Intelli -Pantry

Smaran Pandey



Stand Number - 2206

Sign here:

Contents

[Summary: 4](#_Toc123567252)

[Introduction: 5](#_Toc123567253)

[Experimental Methods: 6](#_Toc123567254)

[Results: 20](#_Toc123567255)

[Conclusions and recommendations: 21](#_Toc123567256)

[Acknowledgments: 24](#_Toc123567257)

[Appendices: 25](#_Toc123567258)

[References: 26](#_Toc123567259)

# 

# Summary:

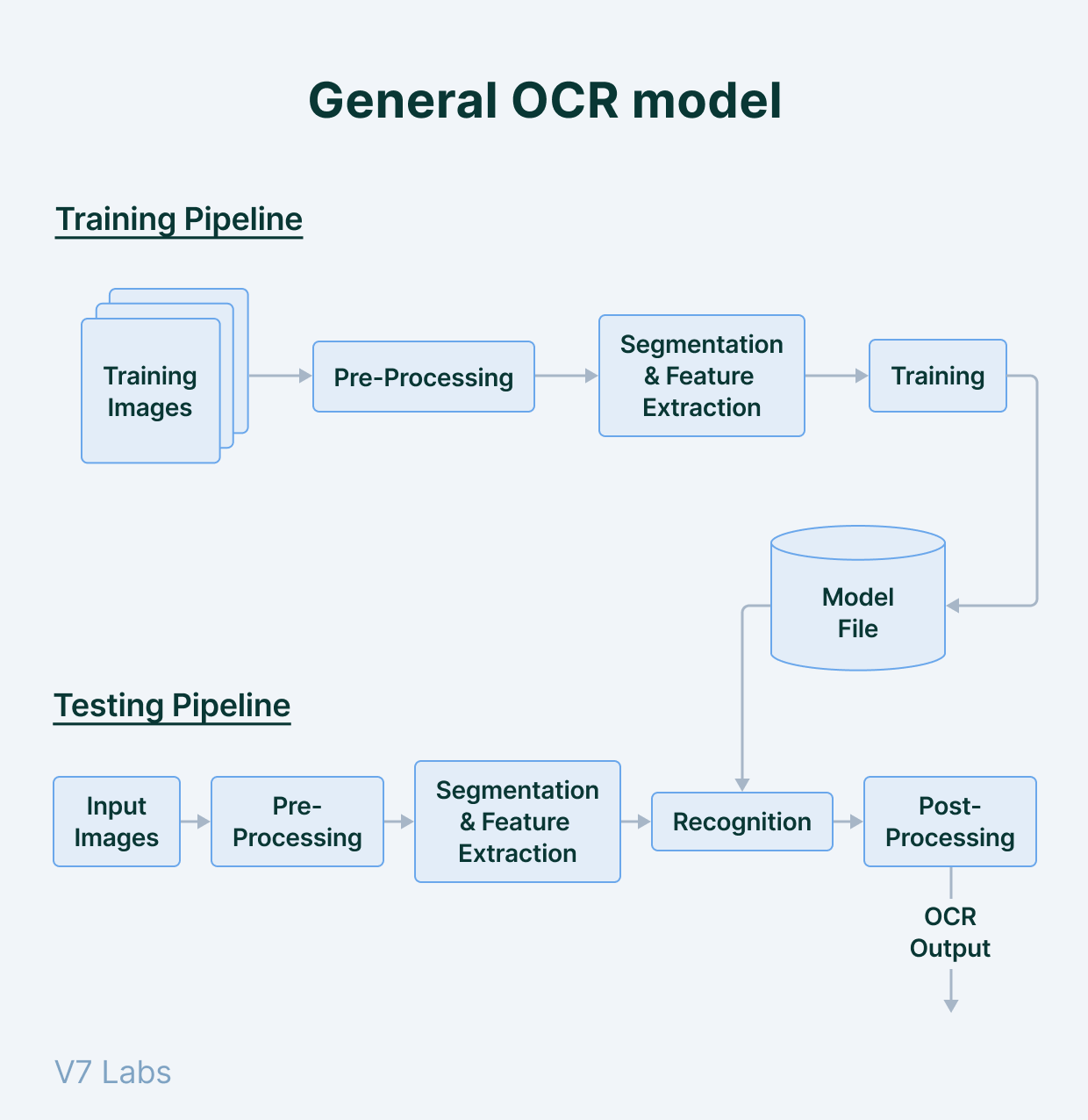
My product takes an image of the consumable with the best before date on it. Converts the date into text for the computer to read using optical character recognition. The data is then stored in excel because excel helps in storing a large number of dates so it will act like a database. We will extract the date once it is stored in excel and then find the current date. Keep in mind, the current date will keep on changing so the difference between the best before date and the current date keeps on increasing. Once we have found the current date, we will find the difference between the best before date and the current date to find out when the product will expire. If the difference is 1 then we have one day to finish the product. If this is the case, then an e-mail will be sent to a Gmail account which will state that your product is about to expire. The client will be alerted through whichever device he/she has logged his/her Gmail account into. This will make sure that the client consumes the product before the expiry date and doesn’t throw it away, which is the main purpose of this project. This is what I set out to achieve. I started to code this on my computer and after several days of trial and error, the results and what I set out to achieve were similar. The best before date was stored in a notepad file because it is a .txt format. It was then stored in excel so we could easily extract the date later when we needed to. The current date and best before date was compared and there were no problems there. Although, The code for the E-mail had several issues, mostly that it didn’t get sent to the desired Gmail account but was solved within two days of debugging. There were some issues, but they were solved. I also kept on enhancing the code like I decreased the number of lines of code so I could easily understand it and separate out code that performed different functions like the code for running the web camera on the computer was separated out with the main python code as I didn’t want the camera to run every time, I executed the code. I also decreased the amount of libraires I used without changing the function of the product like SSL library in python which secures the connection between python and the internet. Web browsers on the computer like Google Chrome, Firefox, Brave or if you are on any mobile devices then Safari, Samsung Internet, Opera GX or any other official web browsers already make the connection secure so there was no point in using it. Overall, the results matched what I was setting out to achieve. However, there were some errors which I fixed, and I improved the code as mentioned above.

# Introduction:

The main purpose of this project was to make sure that the client knows when their food gets expired and consume it before the expiry date to prevent food being wasted. The idea for this project sprang to my mind when I was asking my mother for a piece of chocolate from a chocolate bar. She opened the fridge and before she handed the chocolate bar to me, she checked its expiry date, and it was about to expire. She proceeded to tell me that the food was expired and simply threw it away. This made me think how this could be prevented because we just wasted food and we would have to buy a new chocolate bar. I then thought of how I could prevent this and decided that it was going to be in the form of a device. I then started to think what the device was going to be made up of as it should be compact. I then thought of what the product was going to do. I brainstormed several ideas and then came up with the conclusion with research to back me up that there was going be a small computer and a camera which would record the expiry date and tell the user when it will expire. That still wasn’t enough, but I made progress. I then researched compact computers that can handle running and stumbled across the Raspberry Pi. I decided to use the latest Raspberry Pi Model 4 B along with a camera from its official website as the camera will be compatible with the Raspberry Pi. I kept on researching how I would record and send the date. After several weeks of researching, I came to a conclusion that I would use a computer and its web camera. I will use a programming language called python. I will take a picture from the web camera, Turn the picture into characters using OCR and then make sure Python can read the date. Then I will store int into excel as it will act like a database, Read the date from it and find the current date which will obviously keep on changing. After that, I will compare the expiry date with the current date. Note that the difference will keep on increasing because the current date keeps on increasing too and the expiry date stays the same. Finally, I will send four emails to the client’s desired Gmail account to alert the client three days, two and one day before the expiry date and finally when the product has expired the email will tell the client to throw it away and send the E-mail using an internet connection. This way my initial goal was achieved which was to make sure the client knows a couple of days prior to his/her product being expired and when it has expired to just throw it away. If this was implemented in every household in Ireland and any other country, then the food wastage rate would decline rapidly, and people would save a lot of money because they wouldn’t have to pay so much for electricity and not have to buy the product again. In conclusion, the initial goal I wanted to achieve has been fulfilled with this product.

How does tesseract work?

OCR stands for optical character recognition. It converts images to text for the computer to read. Tesseract is a module which performs OCR. Tesseract first finds the text and finds out whether they are fixed pitched or not. Fixed pitched is when the characters in the text occupy the same number of spaces between each of them. Then if they are fixed pitched ,which is mostly the case unless it is handwritten, tesseract cuts the text into characters and then converts the characters into strings. Finally, Tesseract converts the strings into their original form by turning the string back into fixed pitch and then displaying the text. Here is a flowchart of how OCR works with the help of machine learning:

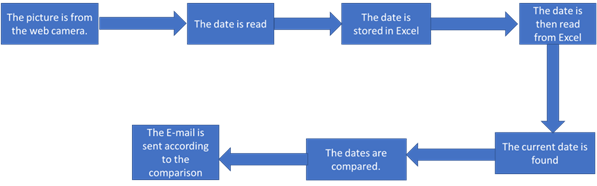


# 

To make OCR more accurate, Machine learning can be introduced when OCR uses its pattern-based algorithms, to compare the text with the images to its database. That database can be trained with the help of machine learning.

# Experimental Methods:

Here is another diagram which shows the steps of the code:



Now I will be explaining all the code I wrote section wise:

Step 1:

First of all, I will download tesseract from the official website as this will help me complete Optical character recognition which converts all the characters found in images to text saved in a notepad file so that the computer can identify the text which will be the best before date in this case for further use. Here is where I got the installer from.

<https://github.com/UB-Mannheim/tesseract/wiki>

Graphical user interface, text, application, email

Description automatically generated

I chose the 64-bit installer and downloaded tesseract.

Step 2:

Once I have downloaded the tesseract library, I will use it to convert a date let’s say the date 26 of December 2022. I will run a command in visual studio code which will convert it. The command is as follows: tesseract.exe Can.png output -l eng.

Here is how it will look like:

Text

Description automatically generated

After this, a text file will be made in the directory which is C:\Users\Sam\Downloads\Date. The text file will be named as output.txt and here is how it will look like:

A picture containing graphical user interface

Description automatically generated

Here is the original image for comparison and it will be sent as an attachment later on:

Text, letter

Description automatically generated

Step 3:

Once the characters are converted to text. I can start coding. I will have to first import the modules to run the code like so:

Text

Description automatically generated

The module on line 4 will help me find the current date. The module on line 5 and 6 will help will store the date into excel. The module on line 7 will help me extract the date from excel and help store it in a variable. The modules from lines 8 to 13 will help me send the emails. In more detail, Line 8 will help in sending the email by defining a client session of STMP and then send the e-mail. STMP stands for Simple Mail Transfer Protocol. Lines 9 to 12 will help attach images and text to my Email and finally line 13 will find the path of the image I want to attach.

Step 4:

This is a quick step and all it does is find the current date, which automatically changes over time, and store it in a variable today. Here is the code:

Text

Description automatically generated

Step 5:

In this step, the code will read the date from a notepad file, format it and store it in a variable in date\_obj. More specifically, line 19 will find the text file from the path provided and convert it as a string using the ‘r’ command after the comma. It will store the string in a variable d. Line 20 will find out the first line as a string from the output file which is BEST BEFORE. However, we don’t need the first line because it is not the date we are looking for. We need the second line, Line 21 will the text as a list because we use the redlines function. Since we extracted the text as a list, there are going to be characters we don’t need like /n because we can’t format it later on and they are not part of the date. Line 22 will replace the /n character with a blank space character. And store it back in line\_2 variable. After that, line 23 will find the first element of the list of lines\_2 and store it as a string. Now we have the date, but we need to format it into year-month-day. We can do this be using the strptime() which will help format the string which is the date. Line 24 will use the function to format the string and store it back in the date\_obj variable. Now that everything is formatted, line 25 will simply print or display in the terminal the date we have so we can see if it matches the date in output.txt.

Text

Description automatically generated

It does match the date:

Text

Description automatically generated

\*Note that the rest of the lines underneath ‘date read before being stored in excel 2022-12-26 00:00:00’ highlighted in red will be explained later.

The date (2022-12-26 00:00:00) does in fact match the date stored in the output.txt file(22 12 26) so there are no logic errors in our code so far and everything is working as planned. The date is more enhanced as it is formatted and there is an hour:minute:second timer added beside the date for more accuracy. Since there was no hour, minute, second in our screenshot it took. It sets to 00:00:00 by default.

Step 6:

Now that we have formatted the date, It is now ready to be stored in our database excel. Here is how the code looks like:

Graphical user interface, text

Description automatically generated

Going into more detail, the workbook class in line 28 represents one sheet of the spreadsheet in excel and it is imported from the openpyxl module we imported earlier. We will store it in the workbook variable. In Line 30, the workbook.active helps give the title of the first spreadsheet we created in line 28 which is named as Sheet and stores it in the worksheet variable. Naming it is important so that when we read from the excel file or extract the text, we can find out which sheet we want to extract the text from. In Line 32, we add the final date we formatted into the worksheet we previously named as Sheet. In Line 34, we add the date into the specific cell we want to, in this case we will store it in index 2 and at the first column which is A in excel. It is stored in index 2 because by default python starts to count from 0 so when we said index 2 it will actually go to index 3 which is not what we want or we say index 0 then it will give an error saying that the count is out of bounds. Finally, the date will be stored in A2 of the spreadsheet. Next in Line 36, we will save it in /Users/Sam/Downloads/insert\_row.xlsx or in other words the downloads folder and name it as insert\_row.xlsx so that we can access and extract the date later on when we need to.

Graphical user interface, application, table, Excel

Description automatically generatedHere is where it is stored in excel:

Step 7:

Now that we have stored the date in excel, we need to extract the date from our database. Here is how the code for read the date from excel looks like:

Text

Description automatically generated

For this section of our code, we are using the pandas module in python we imported earlier. First of all, we need to write Date on A1 on excel so that the program knows in which column to find the date. In line 40 we are finding were our excel file is which has the date located in it. In line 41, we are going to sheet1 or in excel it is just referred to Sheet because that is where our date is specifically located. Then, In line 42, in index 0 we can find the best before date and store it in the variable Best\_Before\_date. Finally, in line 43 we are printing the best before date along with the text. Here is how the excel file look like:

A screenshot of a computer

Description automatically generated with medium confidence

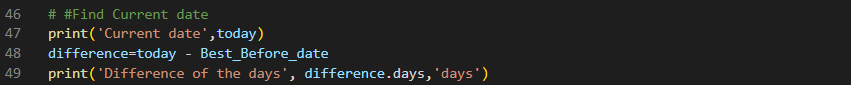
Here is what happens when we print the Best\_before\_date variable:

Text

Description automatically generated with low confidence

As we can see that our previous variable date\_obj wasn’t stored in excel when it was made and our new variable Best\_before\_date was stored in excel. There are no changes made to the dates so this means we can continue on without code without any errors. As I mentioned, the rest of the lines underneath will be explained later on.

Step 8:



Now that we have extracted the date from excel. We will find the difference between today’s date (at the time of coding this) and the best\_before\_date. This is what the code will be doing. In line 47, We will first display the current date at the time of coding this. Then, In line 48 we will subtract today(today stores the current date) from the best before date and store it in the variable difference. Finally In line 49 we display the difference in days. Here is the output:

Text

Description automatically generated

We can see that the current date for me is the first of January, but it will change once I am presenting. The subtraction is done right so we can continue on with the code. Finally, please note that I will explain how I got the last line.

Step 9:

For the final section of the code, we will be sending the E-mail using the following code and since this section is too big, I will be dividing this section of code but he is the fundamental code:

Text

Description automatically generated

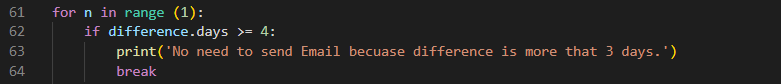
Here is Can.png:

A picture containing text

Description automatically generated

All of the code I am going to talk about in this section, the modules are from line 8 to line 13 from when I previously mentioned. In line 52, we are simply finding and opening the file called as Can.png. Then for Line 53, we are saying to the variable message that we are going to be adding attachments to the E-mail. Then for Line 54, we are saying which E-mail will be sending the Email and for Line 55, we are saying to who we are going to send this E-mail. Then in line 55, we are going to say what the subject of this Email is. Then for line 56, we are going to attach a message which is essentially the body of the message. Finally, for line 57, we are going to attach an image as an attachment. This is the fundamentals of the code which almost all the if loops will have.

Now we are going to compare the dates:



The part above basically says that if the difference is greater than four because we are going the compare the dates 4 times then there will be no need to send the message so we can break out of the loop. The for loop just runs all the if statements one time.

Here are the four if statements:

Three days before the product expires:

A screenshot of a computer

Description automatically generated with medium confidence

Two days before the product expires:

Text

Description automatically generated

One day before the product expires:

Graphical user interface, text

Description automatically generated

When the product finally expires:

Text

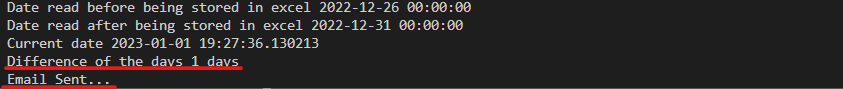
Description automatically generated

Here is how the results looks like if we keep the date read the same:

Text

Description automatically generated

Here is how the result looks like if we changed the date read to the 31 of December 2022 because the current date of when I am writing this report is the 1 of January 2023 so the difference will be one which means one of the if statements should work:



Here is the E-mail being sent to [samisnoice52@gmail.com](mailto:samisnoice52@gmail.com):

Graphical user interface, text, application

Description automatically generated

When I open it:

Graphical user interface, text, application, email

Description automatically generated

How does the code work:

Once I had the fundamentals written I simply implemented then in each of the If statements and it would do send the Email. I wrote the fundaments, set the connect and simply sent the E-mail. Here is how I set the connection:

A screenshot of a computer

Description automatically generated

I first say that I will need to send an E-mail in line 113. In line 114, I will say that I have an e-mail to send. Then in line 115, I will start the connection. After that, In line 116 I will login to the E-mail I will be sending through and the unique app password for my computer. In line 117, I will say that the send the message in this case I will say that the product has expired. Finally, in line 118, I will send the attachment and in line 119 I will send the E-mail. In addition, for line 120 I will basically display that the E-mail is sent so the user knows that. Now that everything has done in the main file. I will need to write the code for the camera so that the user can take a screenshot. Finally, note that when we send the attachment it didn’t change as when I changed the date read it was only a test and the date read will be reverted back to normal.

Step 10:

Here is how the code looks like for the camera:

Shape

Description automatically generated with medium confidence

Text

Description automatically generated

First of all, in line 2 we import the module cv2 which will help take pictures from the camera and in line 3 we import the module time which in this case will essentially delay some parts of the code.

Now that we have imported the modules, in lines 7 – 17. We will need to capture a picture from the web camera and store it in the variable cam. Then we simply set the length and width of the screen that will appear to take the image along with its encoding and how many frames it should run. Then in the last line we set the image counter to zero because this code is capable of capturing multiple images.

Text

Description automatically generated

Now we will open the test window and error handle it:

Text

Description automatically generated

In a summary from lines 20-29. We are opening a window and calling it a test window. Then we will simply run a loop to capture a screenshot from the web camera and if it can’t capture anything then delay the error handling code by 5 seconds and display the error that the camera couldn’t capture anything. Then break out of the loop. Then show the image. Finally, we will wait for one second. For the final section of this code, we will set some keys to take a screenshot and exit out of the test window:

Text

Description automatically generated

From lines 34 to 37, we will first see if the user has pressed the escape key and if he has then wait for 5 seconds and say that the escape key is pressed so the test window will close. Finally, from lines 38 to 42 we will see if the user has pressed the space key and if they have then save the screenshot as ‘opencv\_frame\_{screenshot name}.png and then show the image name and which screenshot number it is for the name of the screenshot. Then it will display the text that the screenshot has been taken and add the image counter to one because one screenshot has been taken. Here is how it will look like:

Here is the window:

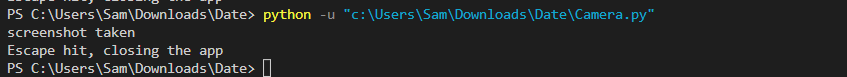
A screenshot of a computer

Description automatically generated with medium confidence

Here is the screenshot which is being taken:

A screenshot of a computer

Description automatically generated



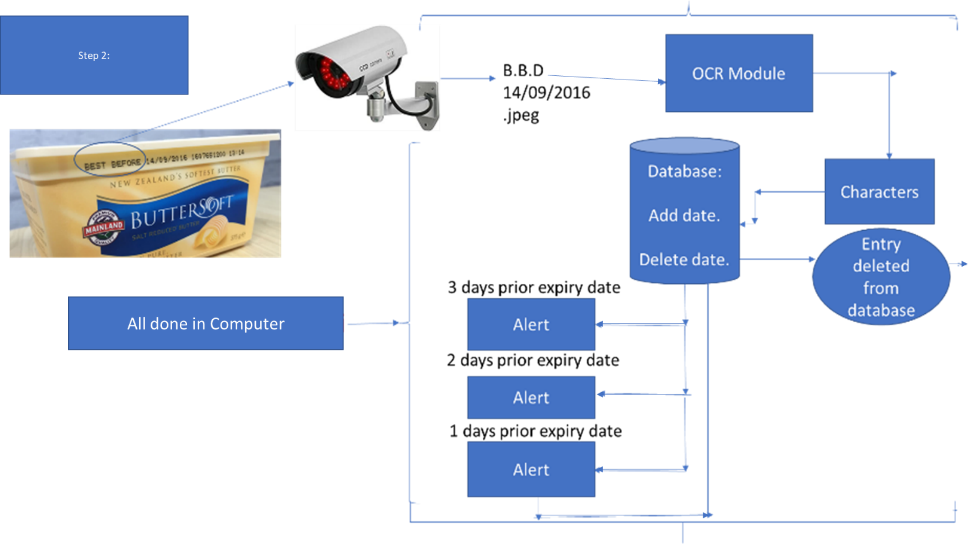
It also says that the escape button has been pressed and that the app will be closing. Now that all the code is written here are diagrams which will help explain how my code works:

All the products from the pantry’s image are taken from the camera:

A picture containing text, indoor, open, opened

Description automatically generated

Then each individual product, let’s say butter in this case, best before date is taken and then how the code work is explained:



# Results:

Now I will explain the results I got from the code written in the main python file I wrote:

The results I got for my code are the following below (Note that the current date for when I was writing this report book was 1 first of January 2023) :

Text

Description automatically generated

You can see that the best before date wasn’t changed after it was stored in excel and that the difference was greater than 4 days so there was no need to send the E-mail.

# Conclusions and recommendations:

Now that all the planning is done and all the code is written, it is time to come to a conclusion about the work I did. I am not the first person to come up with this invention as there are several other projects similar to mine however they are slightly different and I will explain why. There is a Smart Tagging platform called as Ovie Smarterware which places a SmartTag in the food products so it can track the product and see it’s expiry date, what the product is and what you can make out of it. There is not much detail is to how it works but the part where this differs from my product is that you don’t have to place tags on the product and can simply take a picture of the product’s expiry date. There is not need to pay 245 dollars for only 12 Ovie smart tags because one tag would be about 20 dollars and what if you have more than twelve products to store in the fridge. There are also other products you get in the bundle line cup containers, SmartWare universal connectors, clips and an Ovie hub which lets the tags communicate to the Ovie app. This quite expensive and not practical as the tags are not in high quantities and it can’t stick in unpackaged products like lettuce, tomatoes and etc as it might slip and come off.

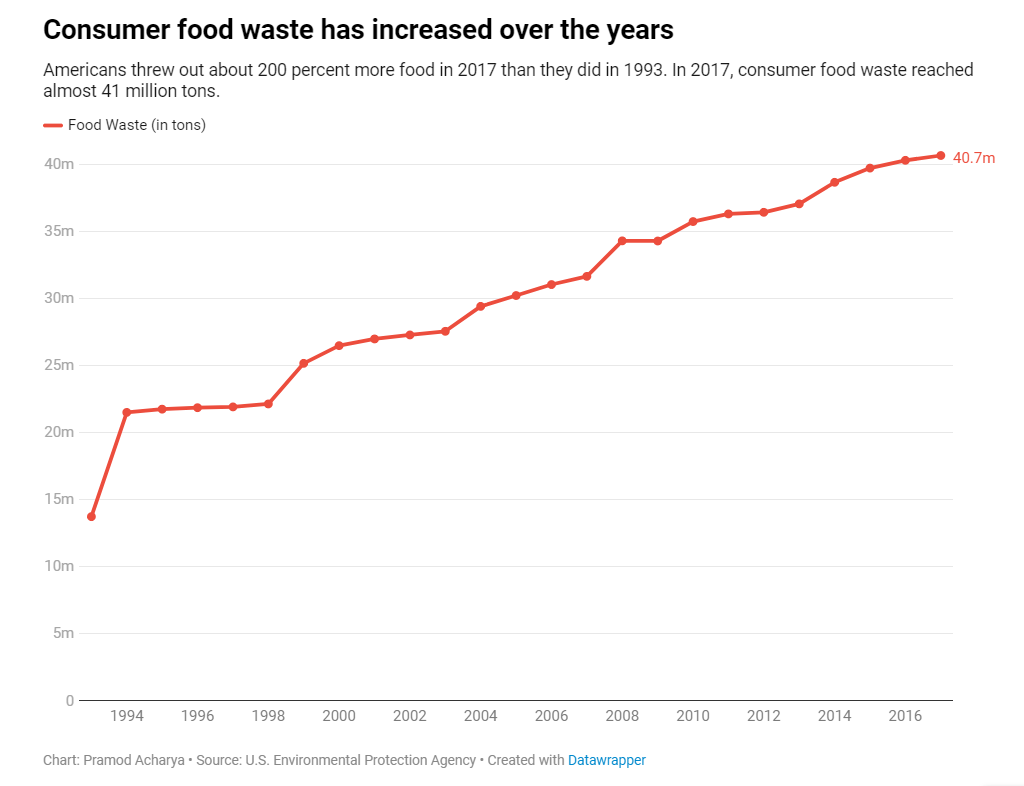
There is also another product which is similar to mine and it is an app called as Fridgely. All you have to do in that app is scan your product’s barcode and it will automatically figure out the product’s name and estimate the expiration date. It then sends an alert when the product is going to expire. Like the previous product, it didn’t explain the fundamentals behind the product but there are a lot of flaws in this app. For the app to estimate the product’s name and expiry date. It needs to have a database which contains all the consumable’s name, how they look like and their expiry date which doesn’t make sense and isn’t practical because this will require a lot of storage and expiry date’s vary depending on the product and its brand. Besides, if the barcode is used for finding the information it won’t be specific for finding the product’s name, where it was manufactured and it’s package size. It doesn’t find the product expiry date. If this isn’t sufficient to prove my point , a president of management strategies of a company called as Kent Aldershof says it himself ‘ When the bar code is scanned, as at a supermarket checkout, that information goes to a computer. The machine knows what is the current price for that item, which is fed back and becomes part of your checkout bill and your receipt.

But that does not include an expiration date.’ The question was ‘Can I get an expiration date from a barcode?’. This forum was found in Quora and here is the link for more information and credibility :

<https://www.quora.com/Can-I-get-an-expiration-date-from-a-barcode>

Most of the answers say no to the question in the forum. Overall, my product does significantly vary from all the products out there and it is more practical and cost effective as my product doesn’t cost much to make so if I were to sell this product then I wouldn’t charge the client that much. I would probably charge them 20 or 30 dollars which is the price of a single tag as I mentioned before.

The data (which is the date) I got from my code is accurate, the date is down to the exact hour, minute and second. The strong point of my code is that it performs the function that I want without any errors as it has been tested and debugged for hours. In contrast, the weak point is that I need to click a lot of buttons to make the program work for instance I need to copy and paste the program for the tesseract command to run and run it in the terminal, then I need to add the attachment into the folder which has my code. It also takes a lot of time to run the camera and is unreliable sometimes as the test window shuts down but if I have a longer time frame as 10 to 20 days isn’t enough to fully complete this project and 1 to 2 years is then I can bring this project to its full potential and it can be improved a lot but the 10 to 20 days timeframe is good enough to showcase a demo of the project and like any project it can be improved further. Speaking of improving it further, there are several ways to improve this project. I can add all the code I have and store it into the raspberry Pi and use the camera to capture the images, I can use machine learning which will help in finding out pattern based algorithms for OCR to use by training it frequently, I can make the tesseract convert the image to text autonomously, I can have an actual database like Oracle if I have the funding which stores the dates instead of Excel as it can’t store more than 1 million entries of data finally I could make the code run every day automatically instead of pressing the hit button as the client might forget and this will make the code completely automatic. As I keep on developing my project, I will find things to improve but these were the major ones I have to improve on. Finally, this product is changing the world because this will drastically reduce food wastage. If the consumer knows when the product will be wasted, then they will consume it before the product expires. This will make sure they don’t throw out the product. If this is implemented in every household then the food wastage rate would decline rapidly, here is a graph of food wastage rates in America:



As we can see that this is rising and if the product was implemented in households, then the amount of food wasted would decline rapidly. Now the food wastage is at 40.7 million tons. If food wasn’t wasted that much in America and all around the world then the land and soil wouldn’t be degraded, there wouldn’t be water pollution, and this is being caused because the resources are being used faster than they are being replaced. This can also stop global warming as the wasted food goes to the landfills and rots which produces methane, and that gas is a greenhouse gas and greenhouse gases cause global warming. In conclusion, with a small invention like mine can greatly impact the world in a positive way as my product drastically reduces food wastage. Finally, here are some major changes I made after I wrote the code and I explain why:

|  |  |
| --- | --- |
| What it initially was going to be | What change I made and why? |
| I was initially going to have my code in a Raspberry Pi Model B as it is more compact and can be implemented in fridges and pantries. It also supported the desired programming language for this project which was python. | I changed this because the libraires which would help run the code, especially tesseract couldn’t be downloaded onto the Raspberry Pi due to its file size. The raspberry Pi has 32GB micro-SD card and 28GB was used to download the Raspian OS and the rest 4 Gb were insufficient to download the libraries as the libraires were 5 to 10 GB. I couldn’t get another card as the delivery time on Amazon, DID, Euronics and electrocity were too much and some of them didn’t supply the card. The computer was capable of handling the file size of the libraries and could easily run the code, which led me to make the decision. |
| I was also going the print the hour, minute and second along with the date for when I was finding the current date and the expiry date | After I coded everything, I ran the code and it gave me a logic error which was when I was finding the difference it reduced a day. When I said the date read was the 14/01/2023 and the current date was 05/01/2023 it gave me a difference of 8 days and not 9 days because the hour minute and second where involved and I ran the code as 6 o clock pm so it didn’t qualify as a day so it reduced a day when finding the difference. I didn’t need this amount of accuracy as I was just dealing with days so I set the hour minute and second to all zeros and now when I ran the code again the difference was 9 days and not 8 days. |

# Acknowledgements:

I have consulted my Physics and Chemistry teacher which is Miss Carla for guidance as I needed to know which libraries I needed to run my code and checked all the code with her to see if it was right.

# Appendices:

I would just like to remind the panel that this is a demo of the project and if I do get selected, I will improve the code further on. This demo will just make sure that the panel know the main function of this project and I haven’t developed the full product because the time frame was too limited to complete this entire project like I mentioned earlier.

# References:

Here are my references in a link format:

<https://www.quora.com/Can-I-get-an-expiration-date-from-a-barcode>

<https://flatlandkc.org/eats-drinks/food-waste-piling-up-more-than-ever/>

<https://github.com/UB-Mannheim/tesseract/wiki>

[https://aws.amazon.com/what-is/ocr/#:~:text=A%20simple%20OCR%20engine%20works,is%20called%20optical%20word%20recognition](https://aws.amazon.com/what-is/ocr/" \l ":~:text=A simple OCR engine works,is called optical word recognition).

<https://www.baeldung.com/cs/ocr>

<https://www.youtube.com/watch?v=jO-1rztr4O0&t=212s>

<https://www.youtube.com/watch?v=ZNrteLp_SvY>

<https://www.geeksforgeeks.org/reading-text-from-the-image-using-tesseract/>

<https://nanonets.com/blog/ocr-with-tesseract/>

<https://www.projectpro.io/article/how-to-train-tesseract-ocr-python/561>

<https://pypi.org/project/pytesseract/>

<https://www.youtube.com/watch?v=4uWp6dS6_G4>

<https://www.youtube.com/watch?v=PY_N1XdFp4w>

<https://www.youtube.com/watch?v=JkzFjj2hjtw>

<https://www.youtube.com/watch?v=oyqNdcbKhew>

[https://fridgelyapp.com/#:~:text=Fridgely%20tells%20you%20when%20your,food%20is%20close%20to%20expiring.&text=Stop%20wasting%20food-,The%20average%20family%20of%20four%20wastes,of%20their%20food%20each%20year](https://fridgelyapp.com/" \l ":~:text=Fridgely tells you when your,food is close to expiring.&text=Stop wasting food-,The average family of four wastes,of their food each year).