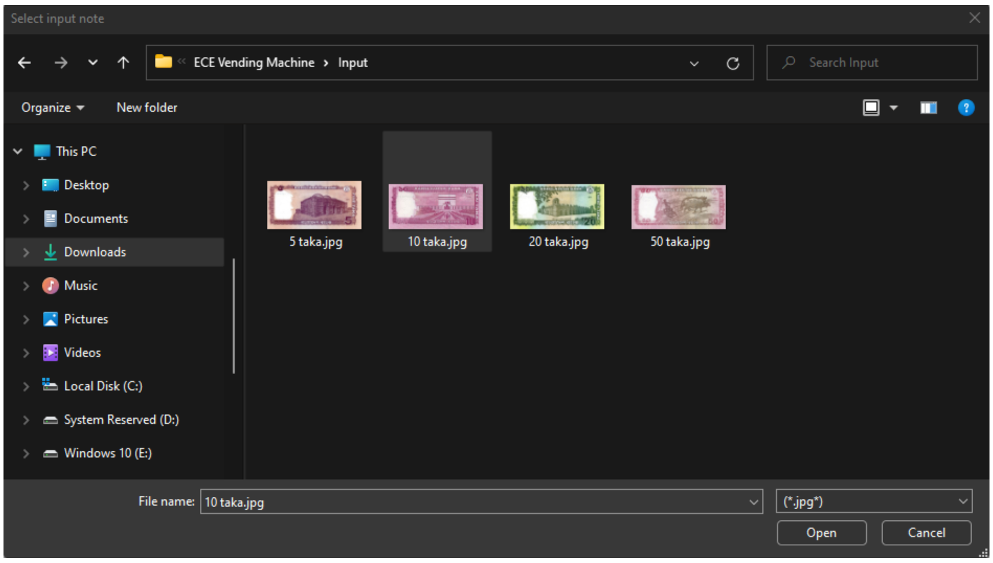
Only 5 ৳, 10 ৳, 20 ৳ and 50 ৳ notes are allowed to pay the price.

Total Paid ৳ 0

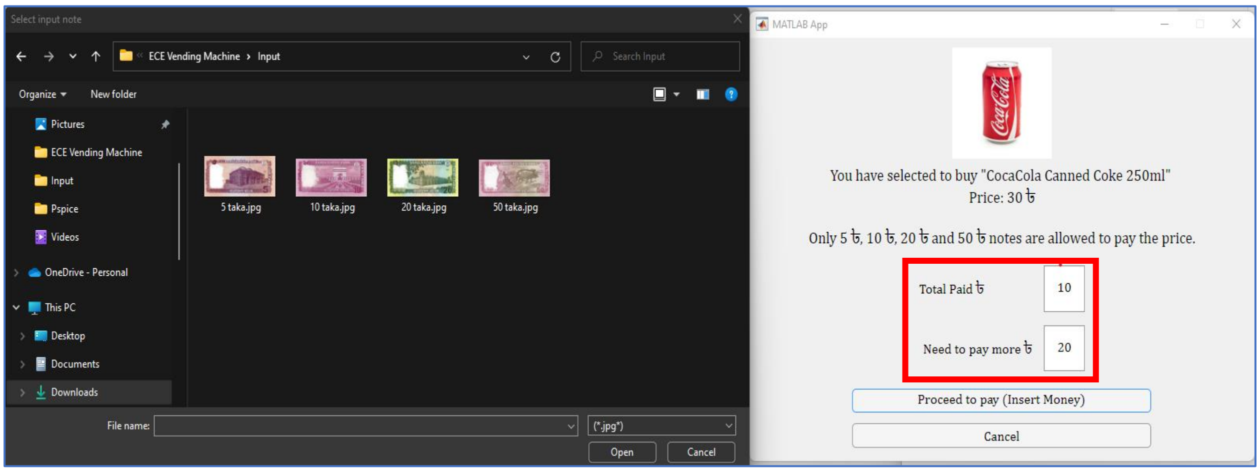
Need to pay more ৳ 30

Figure 6: GUI window of “CocaCola Canned Coke 250ml (Price: 30 ৳)”

3. Pressing “Proceed to pay (Insert Money)” and inserting 10 taka as input



4. Additional taka required to buy the product will be shown in “Need to pay more ৳” Numeric Field of the same window. Also, total paid taka is also shown in “Total paid ৳” Numeric Field of the same window. Moreover, “Select Input Note” file manager window will also pop up.



You have selected to buy "CocaCola Canned Coke 250ml"
Price: 30 ৳

Only 5 ৳, 10 ৳, 20 ৳ and 50 ৳ notes are allowed to pay the price.

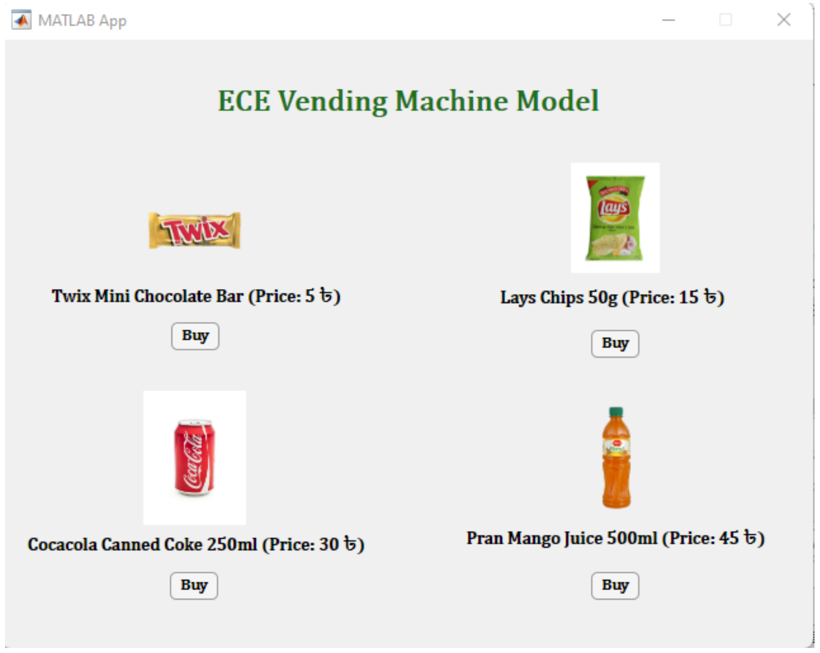
Total Paid ৳ 10

Need to pay more ৳ 20

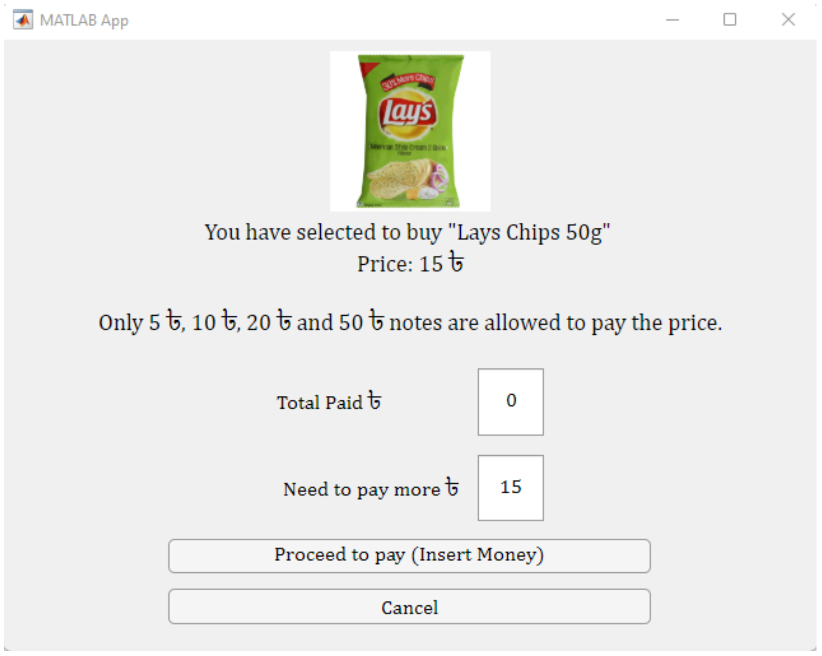
Figure 7: Change in numeric edit field in GUI window

5.3. Case 3: If input money is greater than price of the product:

1. Running “vendingmachine.mlapp”


MATLAB App
ECE Vending Machine Model
Twix Mini Chocolate Bar (Price: 5 ৳)
Buy
Lays Chips 50g (Price: 15 ৳)
Buy
Cocacola Canned Coke 250ml (Price: 30 ৳)
Buy
Pran Mango Juice 500ml (Price: 45 ৳)
Buy

2. Selecting to buy “Lays Chips 50g (Price: 15 ৳)”



You have selected to buy "Lays Chips 50g"
Price: 15 ৳

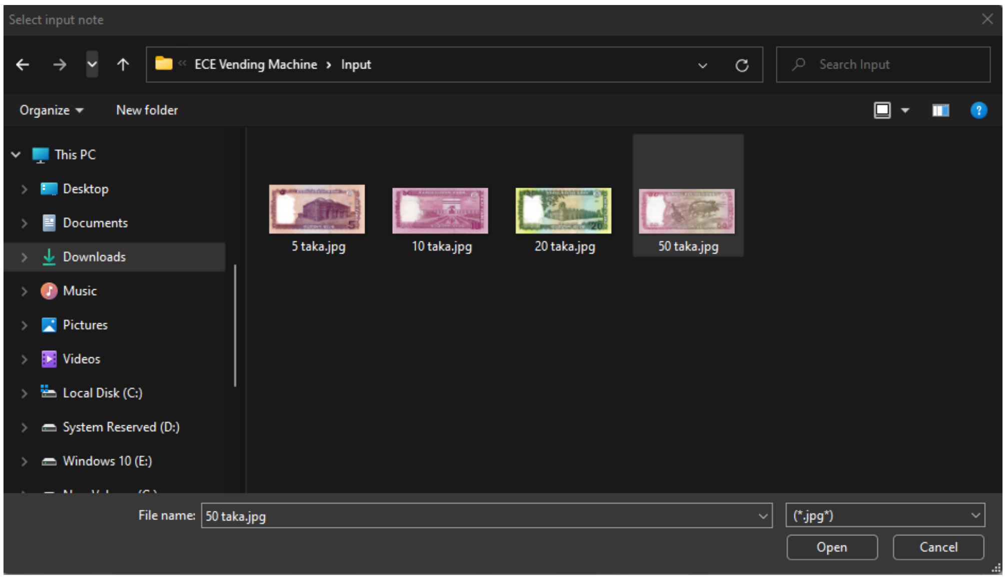
Only 5 ৳, 10 ৳, 20 ৳ and 50 ৳ notes are allowed to pay the price.

Total Paid ৳ 0

Need to pay more ৳ 15

Figure 8: GUI window of “Lays Chips 50g (Price: 15 ৳)”

3. Pressing “Proceed to pay (Insert Money)” and inserting 50 taka as input



4. Total paid taka is shown and returned taka , product picture is shown in two different figure windows as output

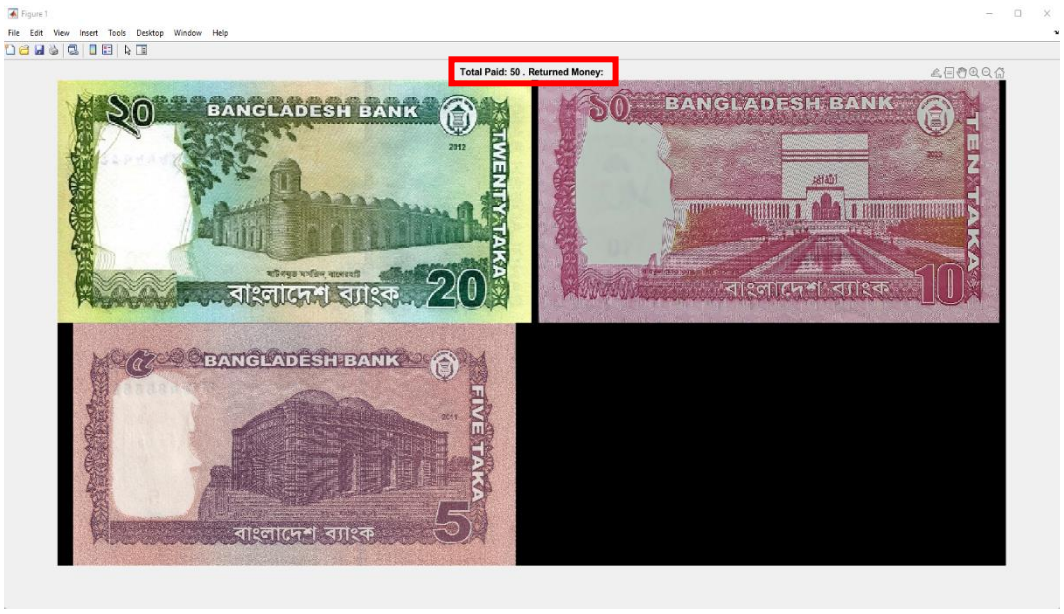


Figure 1
File Edit View Insert Tools Desktop Window Help
Total Paid: 50 . Returned Money:

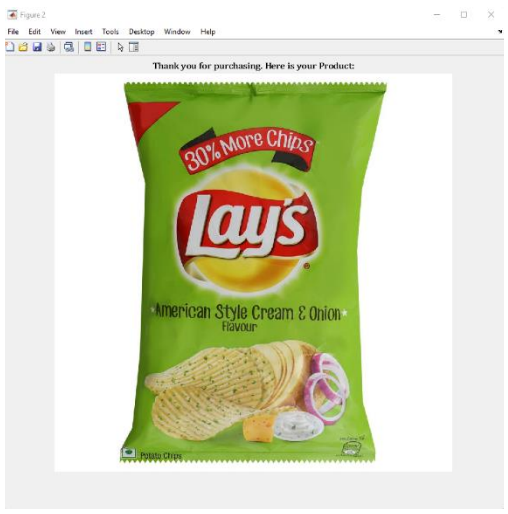


Figure 2
File Edit View Insert Tools Desktop Window Help
Thank you for purchasing. Here is your Product:

Figure 9: Returned money and output product is shown in new different windows

6. Applications of our project:

Our project is mainly based on image processing with calculations. We can use our project in the following cases:

- ❖ As the project of our name is "ECE Vending Machine Model", this project can be used to form the algorithm of a vending machine. Bangladesh is importing vending machines from foreign countries. These vending machines cost a lot. However, if we can create the body parts of a vending machine, our project can be used in the software and algorithm part.
- ❖ Vending machine which are available in our country can only scan the front side of a note. Whereas our vending machine program can scan the back side of a note. Through our project, we have found that scanning the back part of a note is more efficient and reliable to identify the program. So, this can be adopted by vending machine manufacturers so that machines can scan both sides of any notes.

7. Conclusion:

After hours and hours of dedication, our project has been completed successfully. We thank our respectful teachers as they granted our project proposal. We had to give a lot of effort to this project as this is a complete unique task and not many helpful resources could not be found on the internet directly related to our project. Though our program gives accurate output every time, our project is not completely flawless. These are the shortcomings we could find while we worked for our project:

1. Our program can only detect 4 types of currency notes, which are 5 taka, 10 taka, 20 taka and 50 taka. It cannot detect bigger notes like 100 taka, 200 taka and 500 taka etc. We could add 100 taka, 200 taka and 500 taka notes in the database and upgrade the code to improve our program.
2. If any picture other than the "input notes" are given as input, error messages would be shown in console window. However, it could not display a separate error message window. We could build a separate GUI for this error message.
3. Our program could only detect notes by the back side picture of the notes, not front side picture of notes. If we upgrade our image processing algorithm, the program can scan both sides of notes.

We had about five to six weeks to work on our project. However, to develop our program, we needed to learn image processing, calculation algorithm etc., which were completely new things to us. So, we delivered our best to this project. We hope to overcome these flaws of our project in future development. Finally, we humbly apologize to our course teachers for any kind of mistakes.