



FACULTY OF  
**AUTOMATIC CONTROL  
AND COMPUTERS**  
University POLITEHNICA of Bucharest

## *OOP Project 2020-2021*

*Object oriented programming*

*Teacher: Dan Mihail CARAMIHAI*

*University Polytechnic of Bucharest, Faculty of  
Automatic Control and Computers*

*Risk assessment in oncological disease by determining  
the surface area of malignant tumors*

## *Contents:*

- § Information about the team of students*
- § Cancer, what is it, breast cancer*
- § The program to determine the stage of cancer  
(approximative)*
- § What is OpenCV*
- § The interface*
  - What options were considered*
  - What option did we settle on?*
- § Art direction*
- § Bibliography*

## *Who are we?*

*We are a group of students from the faculty of Automatic Control and Computers our coordinating teacher is Dan Mihail Caramihai and under his guidance we started this project with the purpose of extending our programming knowledge and of determining whether the breast cancer of certain persons and their condition may worsen or improve.*



RUCAREANU CATALIN-STEFAN:

FLECTERE SI NEQUEO SUPEROS, ACHERONTA  
MOVEBO.

LEAD PRODUCER, ART DIRECTION AND MUSICIAN,  
LEAD PROGRAMMER, INTERFACE DESIGNER,  
ASSESSMENT ASSISTANT, LEAD RESEARCHER



OLOGU CATINCA IOANA:

FINIS CORONAT OPUS.

TEAM LEADER, LEAD COORDINATOR, LEAD  
RESEARCHER, LEAD EVALUATOR, APPLICATION  
TESTER

*Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu,  
Madalin Tilmaciu, Florean Edi Andrei*



TILMACIU CONSTANTIN MADALIN:

MALUM CONSILIUM QUOD MUTARI NON  
POTEST.

DOCUMENTATION, IDEA GENERATOR, TESTING

FLOREAN EDI ANDREI:

DULCE PERICULUM.

DOCUMENTATION, IDEA GENERATOR, TESTING



STANOIU ANDREI:

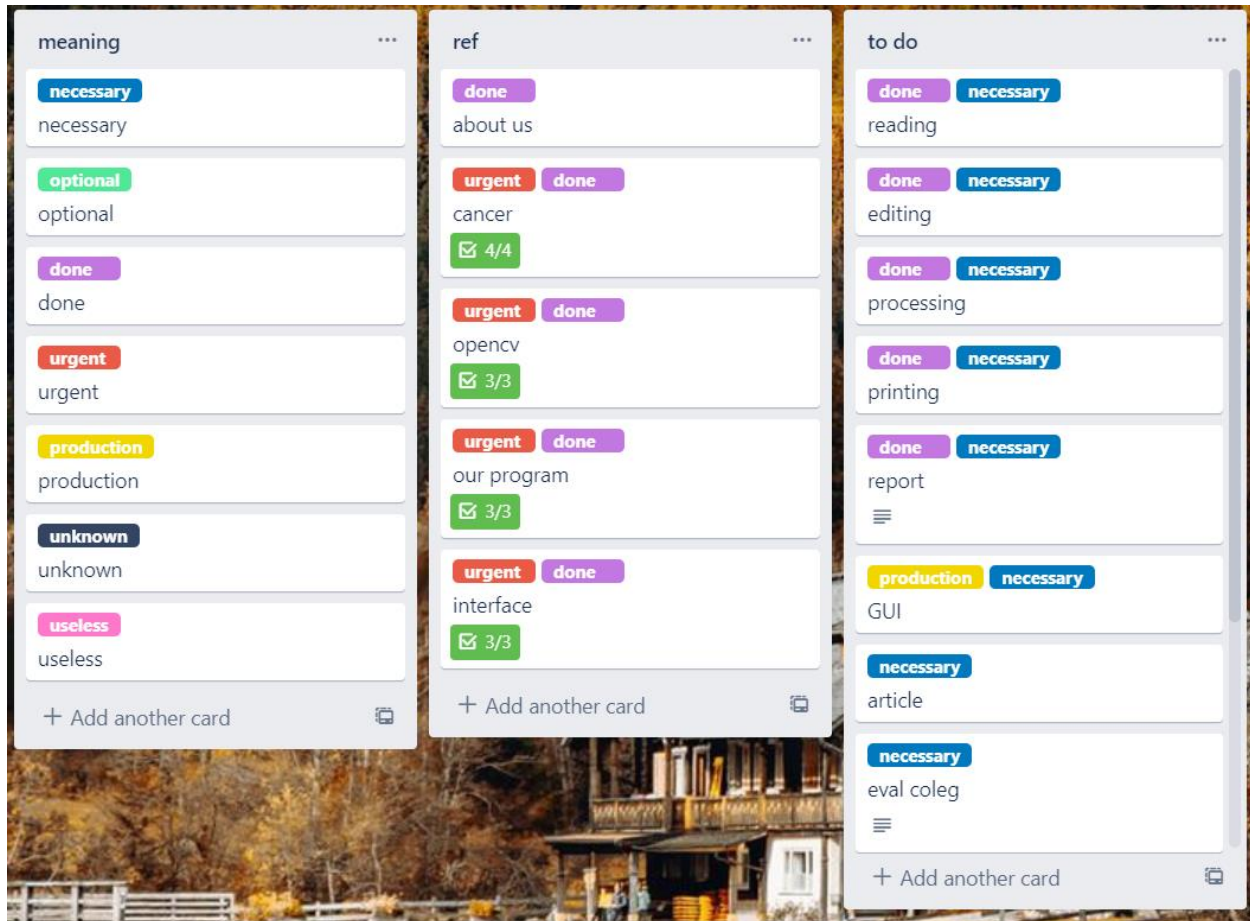
CONDEMNANT QUO NON INTELLEGUNT.

DOCUMENTATION, IDEA GENERATOR, TESTING



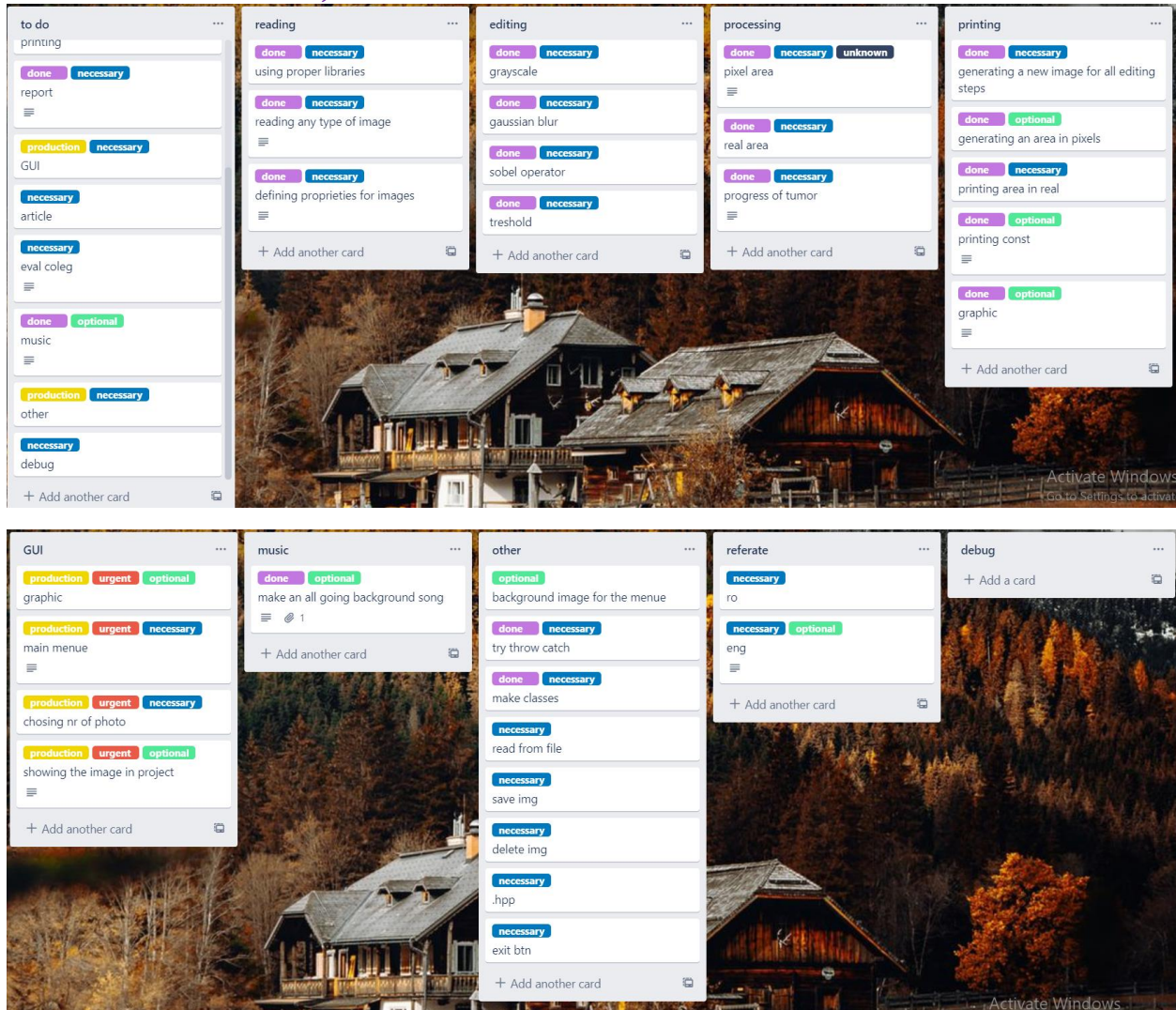
*How we organized ourselves:*

We have used the site trello for organizing our work:





## *Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei*



## *What is cancer?*

Cancer is the name given to a collection of related diseases. In all types of cancer, some of the body's cells begin to divide without stopping and spread into surrounding tissues.

Cancer can start almost anywhere in the human body, which is made up of trillions of cells. Normally, human cells grow and divide to form new cells as the body needs them. When cells grow old or become damaged, they die, and new cells take their place.

## *Faculty of Automatic Control and Computers*

When cancer develops, however, this orderly process breaks down. As cells become more and more abnormal, old or damaged cells survive when they should die, and new cells form when they are not needed. These extra cells can divide without stopping and may form growths called tumors.

Many cancers form solid tumors, which are masses of tissue. Cancers of the blood, such as leukemias, generally do not form solid tumors.

The most dangerous types of cancer are:

1. ***Lung and bronchial cancer:*** 792,495 lives Lung and bronchial cancer is the top killer cancer in the United States. Smoking and use of tobacco products are the major causes of it, and it strikes most often between the ages of 55 and 65, according to the NCI.
2. ***Colon and rectal cancer:*** 268,783 lives Colon cancer grows in the tissues of the colon, whereas rectal cancer grows in the last few inches of the large intestine near the anus, according to the National Cancer Institute.
3. ***Breast cancer:*** 206,983 lives Breast cancer is the second most common cancer in women in the United States, after skin cancer, according to the Mayo Clinic.
4. ***Pancreatic cancer:*** 162,878 lives Pancreatic cancer begins in the tissues of the pancreas, which aids digestion and metabolism regulation.
5. ***Prostate cancer:*** 144,926 lives This cancer is the second-leading cause of cancer deaths in men, after lung and bronchial cancer, according to the NCI.
6. ***Leukemia:*** 108,740 lives There are many types of leukemia, but all affect the blood-forming tissues of the body, such as the bone marrow and the lymphatic system, and result in an overproduction of abnormal white blood cells, according to the NCI.

***Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei***

Since breast cancer is in the top of most devastating types of cancer, we must put our focus on it, being one of the few sicknesses that affects the fairer gender.

In the beginning stage of our project we have talked with a couple of well informed people in both medicine and in programming, the main goal that we had in mind while we discussed with those persons was to find more information about cancer, breast cancer and also about processing of images and about visual graphic user interface.

Main reason for talking with those people was to base our studies on real physical evidence and to be sure that our work is as fail-proof as possible.

In that order of thought, we have talked with: ***Bill Pomidor, Physician/MD and Unity Developer: Tallahassee, Florida, USA;*** and ***Paolo Alejandro Catilo deep tech entrepreneur Geneva, Canton of Geneva, Switzerland*** on the basis of the medical side of our project and for the programming part we talked with: ***Iulia-Lidia IACOB, Stefan Alexandru MOCANU, teachers and laboratory assistants at Polytechnic of Bucharest in the field of Computer Programming,*** also we got help from our teacher ***Dan Mihail Caramihai,*** from whom we received the photos we had to evaluate and our laboratory assistant who gave us some ideas and nudged us in the right direction, ***Daniel-Ioan Chis.***





***Breast cancer:***

All types of cancer are being classified in a couple of different stages, depending on their size and on their visibility on the top layer of skin as follows.

***TNM Classification for Breast Cancer***

The ***TNM Classification of Malignant Tumors (TNM)*** is a globally recognized standard for classifying the extent of spread of cancer. It is a classification system of the anatomical extent of tumor cancers. It has gained wide international acceptance for many solid tumor cancers but is not applicable to leukemia and tumors of the central nervous system. Most common tumors have their own TNM classification. Sometimes also described as the AJCC system.



<b>Tumor size</b>  <b>T</b>	Tumor size < 2 cm  <b>T1</b>	Tumor size 2-5 cm  <b>T2</b>	Tumor size > 5 cm  <b>T3</b>	Tumor extends to skin or chest wall  <b>T4</b>
<b>Lymph Nodes</b>  <b>N</b>	<b>N0</b> No lymph node metastasis	<b>N1</b> Metastasis to ipsilateral, movable, axillary LNs	<b>N2</b> Metastasis to ipsilateral fixed axillary, or IM LNs	<b>N3</b> Metastasis to infraclavicular/supraclavicular LN, or to axillary and IM LNs
<b>Metastasis</b>  <b>M</b>	<b>M0</b> No distant metastasis	<b>M1</b> Distant metastasis	LNs= Lymph Nodes; IM= Internal Mammary	

### 1. *Primary Tumor (T)*

- T1:  $\leq 2$  cm (20 mm)
  - mi:  $\leq 1$  mm
  - a: 1-5 mm
  - b: 5-10 mm
  - c: 10-20 mm
- T2: 2-5 cm
- T3:  $> 5$  cm
- T4: Direct extension to chest wall and/or skin
  - a: Chest wall (exclude only pectoralis muscle adherence/invasion)
  - b: Ipsilateral ulceration, satellite nodules or peau'd orange
  - c: Both a and b
  - d: Inflammatory carcinoma

### 2. *Regional Lymph Node (N)*

- N1: Ipsilateral and mobile level I and II axillary nodes
- N2:

*Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei*

- a: Ipsilateral and matted level I and II axillary nodes
  - b: Ipsilateral internal mammary nodes only
  - N3:
    - a: Ipsilateral level III axillary nodes (infraclavicular node)
    - b: Ipsilateral internal mammary nodes + level I, II nodes
    - c: Ipsilateral supraclavicular nodes
3. *Distant Metastasis (M)*
- cMo (i+): Clinically and radiologically normal but  $\leq 0.2$  mm tumor cells in blood, bone marrow or other nonregional nodal tissues
  - M1:  $>0.2$  mm metastases

***Mandatory parameters***

- **T:** size or direct extent of the primary tumor
  - **T0:** no evidence of tumor
  - **T1, T2, T3, T4:** size and/or extension of the primary tumor
- **N:** degree of spread to regional lymph nodes
  - **N0:** no regional lymph nodes metastasis
  - **N1:** regional lymph node metastasis present; at some sites, tumor spread to closest or small number of regional lymph nodes
  - **N2:** tumor spread to an extent between N1 and N3 (N2 is not used at all sites)
  - **N3:** tumor spread to more distant or numerous regional lymph nodes (N3 is not used at all sites)
- **M:** presence of distant metastasis
  - **M1:** metastasis to distant organs (beyond regional lymph nodes)

## *Faculty of Automatic Control and Computers*

	N0	N1	N2	N3	M1
T1	I	IIA	IIIA	IIIC	IV
T2	IIA	IIB	IIIA	IIIC	IV
T3	IIB	IIIA	IIIA	IIIC	IV
T4	IIIB	IIIB	IIIB	IIIC	IV

M1 = Stage IV

T4 = Stage III – b or c

N3 = Stage III – c

N2 = Stage III – a or b

T4N2 = Stage IIIB

T 1,2,3 N2 = Stage IIIa

Add up to make 1: Stage I (T1No)

Add up to make 2: Stage IIa (T1N1 or T2N0)

Add up to make 3: Stage IIb (T3N0 or T2N1)

Add up to make 4: Stage IIIa (T3N1)

### *OpenCV:*

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire

*Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei*

scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

For our program we have used a couple of interesting functions that came from OpenCV: reading and writing of images, grayscale filter, blurring filter, Sobel filter and the Threshold filter. The other parts of our algorithm were hand made.

*Reading image:*

```
image = imread("res/" + name + ".jpg", IMREAD_COLOR);
try
{
    if (!image.data) {
        throw - 1;
    }
    else {
        ...
    }
}
catch (const int impos)
{
    cout << "Could not open the image file" << endl;
    return impos;
}
```

*Grayscale filter:*

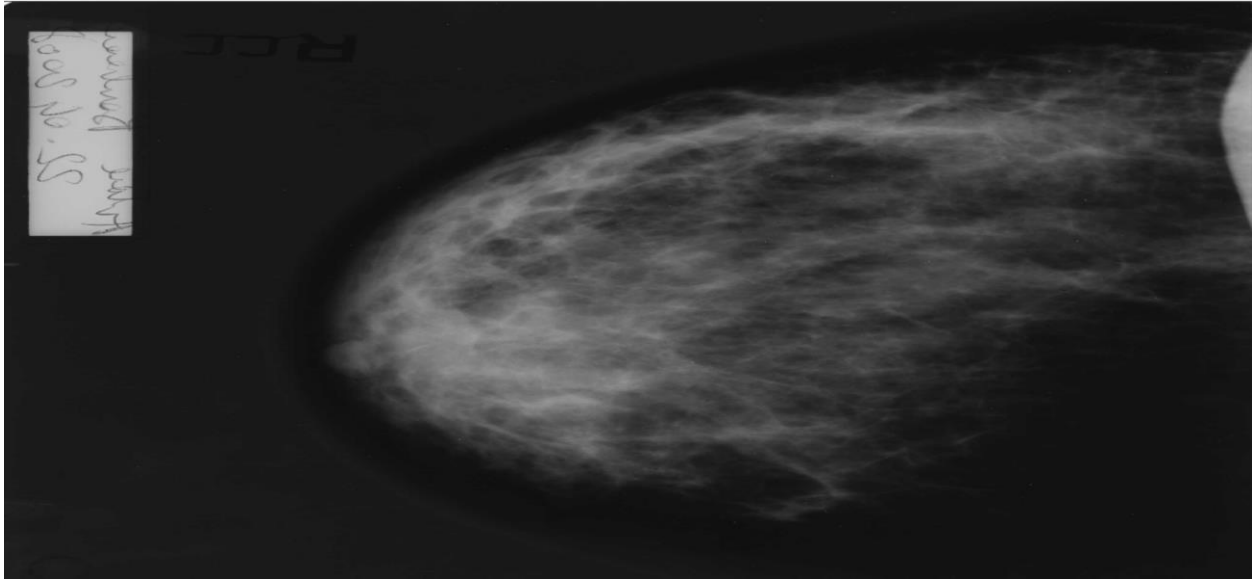
To grayscale an image a person has to take all the three color fields of every pixel and use a simple formula to determine the gray value of that certain pixel, after one pixel is ready we move to the next and do the same; the formula to use is:

*GrayValue = RedValue\*0.2126+GreenValue\*0.7152+BlueValue\*0.0722*

But with the use of OpenCV there is no need of doing that by ourselves, we can use the function already made:

```
height = image.rows;
width = image.cols;
cout << "height=" << height << endl;
cout << "width=" << width << endl;
cvtColor(image, grayImage, COLOR_BGR2GRAY);
```





*Write image:*

```
void Write()
{
    image = imwrite("res/" + name + "_Gray_blur.jpg", blurredGrayImage);

    image = imwrite("res/" + name + "_Sobel.jpg", grad);

    image = imwrite("res/" + name + "_Tresh.jpg", dst);
}
```

*Gaussian Blur:*

In image processing, a Gaussian blur (also known as Gaussian smoothing) is the result of blurring an image by a Gaussian function (named after mathematician and scientist Carl Friedrich Gauss).

It is a widely used effect in graphics software, typically to reduce image noise and reduce detail. The visual effect of this blurring technique is a smooth blur resembling that of viewing the image through a translucent screen, distinctly different from the bokeh effect produced by an out-of-focus lens or the shadow of an object under usual illumination.

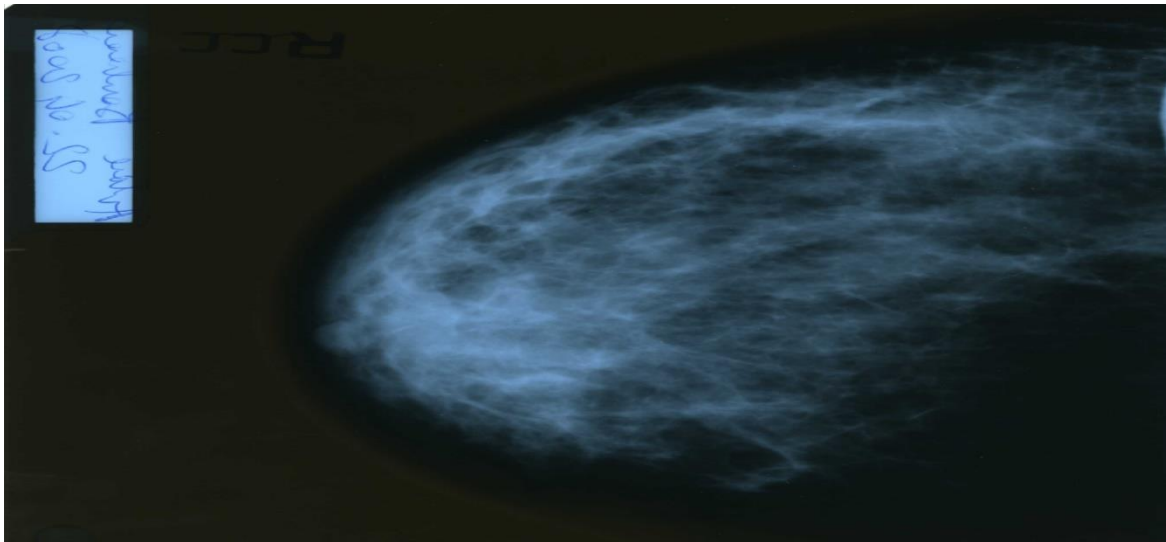
In this method, instead of a box filter, a Gaussian kernel is used. It is done with the function, `cv.GaussianBlur()`. We should specify the width and height of the kernel which should be positive and odd. We also should specify the standard deviation in the X and Y directions, `sigmaX` and `sigmaY` respectively. If only `sigmaX` is

*Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei*

specified, sigmaY is taken as the same as sigmaX. If both are given as zeros, they are calculated from the kernel size. Gaussian blurring is highly effective in removing Gaussian noise from an image.

And thus gaussian filtering is made by the convolution of our image (taken as a matrix) with our gaussian matrix such as the values of our matrix add up to 1, with the element in the center being the biggest and getting smaller by moving away from the center of the convolutional matrix

```
GaussianBlur(grayImage, blurredGrayImage, Size(BLUR_RATIO, BLUR_RATIO), 0, 0);  
GaussianBlur(image, blurredImage, Size(BLUR_RATIO, BLUR_RATIO), 0, 0);
```



*Sobel filtering:*

This filter necessitates again the convolution of our image matrix with a filtering matrix.

Getting the bidirectional filtering of our image needs to get first the X directional filtering of the image, then the Y directional, and after that we must add those 2 together.

The Sobel operator, sometimes called the Sobel–Feldman operator or Sobel filter, is used in image processing and computer vision, particularly within edge detection algorithms where it creates an image emphasizing edges. It is named after Irwin Sobel and Gary Feldman, colleagues at the Stanford Artificial Intelligence Laboratory (SAIL).

## *Faculty of Automatic Control and Computers*

the operator uses two  $3 \times 3$  kernels which are convolved with the original image to calculate approximations of the derivatives – one for horizontal changes, and one for vertical. If we define A as the source image, and Gx and Gy are two images which at each point contain the horizontal and vertical derivative approximations respectively, the computations are as follows:

$$Gx = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} * image$$
$$Gy = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * image$$

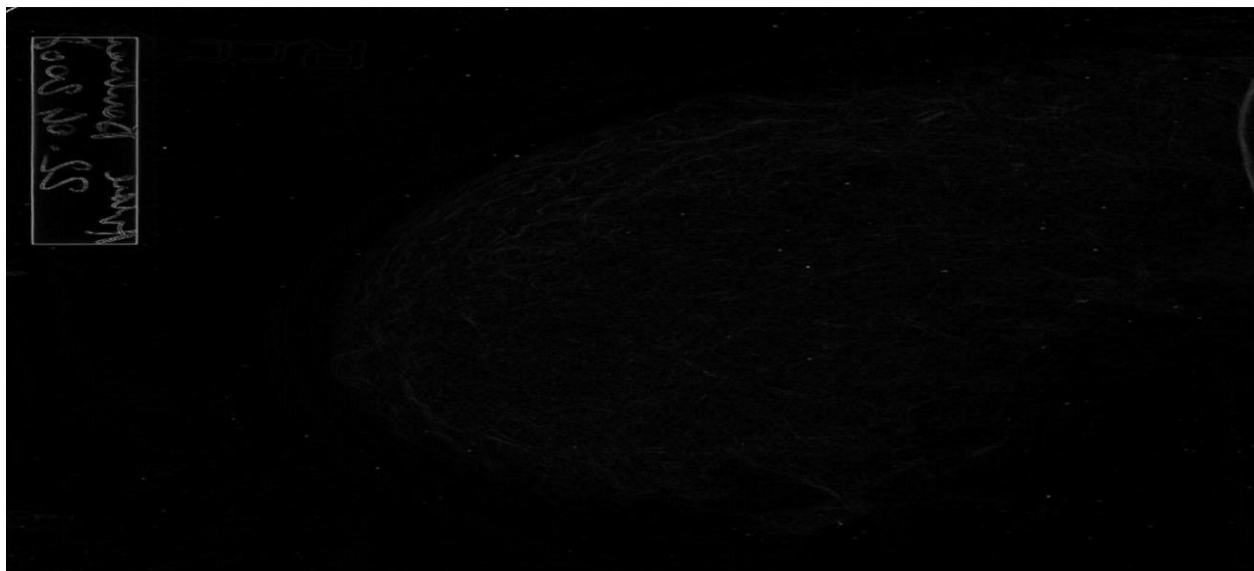
At each point in the image, the resulting gradient approximations can be combined to give the gradient magnitude, using:

$$G = \sqrt{Gx + Gy}$$

```
Sobel(blurredGrayImage, grad_x, CV_16S, 1, 0, 3, 1, 0, BORDER_DEFAULT);
Sobel(blurredGrayImage, grad_y, CV_16S, 0, 1, 3, 1, 0, BORDER_DEFAULT);

convertScaleAbs(grad_x, abs_grad_x);
convertScaleAbs(grad_y, abs_grad_y);

addWeighted(abs_grad_x, 0.5, abs_grad_y, 0.5, 0, grad);
```



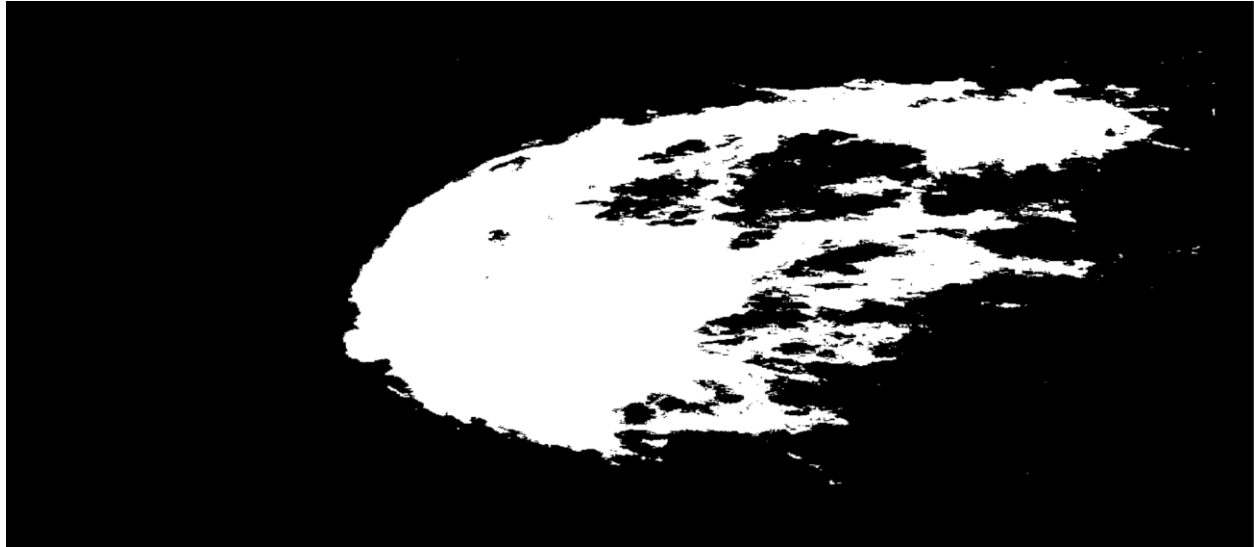
*Threshold filter*

*Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei*

In digital image processing, thresholding is the simplest method of segmenting images. From a grayscale image, thresholding can be used to create binary images.

The simplest thresholding methods replace each pixel in an image with a black pixel if the image intensity is less than some fixed constant  $T$ , or a white pixel if the image intensity is greater than that constant.

```
threshold(grayImage, dst, thresh, maxValue, THRESH_BINARY);
```



For a faster processing of the images we have made a txt file named sani.txt to write the names of the images to process them all together

The code to read from a file all the names of the images is:



```
vector<string> name;
string n,fisier= "res/sani.txt";
ifstream m1(fisier);
try
{
    if (m1.is_open())
    {
        while (getline(m1,n))
        {
            name.push_back(n);
        }
    }
    else
    {
        throw fisier;
    }
}
catch (const string fisier)
{
    cout << "Nu exista fisierul cu denumirea " << fisier << endl;
    exit(404);
}
double ariaCm, alfa, ariaPozCm, ariaPx, ariaPoz;
int position;
cout << "\nSe proceseaza " << name.size() << " imagini\n";
```

After we have made the threshold image, we had to count all the white pixels and get the discreet area in pixels and then to transform it in centimeters.

We have used the following code to do all that work:

```
ariaPoz = this->width * this->height;

for (i = 0; i < dst.cols; i++) {
    for (k = 0; k < dst.rows; k++) {
        if (dst.at<uchar>(k, i) == 255)
        {
            ariaPx++;
        }
    }
}

beta = 0.17;
alfa = beta * 10 * 3 * 3.14 / ariaPoz;
ariaCm = ariaPx * alfa;
ariaPozCm = ariaPoz * alfa;
```

ariaPoz is the area in pixels of the image that is being processed

Alfa is the constant with which we get the area in centimeters as based on the area of a medium sized breast of diameter 10.2 cm

ariaCm is the area of the malignant cancer in centimeters

ariaPx is the area of the malignant cancer in pixels

ariaPozCm is the area in centimeters of the image that is being processed

```
double getArea()  
{  
    return this->ariaPx;  
}  
  
double getAreaCm()  
{  
    return this->ariaCm;  
}  
  
double getAreaPoz()  
{  
    return this->ariaPoz;  
}  
  
double getAreaPozCm()  
{  
    return this->ariaPozCm;  
}  
  
double getAlfa()  
{  
    return this->alfa;  
}
```

Those functions return the values that are necessary to print on the screen.

```
PlaySound(L"res/snd2.wav", NULL, SND_LOOP | SND_ASYNC);
```

Using this line of code, we play the song made special for this program, more details in the Art Direction chapter.

Displaying of the image is made by using the code:

*Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei*

```
void Display()
{
    namedWindow("Display window", WINDOW_NORMAL);
    imshow("Display window", image);

    namedWindow("Gray Image", WINDOW_NORMAL);
    imshow("Gray Image", grayImage);

    namedWindow("Blured Gray Image", WINDOW_NORMAL);
    imshow("Blured Gray Image", bluredGrayImage);

    namedWindow("Blured Image", WINDOW_NORMAL);
    imshow("Blured Image", bluredImage);

    namedWindow("Sobel Image", WINDOW_NORMAL);
    imshow("Sobel Image", grad);

    namedWindow("Tresh Image", WINDOW_NORMAL);
    imshow("Tresh Image", dst);

    namedWindow("Graph Image", WINDOW_NORMAL);
    imshow("Graph Image", graphic);
}
```

### *Interface:*

Every person in our team has had a different way to produce the interface but we have kept only managed C++ and the others were discarded.

Rucareanu Catalin-Stefan: *C#, managed C++*

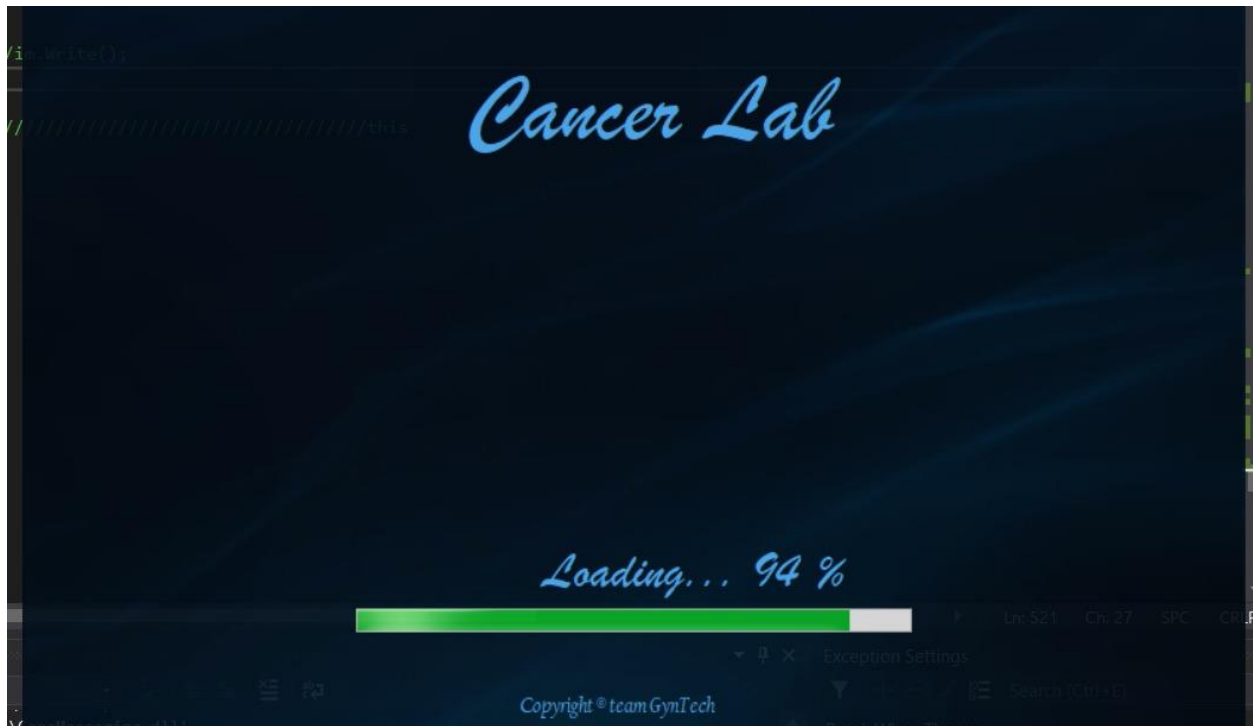
Ologu Catinca Ioana: *C#, managed C++*

Andrei Stanoiu: *Java*

Madalin Tilmaciu: *openGL*

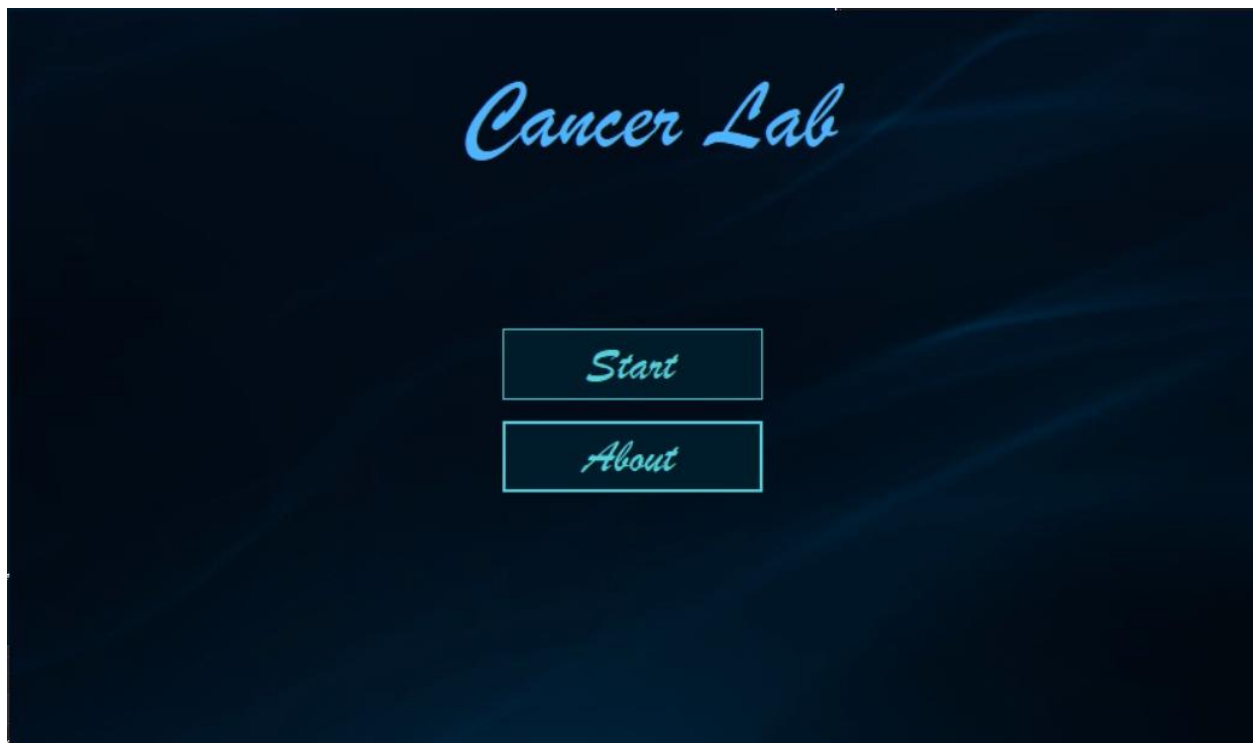
Florean Edi Andrei: *QT*





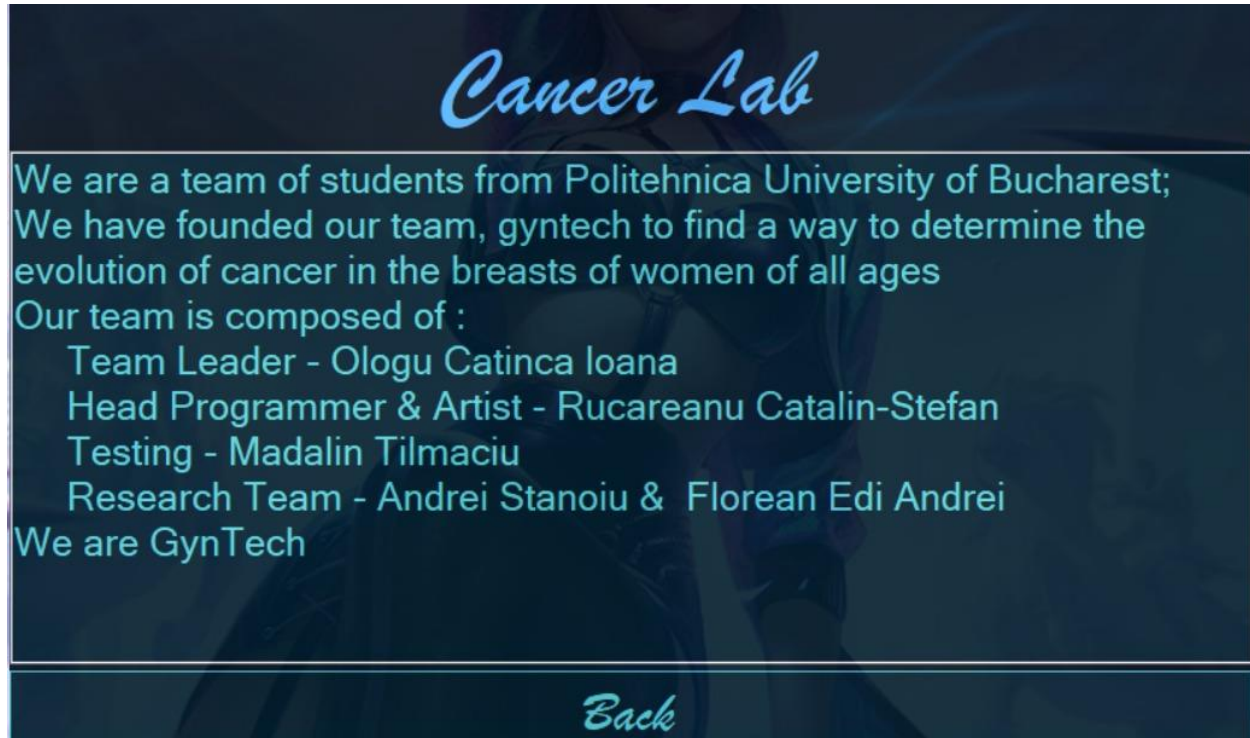
This is the loading screen, after which comes a short time of approximately 5 minutes of processing the backend, and of explaining shortly what our program does.

After the time has elapsed, we are presented with the next screen:



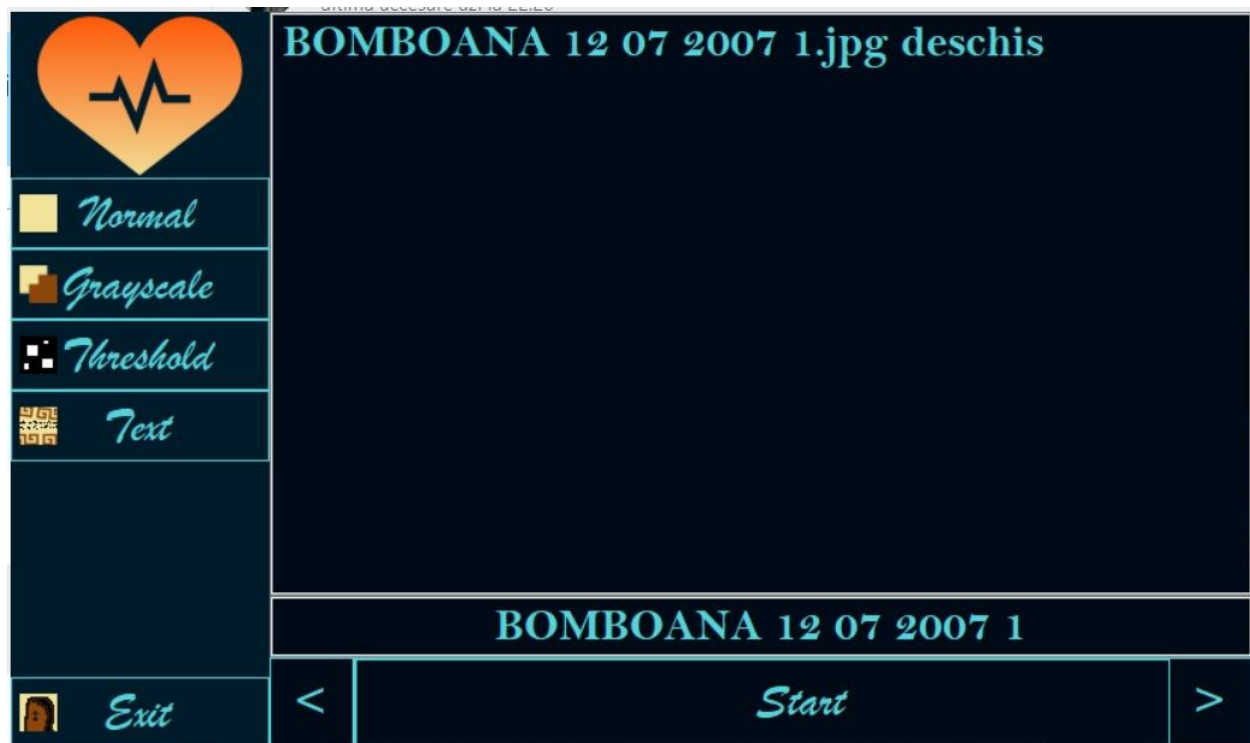
*Rucareanu Catalin-Stefan, Ologu Catinca Ioana, Andrei Stanoiu, Madalin Tilmaciu, Florean Edi Andrei*

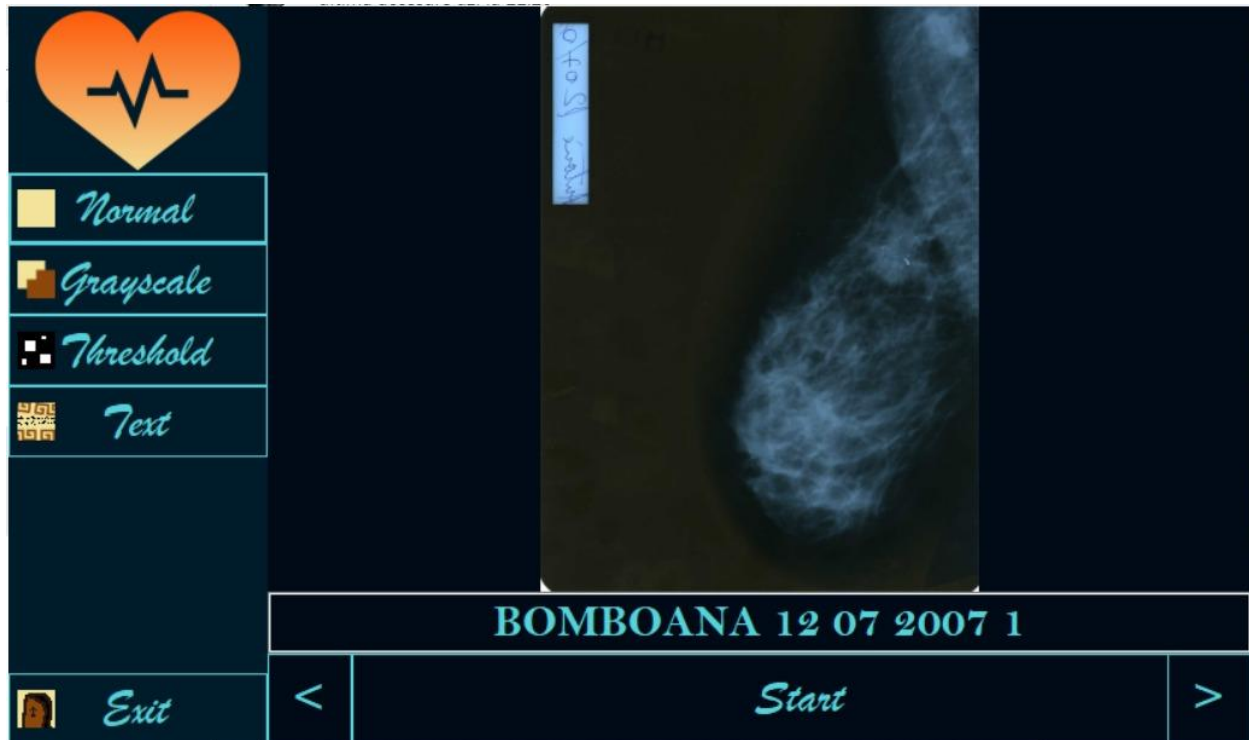
The about button shows information about us:



Here the back button gets us back to the previous image;

And the start button shows us the next screen.





The button Normal shows the original image

The button Grayscale shows the grayscale filter applied over the original image

The button Sobel shows the Sobel filter applied over the original image (it has been removed in a later version)

The button Graph shows the Graph as shown in the next image (a similar one, also removed):



The button Text shows the text as gotten from the main program

```
height=2784
width=2094
Aria in pixeli este: 289943
Aria pozei in pixeli este: 582970
Aria pozei in Cm este: 16.014
Aria in Cm este: 0.796465
Indicele de scalare este: 1/364037 (cm^2/px^2)

T1, b, N1

Stage IIA
Starea se va agrava

S-a procesat imaginea cu aria: 0.796465

Imaginea a fost eliberata
```

```
Aria in Cm este: 0.417977

STAN evolueaza in urmatorul fel:
0.713869 cm - 0.565547 cm - 0.449724 cm - 0.0376417 cm - 0.569247 cm - 0.757755 cm - T1, a, N0

Stage IIA
```



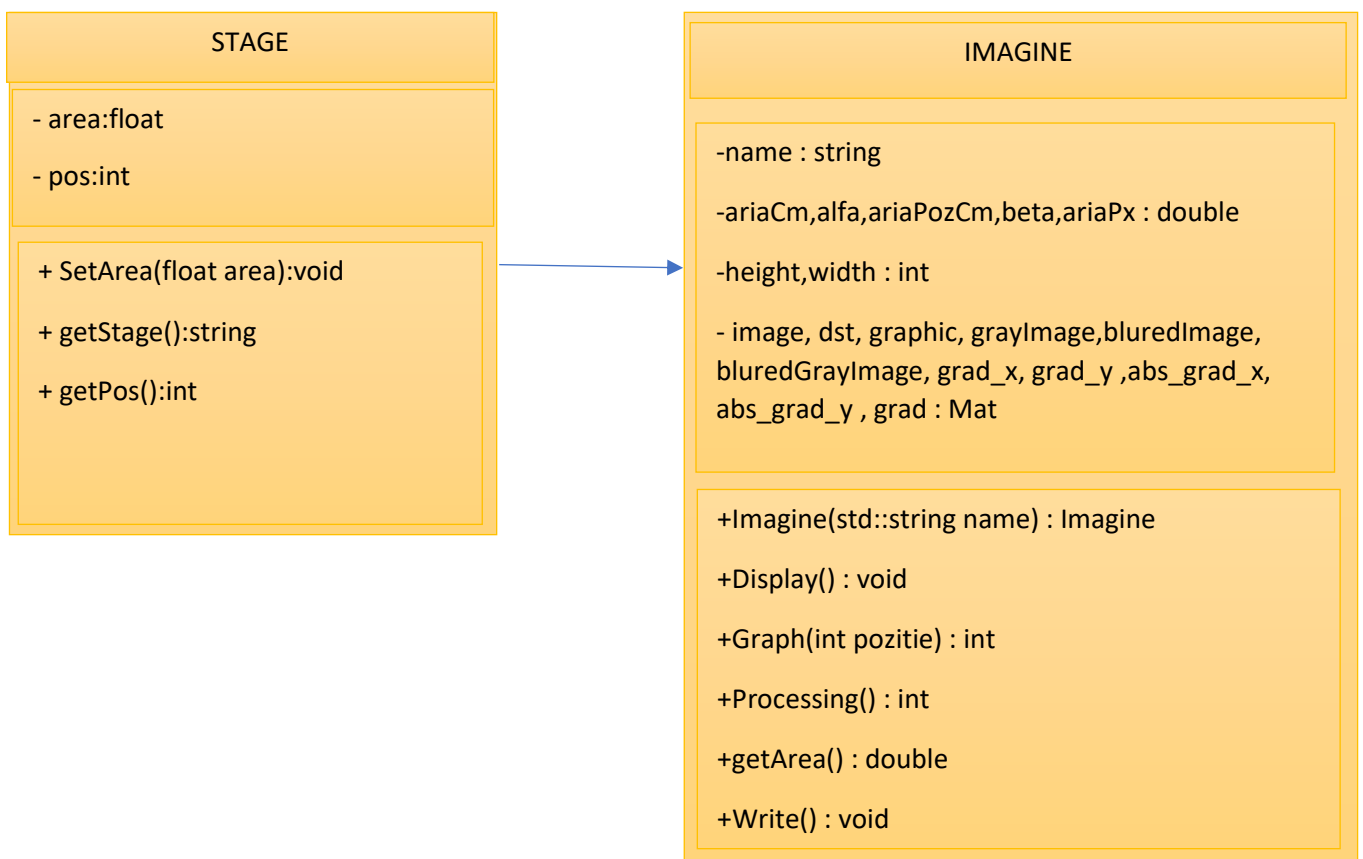
## *Faculty of Automatic Control and Computers*

The button Exit closes the interface

The buttons < and > cycles through the images

And the button Start pressed after either writing the name of a valid name of a picture, after one of the buttons < or > has been pressed or at the beginning, will show the text imageName.jpg, as taken from the text box, opened.

### *UML diagrams*



### *Art direction*

All the art pieces shown in the project, starting with the graph image and ending with the background song have been made by *Rucareanu Catalin-Stefan* in the software: FireAlpaca for the images and Bandlab for the song.

## *Bibliography*

1. <https://trello.com/b/14svVpYD/poo-project>
2. <https://www.cancer.gov/about-cancer/understanding/what-is-cancer#:~:text=Cancer%20is%20the%20name%20given,up%20of%20trillions%20of%20cells>.
3. <https://www.livescience.com/11041-10-deadliest-cancers-cure.html>
4. [https://en.wikipedia.org/wiki/TNM\\_staging\\_system](https://en.wikipedia.org/wiki/TNM_staging_system)
5. <https://epomedicine.com/medical-students/tnm-staging-breast-cancer-simplified/>
6. <https://opencv.org/about/>
7. [https://docs.opencv.org/master/d4/d13/tutorial\\_py\\_filtering.html](https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html)
8. [https://en.wikipedia.org/wiki/Gaussian\\_blur#:~:text=In%20image%20processing%2C%20a%20Gaussian,image%20noise%20and%20reduce%20detail](https://en.wikipedia.org/wiki/Gaussian_blur#:~:text=In%20image%20processing%2C%20a%20Gaussian,image%20noise%20and%20reduce%20detail)
9. [https://en.wikipedia.org/wiki/Sobel\\_operator](https://en.wikipedia.org/wiki/Sobel_operator)
10. [https://en.wikipedia.org/wiki/Thresholding\\_\(image\\_processing\)](https://en.wikipedia.org/wiki/Thresholding_(image_processing))
11. <https://www.bandlab.com/catuta>