

P R O J E C T H A N D B O O K

**Project: Wound Analysis Based On Image Processing**

Course: Information Engineering and Computer Science

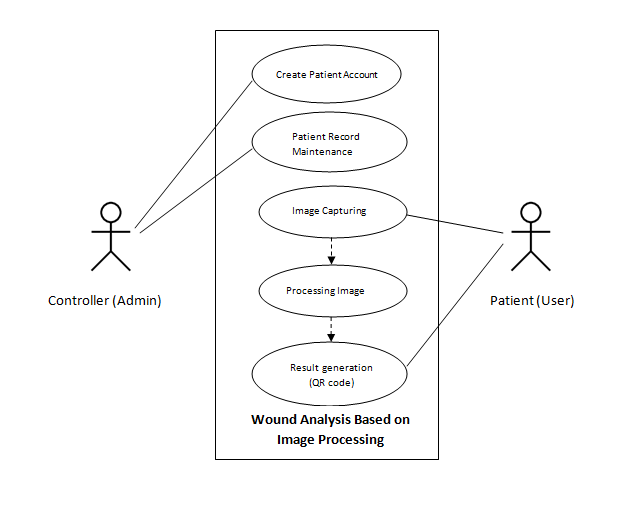
Semester: 2

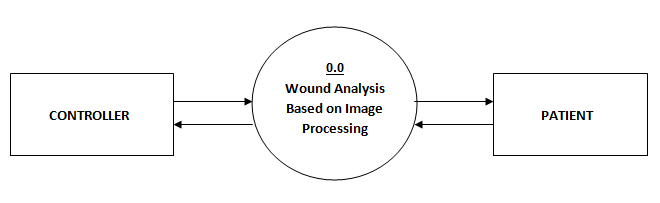
Teacher: Dr. Timo Kahl

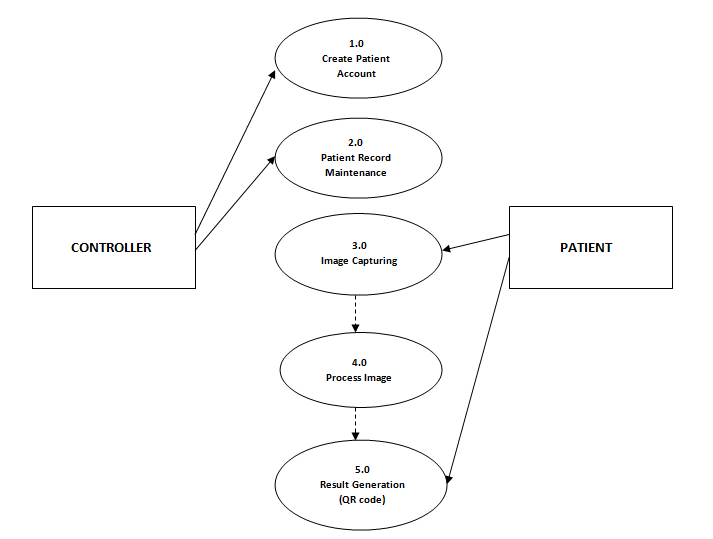
Project leader:

Project members:Qurratul Ain Abid (Matriculation no. 21037)

**3. Project Organization**

*Fig: Use Case Diagram*

*Fig: Level 0 DFD*



*Fig: Level 1 DFD*



1. **WSP Specification/ Description**

Describe the most important work packages of your WSP.

|  |  |  |
| --- | --- | --- |
|  |  | **WP Specification** |
| **Work Package:** 1.0 Research | |  |
| WP Content / Results: |  |  Research on image processing algorithm, hardware, types of |
|  |  | wound, design and development tools. |
|  |  |  |
| Responsible Person: |  | All group members |
|  |  |  |
| Progress since last status |  | Algorithm research is completed and working on implementation. |
| report: |  |  |
|  |  |  |
|  |  |  |
| Open issues: |  | Issues regarding integration and importing library files for the program. |
|  |  |  |

|  |  |
| --- | --- |
|  | **WP Specification** |
| **Work Package:** 2.0 User Interface Design and Development | |
| WP Content / Results: |  User Interface Design and Coding |
|  |  |
| Responsible Person: | Anith and Amir |
|  |  |
| Progress since last status | Main login page completed;Login for patient and operator in progress |
| report: |  |
|  |  |
| Open issues: | Nill |
|  |  |

|  |  |
| --- | --- |
|  | **WP Specification** |
| **Work Package:** 3.0 Image Processing and Hardware Integration | |
| WP Content / Results: |  Implementation of image processing algorithm and hardware. |
|  |  |
| Responsible Person: | Qurratul Ain and Arun |
|  |  |
| Progress since last status | In progress; Hardware request mail send. |
| report: |  |
|  |  |
| Open issues: | Integration issue with website. |
|  |  |

|  |  |  |
| --- | --- | --- |
|  | **WP Specification** |  |
| **Work Package:** 4.0 Databases | |  |
| WP Content / Results: |  Database designing and development |  |
|  |  |  |
| Responsible Person: | Jeff and Mayuri |  |
|  |  |  |
| Progress since last status | Table creation and finalization of entities in progress. |  |
| report: |  |  |
|  |  |  |
| Open issues: | Can be set after algorithm implementation phase. |  |
|  |  |  |
| IP\_Project-Handbook\_V001\_KN.doc | 05.11.2017 | !8 |

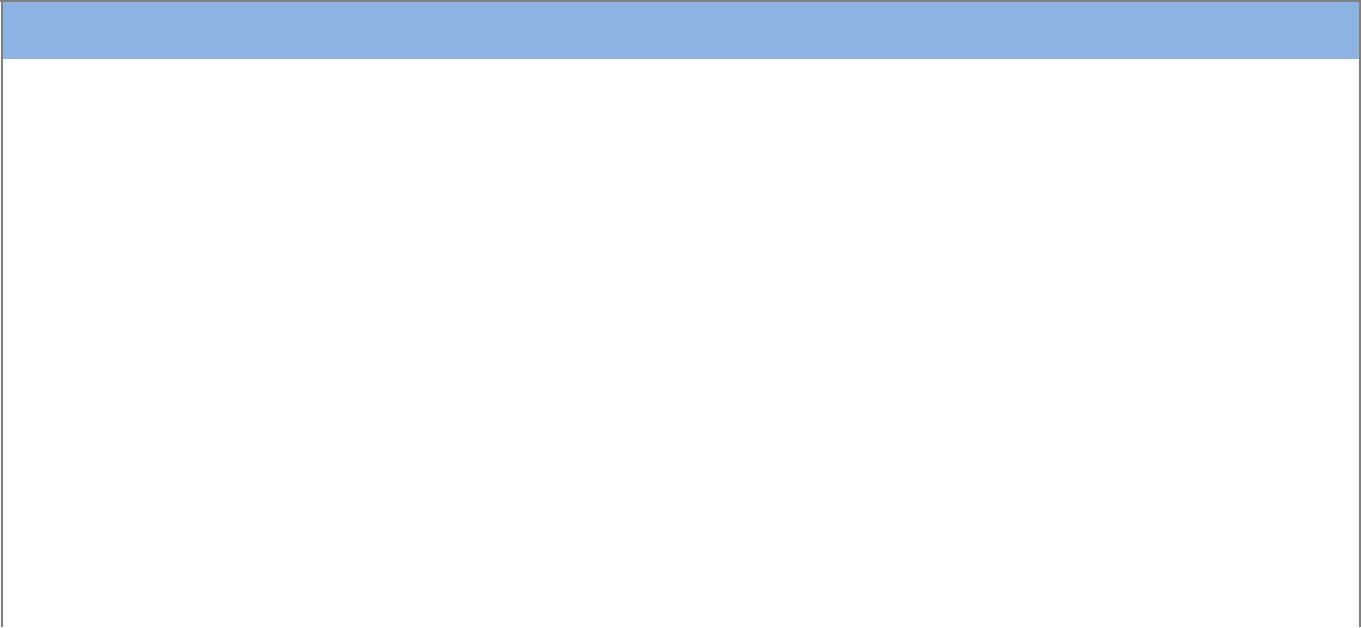


|  |  |
| --- | --- |
|  | **WP Specification** |
| **Work Package:** 5.0 Testing and Deployment | |
| WP Content / Results: |  Testing and Final Deployment |
| Responsible Person: | All group members |
|  |  |
| Progress since last status | Currently working on algorithm implementation phase. |
| report: |  |
|  |  |
| Open issues: | Nill |
|  |  |

|  |  |  |
| --- | --- | --- |
| IP\_Project-Handbook\_V001\_KN.doc | 05.11.2017 | !9 |



1. **Project Milestones**

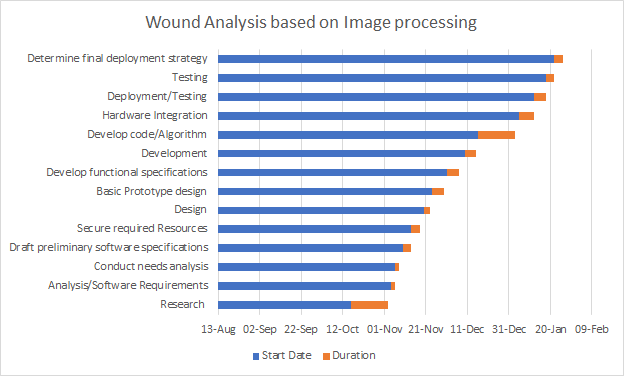


**Milestones**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **WSP-Code** | **Milestone-** |  | **PLANNED-DATE** |  | **IS-DATE** |  |
|  | **Name** |  |  |  |
|  |  |  |  |  |  |  |
| 1.0 | Initial Research | Research | 16 | October | 4 November | |  |
|  | |  |  | |  |  |  |
| 2.0 Analysis/Software | | Software Requirement | 4 November | | 14 | November |  |
| Requirements | | Specification |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| 3.0 | Design | UI Design and Code | 20 | November | 15 | December |  |
|  |  |  |  | |  |  |  |
| 4.0 | Development | Image processing and | 5 December | | 10 | January |  |
| (Phase 1) | | hardware integration |  |  |  |  |  |
|  |  |  |  | |  | |  |
| 4.0 | Development | Databases | 5 December | | 5 January | |  |
| (Phase 2) | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| 5.0 | Deployment | Deployment and | 12 | January | 24 | January |  |
|  |  | Testing |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

1. **Project Schedule Plan**

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Start Date** | **Days to Complete** | **Task Names as in WBS** |
| T\_1.0 | 16-Oct | 18 | Research |
| T\_2.0 | 04-Nov | 2 | Analysis/Software Requirements |
| T\_2.1 | 06-Nov | 2 | Conduct needs analysis |
| T\_2.2 | 10-Nov | 4 | Draft preliminary software specifications |
| T\_2.3 | 14-Nov | 4 | Secure required Resources |
| T\_3.0 | 20-Nov | 3 | Design |
| T\_3.1 | 24-Nov | 6 | Basic Prototype Design |
| T\_3.2 | 01-Dec | 6 | Develop functional specifications |
| T\_4.0 | 10-Dec | 5 | Development |
| T\_4.1 | 16-Dec | 18 | Develop code/Algorithm |
| T\_4.2 | 05-Jan | 7 | Hardware Integration |
| T\_5.0 | 12-Jan | 6 | Deployment/Testing |
| T\_5.1 | 18-Jan | 4 | Testing |
| T\_5.2 | 22-Jan | 4 | Determine final deployment strategy |
|  |  |  |  |
|  | Task schedule based on WBS |  |  |



*Fig: Gantt Chart*

1. **Project Costs and Risks**

|  |  |  |
| --- | --- | --- |
| *Work Package / Milestone* | *Type of Costs* | *Amount/Effort* |
| *Name* | *e.g Personal, material* | *e.g 5 hours* |
| **Barcode scanner** | **material** | **15 hours** |
| **camera** | **material** | **5 hours** |
| **Clouding system** | **material** | **15 hours** |

|  |  |  |  |
| --- | --- | --- | --- |
| *Risk* | *Event risk* | *Impact* | *measures* |
| *NAME AND DISCRIPTION* | *e.g in %* | *e.g quality cost date* | *Wbs code name* |
| **Integration of hardware and software** | **20%** | **Barcode scanner** |  |
| **Software specifications** | **20%** | **R coding / phython** |  |
| **Camera of choice** | **10%** | **Digital camera/ msart phone camera** |  |
| **Integration of database to cloud** | **30%** | **Azure/ docker cloud** |  |
| **Accurate image detector** | **20%** |  |  |

|  |  |  |
| --- | --- | --- |
| IP\_Project-Handbook\_V001\_KN.doc | 05.11.2017 | 10! |

**1. Research Topics Detail**

|  |  |  |
| --- | --- | --- |
| 1. | Anith | Research on different types of Image comparison Algorithms |
| 2. | Qurratul Ain | Research on different types of Image comparison Algorithms |
| 3. | Arun | Research on different types of Image comparison Algorithms |
| 4. | Jeff | Searching types of Wounds |
| 5. | Mayuri | Searching types of Wounds |
| 6. | Amir | Research on hardware device for Images |

**2. Task Distribution List**

1. Architecture design.
2. Layering the system.
3. Front end design.
4. Back end design.
5. Algorithm integration.
6. Code testing in R, Python, C#, MAT lab.
7. Wound Image collection for training the system.
8. Hardware integration.
9. Entity search and finalisation.
10. Database SQL server set up and implementation(primary publishing)
11. IBM cloud set up and implementation (secondary publishing)
12. Testing.

|  |  |  |
| --- | --- | --- |
| # | Tasks | Assigned to |
| 1 | Architecture design. | Anith |
| 2 | Layering the system. | Anith |
| 3 | Front end design. | Qurratul Ain |
| 4 | Back end design. | Amir |
| 5 | Algorithm integration methods. | Arun, Qurratul Ain |
| 6 | Image comparison Code testing in C#. | Anith |
| 7 | Image comparison Code testing in Python | Arun,Jeff |
| 8 | Image comparison Code testing in Java Script. | Amir |
| 9 | Image comparison Code testing in MAT lab | Mayuri, Qurratul Ain |
| 10 | Wound Image collection for training the system. | Jeff,Mayuri |
| 11 | Entity search and finalisation. | Qurratul Ain,Arun |
| 12 | Hardware integration. | Amir,Arun |
| 13 | Database sql server set up and implementation(primary publishing) | Anith,Amir |
| 14 | IBM cloud set up and implementation (secondary publishing) | Jeff,Arun |
| 15 | Testing . | Mayuri, Jeff |

**3. Flow chart (Function flow)**

1. **Operator**

Start

End

Send email with rejection reason

Send email with account details

Verify Patient details

View Patient Details

Delete Patient Record

Requested Patient List

Enrolled Patient List

Website Tab 2

Website Tab 1

Operator Login

Research for implementation in c#

* Algorithm shortlisted and used:

### K Nearest Neighbours – Classification

### K nearest neighbours: stores all available cases and classifies new cases based on a similarity measure.

### SURF Algorithm

### Three main parts: interest point detection, local description and matching.

### In the sample program:

### Clustering algorithm and Speeded up Robust Features (SURF) is used to perform image classification.

### A set of images are initially loaded and put it into different containers.

* Use SURF, so in order to use a standard clustering algorithm that is based on Euclidean distances.
* A good algorithm for clustering code word is the Binary Split variant of the K-Means algorithm.
* Training based on the set of images available or pre loaded into the system
* Use it to extract representations for each of the images in both training and testing sets.
* Creates the Support Vector Machines that will identify images based on their Bag-of-Visual-Words feature vector representation.
* Automatically estimates a good starting point for Gaussian's sigma parameter using initialization heuristics.
* Method automatically estimates a good starting point for the complexity parameter (C) of the SVM learning algorithm.
* Classifies images into one of the possible classes using the Support Vector Machines learned in the previous steps.

All the methods used have inbuilt files or algorithm used in Visual studio in the form of classes. But still has some error facing for the integration into our system.

**Research implementation in MATLAB**

* **Algorithm shortlisted and used:**

First part of image comparison includes detecting edges of the image for this canny edge detection algorithm is used.

### Canny Edge Detection Algorithm

The Canny algorithm is flexible to different environments. Its parameters allow it to be tailored to detection of edges of differing characteristics depending on the specific requirements.

### 

The Process of Canny edge detection algorithm can be broken down to 4 different steps:

**1.) Smooth image with a Gaussian**

Since edge detection results are easily affected image noise, so Gaussian filter is applied to smooth the image in order to remove noise to prevent false detection.

**2).Find the intensity gradients of the image**

An edge in an image may points in different directions, so the Canny algorithm uses filters to detect horizontal, vertical and diagonal edges.

**3). Applying non-maximum suppression**

Thin edges by applying non-maxima suppression to the gradient magnitude to get rid of spurious response to edge detection

**4). Hysteresisthreshold**

Next is to filter out edge pixel with weak and strong gradient value. If an edge pixel’s gradient value is higher than the high threshold value, it is marked as a strong edge pixel. If an edge pixel’s gradient value is smaller than the high threshold value and larger than the low threshold value, it is marked as a weak edge pixel.Weak edges are discarded in the end.

**Wound edge detection using JavaScript**

Using this JavaScript algorithm(Figure 1) we can detect edge of any wound.This code is compatible with Visual studio.

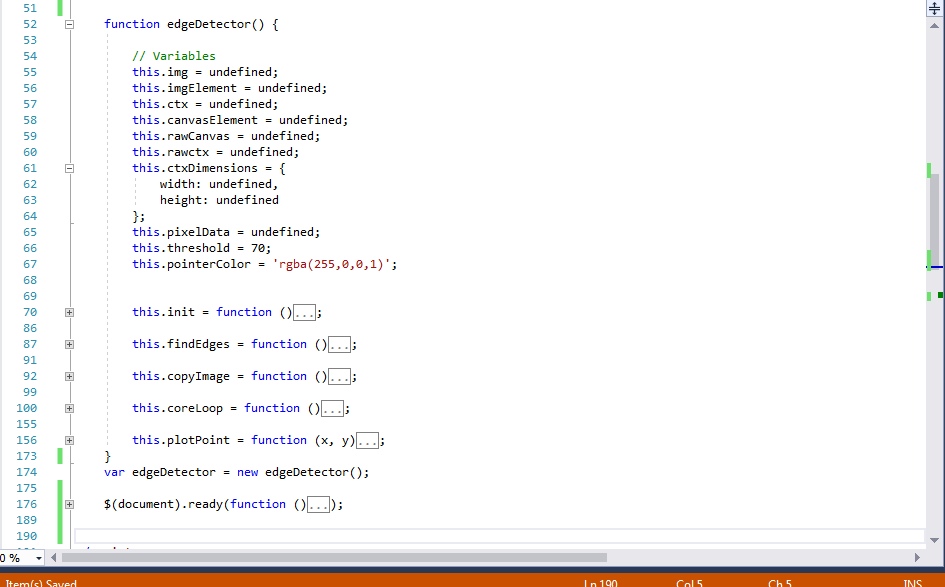


Figure 1: JavaScript Algorithm for wound detection

Detection edge is one of the important parts of our research project. Therefore we can use this algorithm for our web application where user will upload their wound image. After uploading their image we can show the user the exact edge of the wound (Figure 2).

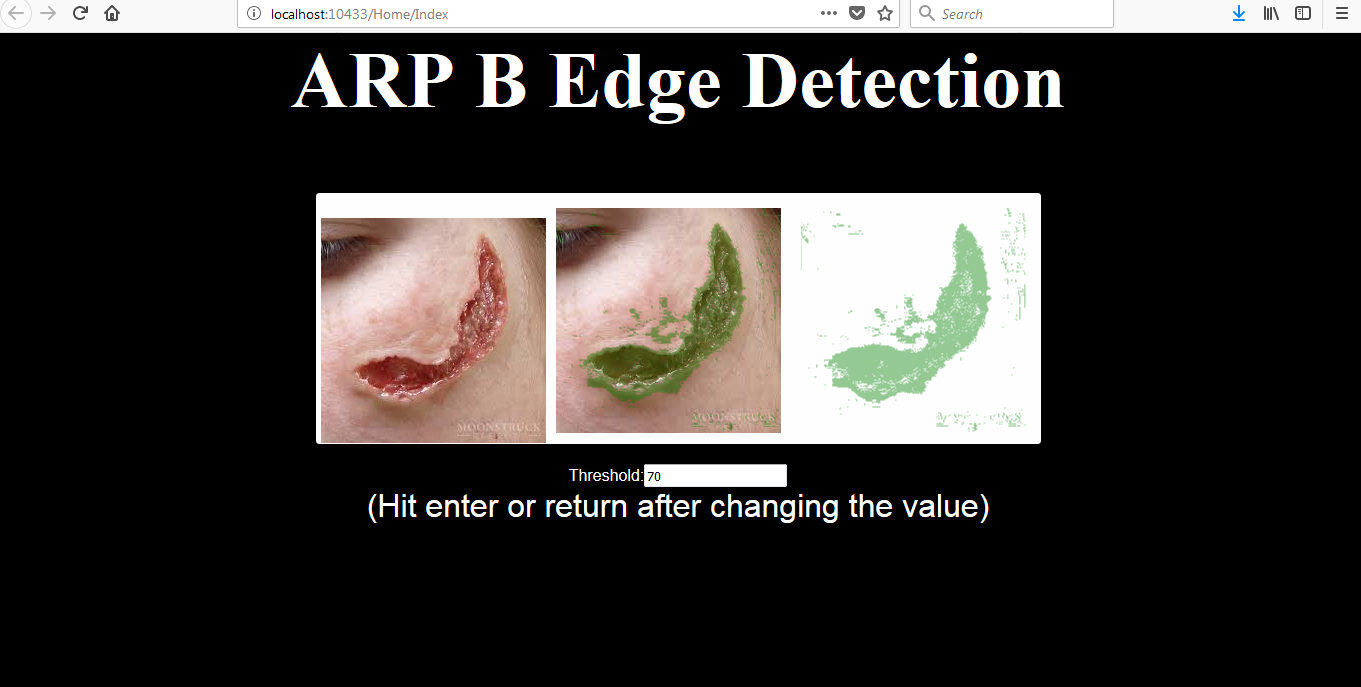


Figure2: Detect wound edge of the user uploaded image(first one is user uploaded image)

However user can change threshold to make the detection more perfect. There is an input field (Figure 3) where user can change threshold value and press enter to view the change. When the user is done with the threshold value, he can click on a submit button and the system will store uploaded the image with only edge.

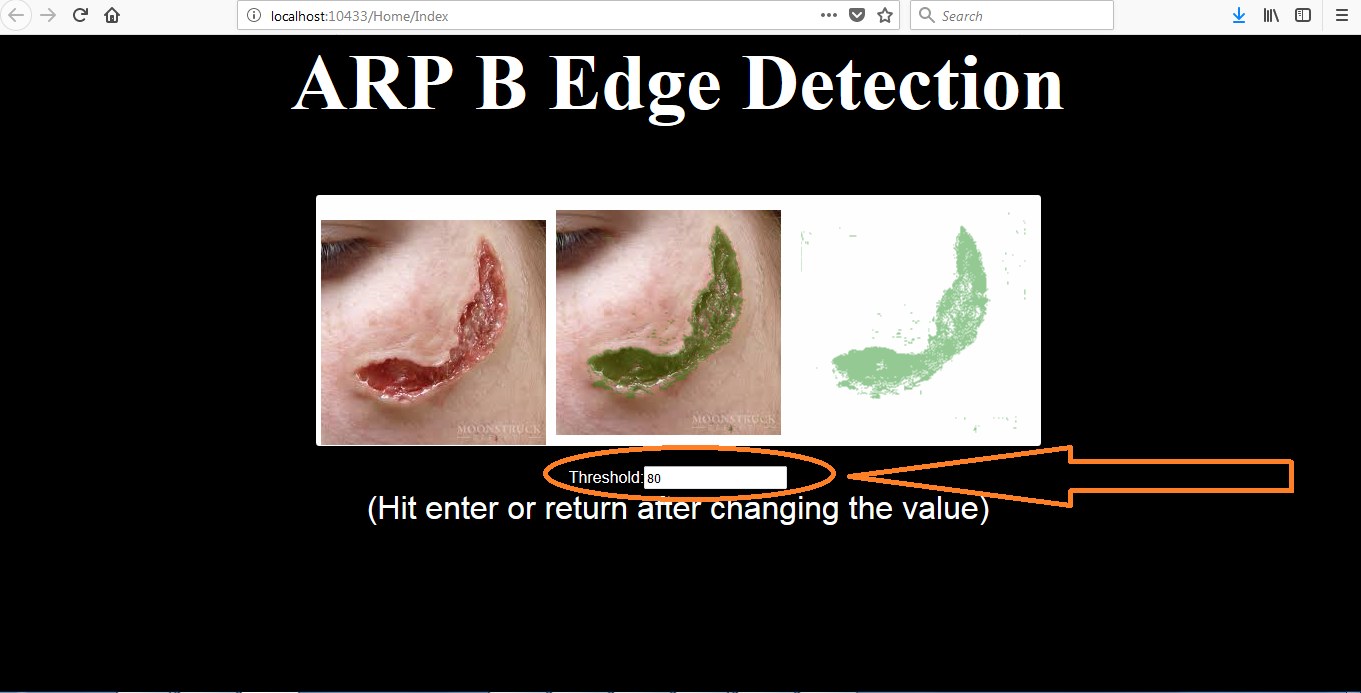


Figure 3: User can change threshold to make the wound edge detection perfect