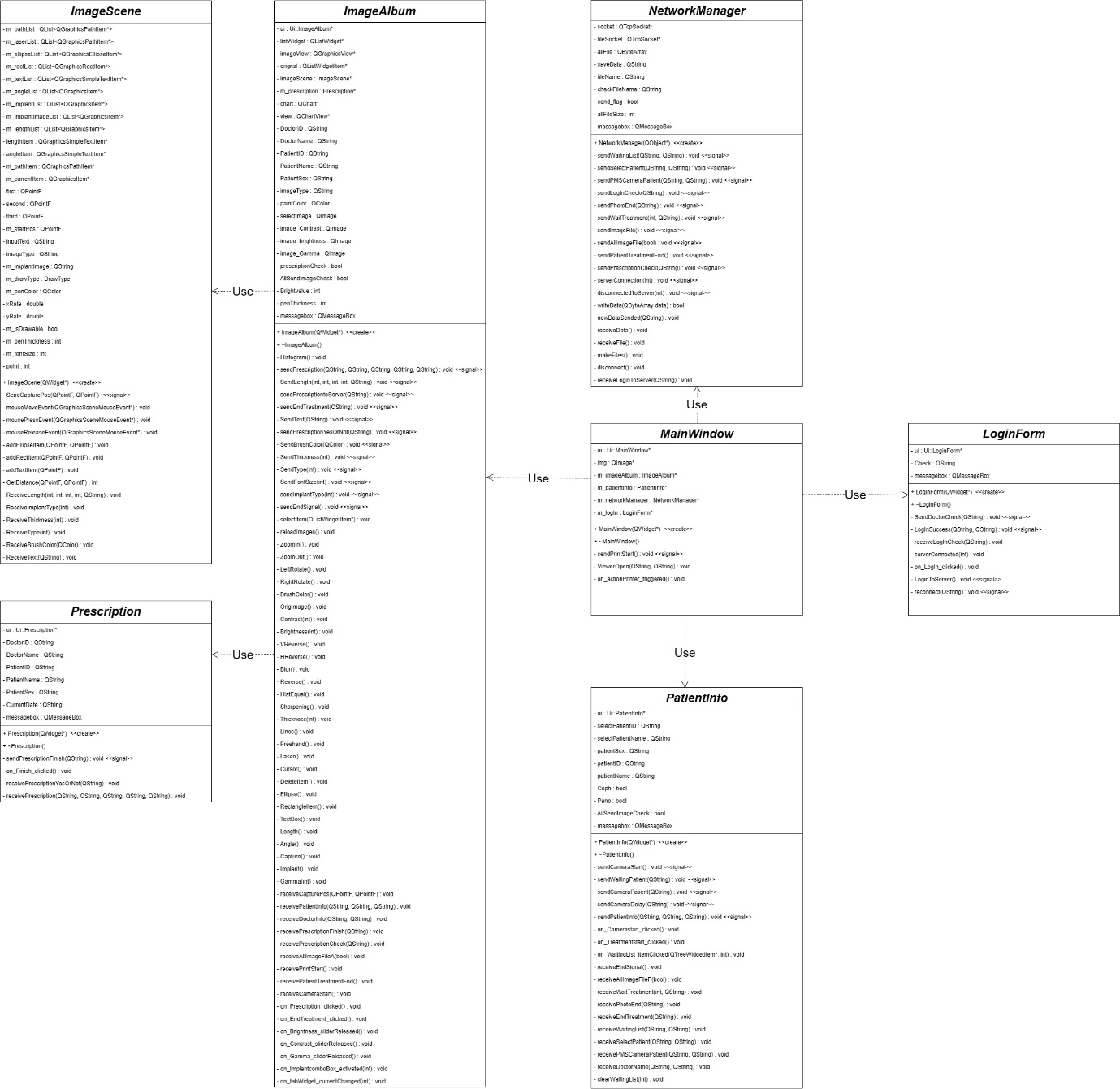
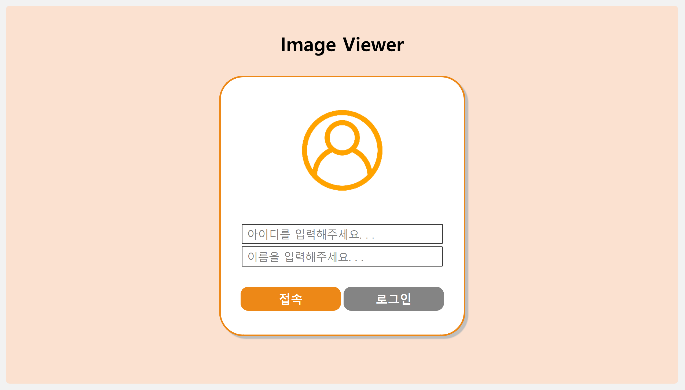
**1. UML 다이어그램**



텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명**2. 영상 뷰어 SW 개요**

- 영상 뷰어SW는 치과에서 진료를 위해 촬영한 영상을 통해 의사가 직접 진단하고 치료 계획을 수립하는데 도움을 주고자 하는 목적을 가지고 있다. 실 사용자는 의사이며, 환자에게 촬영 영상을 직접 보여주며 치아 상태에 대한 설명 및 치료에 대한 구체적인 계획을 수립하는데 도움을 주는 프로그램이다.

해당 SW의 전체적인 흐름도를 살펴보면, 대기 중인 환자 선택 - 촬영 종류 선택 및 촬영 요청 - 촬영된 영상 분석 및 치료 계획 진단 - 처방전 작성 후 환자 관리SW에 진료 종료된 환자 전달 순으로 진행된다. 사용자가 최대한 쉽고 간편하게 사용할 수 있도록 직관적으로 UI를 구성했고, 병원 내 진료 프로세스의 효율성을 위해 진료과정 전체에 대한 체계적인 All-In-One 서비스를 제공하기 위해 타 SW간의 연동 과정에 중점을 두었다.

**3. 전체적인 구성도**

- 전체적인 UI 구성도를 살펴보면 크게 MainWindow 안에서 StackedWidget을 통해 로그인 페이지와 영상 뷰어 페이지를 구분하여 구성하였다.

**2-1. 로그인 페이지**

로그인 페이지에서는 담당 의사가 아이디와 이름을 통해 로그인하는 페이지를 뜻한다.

처음 프로그램 시작 시 로그인 페이지가 보여지고, 접속 버튼을 눌러 서버와 접속 후 담당의사의 아이디와 이름을 입력하여 로그인 버튼 클릭 시 서버에서 해당 담당의사의 정보가 맞는지 확인 절차를 거친다.

접속 버튼과 로그인 버튼을 분리하여 구성한 것은 서버와 접속이 갑작스레 끊어졌을 경우 프로그램이 종료되지 않고 로그인 페이지로 돌아와 서버에 재접속을 할 수 있도록 하기 위함이다.

**2-2. 영상 뷰어 페이지**

영상 뷰어 페이지에서는 환자 클래스와 영상 처리 클래스로 구분하여 구성하였다.

이미지의 타입은 gray-scale 8bit로 입력되고 출력되는 방식이고, 프로그램을 시작하면 listWidget에서 이미지를 선택하기 전에는 이미지 효과 및 그리기를 적용할 수 없도록 구성하였다.

1. 담당의사 : 우선 환자 클래스부터 살펴보면 상단에 담당의사의 이름이 보여지는데, 이는 앞서 Log-in 페이지에서 로그인 성공 시 입력했던 담당의사의 이름이 출력되도록 구성하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 대기 환자 리스트 : 대기 환자 리스트는 TreeWidget으로 구성하였고 PMS로부터 받아온 현재 대기중인 환자의 리스트를 띄워 보여준다. PMS에서 환자를 대기 명단에 추가 시 서버를 통해 “AWL”이라는 Event를 받아와 해당 리스트에서도 바로 해당 환자가 추가되도록 구성하였다.

테이블이(가) 표시된 사진

자동 생성된 설명

1. 진료 시작 버튼 : 대기 환자 리스트에서 환자 중 1명을 선택한 후 진료 시작 버튼을 클릭하면 “VTS”라는 Event로 해당 환자 정보를 서버로 보내줌과 동시에 서버로부터 해당 환자 정보를 받아와 환자 정보 TableWidget에 띄워주었다.

진료 시작 버튼을 누르게 되면 촬영 의뢰 버튼 활성화, 진료 종료 버튼 활성화, 처방전 작성 버튼 활성화, 진료 시작 버튼 비활성화 처리를 해주었다.

해당 부분에서 이미 진료중인 환자가 있을 때 진료 시작 버튼을 누를 경우, 이미 진료중인 환자가 있다는 경고 메시지와 함께 처리되지 않도록 예외처리 해주었고, 이미 촬영중인 환자를 선택 후 진료 시작 버튼을 누를 경우엔 해당 환자는 촬영 중이라는 경고 메시지와 함께 예외처리를 해주었다.

1. 환자 정보 : 환자 정보는 TableWidget으로 구성하였고 진료 시작 버튼 클릭 시 해당하는 환자 정보를 해당 테이블에 띄워준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 촬영 의뢰 버튼 : 촬영 의뢰 버튼 클릭 시 Ceph과 Pano의 체크 상태를 확인 후 촬영 종류가 Ceph인지, Pano인지, Both인지를 함께 촬영 SW에 요청한다.

그러나 촬영 SW가 동작 중이지 않을 경우 계속 기다리거나, 다음 환자를 먼저 진료할 것인지 선택할 수 있도록 구성하였다. 계속 기다리는 경우 다음 동작을 수행하지 않고 기다리는 상태가 되고, 다음 환자를 먼저 진료하게 되는 경우 대기 환자 리스트에서 해당 환자의 진행 상황이 진료 중에서 진료 대기로 변경되고 진료 시작 버튼이 활성화되며 다시 진료 시작을 할 수 있는 상태로 변경된다.

해당 버튼을 누르게 되면 진료 시작 버튼만 활성화, 나머지 버튼 다 비활성화 처리해주었다.

1. listWidget 및 GraphicsView : 진료 시작 버튼 클릭 시 서버의 스토리지에 저장되어 있는 선택한 환자에 해당하는 이미지 파일들을 서버로부터 받아와 listWidget에 띄워지도록 구성하였다.

텍스트, 사람, 실내, 머리카락이(가) 표시된 사진

자동 생성된 설명

촬영 의뢰를 요청하거나 진료 종료를 누르게 되면 해당 환자의 이미지 파일은 삭제되고 다시 진료 시작을 눌렀을 경우에 해당 환자의 이미지 파일이 생성된다.

1. 이미지 정보 및 촬영 날짜 : 이미지 파일명은 CEPH\_yyyymmdd.jpg 형태로 촬영 SW로부터 수신되기 때문에 파일명에서 이미지 정보와 촬영 날짜를 가져와 lineEdit을 이용해 띄워준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. **효과 tab**



효과와 그리기 모든 기능은 listWidget에서 이미지가 선택되어 있을 때 작동한다.

- **reset** : listWidget에서 더블 클릭해서 GraphicsView에 이미지가 띄워질 당시의 이미지를 original로 저장해둔 후 해당 이미지를 띄워준다.

reset시에는 이미지 효과 뿐만 아니라 각종 도형이나 선도 모두 삭제되고 초기화된다.

- **vertical reverse** : GraphicsView를 scale 함수를 활용하여 vertical을 기준으로 이미지를 반전시키게 된다. Scale 함수의 인자를 (-1, 1)로 설정하여 x방향으로 이미지가 반전되게 된다.

- **horizontal reverse** : GraphicsView를 scale 함수를 활용하여 horizontal을 기준으로 이미지를 반전시키게 된다. Scale 함수의 인자를 (1, -1)로 설정하여 y방향으로 이미지가 반전되게 된다.

- **Capture** : mouse Event를 통해 첫 좌표와 마지막 좌표까지 드래그한 영역의 이미지를 자른 후 GraphicsView에 띄워준다. 해당 영역에서 효과나 그리기 적용도 가능하다.

- **Zoom-in, Zoom-out** : GraphicsView를 scale 함수를 활용하여 x1.25배 확대, x0.8배 축소하는 기능이다.

- **Laser** : GraphicsPathItem을 통해 Dot-Line 형태의 선 그리기를 구현하였고 해당 기능은 GraphicsView에 기록되거나 저장되지 않고 다른 동작 수행 시 바로 삭제된다.

- **Left Rotate, Right Rotate** : GraphicsView를 rotate 함수를 통해 왼쪽으로 회전, 오른쪽으로 회전하는 기능이다.

- **Sharpen** : Sharpen 기능은 영상의 경계선, 세밀한 부분을 더욱 강조해서 위해 OpenCV를 활용하여 구현하였다. Mat 클래스를 통해 입력 이미지를 해당 이미지 사이즈의 행렬로 표현 후 GaussianBlur 함수를 통해 블러링된 이미지를 생성하는데 이 이미지는 주변 픽셀 간 급격한 변화, 즉 엣지를 억지 시키게 된다.

따라서 원본 이미지에서 블러링된 이미지를 빼면 나오는 값이 엣지가 되는데 이 값을 원본 이미지에 더해줌으로써 샤프닝을 구현하였다. 여기서 엣지 값에 α라는 가중치를 곱해서 샤프닝의 정도를 조절할 수 있는데 해당 프로젝트에서는 가중치를 1.0으로 설정하였다.

|  |  |
| --- | --- |
| 원본 | α = 0.5 |
| 사람, 실내, 머리카락, 흐림이(가) 표시된 사진  자동 생성된 설명 |  |
| α = 1.0 | α = 1.5 |
| 선화이(가) 표시된 사진  자동 생성된 설명 | 텍스트이(가) 표시된 사진  자동 생성된 설명 |

- **Histogram Equalization** : Histogram Equalization 기능은 선택한 이미지 픽셀의 분포를 고르게 펼쳐 줌으로써 이미지의 대비를 향상시키고자 하는 목적이다. 해당 기능은 OpenCV를 활용하여 구현하였고, Mat 클래스를 통해 입력 이미지를 해당 이미지 사이즈의 행렬로 표현 후 equlizeHist 함수를 활용하여 히스토그램 평활화를 구현하였다.

- **blur** effect : Blur 기능은 선택된 이미지의 픽셀 값을 Mat 클래스를 통해 행렬로 표현 후 Blur 함수를 통해 구현하였다. 여기서 Blur의 커널 사이즈는 5X5로 설정해주었다.

- **negative effect** : 이미지 반전 기능은 선택된 이미지의 픽셀 값을 Mat 클래스를 통해 행렬로 표현 후 픽셀의 최댓값인 255에서 빼서 어두운 영역은 밝게, 밝은 영역은 어둡게 변경되도록 구현하였다.

- **밝기(Brightness)** : 밝기 기능은 선택된 이미지의 픽셀 값에 슬라이더를 통해 더하거나 빼 줌으로써 이미지의 밝기를 조절할 수 있도록 구현하였다.

- **감마(Gamma)** : 감마 기능은 선택된 이미지의 픽셀 값에 슬라이더를 통해 γ(감마) 값을 조절하여 이미지 픽셀 값을 비선형적으로 밝기를 보정할 수 있도록 구현하였다.

<식 4.1> Gamma correction

γ > 1이면 어두운 픽셀의 변화폭은 커서 밝은 픽셀이 급격하게 밝아지지 않고 유지되면서, 어두웠던 픽셀만 잘 보이게 되고, 0 < γ < 1이면 밝은 픽셀의 변화폭이 커서 어두운 픽셀이 급격하게 어두워지지 않은 채로, 밝은 픽셀만 잘 보이게 된다.

위의 수식을 활용하여 입력된 픽셀 값과 매핑된 출력 픽셀 값 사이의 비선형 변환을 사용하여 보정하는 방식으로 구현하였다.

- **대비(Contrast)** : 대비 기능은 선택된 이미지의 픽셀 값에 슬라이더를 통해 0.0~1.5까지의 값을 곱해주어 밝은 값은 더 밝게, 어두운 값은 더 어둡게 구현하였다. 여기서 값을 곱할 때 입력 이미지에 α값을 바로 곱하는 방식이 아닌 밝은 부분은 양수, 어두운 부분은 음수를 곱한 후 원본 이미지 값에 더해주어 Contrast를 구현하였다.

1. 그리기 tab

텍스트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

- **Cursor** : 커서 모드는 삽입된 도형이나 텍스트를 이동할 수 있는 기능이다. 해당 모드가 되면 모든 도형 및 텍스트가 이동할 수 있고 다른 모드 선택 시 이동할 수 없도록 설정하였다.

- **Lines** : 곡선 모드는 GraphicsPathItem을 이용해 자유로운 곡선을 그릴 수 있는 기능이다. MouseMoveEvent에 따라 마우스가 이동되는 경로에 곡선이 그려지게 된다.

- **FreeHand** : 폐곡선 모드는 GraphicsPathItem을 이용해 폐곡선을 그릴 수 있는 기능이다.

MousePressEvent 시에 처음 좌표를 저장해둔 후 MouseReleaseEvent 시에 처음 좌표와 연결되는 곡선이 그려져 닫힌 곡선을 그릴 수 있도록 구현하였다.

- **Ellipse** : 원형 모드는 GraphicsEllipseItem을 이용해 타원을 그릴 수 있는 기능이다.

MousePressEvent 시에 처음 좌표를 저장해둔 후 MouseMoveEvent를 이용해 원을 그릴 영역을 드래그하여 설정하면 MouseReleaseEvent에 마지막 좌표를 저장하여 top-left, bottom-right에 해당하는 영역에 원이 그려지도록 구현하였다.

- **Rectangle** : 사각형 모드는 GraphicsEllipseItem을 이용해 사각형을 그릴 수 있는 기능이다.

MousePressEvent 시에 처음 좌표를 저장해둔 후 MouseMoveEvent를 이용해 사각형을 그릴 영역을 드래그하여 설정하면 MouseReleaseEvent에 마지막 좌표를 저장하여 top-left, bottom-right에 해당하는 영역에 사각형이 그려지도록 구현하였다.

원형 모드와 함께 사각형 모드 시 좌->우, 우->좌의 방향에 상관없이 그려질 수 있도록 normalized() 함수를 통해 넓이가 음수이면 왼쪽과 오른쪽 모서리를 바꿔주고, 높이가 음수이면 위쪽과 아래쪽 모서리를 바꾸어 그려지도록 설정해주었다.

- **Text** : 텍스트 모드는 GraphicsSimpleTextItem을 이용해 원하는 텍스트를 삽입할 수 있는 기능이다. lineEdit에 원하는 텍스트 입력 후 입력 버튼을 누르면 원하는 위치를 선택할 수 있게 된다. 이후 원하는 위치에서 마우스 클릭을 하게 되면 입력한 텍스트가 삽입되도록 구현하였다.

- **Delete** : 삭제 모드는 삽입되어 있는 도형이나 텍스트를 삭제할 수 있는 기능이다.

삭제를 원하는 도형을 클릭하고 삭제 버튼을 누르게 되면 해당 도형이 삭제된다. Ctrl 버튼을 누른 후 다중 도형을 선택하고 삭제 버튼을 누르게 되면 여러 개의 도형이 삭제된다.

- **Length** : 길이 측정 모드는 2개의 좌표 사이의 거리를 통해 실제 길이를 측정할 수 있는 기능이다.

MousePressEvent를 통해 A좌표와 B좌표를 찍으면 GraphicsScene 위에서의 좌표를 구하고, GraphicsScene의 실제 높이 및 넓이 길이와 scaled된 높이 및 길이를 구하여 x의 비율, y의 비율을 구함으로써 픽셀에 따른 길이를 측정한다.

이후 A좌표와 B좌표 사이의 길이를 구할 때 해당 비율을 곱하여 구한 후 픽셀의 길이와 확대율의 관계를 이용해 Ceph인지 Pano인지에 따라 다르게 계산해주었다.

실제 CT내의 Detector에 따라 Pano일 때는 1Pixel당 0.096358mm, Ceph일 때는 1Pixel당 0.08733mm의 비율을 곱하여 최종 길이를 측정하였다.

- **Angle** : 각도 측정 모드는 3개의 좌표를 통해 사잇각을 측정할 수 있는 기능이다.

MousePressEvent를 통해 A좌표, B좌표, C좌표를 찍으면 A-B직선과 B-C직선이 이루는 사잇각을 측정하여 출력해준다.

- **Implant** : 임플란트 측정 모드는 3개의 좌표를 통해 임플란트를 식립하는 기능이다.

MousePressEvent를 통해 A좌표, B좌표, C좌표를 찍으면 각각이 first, second, third 점이 된다. 이때 first와 second 사이의 길이를 높이, first-second를 지나는 직선과 third 사이의 수직인 직선의 길이를 넓이로 설정하였다.

First 좌표는 삽입될 임플란트 이미지의 top-left 좌표가 되고 first-second를 지나는 직선과 y축 사이의 각도만큼 회전시켜 임플란트 이미지를 삽입해준다.

이미지를 삽입할 때 QRectF를 활용하여 top-left, width, height를 통해 사각형의 영역을 구하고 그 영역안에 임플란트 이미지를 삽입해주는데 이 경우엔 임플란트 이미지가 top-left 좌표를 기준으로 항상 오른쪽으로 그려지게 된다.

따라서 이를 해결하기 위해 first-second 좌표를 이은 직선에 third 좌표를 y방향으로 이어 vertical\_point라는 좌표를 구해주었다. first의 x좌표가 second의 x좌표보다 클 때, y좌표를 기준으로 vertical\_point 좌표가 third 좌표보다 크면 이미지를 왼쪽으로 반전시켜주었고 반대로 first의 x좌표가 second의 x좌표보다 작을 때, y좌표를 기준으로 vertical point 좌표가 third 좌표보다 작으면 이미지를 왼쪽으로 반전시켜주었다.

이렇게 구현함으로써 first, second 좌표를 찍고 해당 점들을 지나는 직선 기준으로 third 좌표가 왼쪽, 오른쪽 관계없이 해당 방향으로 임플란트가 삽입되게 된다.

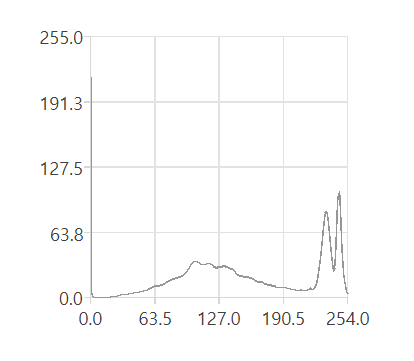
- **컬러 선택 및 크기 선택** : 컬러 선택 기능은 ColorDialog를 통해 원하는 색상을 선택하여 변경할 수 있는 기능이다. 색상 변경을 원하는 도형이나 텍스트를 클릭 후 ColorDialog에서 색상을 선택하면 해당 도형 및 텍스트의 색상이 변경된다.

크기 변경도 마찬가지로 원하는 도형이나 텍스트를 클릭 후 spinBox에서 원하는 사이즈를 입력하거나 버튼을 통해 조절하게 되면 해당 도형 및 텍스트의 크기(두께)가 변경된다.

Ctrl 버튼을 누른 후 다중 도형을 선택하고 컬러 변경 및 크기 변경 시 여러 개의 도형이 색상과 크기가 변경된다.

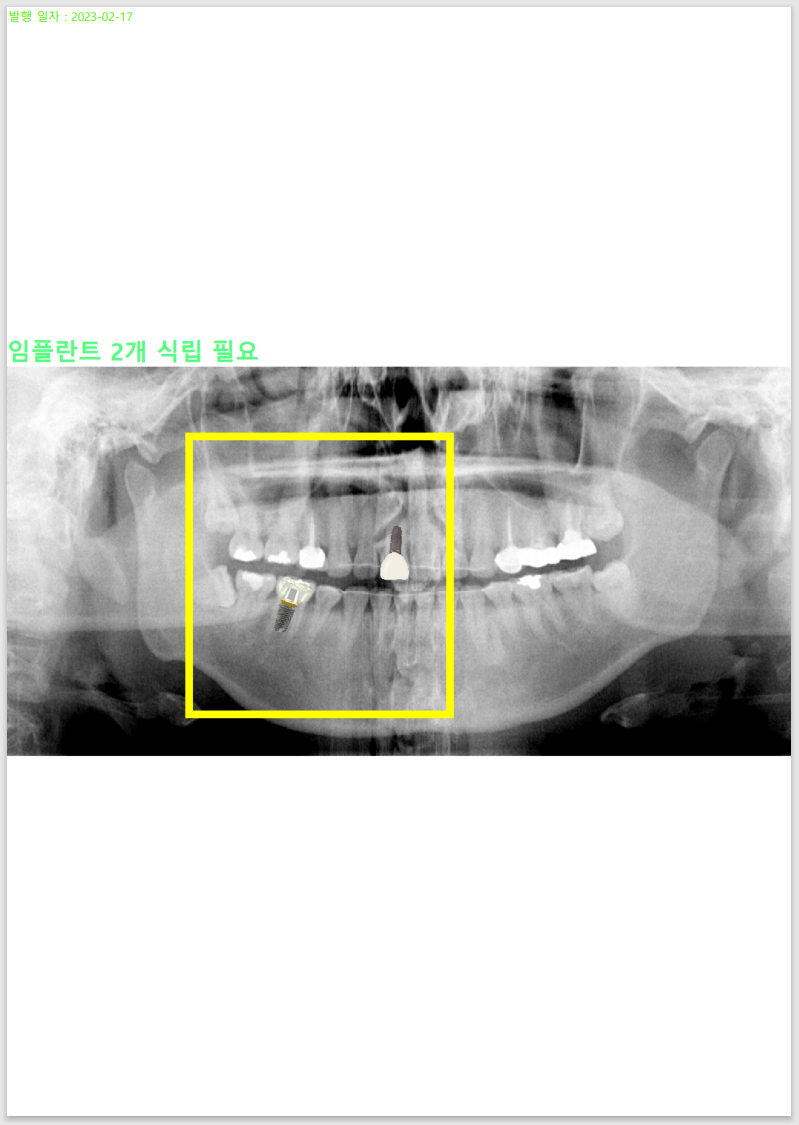
1. **Histogram Chart**

- 히스토그램 차트 기능은 OpenCV의 calcHist() 함수를 사용하여 영상 내에서 각 gray-scale 값에 해당하는 픽셀의 개수를 차트의 형태로 나타내기 위한 것이다. 선택된 이미지의 픽셀 값이 변경될 때마다 해당 함수를 호출하여 이미지 픽셀의 분포도를 확인할 수 있도록 구현하였다.



1. **프린터 버튼**

- ㅇㄹㅇ

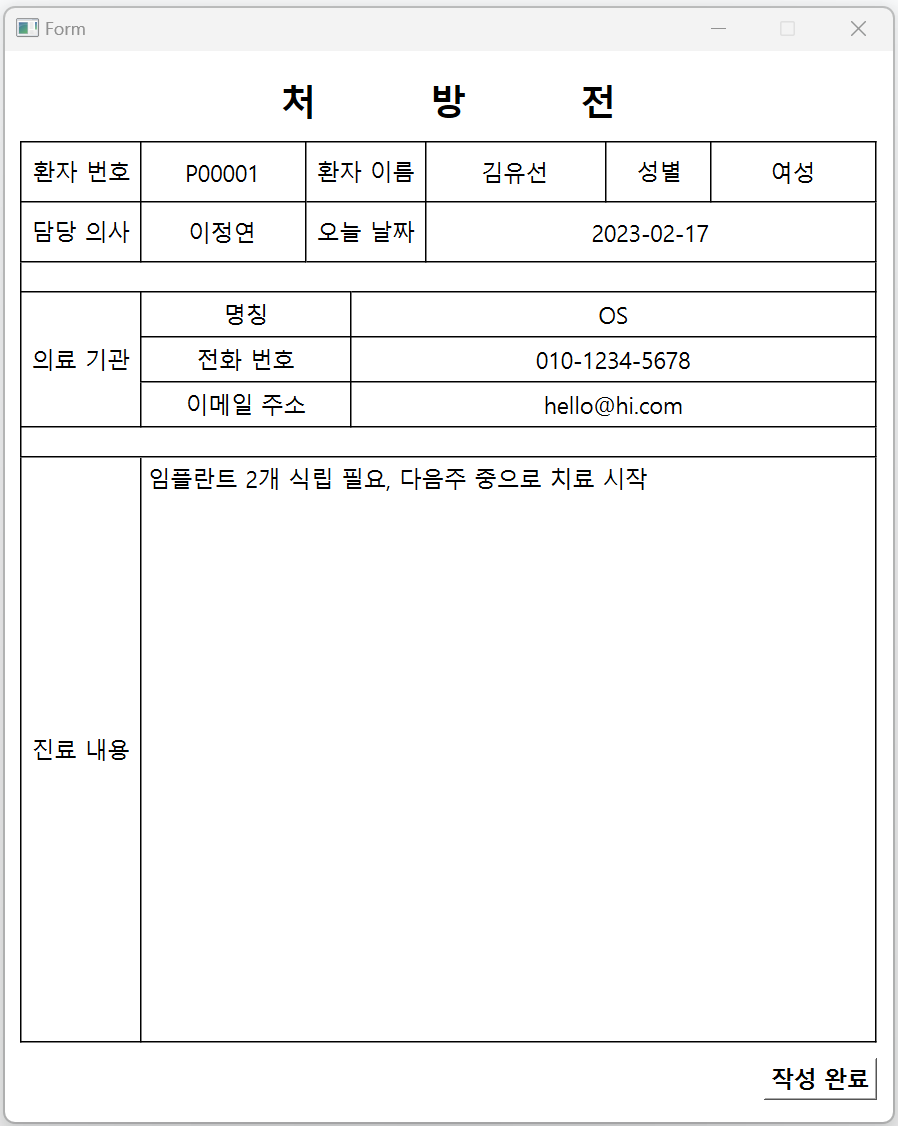


1. **처방전 작성 버튼** : 처방전 작성 버튼 클릭하면 진료 시작 버튼 클릭 시 서버에서 받아온 환자 정보인 환자 번호, 환자 이름, 성별 란에 입력해준다.

담당의사 번호는 로그인 시 입력했던 담당의 ID로 입력해주었고 의료기관 정보는

진료 내용을 제외한 모든 데이터는 읽기 모드로 설정해주었고, 수정되지 않는다.

처방전 작성을 완료하고 작성 완료 버튼 클릭 시 서버로 해당 처방전 데이터가 전송되고 창이 닫히게 된다.



1. **진료 종료 버튼** : 진료 종료 버튼 클릭 시 대기 환자 리스트에서 해당 환자가 삭제되고, 환자 정보란도 비워지게 된다. 또한 GraphicsView, 이미지 정보 및 촬영 일자, 해당 환자의 이미지 파일 등등 모두 삭제된다.

서버에는 해당 환자의 정보를 “VTF”라는 Event로 보내주어서 환자의 진료가 종료되었다는 것을 알려준다.