**ADVANCE DIGITAL RADIOVISOGRAPHY IN DENTISTRY**

**A PROJECT REPORT**

***Submitted by***

**POOJA NANDAGOPAL [REGISTER NO: 211417104186]**

***in partial fulfillment for the award of the degree***

***of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE AND ENGINEERING**

**PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.**

**ANNA UNIVERSITY: CHENNAI 600 025**

**APRIL 2021**

**BONAFIDE CERTIFICATE**

Certified that this project report “**ADVANCE DIGITAL RADIOVISOGRAPHY IN DENTISTRY**” is the bonafide work of “POOJA NANDAGOPAL Reg. No: 211417104186” who carried out the project work under my supervision.

|  |  |
| --- | --- |
| **SIGNATURE** | **SIGNATURE** |
| **Dr.S.MURUGAVALLI,M.E.,Ph.D.,**  **HEAD OF THE DEPARTMENT** | **Dr.D.ANURADHA M.E. PGDBA, Ph.D**  **SUPERVISOR** |
| DEPARTMENT OF CSE,  PANIMALAR ENGINEERING COLLEGE, NAZARATHPETTAI,  POONAMALLEE,  CHENNAI-600 123. | DEPARTMENT OF CSE,  PANIMALAR ENGINEERING COLLEGE, NAZARATHPETTAI,  POONAMALLEE,  CHENNAI-600 123. |

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on...........................

|  |  |
| --- | --- |
| **INTERNAL EXAMINER** | **EXTERNAL EXAMINER** |

**ACKNOWLEDGEMENT**

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like express our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI, Dr.C.SAKTHIKUMAR, M.E., Ph.D.,** and **Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our **Principal Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my Project Guide ……………………… and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

**POOJA NANDAGOPAL**

**ABSTRACT**

Radiovisiography (RVG) as the latest imaging technique in dentistry with the minimal radiation exposure of the patient and numerous possibilities to process the images has many advantages over classic radiography. The RadioVisioGraphy (RVG) imaging system commonly used in dentistry to take intraoral radiographs features the latest innovations in digital radiography, delivering the highest image resolution. The results are faster and better with no loss in image quality. The software is designed in such a way to make use of this images captured by the Radiovisography in a reliable, affordable and lesser time frame by applying the need and the thoughts of dentist and the patients.

TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT** | **4** |
|  | **LIST OF TABLES** | **7** |
|  | **LIST OF FIGURES** | **7** |
|  | **LIST OF SYSMBOLS, ABBREVIATIONS,** | **7** |
| **1.** | **INTRODUCTION** |  |
|  | * 1. **Overview**   2. **Problem Definition** | **8**  **10** |
| **2.** | **LITERATURE SURVEY** | **11** |
| **3.** | **SYSTEM ANALYSIS** |  |
|  | **3.1 Existing Systems**  **3.2 Proposed Systems**  **3.3. Requirement Analysis and Specifications**  **3.3.1 Input Requirements**  **3.3.2 Output Requirements**  **3.3.3 Functional Requirements**  **3.4 Hardware Requirements**  **3.5 Software Requirements** | **12**  **12**  **13**  **13**  **14**  **14**  **14** |
| **4** | **SYSTEM DESIGN** |  |
|  | **4.1 ER Diagram**  **4.2 Data Directory**  **4.3 Table Normalization**  **4.4 Data Flow Diagram**  **4.5 UML Diagram** | **15**  **17**  **18**  **19**  **20** |
| **5** | **SYSTEM ARCHITECTURE** |  |
|  | **5.1 Architecture Overview**  **5.2 Module Design Specification**  **5.3 Program Design Language** | **21**  **22**  **24** |
| **6** | **SYSTEM IMPLEMENTATION** |  |
|  | **6.1 Client Side Coding**  **6.2 Server- Side Coding** | **25**  **34** |
| **7** | **SYSTEM TESTING** |  |
|  | **7.1 Unit Testing**  **7.2 Integration Testing**  **7.3 Test Case & Reports / Performance Analysis** | **48**  **48**  **49** |
| **8** | **CONCLUSION** |  |
|  | **8.1 Conclusion and Future Enhancement** | **51** |
|  | **APPENDICES** |  |
|  | **A.1 Sample Screens**  **A.2 Publications** | **52** |
|  | **REFERENCES** | **55** |

**List of Tables**

|  |  |  |
| --- | --- | --- |
| S.No | Descriptions | Page No |
| 1 | Table 1 : Proposed System | 12 |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| S.No | Descriptions | Page No |
| 1 | ER Diagram | 14 |
| 2 | Data Directory | 15 |
| 3 | Table Normalization | 16 |
| 4 | Data Flow Diagram | 17 |
| 5 | UML Diagram | 18 |
| 6 | Architecture Overview | 19 |
| 7 | Module Design Specification | 20 |
| 8 | Program Design Language | 21 |
| 9 | Test Case 1 | 46 |
| 10 | Test Case 2 | 47 |
| 11 | A1 – Sample Screens | 48 |

**LIST OF SYSMBOLS, ABBREVIATIONS,**

|  |  |  |
| --- | --- | --- |
| S.No | Abbreviations | Descriptions |
| 1 | CCD | Charge Coupled Devices |
| 2 | CMOS | Common Metal Oxide Sensors |
| 3 | PSP | Photo Simulator Phosphor Plate |
| 4 | CT-3D | Computed Tomography Three Dimensions |
| 5 | RVG | Radiovisiography |
| 6 | CR | Computed Radiography |
| 7 | DR | Digital Radiography |
| 8 | RIS | Radiology Information System |
| 9 | PACS | Picture Archiving and Communication System |
| 10 | SDK | Software/ Source Development Kit |

1. **INTRODUCTION**
   1. **Overview**

Radiography X Ray Stand for Radiation and found in the Year 1895. Radiography is a imaging technique used to view the internal form of an object. Radiography is used as medical radiology (Diagnostics) and industrial radiology. Digital Radiography is the latest advancement over the Classical Radiography (Film Based X Rays) from the year 2006 and there comes the end of Film Based Radiology

**Comparison of B/W Film Based and Digital Radiography.**

***Film Based Imaging***

Film Based imaging consist of X Ray interaction with electrons in the film emulsion. Through the Clinical Processing we transform the latent image into a visible one. Film is a relatively inefficient radiation detector and, thus, requires relatively high radiation exposure.

***Digital Imaging***

Digital Imaging is the result of X Ray interaction with electrons in electronic sensor pixels. Conversion of analog data to digital data, computer processing, and display of the visible image on a computer screen.

**Types of Digital Radiography**

***Direct Digital Radiography***

* + Sensors is Placed in Patients Mouth
  + Exposed to Radiation
  + Sensor Captures Radiographic Image
  + Transit Image to Computer Monitor using software’s
  + Image appears on the screen within a minute

***Indirect Digital Radiography***

* + Existing X Ray Film Digitized Using CCD Camera
  + Scan to Images
  + Digitized display to the Monitor

**Types of Digital Image receptor**

* *CCD (Charge Coupled Devices)*
* *CMOS (Common Metal Oxide Sensors – Alternative for CCD used for Active Pixel)*
* *PSP (Photo simulated Phosphor Plate)*
* *Computed Tomography ( CT) – 3D Images – Advancement of the above all receptor.*

**Digital Image Processing and Image Storage**

Image processing involves Image restoration, Image Enhancement, Image Analysis, Image compression. The File Size of the images varies considerably ranging from 200 KB to 6 MB. So the Software is designed and developed to reduce the Image Storage.

**Advantages of Digital Imaging**

* + Enhancement of Images
  + 3D Reconstruction
  + Filtration
  + Storage
  + Time
  + Environmental Friendly
  + Tele-Radiology
  + Dark Room is no Longer Needed
  + Dose Reduction & Patient Education

**Disadvantage of Dental Imaging**

* + Cost
  + Medico legal
  + Cross infection Control
  + Sensor Dimensions.
  1. **Problem Definitions**

The increasing demand for radiology services and the limited resources of radiologists have led to a long waiting time for radiology results. Increased radiology turnaround time will cause radiology services become delay, which can affect patient’s complaint. To overcome the problem of delay problem, availability, and affordability for diagnostic radiology services, there needs to be a rapid change in dentistry for the delivery in radiology services.

1. **LITERATURE SURVEY**

**The Advance Indications and Use of this Radiovisography (RVG) in the field of Dentistry is as follows**

* + **Detection of Tumors**
  + **Detection of hard Tissues**
  + **For Implants analysis**
  + **Detection of Carries and Periodontal Diseases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.NO** | **YEAR** | **AUTHOR NAME** | **PROJECT TITLE** | **MERITS** | **DEMERITS** |
| 1. | Published in journal (2020) | Dhia Ayu Salsabila1 | The Hospital Radiology Service Redesign, By Using Business Process Re-engineering and Information Systems Approach | Improved Radiology Services and Time Reduction | Healthcare/ Hospitals / Medical Practicioners has to understand and improve their standards |

1. **SYSTEM ANALYSIS**
   1. **Existing System**

Before modeling the current radiology service process, the entire process of Dental diagnostic radiology services is examined. The entire process information obtained from direct observations and verified by the Dentistry through interviews. The waiting time at radiology service is categorized into two parts, namely the system turnaround time and radiology turnaround time. The waiting time from the radiology “examination request” activity is made by the referral doctor until finally given to the radiology department. The administrative processes outside of radiology is defined as the system turnaround time, while the time from conducting the radiology examination until the availability of “expertise to be given to the patient” or the referral doctor is defined as the radiology turnaround time. The difference between these two categories is that the system turnaround time does not depend on radiology unit performance, while radiology turnaround time is high dependent on the availability and performance of radiologists and radiographers. As-is process and to-be process simulations are performed using Lynxvision software. A set of random numbers will be selected in each replication and used in estimating the probability distribution function in the simulation of the radiology service process. The entire process time included in the radiology service process simulation will be a time span using a uniform distribution so that every possible outcome has the same probability of occurring. Validation of the as-is model is done by using faces and event validity to ensure the as-is process model is made according to the current radiology service process.

* 1. **Proposed System**

The objectives of business process reengineering and the information system design of the to-be process are determined after gathering the weaknesses of the radiology service process. To-be process is designed to provide fast service with good quality for hospital radiology services. The to-be system used by the parties involved in radiology services, such as referral doctors, radiographers, and radiologists with information systems designed to be limited to the radiology unit. There are several ways that can be done in making improvements for business processes in a company, such as benchmarking, ESIA (Eliminate, Simplify, Integrate, Automate), and BPR Best Practice. The improvement of radiology business process system are made by considering the results of simulations and discussions conducted with relevant stakeholders which can be seen in Table 1.

Table 1: Proposed Solution

|  |  |  |
| --- | --- | --- |
| No | Improvement | Description |
| 1 | Design a neat storage area | Dividing the storage place of images and the results that have not finished yet in alphabetical order, month, and year. |
| 2 | Enhanced the features of RIS | Optimizing the use of Radiology Information System (RIS) to increase the effectiveness of radiographers and radiologists |
| 3 | Changing imaging modalities from CR to DR | Simplify radiographers’ work by replacing Computed Radiography (CR) into Digital Radiography (DR) to provide quality improvement and speed up the radiological examination process |
| 4 | Using PACS | Picture Archiving and Communication System (PACS) is a technology that integrated with various imaging modalities to provides storage and easy access |

* 1. **Requirement Analysis and Specification**
     1. **Input Requirements**

A specification of the essential subject matter, business objects, entities, and classes that are germane to the product. It might take the form of a first-cut class model, an object model, or a domain model.

* + 1. **Output Requirements**

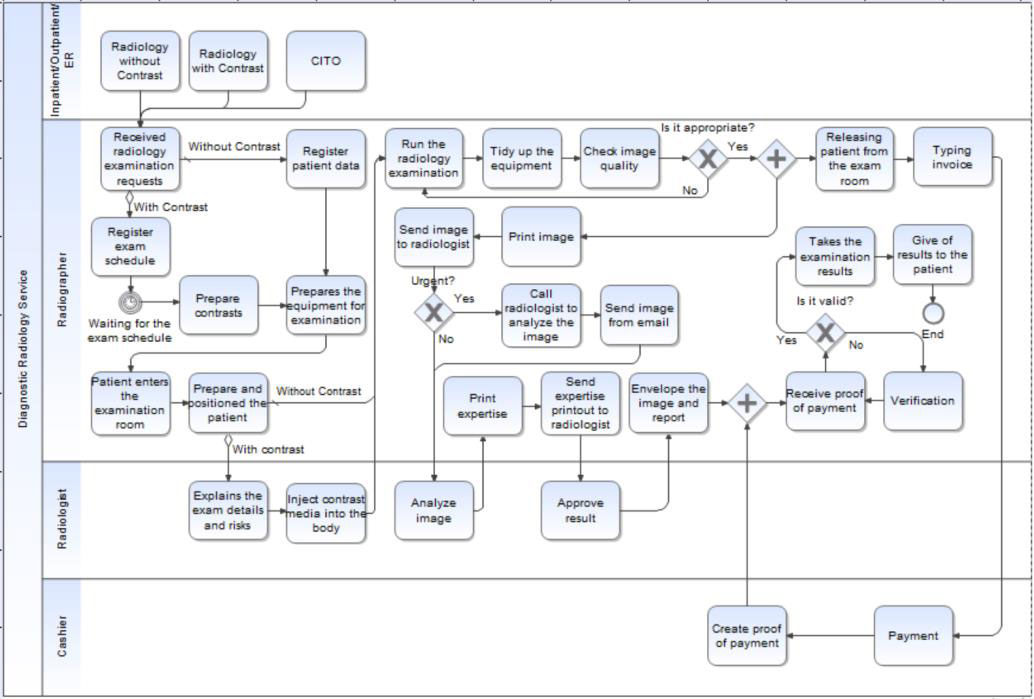
Specifies the amount of time available to complete specified tasks. These requirements often refer to response times. They can also refer to the product’s ability to operate at a speed suitable for the intended environment.

* + 1. **Functional Requirements**

A specification for each functional requirement. As with all types of requirements, use the requirements shell. Each functional requirement should have a fit criterion or a test case. In any event, the fit criterion is the benchmark to allow the tester to determine whether the implemented product has met the requirement.

* 1. **Hardware Requirement**
* Any Desktop / Laptop with 2GB RAM,500GB HDD, 2.0/3.0 USB Driver, 17” Monitor
* S11684 – 12/- 62 - Digital Image receptor
* An Dental X Ray Machine
* Printer
  1. **Software Requirement**
* VB.Net Technology
* SQL Server
* S11684S11685APL - Source From the RVG Manufacturer
* SDK - CMOS\_USB.dll, DynamicDataDisplay.dll
* Functional Library Files for S11684S11685

1. **SYSTEM DESIGN**
   1. **ER Diagram**

****

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships.

ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.

Electromagnetic compatibility is the ability of electronic device elements to correctly interact in an electronic environment. Although the Lynxvision software system was designed according to this compatibility and complies with the electromagnetic interference thresholds established by the regulatory agency, there is no guarantee about interference likely to occur on a particular installation. If the device generates interference with radio communication services (which can be determined by switching it off and on), it is recommended to try to correct this phenomenon by taking whole or part of the following measures:

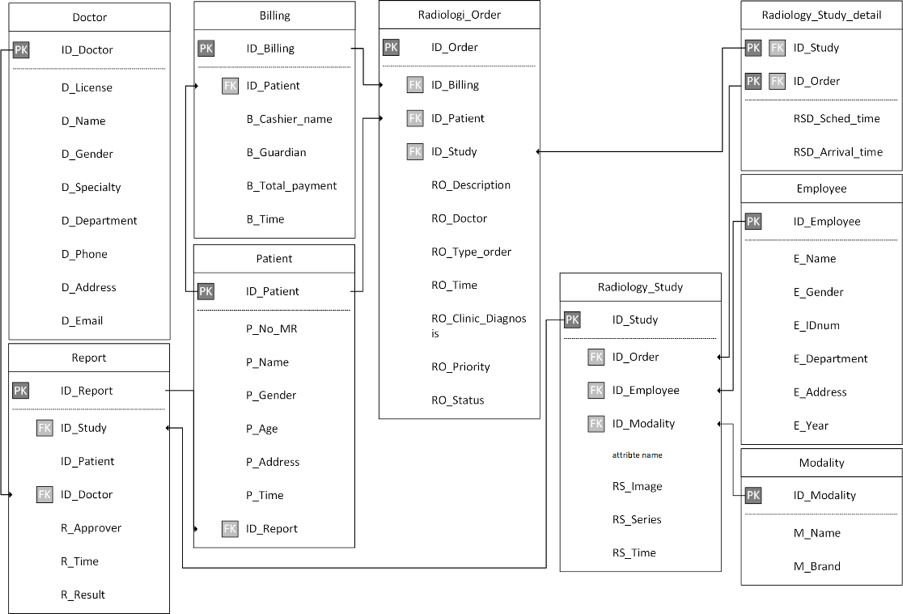
• Change the receiving antenna orientation

• Reposition the product according to the receiver.

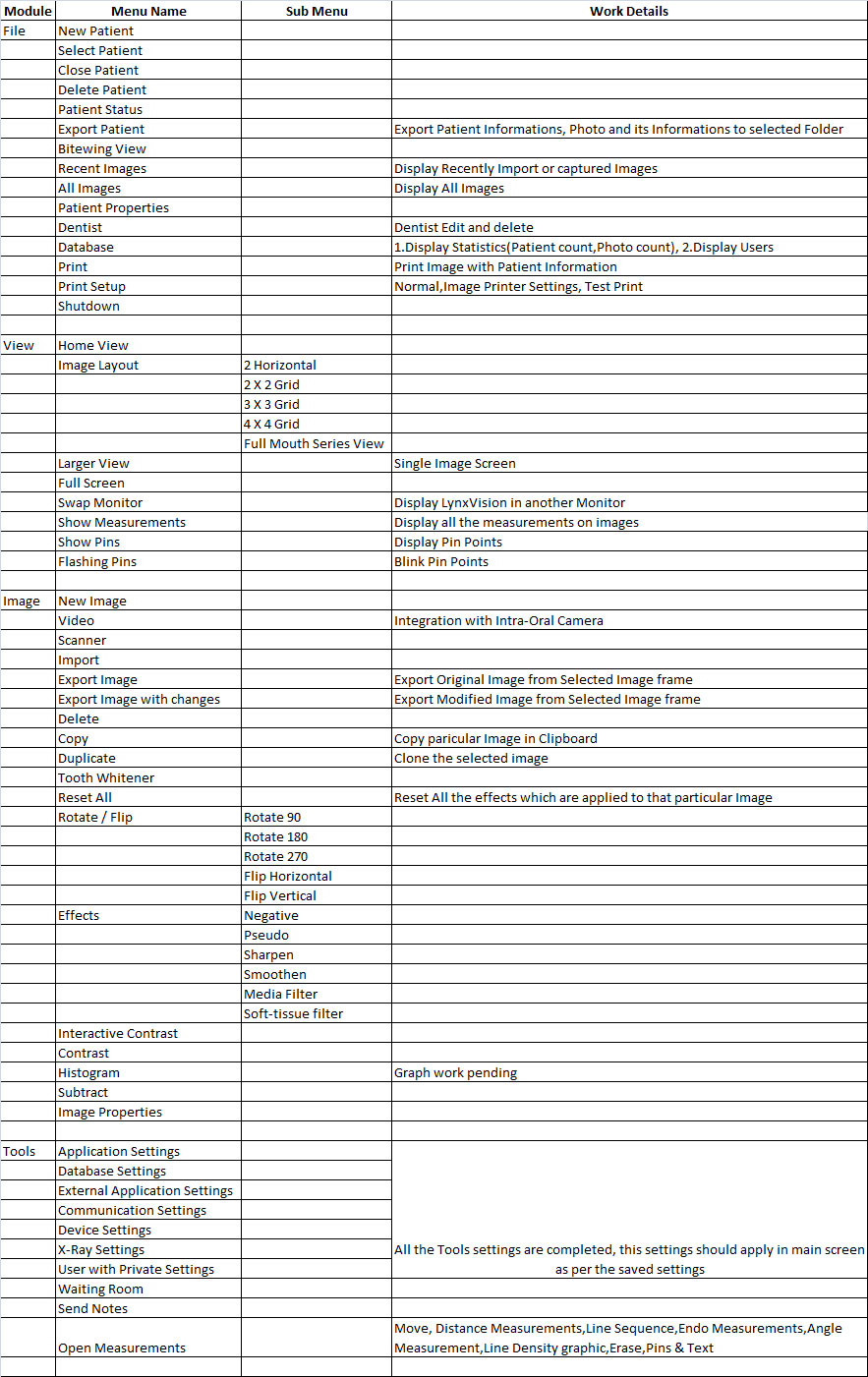
• Take the computer away from the receiver.

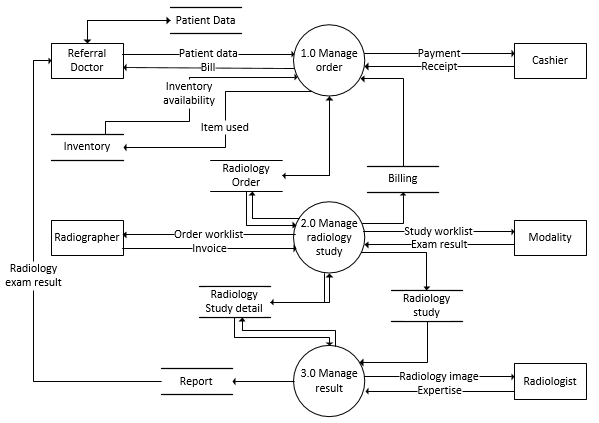
A systemic approach is required for a coherent and well-running system. Bottom-Up or Top-Down approach is required to take into account all related variables of the system. A designer uses the modelling languages to express the information and knowledge in a structure of system that is defined by a consistent set of rules and definitions. The designs can be defined in graphical or textual modelling languages.

* 1. **Data Directory**

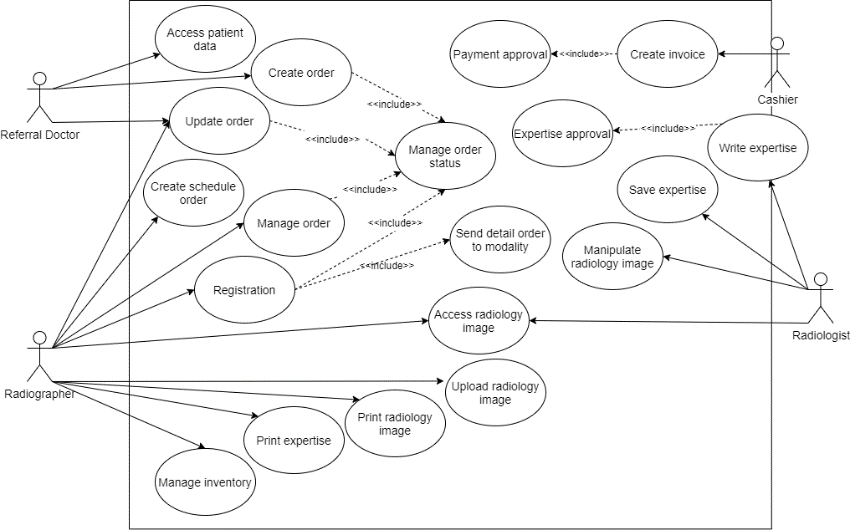
****

Data Directory - An inventory that specifies the source, location, ownership, usage, and destination of all of the data elements that are stored in a database.

* 1. **Table Normalization**
  2. **Data Flow Diagram**

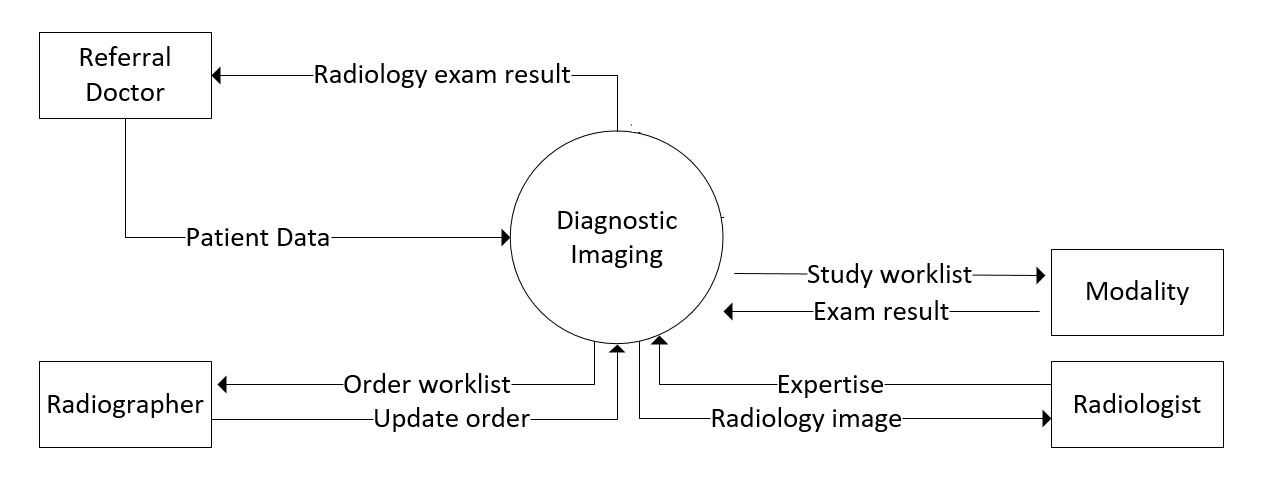
A Data Flow Diagram (DFD) is a graphical representation of the “flow” of data through an information system, modeling its aspects. It is a preliminary step used to create an overview of the system which can later be elaborated DFDs can also be used for visualization of data processing.

* 1. **UML Diagram**

 Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development.

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases

1. **SYSTEM ARCHITECTURE**
   1. **Architecture Overview**

****

* 1. **Module Design Specification**

Click the Lynxvision software icon to launch the software and the active screens are shown below.

* System menu of the program windows
* Menu bar
* Tool bar
* Patient Account List
* Teeth Chart
* Target Frame
* Image Information

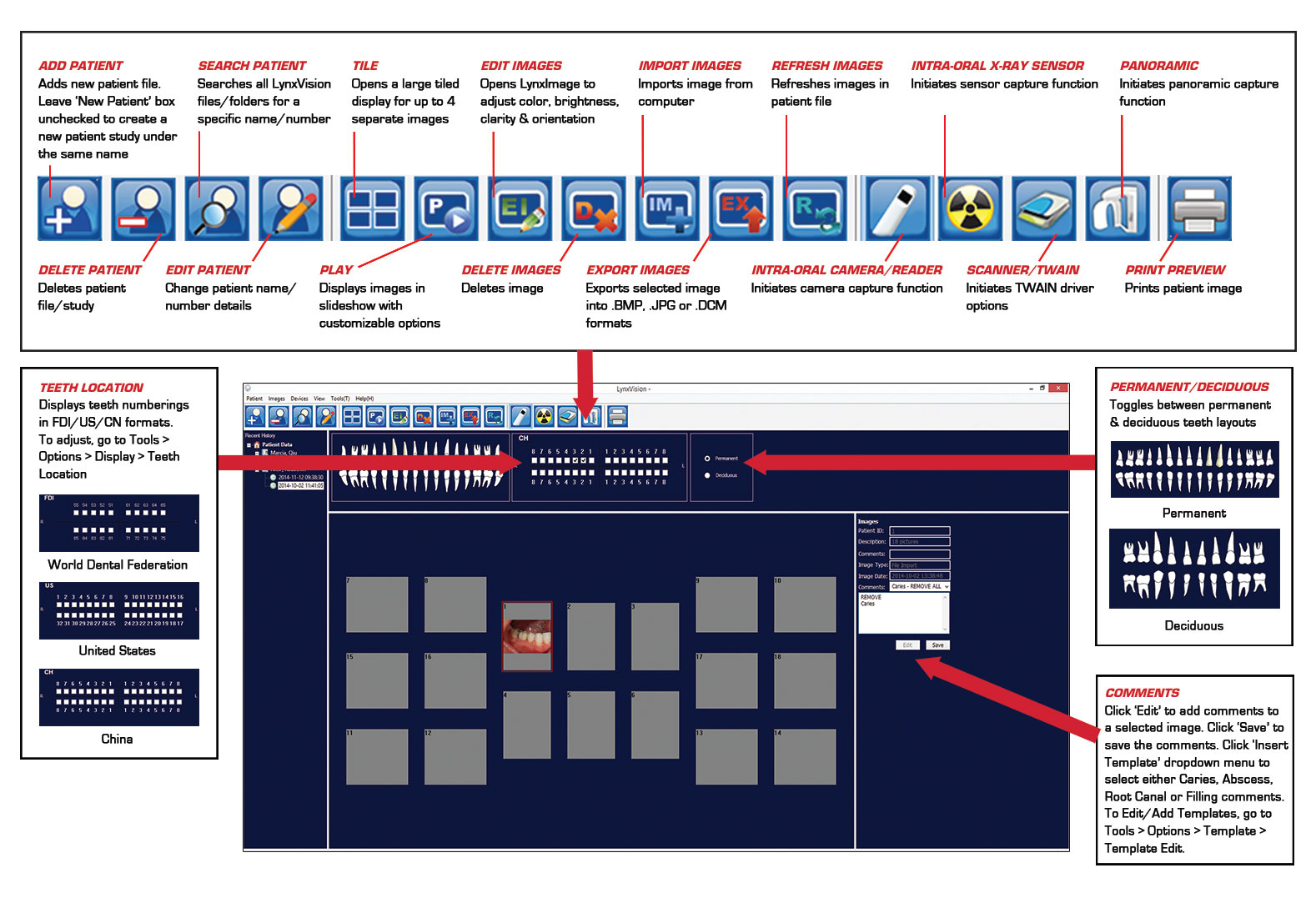
Each item contains a drop-down menu; some functions can be accessed directly from the icon in the toolbar.

Click on Tools (T) on the menu bar and select Options.In the recent history section there are two options, patient details are arranged by first and last name, depending on the country's custom. You can select the Application Language. Teeth Locations as per user convinence and also can select the fonts and font size for easy user interation. This will be updates once the software is restarted.

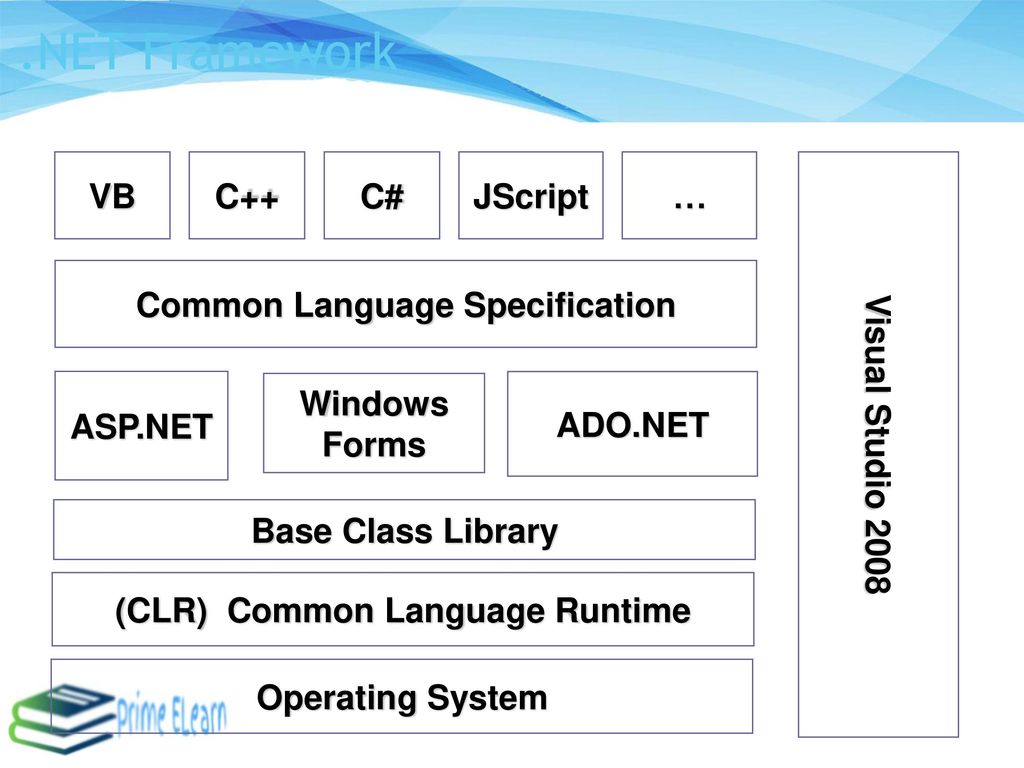
Each button contains and explanation of its function. By moving the mouse over the icon, it will display a short note of its function

Input patient information by adding the new patients from the icons, select picture series and you can also input study comments in the frame and then click ok. A new patient account with the Case study will be created. You can search, update, print and delete the patient information based on the user needs. The software will display a minimum set of records in the front screen and all will be hidden in the screen. To extract more details we can use the search icon.

Now the hardware device RVG is connected with the working software. Before taking and processing the image make sure the Hardware Device RVG is well connected with the software. After taking the image then the software is used with Enhanced Features of Flip, Mirror, Rotate Clockwise and Anti Clock wise, Measurements, Annotations on the image, Negative Images, Zoom particular area and so on.

****

* 1. **Program Design Language**

****

1. **SYSTEM IMPLEMENTATION**
   1. **Client Side Coding**

***Image Default Settings- This coding is used to set the default setting of the Caputred Image from the Intra Oral Sensor.***

<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()> \_

Partial Class frm\_Imagedefaultsettings

Inherits System.Windows.Forms.Form

'Form overrides dispose to clean up the component list.

<System.Diagnostics.DebuggerNonUserCode()> \_

Protected Overrides Sub Dispose(ByVal disposing As Boolean)

Try

If disposing AndAlso components IsNot Nothing Then

components.Dispose()

End If

Finally

MyBase.Dispose(disposing)

End Try

End Sub

'Required by the Windows Form Designer

Private components As System.ComponentModel.IContainer

'NOTE: The following procedure is required by the Windows Form Designer

'It can be modified using the Windows Form Designer.

'Do not modify it using the code editor.

<System.Diagnostics.DebuggerStepThrough()> \_

Private Sub InitializeComponent()

Me.Label1 = New System.Windows.Forms.Label()

Me.chkImgScalefactor = New System.Windows.Forms.CheckBox()

Me.Label2 = New System.Windows.Forms.Label()

Me.NumUpDContrast = New System.Windows.Forms.NumericUpDown()

Me.Label3 = New System.Windows.Forms.Label()

Me.NumUpDBrightness = New System.Windows.Forms.NumericUpDown()

Me.Label4 = New System.Windows.Forms.Label()

Me.Label5 = New System.Windows.Forms.Label()

Me.Button4 = New System.Windows.Forms.Button()

Me.Button5 = New System.Windows.Forms.Button()

Me.Button6 = New System.Windows.Forms.Button()

CType(Me.NumUpDContrast, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.NumUpDBrightness, System.ComponentModel.ISupportInitialize).BeginInit()

Me.SuspendLayout()

'

'Label1

'

Me.Label1.AutoSize = True

Me.Label1.Location = New System.Drawing.Point(44, 33)

Me.Label1.Margin = New System.Windows.Forms.Padding(4, 0, 4, 0)

Me.Label1.Name = "Label1"

Me.Label1.Size = New System.Drawing.Size(168, 14)

Me.Label1.TabIndex = 0

Me.Label1.Text = "Ignore image scalefactor:"

'

'chkImgScalefactor

'

Me.chkImgScalefactor.AutoSize = True

Me.chkImgScalefactor.Checked = True

Me.chkImgScalefactor.CheckState = System.Windows.Forms.CheckState.Checked

Me.chkImgScalefactor.Location = New System.Drawing.Point(220, 33)

Me.chkImgScalefactor.Margin = New System.Windows.Forms.Padding(4, 3, 4, 3)

Me.chkImgScalefactor.Name = "chkImgScalefactor"

Me.chkImgScalefactor.Size = New System.Drawing.Size(15, 14)

Me.chkImgScalefactor.TabIndex = 1

Me.chkImgScalefactor.UseVisualStyleBackColor = True

'

'Label2

'

Me.Label2.AutoSize = True

Me.Label2.Location = New System.Drawing.Point(145, 68)

Me.Label2.Margin = New System.Windows.Forms.Padding(4, 0, 4, 0)

Me.Label2.Name = "Label2"

Me.Label2.Size = New System.Drawing.Size(67, 14)

Me.Label2.TabIndex = 2

Me.Label2.Text = "Contrast:"

'

'NumUpDContrast

'

Me.NumUpDContrast.Location = New System.Drawing.Point(220, 67)

Me.NumUpDContrast.Margin = New System.Windows.Forms.Padding(4, 3, 4, 3)

Me.NumUpDContrast.Name = "NumUpDContrast"

Me.NumUpDContrast.Size = New System.Drawing.Size(101, 22)

Me.NumUpDContrast.TabIndex = 3

'

'Label3

'

Me.Label3.AutoSize = True

Me.Label3.Location = New System.Drawing.Point(133, 107)

Me.Label3.Margin = New System.Windows.Forms.Padding(4, 0, 4, 0)

Me.Label3.Name = "Label3"

Me.Label3.Size = New System.Drawing.Size(79, 14)

Me.Label3.TabIndex = 4

Me.Label3.Text = "Brightness:"

'

'NumUpDBrightness

'

Me.NumUpDBrightness.Location = New System.Drawing.Point(220, 105)

Me.NumUpDBrightness.Margin = New System.Windows.Forms.Padding(4, 3, 4, 3)

Me.NumUpDBrightness.Name = "NumUpDBrightness"

Me.NumUpDBrightness.Size = New System.Drawing.Size(101, 22)

Me.NumUpDBrightness.TabIndex = 5

'

'Label4

'

Me.Label4.AutoSize = True

Me.Label4.Location = New System.Drawing.Point(147, 146)

Me.Label4.Margin = New System.Windows.Forms.Padding(4, 0, 4, 0)

Me.Label4.Name = "Label4"

Me.Label4.Size = New System.Drawing.Size(65, 14)

Me.Label4.TabIndex = 6

Me.Label4.Text = "Sharpen:"

'

'Label5

'

Me.Label5.AutoSize = True

Me.Label5.Location = New System.Drawing.Point(136, 185)

Me.Label5.Margin = New System.Windows.Forms.Padding(4, 0, 4, 0)

Me.Label5.Name = "Label5"

Me.Label5.Size = New System.Drawing.Size(76, 14)

Me.Label5.TabIndex = 7

Me.Label5.Text = "Smoothen:"

'

'Button4

'

Me.Button4.Location = New System.Drawing.Point(439, 65)

Me.Button4.Margin = New System.Windows.Forms.Padding(4, 3, 4, 3)

Me.Button4.Name = "Button4"

Me.Button4.Size = New System.Drawing.Size(125, 25)

Me.Button4.TabIndex = 9

Me.Button4.Text = "OK"

Me.Button4.UseVisualStyleBackColor = True

'

'Button5

'

Me.Button5.Location = New System.Drawing.Point(439, 96)

Me.Button5.Margin = New System.Windows.Forms.Padding(4, 3, 4, 3)

Me.Button5.Name = "Button5"

Me.Button5.Size = New System.Drawing.Size(125, 25)

Me.Button5.TabIndex = 10

Me.Button5.Text = "Cancel"

Me.Button5.UseVisualStyleBackColor = True

'

'Button6

'

Me.Button6.Location = New System.Drawing.Point(439, 129)

Me.Button6.Margin = New System.Windows.Forms.Padding(4, 3, 4, 3)

Me.Button6.Name = "Button6"

Me.Button6.Size = New System.Drawing.Size(125, 25)

Me.Button6.TabIndex = 11

Me.Button6.Text = "Help"

Me.Button6.UseVisualStyleBackColor = True

'

'frm\_Imagedefaultsettings

'

Me.AutoScaleDimensions = New System.Drawing.SizeF(8.0!, 14.0!)

Me.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font

Me.ClientSize = New System.Drawing.Size(592, 256)

Me.Controls.Add(Me.Button6)

Me.Controls.Add(Me.Button5)

Me.Controls.Add(Me.Button4)

Me.Controls.Add(Me.Label5)

Me.Controls.Add(Me.Label4)

Me.Controls.Add(Me.NumUpDBrightness)

Me.Controls.Add(Me.Label3)

Me.Controls.Add(Me.NumUpDContrast)

Me.Controls.Add(Me.Label2)

Me.Controls.Add(Me.chkImgScalefactor)

Me.Controls.Add(Me.Label1)

Me.Font = New System.Drawing.Font("Verdana", 9.0!, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, CType(0, Byte))

Me.Margin = New System.Windows.Forms.Padding(4, 3, 4, 3)

Me.Name = "frm\_Imagedefaultsettings"

Me.Text = "Edit Default image settings"

CType(Me.NumUpDContrast, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.NumUpDBrightness, System.ComponentModel.ISupportInitialize).EndInit()

Me.ResumeLayout(False)

Me.PerformLayout()

End Sub

Friend WithEvents Label1 As System.Windows.Forms.Label

Friend WithEvents chkImgScalefactor As System.Windows.Forms.CheckBox

Friend WithEvents Label2 As System.Windows.Forms.Label

Friend WithEvents NumUpDContrast As System.Windows.Forms.NumericUpDown

Friend WithEvents Label3 As System.Windows.Forms.Label

Friend WithEvents NumUpDBrightness As System.Windows.Forms.NumericUpDown

Friend WithEvents Label4 As System.Windows.Forms.Label

Friend WithEvents Label5 As System.Windows.Forms.Label

Friend WithEvents Button4 As System.Windows.Forms.Button

Friend WithEvents Button5 As System.Windows.Forms.Button

Friend WithEvents Button6 As System.Windows.Forms.Button

End Class

***Image Measurements - Client side coding. This is used to find the measurements of the images in mm/cm based on the user requirements***

<Global.Microsoft.VisualBasic.CompilerServices.DesignerGenerated()> \_

Partial Class frm\_Measurements

Inherits System.Windows.Forms.Form

'Form overrides dispose to clean up the component list.

<System.Diagnostics.DebuggerNonUserCode()> \_

Protected Overrides Sub Dispose(ByVal disposing As Boolean)

Try

If disposing AndAlso components IsNot Nothing Then

components.Dispose()

End If

Finally

MyBase.Dispose(disposing)

End Try

End Sub

'Required by the Windows Form Designer

Private components As System.ComponentModel.IContainer

'NOTE: The following procedure is required by the Windows Form Designer

'It can be modified using the Windows Form Designer.

'Do not modify it using the code editor.

<System.Diagnostics.DebuggerStepThrough()> \_

Private Sub InitializeComponent()

Dim resources As System.ComponentModel.ComponentResourceManager = New System.ComponentModel.ComponentResourceManager(GetType(frm\_Measurements))

Me.PictureBox1 = New System.Windows.Forms.PictureBox()

Me.pic\_distance\_measure = New System.Windows.Forms.PictureBox()

Me.PictureBox3 = New System.Windows.Forms.PictureBox()

Me.PictureBox4 = New System.Windows.Forms.PictureBox()

Me.PictureBox5 = New System.Windows.Forms.PictureBox()

Me.PictureBox6 = New System.Windows.Forms.PictureBox()

Me.PictureBox7 = New System.Windows.Forms.PictureBox()

Me.PictureBox8 = New System.Windows.Forms.PictureBox()

Me.PictureBox9 = New System.Windows.Forms.PictureBox()

Me.PictureBox10 = New System.Windows.Forms.PictureBox()

Me.Label2 = New System.Windows.Forms.Label()

Me.Label1 = New System.Windows.Forms.Label()

Me.HScrollBlack = New System.Windows.Forms.HScrollBar()

Me.HScrollWhite = New System.Windows.Forms.HScrollBar()

CType(Me.PictureBox1, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.pic\_distance\_measure, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox3, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox4, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox5, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox6, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox7, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox8, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox9, System.ComponentModel.ISupportInitialize).BeginInit()

CType(Me.PictureBox10, System.ComponentModel.ISupportInitialize).BeginInit()

Me.SuspendLayout()

'

'PictureBox1

'

Me.PictureBox1.BackgroundImage = CType(resources.GetObject("PictureBox1.BackgroundImage"), System.Drawing.Image)

Me.PictureBox1.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox1.Location = New System.Drawing.Point(0, 0)

Me.PictureBox1.Name = "PictureBox1"

Me.PictureBox1.Size = New System.Drawing.Size(40, 44)

Me.PictureBox1.TabIndex = 0

Me.PictureBox1.TabStop = False

'

'pic\_distance\_measure

'

Me.pic\_distance\_measure.BackgroundImage = CType(resources.GetObject("pic\_distance\_measure.BackgroundImage"), System.Drawing.Image)

Me.pic\_distance\_measure.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.pic\_distance\_measure.Location = New System.Drawing.Point(44, 0)

Me.pic\_distance\_measure.Name = "pic\_distance\_measure"

Me.pic\_distance\_measure.Size = New System.Drawing.Size(40, 44)

Me.pic\_distance\_measure.TabIndex = 0

Me.pic\_distance\_measure.TabStop = False

'

'PictureBox3

'

Me.PictureBox3.BackgroundImage = CType(resources.GetObject("PictureBox3.BackgroundImage"), System.Drawing.Image)

Me.PictureBox3.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox3.Location = New System.Drawing.Point(88, 0)

Me.PictureBox3.Name = "PictureBox3"

Me.PictureBox3.Size = New System.Drawing.Size(40, 44)

Me.PictureBox3.TabIndex = 0

Me.PictureBox3.TabStop = False

'

'PictureBox4

'

Me.PictureBox4.BackgroundImage = CType(resources.GetObject("PictureBox4.BackgroundImage"), System.Drawing.Image)

Me.PictureBox4.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox4.Location = New System.Drawing.Point(132, 0)

Me.PictureBox4.Name = "PictureBox4"

Me.PictureBox4.Size = New System.Drawing.Size(40, 44)

Me.PictureBox4.TabIndex = 0

Me.PictureBox4.TabStop = False

'

'PictureBox5

'

Me.PictureBox5.BackgroundImage = CType(resources.GetObject("PictureBox5.BackgroundImage"), System.Drawing.Image)

Me.PictureBox5.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox5.Location = New System.Drawing.Point(176, 0)

Me.PictureBox5.Name = "PictureBox5"

Me.PictureBox5.Size = New System.Drawing.Size(40, 44)

Me.PictureBox5.TabIndex = 0

Me.PictureBox5.TabStop = False

'

'PictureBox6

'

Me.PictureBox6.BackgroundImage = CType(resources.GetObject("PictureBox6.BackgroundImage"), System.Drawing.Image)

Me.PictureBox6.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox6.Location = New System.Drawing.Point(220, 0)

Me.PictureBox6.Name = "PictureBox6"

Me.PictureBox6.Size = New System.Drawing.Size(40, 44)

Me.PictureBox6.TabIndex = 0

Me.PictureBox6.TabStop = False

'

'PictureBox7

'

Me.PictureBox7.BackgroundImage = CType(resources.GetObject("PictureBox7.BackgroundImage"), System.Drawing.Image)

Me.PictureBox7.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox7.Location = New System.Drawing.Point(352, 0)

Me.PictureBox7.Name = "PictureBox7"

Me.PictureBox7.Size = New System.Drawing.Size(40, 44)

Me.PictureBox7.TabIndex = 0

Me.PictureBox7.TabStop = False

'

'PictureBox8

'

Me.PictureBox8.BackgroundImage = CType(resources.GetObject("PictureBox8.BackgroundImage"), System.Drawing.Image)

Me.PictureBox8.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox8.Location = New System.Drawing.Point(397, 0)

Me.PictureBox8.Name = "PictureBox8"

Me.PictureBox8.Size = New System.Drawing.Size(40, 44)

Me.PictureBox8.TabIndex = 0

Me.PictureBox8.TabStop = False

'

'PictureBox9

'

Me.PictureBox9.BackgroundImage = CType(resources.GetObject("PictureBox9.BackgroundImage"), System.Drawing.Image)

Me.PictureBox9.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox9.Location = New System.Drawing.Point(308, 0)

Me.PictureBox9.Name = "PictureBox9"

Me.PictureBox9.Size = New System.Drawing.Size(40, 44)

Me.PictureBox9.TabIndex = 2

Me.PictureBox9.TabStop = False

'

'PictureBox10

'

Me.PictureBox10.BackgroundImage = CType(resources.GetObject("PictureBox10.BackgroundImage"), System.Drawing.Image)

Me.PictureBox10.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Zoom

Me.PictureBox10.Location = New System.Drawing.Point(264, 0)

Me.PictureBox10.Name = "PictureBox10"

Me.PictureBox10.Size = New System.Drawing.Size(40, 44)

Me.PictureBox10.TabIndex = 1

Me.PictureBox10.TabStop = False

'

'Label2

'

Me.Label2.AutoSize = True

Me.Label2.Font = New System.Drawing.Font("Verdana", 9.75!, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, CType(0, Byte))

Me.Label2.Location = New System.Drawing.Point(12, 111)

Me.Label2.Name = "Label2"

Me.Label2.Size = New System.Drawing.Size(71, 16)

Me.Label2.TabIndex = 24

Me.Label2.Text = "Contrast"

'

'Label1

'

Me.Label1.AutoSize = True

Me.Label1.Font = New System.Drawing.Font("Verdana", 9.75!, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, CType(0, Byte))

Me.Label1.Location = New System.Drawing.Point(12, 58)

Me.Label1.Name = "Label1"

Me.Label1.Size = New System.Drawing.Size(86, 16)

Me.Label1.TabIndex = 23

Me.Label1.Text = "Brightness"

'

'HScrollBlack

'

Me.HScrollBlack.LargeChange = 1

Me.HScrollBlack.Location = New System.Drawing.Point(15, 137)

Me.HScrollBlack.Name = "HScrollBlack"

Me.HScrollBlack.Size = New System.Drawing.Size(294, 17)

Me.HScrollBlack.TabIndex = 22

'

'HScrollWhite

'

Me.HScrollWhite.LargeChange = 1

Me.HScrollWhite.Location = New System.Drawing.Point(15, 83)

Me.HScrollWhite.Name = "HScrollWhite"

Me.HScrollWhite.Size = New System.Drawing.Size(294, 17)

Me.HScrollWhite.TabIndex = 21

'

'frm\_Measurements

'

Me.AutoScaleDimensions = New System.Drawing.SizeF(6.0!, 13.0!)

Me.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font

Me.ClientSize = New System.Drawing.Size(439, 208)

Me.Controls.Add(Me.Label2)

Me.Controls.Add(Me.Label1)

Me.Controls.Add(Me.HScrollBlack)

Me.Controls.Add(Me.HScrollWhite)

Me.Controls.Add(Me.PictureBox9)

Me.Controls.Add(Me.PictureBox10)

Me.Controls.Add(Me.PictureBox8)

Me.Controls.Add(Me.PictureBox7)

Me.Controls.Add(Me.PictureBox6)

Me.Controls.Add(Me.PictureBox5)

Me.Controls.Add(Me.PictureBox4)

Me.Controls.Add(Me.PictureBox3)

Me.Controls.Add(Me.pic\_distance\_measure)

Me.Controls.Add(Me.PictureBox1)

Me.Icon = CType(resources.GetObject("$this.Icon"), System.Drawing.Icon)

Me.Name = "frm\_Measurements"

Me.Text = "Measurements"

CType(Me.PictureBox1, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.pic\_distance\_measure, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox3, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox4, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox5, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox6, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox7, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox8, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox9, System.ComponentModel.ISupportInitialize).EndInit()

CType(Me.PictureBox10, System.ComponentModel.ISupportInitialize).EndInit()

Me.ResumeLayout(False)

Me.PerformLayout()

End Sub

Friend WithEvents PictureBox1 As System.Windows.Forms.PictureBox

Friend WithEvents pic\_distance\_measure As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox3 As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox4 As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox5 As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox6 As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox7 As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox8 As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox9 As System.Windows.Forms.PictureBox

Friend WithEvents PictureBox10 As System.Windows.Forms.PictureBox

Friend WithEvents Label2 As System.Windows.Forms.Label

Friend WithEvents Label1 As System.Windows.Forms.Label

Friend WithEvents HScrollBlack As System.Windows.Forms.HScrollBar

Friend WithEvents HScrollWhite As System.Windows.Forms.HScrollBar

End Class

* 1. **Server Side Coding**

***DeviceManagement.cs – This coding is used to verify the Intra Oral Device is connected with the software using the USB Port***

using System;

using System.Runtime.InteropServices;

//using System.Windows.Forms;

namespace WinUsbSample

{

/// <summary>

/// Routines for receiving device notifications.

/// </summary>

sealed internal partial class DeviceManagement

{

/// <summary>

/// Compares two device path names. Used to find out if the device name

/// of a recently attached or removed device matches the name of a

/// device the application is communicating with.

/// </summary>

///

///

/// <param name="mydevicePathName"> a device pathname returned by

/// SetupDiGetDeviceInterfaceDetail in an SP\_DEVICE\_INTERFACE\_DETAIL\_DATA structure. </param>

///

/// <returns>

/// True if the names match, False if not.

/// </returns>

///

internal bool DeviceNameMatch(IntPtr lParam, String mydevicePathName)

{

Int32 stringSize;

try

{

DEV\_BROADCAST\_DEVICEINTERFACE\_1 devBroadcastDeviceInterface = new DEV\_BROADCAST\_DEVICEINTERFACE\_1();

DEV\_BROADCAST\_HDR devBroadcastHeader = new DEV\_BROADCAST\_HDR();

// The lParam parameter of Message is a pointer to a DEV\_BROADCAST\_HDR structure.

Marshal.PtrToStructure(lParam, devBroadcastHeader);

if ((devBroadcastHeader.dbch\_devicetype == DBT\_DEVTYP\_DEVICEINTERFACE))

{

// The dbch\_devicetype parameter indicates that the event applies to a device interface.

// So the structure in lParam is actually a DEV\_BROADCAST\_INTERFACE structure,

// which begins with a DEV\_BROADCAST\_HDR.

// Obtain the number of characters in dbch\_name by subtracting the 32 bytes

// in the strucutre that are not part of dbch\_name and dividing by 2 because there are

// 2 bytes per character.

stringSize = System.Convert.ToInt32((devBroadcastHeader.dbch\_size - 32) / 2);

// The dbcc\_name parameter of devBroadcastDeviceInterface contains the device name.

// Trim dbcc\_name to match the size of the String.

devBroadcastDeviceInterface.dbcc\_name = new Char[stringSize + 1];

// Marshal data from the unmanaged block pointed to by lParam

// to the managed object devBroadcastDeviceInterface.

Marshal.PtrToStructure(lParam, devBroadcastDeviceInterface);

// Store the device name in a String.

String DeviceNameString = new String(devBroadcastDeviceInterface.dbcc\_name, 0, stringSize);

// Compare the name of the newly attached device with the name of the device

// the application is accessing (mydevicePathName).

// Set ignorecase True.

// mydevicePathName of S11684 and S11685 will vary in acorrdance with the port of PC.

// It is important to use a constant part of mydevicePathName.

if ((String.Compare(DeviceNameString.Substring(0, 24), mydevicePathName.Substring(0, 24), true) == 0))

{

return true;

}

else

{

return false;

}

}

}

catch (Exception)

{

throw;

}

return false;

}

/// <summary>

/// Use SetupDi API functions to retrieve the device path name of an

/// attached device that belongs to a device interface class.

/// </summary>

///

/// <param name="myGuid"> an interface class GUID. </param>

/// <param name="devicePathName"> a pointer to the device path name

/// of an attached device. </param>

///

/// <returns>

/// True if a device is found, False if not.

/// </returns>

internal bool FindDeviceFromGuid(System.Guid myGuid, ref String devicePathName)

{

Int32 bufferSize = 0;

IntPtr detailDataBuffer = IntPtr.Zero;

bool deviceFound;

IntPtr deviceInfoSet = new System.IntPtr();

bool lastDevice = false;

Int32 memberIndex = 0;

SP\_DEVICE\_INTERFACE\_DATA MyDeviceInterfaceData = new SP\_DEVICE\_INTERFACE\_DATA();

bool success;

try

{

// \*\*\*

// API function

// summary

// Retrieves a device information set for a specified group of devices.

// SetupDiEnumDeviceInterfaces uses the device information set.

// parameters

// Interface class GUID.

// Null to retrieve information for all device instances.

// Optional handle to a top-level window (unused here).

// Flags to limit the returned information to currently present devices

// and devices that expose interfaces in the class specified by the GUID.

// Returns

// Handle to a device information set for the devices.

// \*\*\*

deviceInfoSet = SetupDiGetClassDevs(ref myGuid, IntPtr.Zero, IntPtr.Zero, DIGCF\_PRESENT | DIGCF\_DEVICEINTERFACE);

deviceFound = false;

memberIndex = 0;

// The cbSize element of the MyDeviceInterfaceData structure must be set to

// the structure's size in bytes.

// The size is 28 bytes for 32-bit code and 32 bits for 64-bit code.

MyDeviceInterfaceData.cbSize = Marshal.SizeOf(MyDeviceInterfaceData);

do

{

// Begin with 0 and increment through the device information set until

// no more devices are available.

// \*\*\*

// API function

// summary

// Retrieves a handle to a SP\_DEVICE\_INTERFACE\_DATA structure for a device.

// On return, MyDeviceInterfaceData contains the handle to a

// SP\_DEVICE\_INTERFACE\_DATA structure for a detected device.

// parameters

// DeviceInfoSet returned by SetupDiGetClassDevs.

// Optional SP\_DEVINFO\_DATA structure that defines a device instance

// that is a member of a device information set.

// Device interface GUID.

// Index to specify a device in a device information set.

// Pointer to a handle to a SP\_DEVICE\_INTERFACE\_DATA structure for a device.

// Returns

// True on success.

// \*\*\*

success = SetupDiEnumDeviceInterfaces

(deviceInfoSet,

IntPtr.Zero,

ref myGuid,

memberIndex,

ref MyDeviceInterfaceData);

// Find out if a device information set was retrieved.

if (!success)

{

lastDevice = true;

}

else

{

// A device is present.

// \*\*\*

// API function:

// summary:

// Retrieves an SP\_DEVICE\_INTERFACE\_DETAIL\_DATA structure

// containing information about a device.

// To retrieve the information, call this function twice.

// The first time returns the size of the structure.

// The second time returns a pointer to the data.

// parameters

// DeviceInfoSet returned by SetupDiGetClassDevs

// SP\_DEVICE\_INTERFACE\_DATA structure returned by SetupDiEnumDeviceInterfaces

// A returned pointer to an SP\_DEVICE\_INTERFACE\_DETAIL\_DATA

// Structure to receive information about the specified interface.

// The size of the SP\_DEVICE\_INTERFACE\_DETAIL\_DATA structure.

// Pointer to a variable that will receive the returned required size of the

// SP\_DEVICE\_INTERFACE\_DETAIL\_DATA structure.

// Returned pointer to an SP\_DEVINFO\_DATA structure to receive information about the device.

// Returns

// True on success.

// \*\*\*

success = SetupDiGetDeviceInterfaceDetail

(deviceInfoSet,

ref MyDeviceInterfaceData,

IntPtr.Zero,

0,

ref bufferSize,

IntPtr.Zero);

// Allocate memory for the SP\_DEVICE\_INTERFACE\_DETAIL\_DATA structure using the returned buffer size.

detailDataBuffer = Marshal.AllocHGlobal(bufferSize);

// Store cbSize in the first bytes of the array. The number of bytes varies with 32- and 64-bit systems.

Marshal.WriteInt32(detailDataBuffer, (IntPtr.Size == 4) ? (4 + Marshal.SystemDefaultCharSize) : 8);

// Call SetupDiGetDeviceInterfaceDetail again.

// This time, pass a pointer to DetailDataBuffer

// and the returned required buffer size.

success = SetupDiGetDeviceInterfaceDetail

(deviceInfoSet,

ref MyDeviceInterfaceData,

detailDataBuffer,

bufferSize,

ref bufferSize,

IntPtr.Zero);

// Skip over cbsize (4 bytes) to get the address of the devicePathName.

IntPtr pDevicePathName = new IntPtr(detailDataBuffer.ToInt32() + 4);

// Get the String containing the devicePathName.

devicePathName = Marshal.PtrToStringAuto(pDevicePathName);

deviceFound = true;

}

memberIndex = memberIndex + 1;

}

while (!((lastDevice == true)));

return deviceFound;

}

catch (Exception)

{

throw;

}

finally

{

if (detailDataBuffer != IntPtr.Zero)

{

// Free the memory allocated previously by AllocHGlobal.

Marshal.FreeHGlobal(detailDataBuffer);

}

if (deviceInfoSet != IntPtr.Zero)

{

// \*\*\*

// API function

// summary

// Frees the memory reserved for the DeviceInfoSet returned by SetupDiGetClassDevs.

// parameters

// DeviceInfoSet returned by SetupDiGetClassDevs.

// returns

// True on success.

// \*\*\*

SetupDiDestroyDeviceInfoList(deviceInfoSet);

}

}

}

/// <summary>

/// Requests to receive a notification when a device is attached or removed.

/// </summary>

///

/// <param name="devicePathName"> handle to a device. </param>

/// <param name="formHandle"> handle to the window that will receive device events. </param>

/// <param name="classGuid"> device interface GUID. </param>

/// <param name="deviceNotificationHandle"> returned device notification handle. </param>

///

/// <returns>

/// True on success.

/// </returns>

///

internal bool RegisterForDeviceNotifications(String devicePathName, IntPtr formHandle, Guid classGuid, ref IntPtr deviceNotificationHandle)

{

// A DEV\_BROADCAST\_DEVICEINTERFACE header holds information about the request.

DEV\_BROADCAST\_DEVICEINTERFACE devBroadcastDeviceInterface = new DEV\_BROADCAST\_DEVICEINTERFACE();

IntPtr devBroadcastDeviceInterfaceBuffer = IntPtr.Zero;

Int32 size = 0;

try

{

// Set the parameters in the DEV\_BROADCAST\_DEVICEINTERFACE structure.

// Set the size.

size = Marshal.SizeOf(devBroadcastDeviceInterface);

devBroadcastDeviceInterface.dbcc\_size = size;

// Request to receive notifications about a class of devices.

devBroadcastDeviceInterface.dbcc\_devicetype = DBT\_DEVTYP\_DEVICEINTERFACE;

devBroadcastDeviceInterface.dbcc\_reserved = 0;

// Specify the interface class to receive notifications about.

devBroadcastDeviceInterface.dbcc\_classguid = classGuid;

// Allocate memory for the buffer that holds the DEV\_BROADCAST\_DEVICEINTERFACE structure.

devBroadcastDeviceInterfaceBuffer = Marshal.AllocHGlobal(size);

// Copy the DEV\_BROADCAST\_DEVICEINTERFACE structure to the buffer.

// Set fDeleteOld True to prevent memory leaks.

Marshal.StructureToPtr(devBroadcastDeviceInterface, devBroadcastDeviceInterfaceBuffer, true);

// \*\*\*

// API function

// summary

// Request to receive notification messages when a device in an interface class

// is attached or removed.

// parameters

// Handle to the window that will receive device events.

// Pointer to a DEV\_BROADCAST\_DEVICEINTERFACE to specify the type of

// device to send notifications for.

// DEVICE\_NOTIFY\_WINDOW\_HANDLE indicates the handle is a window handle.

// Returns

// Device notification handle or NULL on failure.

// \*\*\*

deviceNotificationHandle = RegisterDeviceNotification(formHandle, devBroadcastDeviceInterfaceBuffer, DEVICE\_NOTIFY\_WINDOW\_HANDLE);

// Marshal data from the unmanaged block devBroadcastDeviceInterfaceBuffer to

// the managed object devBroadcastDeviceInterface

Marshal.PtrToStructure(devBroadcastDeviceInterfaceBuffer, devBroadcastDeviceInterface);

if ((deviceNotificationHandle.ToInt32() == IntPtr.Zero.ToInt32()))

{

return false;

}

else

{

return true;

}

}

catch (Exception)

{

throw;

}

finally

{

if (devBroadcastDeviceInterfaceBuffer != IntPtr.Zero)

{

// Free the memory allocated previously by AllocHGlobal.

Marshal.FreeHGlobal(devBroadcastDeviceInterfaceBuffer);

}

}

}

/// <summary>

/// Requests to stop receiving notification messages when a device in an

/// interface class is attached or removed.

/// </summary>

///

/// <param name="deviceNotificationHandle"> handle returned previously by

/// RegisterDeviceNotification. </param>

internal void StopReceivingDeviceNotifications(IntPtr deviceNotificationHandle)

{

try

{

// \*\*\*

// API function

// summary

// Stop receiving notification messages.

// parameters

// Handle returned previously by RegisterDeviceNotification.

// returns

// True on success.

// \*\*\*

// Ignore failures.

DeviceManagement.UnregisterDeviceNotification(deviceNotificationHandle);

}

catch (Exception)

{

throw;

}

}

}

}

***CMOS\_USB.CS – This coding will help the Software to understand that the Intra Oral Sensor is connected to the computer or laptop and also commands the software that the Intra Oral Sensor is ready to take images using X-Ray and the Data is ready is communicate to the software.***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Runtime.InteropServices;

namespace CMOS\_USB\_DemoApplication

{

class CMOS\_USB

{

//Error handle

public static readonly IntPtr INVALID\_HANDLE\_VALUE = new IntPtr(-1);

[StructLayout(LayoutKind.Sequential)]

public struct UnitIntegrationParameter

{

public ushort Xray\_Incident\_Threshold;

public ushort Xray\_Integration\_End\_Threshold;

public double Integration\_Time;

}

[StructLayout(LayoutKind.Sequential)]

public struct UnitXrayImage

{

public byte Mode;

public UnitIntegrationParameter IntegParam;

}

[StructLayout(LayoutKind.Sequential)]

public struct UnitSensorInformation

{

public UnitIntegrationParameter IntegParam;

public ushort Lot\_Serial\_No;

public byte Sensor\_Type;

public byte Firmware\_Version;

}

//Functions

[DllImport("CMOS\_USB.dll")]

public extern static System.IntPtr USB\_OpenDevice(ushort ProductID);

[DllImport("CMOS\_USB.dll")]

public extern static System.IntPtr USB\_OpenTargetDevice(ushort ProductID, ushort SerialNo);

[DllImport("CMOS\_USB.dll")]

public extern static void USB\_CloseDevice(System.IntPtr DeviceHandle);

[DllImport("CMOS\_USB.dll")]

public extern static System.IntPtr USB\_OpenPipe(System.IntPtr DeviceHandle);

[DllImport("CMOS\_USB.dll")]

public extern static uint HPK\_GetXrayImage(IntPtr DeviceHandle, byte[] ImageData, out ulong bufferLength, ref UnitXrayImage XrayImage);

[DllImport("CMOS\_USB.dll")]

public extern static uint HPK\_GetTrigPdData(IntPtr DeviceHandle, double AcquisitionTime, ushort[] TrigPdData, out ulong bufferLength, ref UnitIntegrationParameter IntegParam);

[DllImport("CMOS\_USB.dll")]

public extern static uint HPK\_GetSensorInformation(IntPtr DeviceHandle, ref UnitIntegrationParameter IntegParam, out UnitSensorInformation SensorInfo);

[DllImport("CMOS\_USB.dll")]

public extern static uint HPK\_StopTrigPdData(IntPtr DeviceHandle, byte[] Buffer, out ulong bufferLength, ref UnitIntegrationParameter IntegParam);

[DllImport("CMOS\_USB.dll")]

public extern static uint HPK\_ForceTrigAndGetDummy(IntPtr DeviceHandle, byte[] ImageData, out ulong bufferLength, ref UnitXrayImage XrayImage);

[DllImport("CMOS\_USB.dll")]

public extern static uint HPK\_AbortBulkPipe(IntPtr DeviceHandle);

[DllImport("CMOS\_USB.dll")]

public extern static uint HPK\_GetXrayCorrectionImage(IntPtr DeviceHandle, byte[] ImageData, out ulong bufferLength, ref UnitXrayImage XrayImage);

[DllImport("CMOS\_USB.dll")]

public extern static uint USB\_SuspendDevice(IntPtr DeviceHandle, uint SuspendDelay);

[DllImport("CMOS\_USB.dll")]

public extern static uint USB\_ResumeDevice(IntPtr DeviceHandle);

}

}

1. **SYSTEM TESTING**
   1. Unit Testing

**UNIT TESTING** is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

* 1. Integration Testing

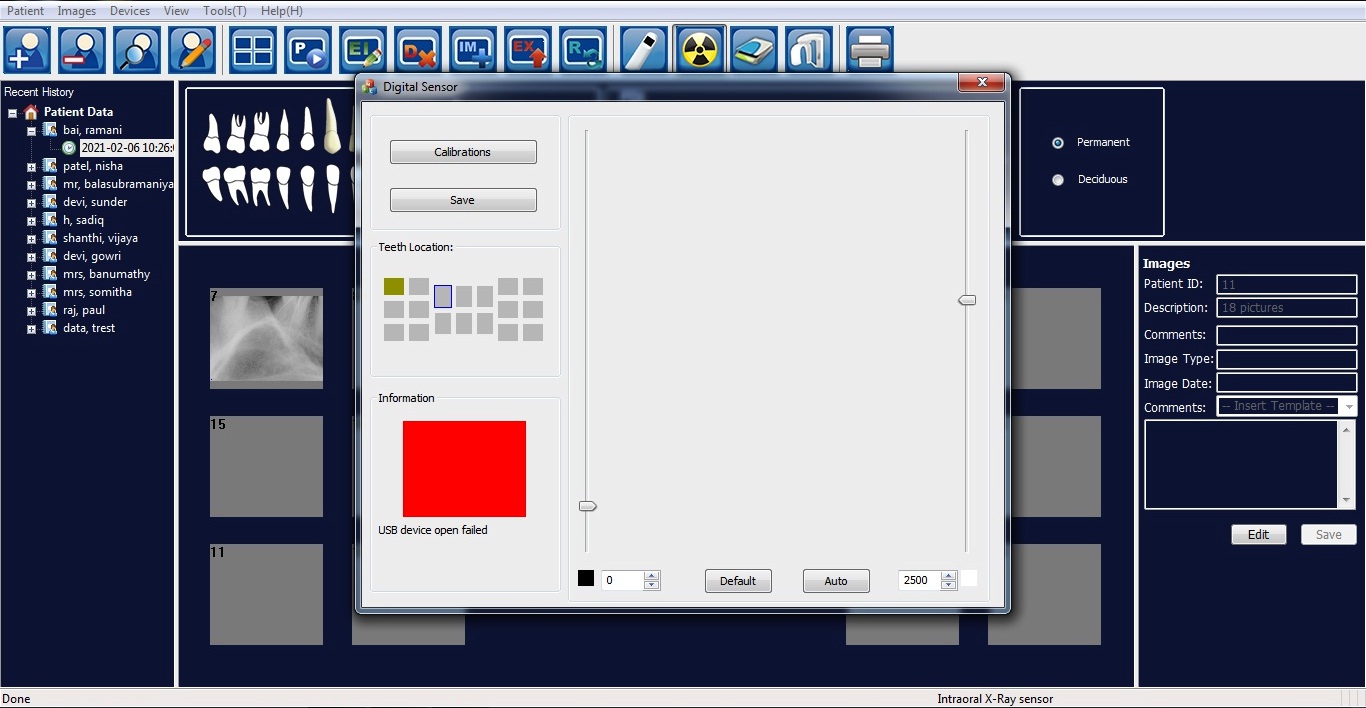
**Integration testing** (sometimes called **integration and testing**, abbreviated **I&T**) is the phase in software testing in which individual software modules are combined and tested as a group. Integration testing is conducted to evaluate the compliance of a system or component with specified functional requirements. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

* 1. Test Case & Reports / Performance Analysis

All the Test Case and Corresponding Reports are analysed in this project. The Entire Project is monitored in their Client Side and the Performace of the project is analysied every 15 days to improve the performace of the project/product.

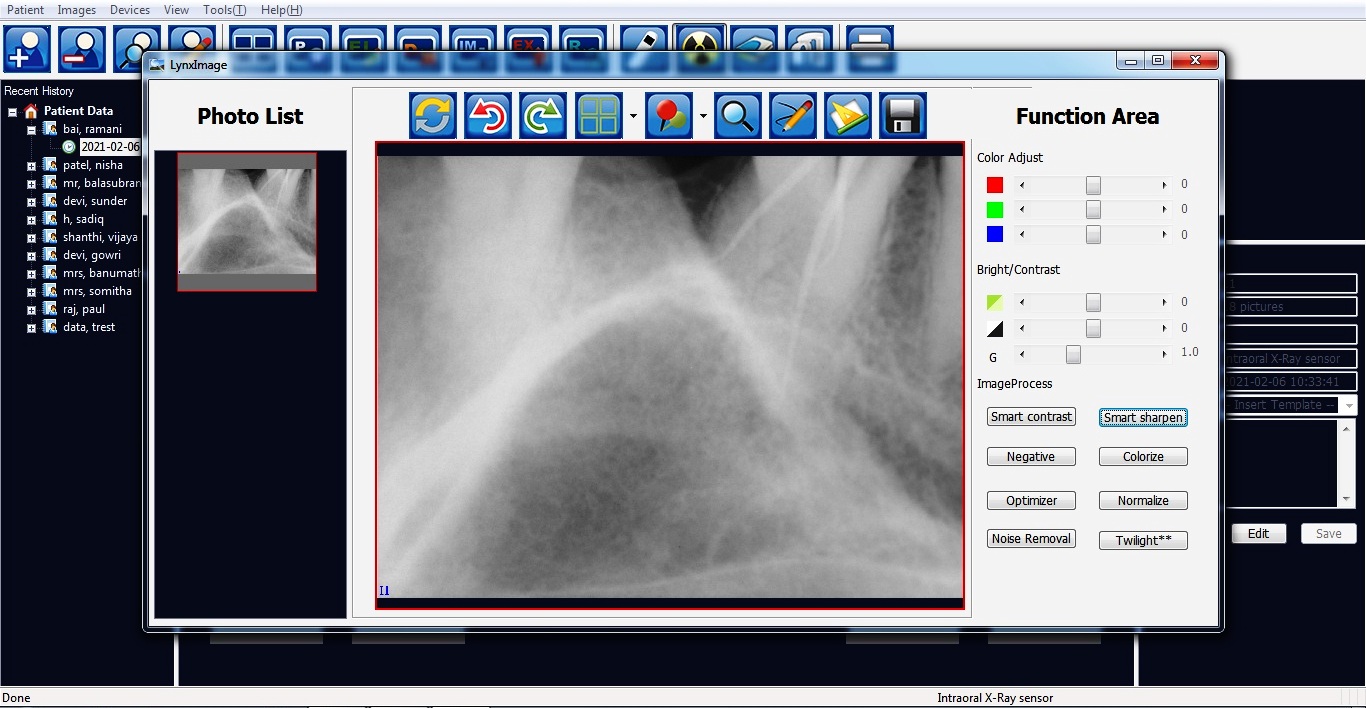
***Test Case 1 – Verifing the Device is connected with the Software. (Server Side)***

Red Color – Indicates the Devices is Not Connected with the Software.

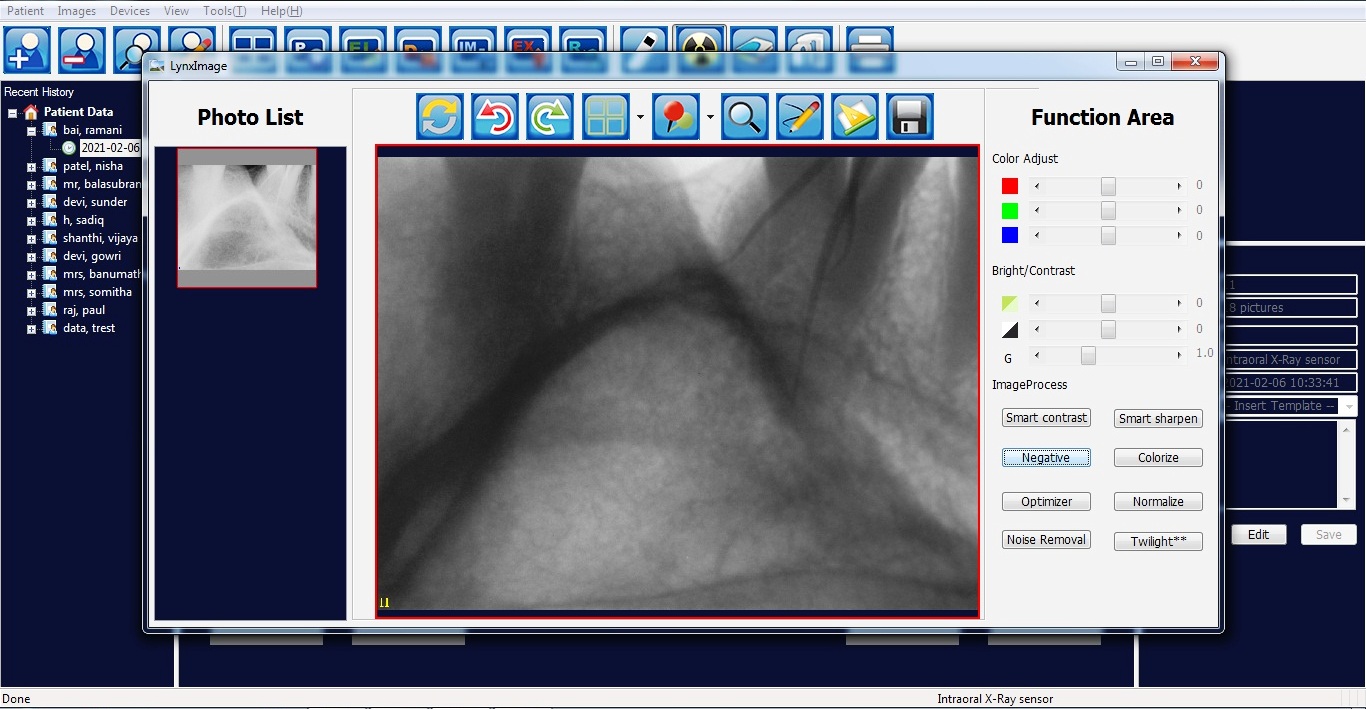
Green Color – Indicates the Devices is Connected with the Software and ready to take X Ray.

***Test Case 2 – Image Default Properties and their corresponding changes after applying the functions for the orginal Images (Functional Area of the Images)***

***Default Capture Image***



***After applying the Function “Negative” the result of the image is as follows***



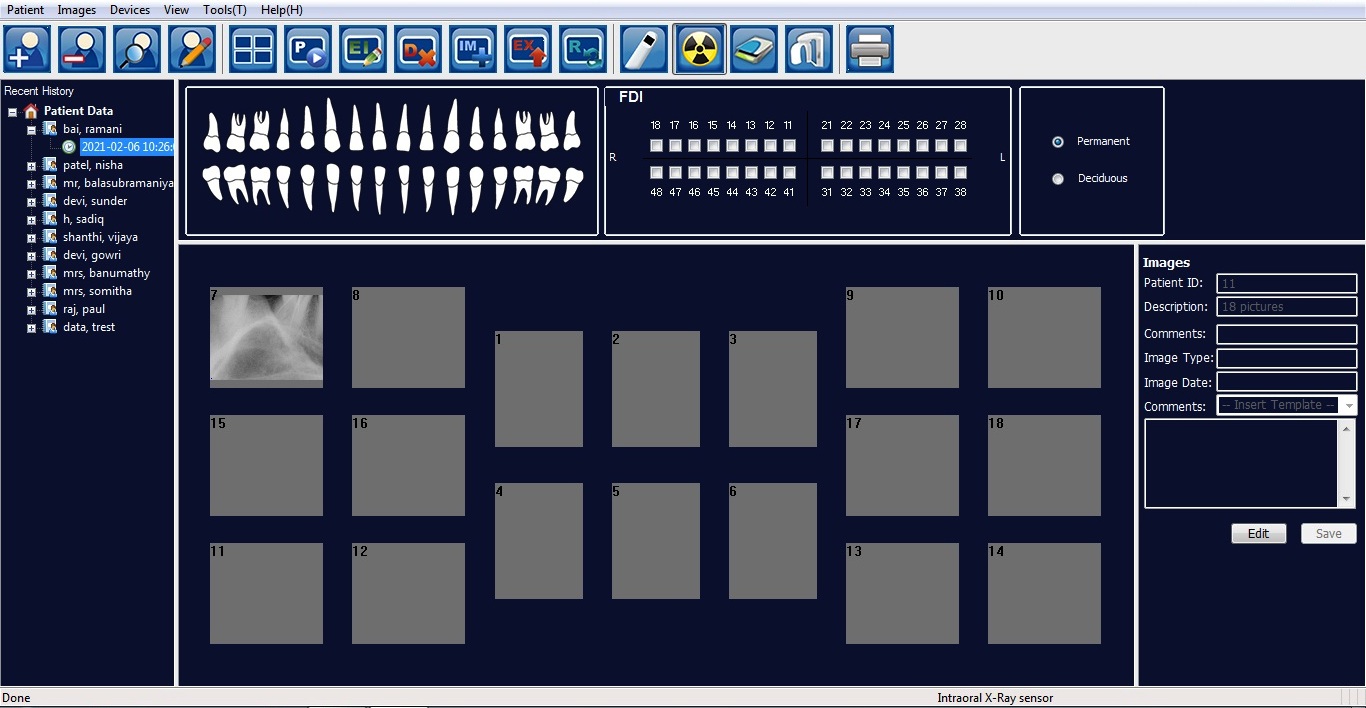
1. **CONCLUSION**

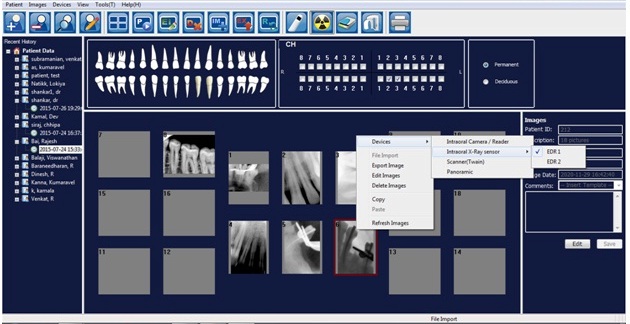
**8.1 Conclusion and Future Enhancement**

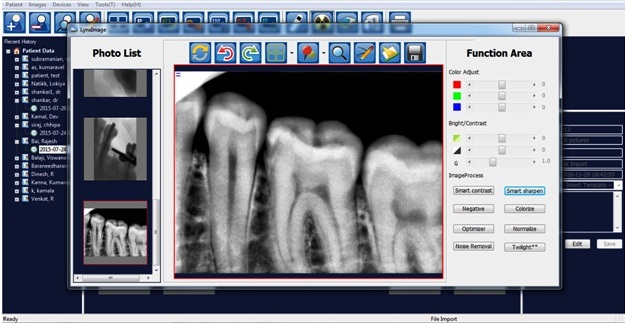
The reduction in average overall time for this combination of improvements is 83%, with a reduction in average working time of 12% and a reduction in average waiting time by 97%.

By applying Intra Oral Sensor (RVG) technique the time for diagnostic procedure is much shorter in comparison with traditional dental radiography enabling archiving and follow-up the presented case in the course of time. The results are faster and better with no loss in image quality.

**A.1 Sample Screens**







**REFERENCES**.

* Dhia Ayu Salsabila1, Muhammad Dachyar from Department of Industrial Engineering, Universitas Indonesia, Depok - The Hospital Radiology Service Redesign, By Using Business Process Re-engineering and Information Systems Approach (2020) pp 3623-3631
* L. Keaver, C. O’Meara, M. Mukhtar, and C. McHugh, "Providing Nutrition Care to Patients with Chronic Disease: An Irish Teaching Hospital Healthcare Professional Study", Journal of Journal of Biomedical Education, vol. 2018, (2018), pp. 1–7.
* M. Lopreite and M. Mauro, "The Effects of Population Ageing on Health Care Expenditure: A Bayesian VAR Analysis Using Data from Italy", Journal of Health Policy, vol. 121, no. 6, (2017), pp. 663–674.
* L. Borgermans, Y. Marchal, L. Busetto, et al., "How to Improve Integrated Care for People with Chronic Conditions: Key Findings from EU FP-7 Project INTEGRATE and Beyond", Journal of International Journal of Integrated Care, vol. 17, no. 4, (2017), pp. 1–12.
* E. OZCETE, B. BOYDAK, M. ERSEL, S. KIYAN, I. UZ, and O. CEVRIM, "Comparison of Conventional Radiography and Digital Computerized Radiography in Patients Presenting to Emergency Department", Journal of Turkish Journal of Emergency Medicine, vol. 15, no. 1, (**2015**), pp. 8–12.
* L. Brandenburg, P. Gabow, and G. Steele, "Innovation and Best Practices in Health Care Scheduling - Discussion Paper", Journal of Institute of Medicine, (2015).
* V. Saliba, H. Legido-Quigley, R. Hallik, A. Aaviksoo, J. Car, and M. McKee, "Telemedicine across Borders: A Systematic Review of Factors That Hinder or Support Implementation", Journal of International Journal of Medical Informatics, vol. 81, no. 12, (2012), pp. 793–809.
* J.W.M. Hoe, "Service Delivery and Service Quality in Radiology", Journal of Journal of the American College of Radiology, vol. 4, no. 9, (**2007**), pp. 643–651.