

# 배터리팩 품질 관리 프로그램

C# 프로그래밍

안위재 이유탁 신선호 신중하

# 목차

- 1) 프로젝트 개요
- 2) 팀 구성 및 역할
- 3) 프로젝트 수행 절차
- 4) 프로젝트 수행 결과
- 5) 자체 평가

# 1) 프로젝트 개요

## 전자부품(배터리팩)의 품질관리

- **csv**파일 형태의 빅 데이터를 활용하는 프로그램
- 공정에 대한 분석을 통해 배터리팩의 품질관리를 강화
- 배터리팩 품질에 영향을 미치는 용접 설비를 대상
- 용접 순서, 날짜, 출력 등의 기준으로 양품과 불량품을 파악

# 1) 프로젝트 개요

## 활용 데이터

제조AI데이터셋

### 전자부품(배터리팩) 예지보전 AI 데이터셋

전기차용 배터리모듈 용접 불량 분석을 위한 레이저 용접기 데이터

↓ 제조AI데이터셋

↓ 가이드북

업종	전자부품	유형	CSV
목적	예지보전	제조데이터 등록일	2022.12.23
사용조건	콘텐츠 변경허용	최종 수정일자	2023.08.14
제공기관	스마트제조혁신추진단 (수행기관 : ㈜인터엑스/네스트필드㈜)		
태그	#전기차, # EV, # 전기차 배터리, # 배터리 용접, # 용접기, # 레이저 용접기, # 예지보전, # N-HITS, # AAS		
내용	적용공정	배터리 용접 공정	
	제조AI데이터셋 소개	전기차용 배터리모듈 용접에 사용되는 용접설비 데이터(용접출력, 작업속도, 출력설정, 작업상태)를 기준으로 설비의 이상 발생 징후를 예측하기 위한 제조 AI분석과정을 담은 데이터셋과 가이드북입니다. 용접설비로부터 운영 데이터를 수집하고, 장시간 높은 예측 정확도와 빠른 계산속도를 가지는 시계열 예측 알고리즘 N-HITS를 학습시켜 설비의 정상/비정상 상태 예측을 도모합니다.	

## 2) 팀 구성 및 역할



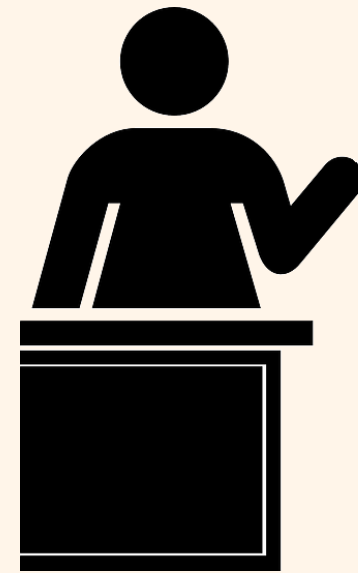
안위재

의견 수렴  
PPT 작성  
코드 작성 및 발표



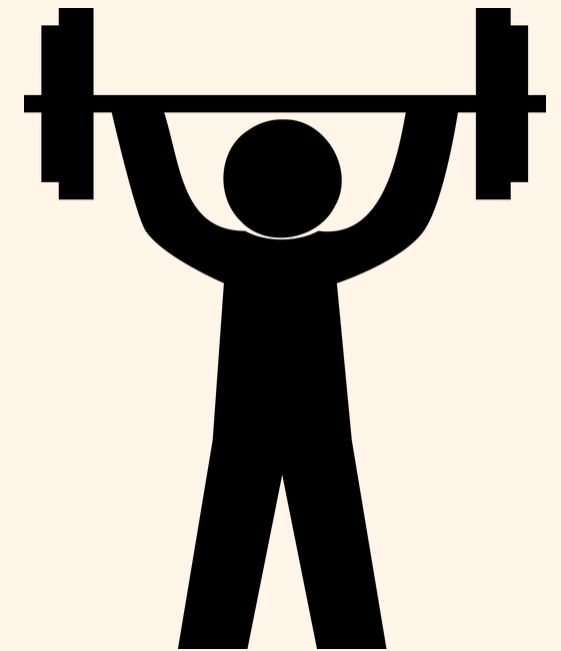
이유탉

코드 작성  
PPT 작성  
오류 수정



신선호

코드 작성  
서류 작성  
오류 수정



신중하

코드 작성  
서류 작성  
오류 수정

# 3) 프로젝트 수행절차

## MSSQL 테이블 데이터 구조

ProductData Table			
논리이름	물리이름	데이터 타입	비고
용접시퀀스	PageNo	NVARCHAR(50)	(Count)
용접속도설정	Speed	NVARCHAR(50)	(mm/s)
용접길이설정	Length	NVARCHAR(50)	(mm)
용접출력	RealPower	NVARCHAR(50)	(W)
발광횟수설정	SetFrequency	NVARCHAR(50)	(Hz)
최대용접출력설정	SetDuty	NVARCHAR(50)	(%)
용접출력설정	SetPower	NVARCHAR(50)	(%)
용접시간	GateOnTime	NVARCHAR(50)	(s)
작업시간	WorkingTime	NVARCHAR(50)	-

# 3) 프로젝트 수행 절차

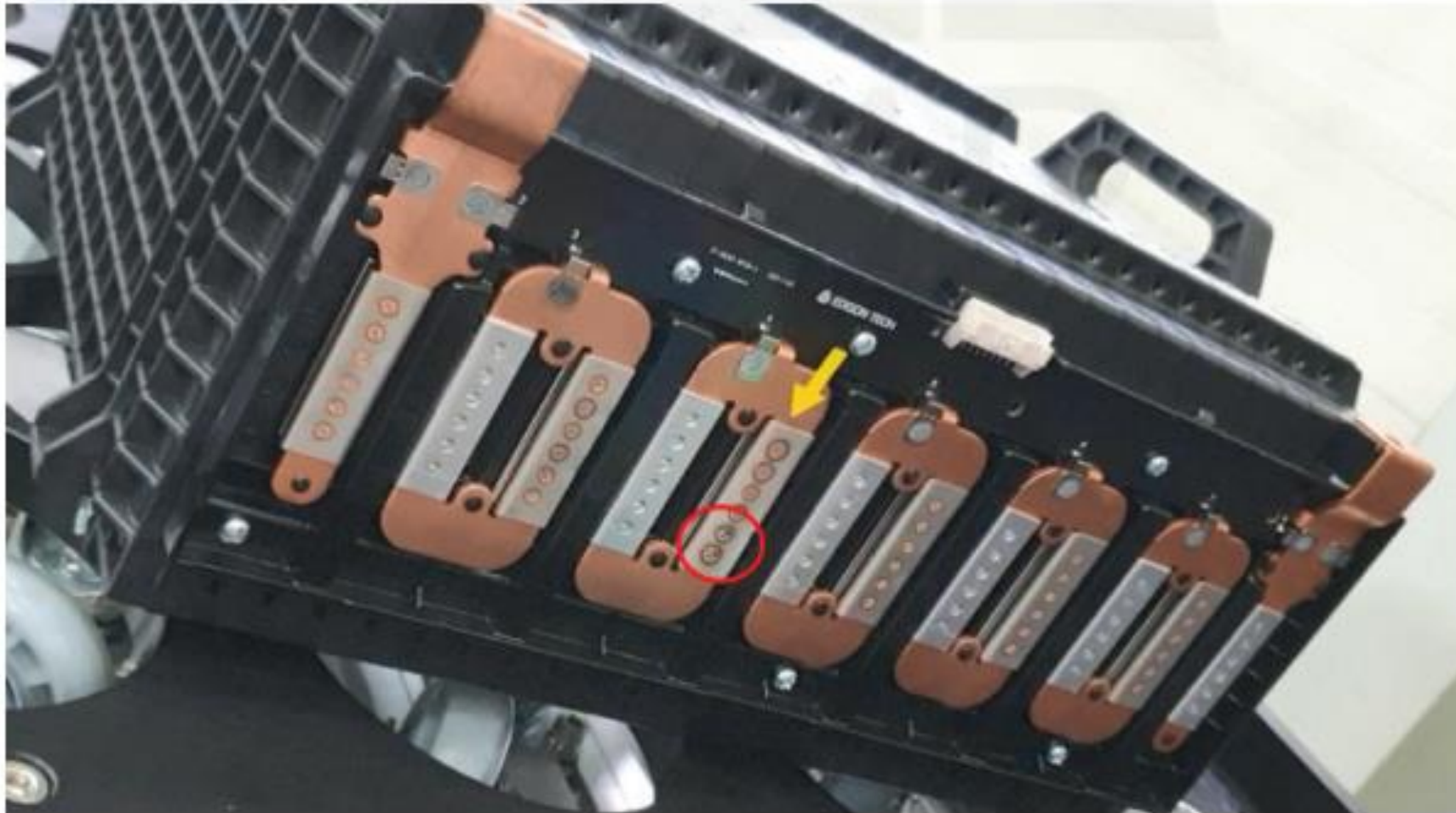
## csv 데이터 유형/구조

	A	B	C	D	E	F	G	H	I	
1	PageNo	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime	WorkingTime	
2	1	250	241.1	1660	1000	100	82	1154	2022-02-11 09:12:28.065	
3	2	250	241.2	1685	1000	100	83	1670	2022-02-11 09:12:31.355	
4	3	250	241.1	1666	1000	100	82	1153	2022-02-11 09:12:34.187	
5	4	250	241.2	1690	1000	100	83	1670	2022-02-11 09:12:37.429	
6	5	250	241.1	1670	1000	100	82	1154	2022-02-11 09:12:40.189	
7	6	250	241.2	1693	1000	100	83	1670	2022-02-11 09:12:43.509	
8	7	250	241.1	1669	1000	100	82	1153	2022-02-11 09:12:49.277	
9	8	250	241.2	1694	1000	100	83	1670	2022-02-11 09:12:52.547	
10	9	250	241.1	1673	1000	100	82	1154	2022-02-11 09:12:55.389	
11	10	250	241.2	1695	1000	100	83	1670	2022-02-11 09:12:58.755	
12	11	250	241.1	1675	1000	100	82	1154	2022-02-11 09:13:01.615	
13	12	250	241.2	1698	1000	100	83	1670	2022-02-11 09:13:04.827	
14	13	30	19.4	682	1000	100	38	650	2022-02-11 09:13:09.307	
15	14	30	19.4	680	1000	100	38	650	2022-02-11 09:13:11.019	
16	15	30	19.4	682	1000	100	38	650	2022-02-11 09:13:13.949	
17	16	30	19.4	680	1000	100	38	650	2022-02-11 09:13:17.335	

# Winform 구성

날짜와 PageNo별로 제조데이터를  
확인합니다.  
불량품과 양품의 데이터 차이를  
PageNo별로 비교할 수 있습니다.  
AI가 불량품과 양품을 구별하는데  
필요한 학습 자료가 됩니다.

품질 관리  
양품과 불량품  
비율로 분석  
있습니다



PageNo는 용접 공정 번호를 뜻합니다.  
저희 자료의 배터리 셀에는 총 39개의 용접  
포인트가 있고,  
PageNo는 총 39번까지 있습니다.

기록한  
! 때  
들을  
;입니다.

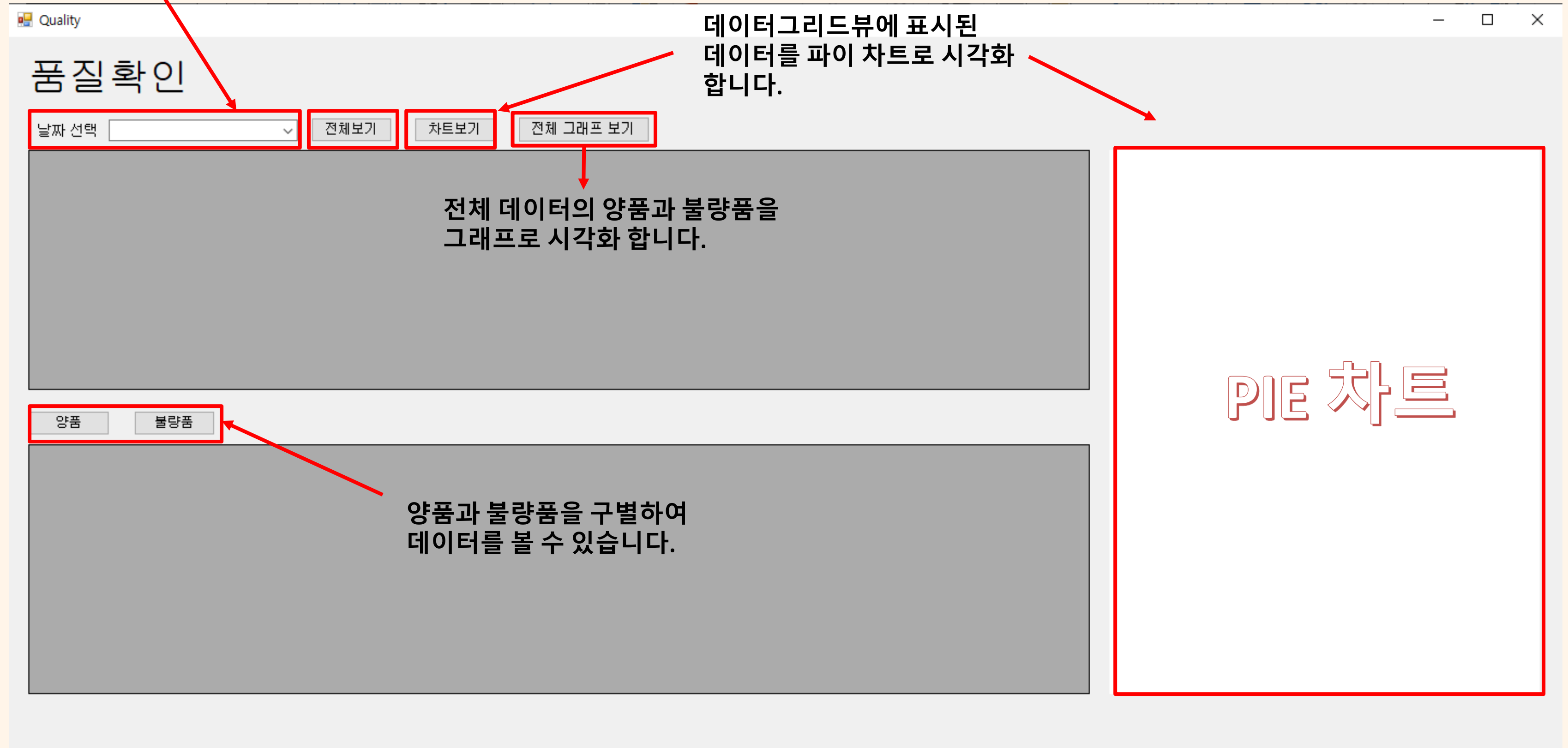
CSV파일을 DB에  
저장합니다.





날짜 선택 후 전체 보기를  
누르면 해당 날짜의  
데이터를 데이터 그리드  
뷰로 불러옵니다

# Winform구성

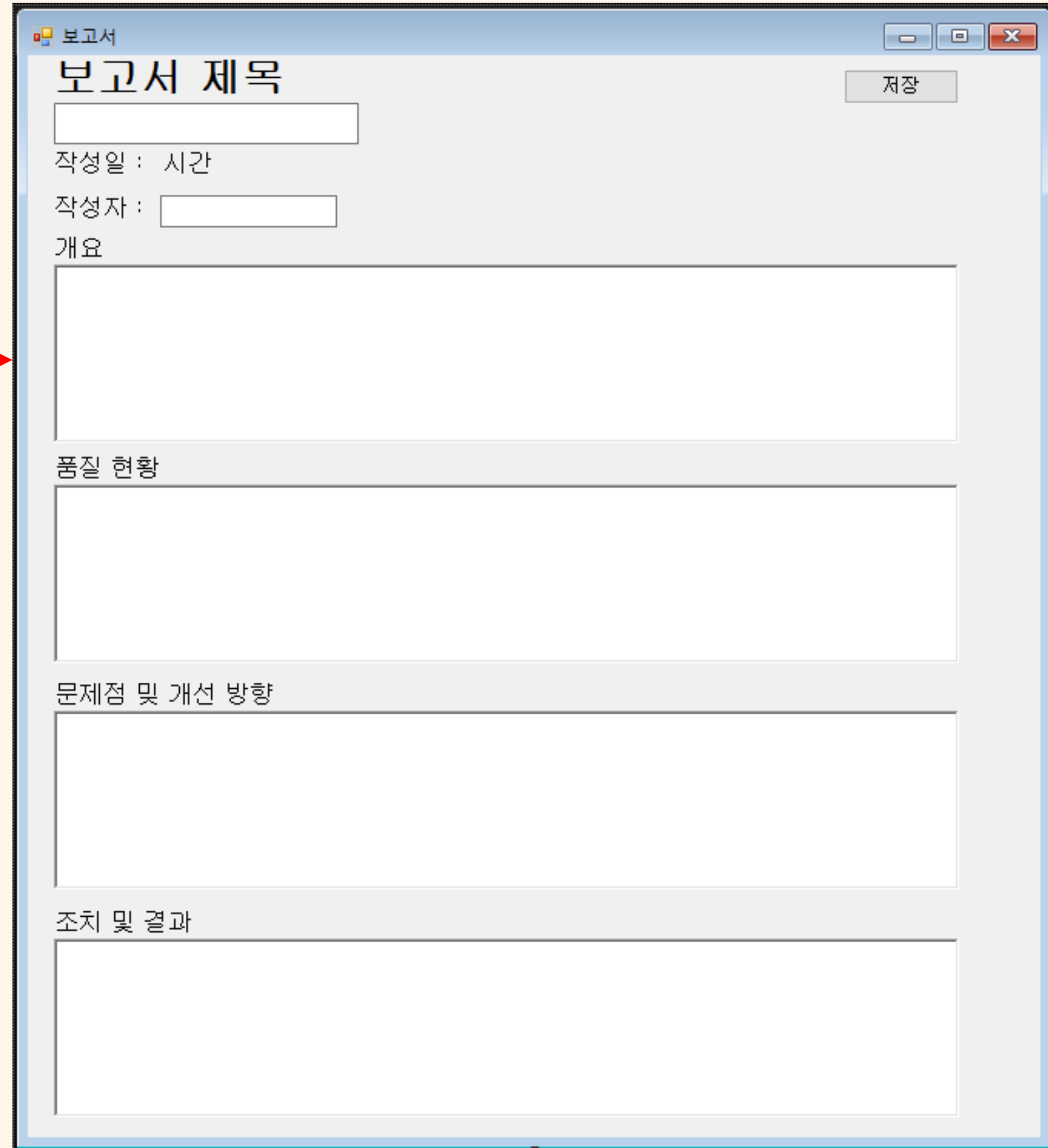


# Winform구성



A Winform interface with two buttons at the top: '양품' (Good) and '불량품' (Defective). Below the buttons is a large gray rectangular area representing a grid view. A red border highlights this grid area, and a red arrow points from its right side towards the report form on the right.

그리드 뷰를 클릭하면 보고서  
폼이 등장합니다. 문제가 있는  
데이터에 대한 보고서를  
작성하고 txt파일로 저장합니다.

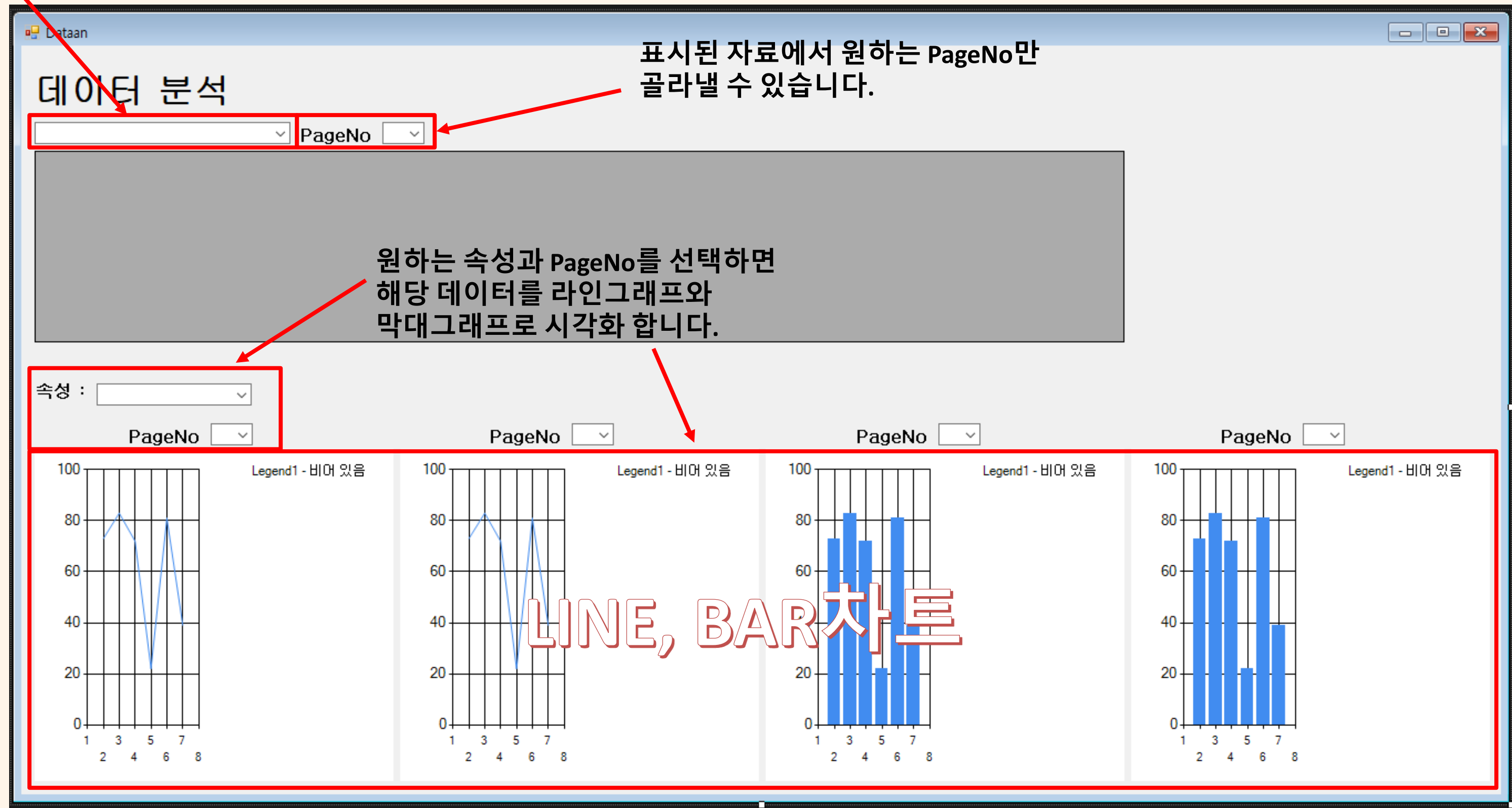


A report form titled '보고서' (Report) with a '저장' (Save) button. The form contains the following fields:

- 보고서 제목 (Report Title):
- 작성일 : 시간 (Date/Time):
- 작성자 : (Author):
- 개요 (Summary):
- 품질 현황 (Quality Status):
- 문제점 및 개선 방향 (Problem points and improvement direction):
- 조치 및 결과 (Action and result):

# Winform구성

원하는 날짜를 선택하면  
해당 날짜의 데이터를  
데이터 뷰로 표시합니다.

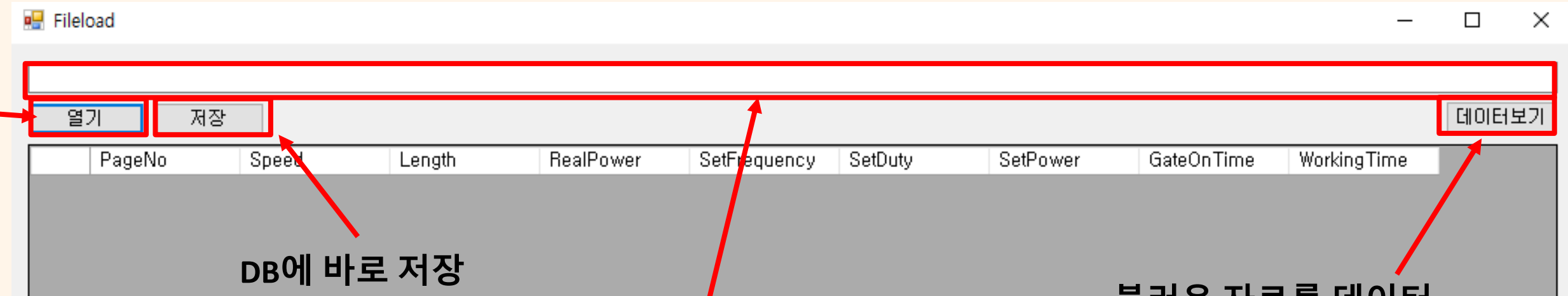


# Winform구성

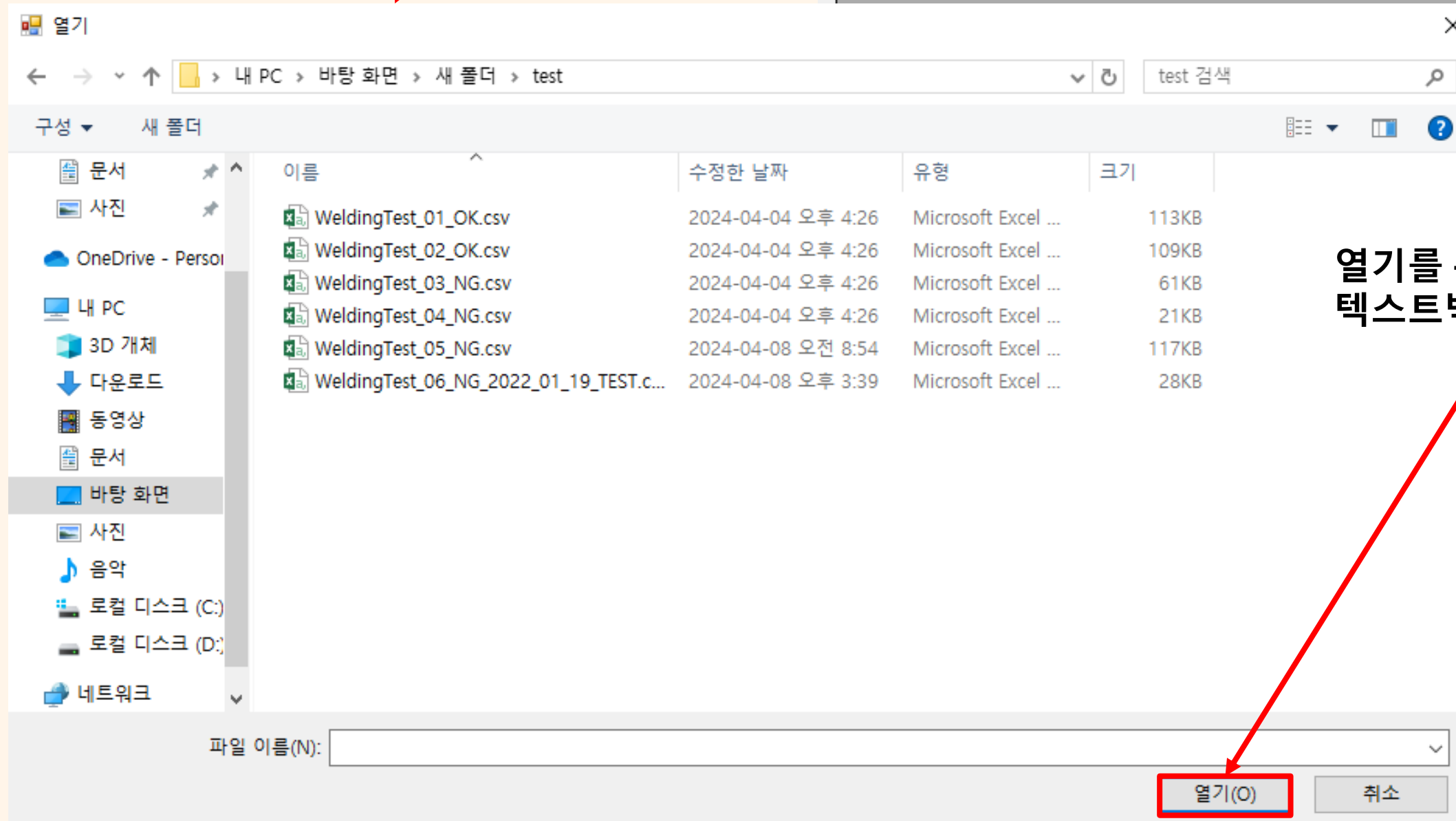


# Winform구성

열기 버튼을 누르면 파일  
탐색기를 열어 원하는  
자료를 찾을 수 있습니다.



불러온 자료를 데이터  
그리드 뷰에 출력합니다.



열기를 누르면 데이터 경로가  
텍스트박스에 입력됩니다.

# 3) 프로젝트 수행 절차

## 코드

```
32 string[] csv = File.ReadAllLines(file);
33 string[] data = csv.Skip(1).ToArray(); // 첫번째 줄 생략
34
35 foreach (string item in data)
36 {
37     string[] value = item.Split(',');
38     sql = "insert into productdata values (@val1, @val2, @val3, @val4, @val5, @val6, @val7, @val8, @val9)";
39     cmd.Parameters.AddWithValue("@val1", value[0]);
40     cmd.Parameters.AddWithValue("@val2", value[1]);
41     cmd.Parameters.AddWithValue("@val3", value[2]);
42     cmd.Parameters.AddWithValue("@val4", value[3]);
43     cmd.Parameters.AddWithValue("@val5", value[4]);
44     cmd.Parameters.AddWithValue("@val6", value[5]);
45     cmd.Parameters.AddWithValue("@val7", value[6]);
46     cmd.Parameters.AddWithValue("@val8", value[7]);
47     // 시간 없애고 날짜만 저장하는 코드
48     string date = "";
49     for (int i = 0; i < value[8].Length - 13; i++)
50     {
51         date += value[8][i];
52     }
53     value[8] = date;
54     cmd.Parameters.AddWithValue("@val9", value[8]);
55
56     cmd.CommandText = sql;
57     cmd.ExecuteNonQuery();
58     cmd.Parameters.Clear();
59 }
60 System.Windows.Forms.MessageBox.Show("저장이 완료되었습니다");
61 }
```

데이터베이스에 날짜를  
저장할때 시간을  
제외하고 저장하기 위해  
문자열을 차례로읽어  
뒷자리를 제외한 후  
데이터 저장

WorkingTime
2022-02-11 09:12:28.065
2022-02-11 09:12:31.355

WorkingTime
2022-02-10
2022-02-10



# 3) 프로젝트 수행 절차

```
public Fileload()
{
    InitializeComponent();
}

//파일 불러오기
참조 1개
private void button1_Click(object sender, EventArgs e)
{
    OpenFileDialog ofd = new OpenFileDialog();
    DialogResult dr = ofd.ShowDialog();
    if (dr == DialogResult.OK)
    {
        string fileFullName = ofd.FileName;
        textBox1.Text = fileFullName;
    }
}

//파일 저장
참조 1개
private void button2_Click(object sender, EventArgs e)
{
    if (textBox1.Text == "")
    {
        MessageBox.Show("파일을 선택해 주세요");
    }
    else
    {
        DBManager db = new DBManager();
        db.DBSave(textBox1.Text);
    }
}
```

열기

마리 볼 파일을 선택하십시오.

```
//불러온 파일의 데이터보기
참조 1개
private void button3_Click(object sender, EventArgs e)
{
    if (textBox1.Text == "")
    {
        MessageBox.Show("파일을 선택해 주세요");
    }
    else
    {
        dataGridView1.Rows.Clear();
        dataGridView1.DataSource = null;
        string[] csv = File.ReadAllLines(textBox1.Text);
        string[] data = csv.Skip(1).ToArray();
        foreach (string item in data)
        {
            string[] value = item.Split(',');
            string date = "";
            for (int i = 0; i < value[8].Length - 13; i++)
            {
                date += value[8][i];
            }
            value[8] = date;
            dataGridView1.Rows.Add(value[0], value[1], value[2], value[3], value[4], value[5], value[6], value[7], value[8]);
        }
    }
}
```

D:\후반\VS C#\제조데이터기반 시각화\ProductDataProject\Dataset\_전자부품(배터리팩) 예지보전 AI 데이터셋\data\raw\_data\test\WeldingTest\_03\_NG.csv

열기 저장 데이터보기

	PageNo	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime	WorkingTime
▶	1	250	241.1	1660	1000	100	82	1154	2022-08-03
	2	250	241.2	1685	1000	100	83	1670	2022-08-03
	3	250	241.1	1666	1000	100	82	1153	2022-08-03
	4	250	241.2	1688	1000	100	83	1670	2022-08-03

csv파일 데이터를 읽어 올때도 마찬가지로 날짜만 보이게 함

# 3) 프로젝트 수행 절차

## 코드

```
56 ConnectDB();
57 SqlCommand cmd = new SqlCommand();
58 cmd.Connection = conn;
59 cmd.CommandText = "select Distinct WorkingTime from ProductData order by WorkingTime";
60 SqlDataReader reader = cmd.ExecuteReader();
61 comboBox1.Items.Add("All"); // comboBox에 All 모두보기 추가
62 while (reader.Read())
63 {
64     comboBox1.Items.Add(reader["WorkingTime"].ToString());
65 }
66
67 }
```

Distinct로 DB에 있는  
중복 데이터들을  
제거한 후  
콤보박스에 추가

전체 데이터를 보기위해  
콤보박스에 All을 추가하고  
나머지 날짜는 DB에서 읽어옴

날짜 선택

All
2022-01-08
2022-01-18
2022-01-19
2022-01-20
2022-02-10
2022-02-11

	workingtime
1	2022-01-08
2	2022-01-08
3	2022-01-08
4	2022-01-08
5	2022-01-08
6	2022-01-08
7	2022-01-08
8	2022-01-08

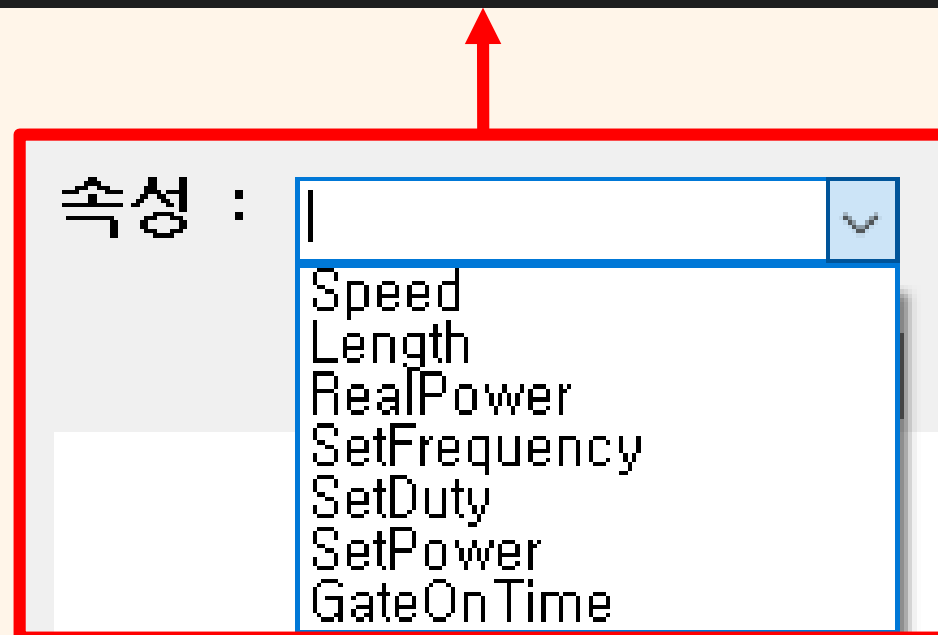
	workingtime
1	2022-01-08
2	2022-01-18
3	2022-01-19
4	2022-01-20
5	2022-02-10
6	2022-02-11
7	2022-02-17
8	2022-02-18



# 3) 프로젝트 수행 절차

## 코드

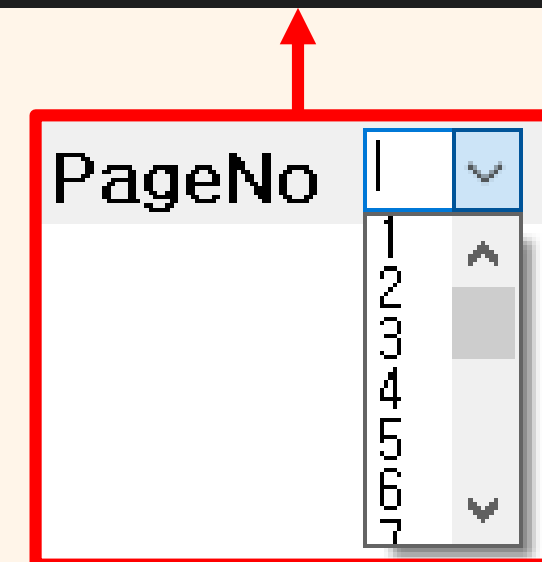
```
/*=====comboBox2
comboBox2.Items.Add("Speed");
comboBox2.Items.Add("Length");
comboBox2.Items.Add("RealPower");
comboBox2.Items.Add("SetFrequency");
comboBox2.Items.Add("SetDuty");
comboBox2.Items.Add("SetPower");
comboBox2.Items.Add("GateOnTime");
```



```
/*=====comboBox3
string[] pagedata = new string[39];

for(int i = 1; i < 40; i++)
{
    pagedata[i - 1] = i.ToString();
}

comboBox3.Items.AddRange(pagedata);
comboBox4.Items.AddRange(pagedata);
comboBox5.Items.AddRange(pagedata);
comboBox6.Items.AddRange(pagedata);
comboBox7.Items.AddRange(pagedata);
}
```



# 3) 프로젝트 수행 절차

```
/*===== 날짜 별 데이터 띄우기 =====*/
참조 1개
private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        ConnectDB();
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = conn;
        SqlParameter data = new SqlParameter("@Wt", comboBox1.Text);
        cmd.Parameters.Add(data);

        // 데이터 모두(All) 보기 추가
        if(comboBox1.Text == "All")
        {
            cmd.CommandText = "SELECT TRIM(PageNo) AS PageNo, TRIM(Speed) AS Speed, TRIM(Length) AS Length, TRIM(RealPower) AS RealPower, TRIM(SetFrequency) AS SetFrequency,"
                               + " TRIM(SetDuty) AS SetDuty, TRIM(SetPower) AS SetPower, TRIM(GateOnTime) AS GateOnTime, TRIM(WorkingTime) AS WorkingTime FROM ProductData order by WorkingTime";
        }
        else
        {
            cmd.CommandText = "SELECT TRIM(PageNo) AS PageNo, TRIM(Speed) AS Speed, TRIM(Length) AS Length, TRIM(RealPower) AS RealPower, TRIM(SetFrequency) AS SetFrequency,"
                               + " TRIM(SetDuty) AS SetDuty, TRIM(SetPower) AS SetPower, TRIM(GateOnTime) AS GateOnTime, TRIM(WorkingTime) AS WorkingTime FROM ProductData"
                               + " WHERE WorkingTime = @Wt";
        }

        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataSet ds = new DataSet();
        da.Fill(ds, "mytest");
        dataGridView1.DataSource = null;
        dataGridView1.DataSource = ds;
        dataGridView1.DataMember = "mytest";
    }
    catch (Exception ex)
    {
        MessageBox.Show("오류 발생: " + ex.Message);
    }
    finally
    {
        conn.Close();
    }
}
```

	PageNo	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime	WorkingTime
▶	1	250	241.1	1688	1000	100	82	1154	2022-01-20
	2	250	241.2	1713	1000	100	83	1670	2022-01-20
	3	250	241.1	1695	1000	100	82	1153	2022-01-20
	4	250	241.2	1717	1000	100	83	1670	2022-01-20
	5	250	241.1	1698	1000	100	82	1154	2022-01-20
	6	250	241.2	1721	1000	100	83	1670	2022-01-20
	7	250	241.1	1695	1000	100	82	1153	2022-01-20

# 3) 프로젝트 수행 절차

```
/*===== 날짜와 페이지별로 자료 띄우기 =====*/
참조 1개
private void comboBox7_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        ConnectDB();
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = conn;
        SqlParameter data = new SqlParameter("@Wt", comboBox1.Text);
        SqlParameter data2 = new SqlParameter("@Pn", comboBox7.Text);
        cmd.Parameters.Add(data);
        cmd.Parameters.Add(data2);

        if (comboBox1.Text == "All")
        {
            cmd.CommandText = "SELECT TRIM(PageNo) AS PageNo, TRIM(Speed) AS Speed, TRIM(Length) AS Length, TRIM(RealPower) AS RealPower, TRIM(SetFrequency) AS SetFrequency,"
                + " TRIM(SetDuty) AS SetDuty, TRIM(SetPower) AS SetPower, TRIM(GateOnTime) AS GateOnTime, TRIM(WorkingTime) AS WorkingTime"
                + " FROM ProductData where PageNo = @Pn order by WorkingTime";
        }
        else
        {
            cmd.CommandText = "SELECT TRIM(PageNo) AS PageNo, TRIM(Speed) AS Speed, TRIM(Length) AS Length, TRIM(RealPower) AS RealPower, TRIM(SetFrequency) AS SetFrequency,"
                + " TRIM(SetDuty) AS SetDuty, TRIM(SetPower) AS SetPower, TRIM(GateOnTime) AS GateOnTime, TRIM(WorkingTime) AS WorkingTime FROM ProductData"
                + " WHERE WorkingTime = @Wt AND PageNo = @Pn";
        }

        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataSet ds = new DataSet();
        da.Fill(ds, "mytest");
        dataGridView1.DataSource = null;
        dataGridView1.DataSource = ds;
        dataGridView1.DataMember = "mytest";
    }
    catch (Exception ex)
    {
        MessageBox.Show("오류 발생: " + ex.Message);
    }
    finally
    {
        conn.Close();
    }
}
```

	PageNo	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime	WorkingTime
▶	12	250	241.2	1723	1000	100	83	1670	2022-01-08
	12	250	241.2	1723	1000	100	83	1670	2022-01-08
	12	250	241.2	1723	1000	100	83	1670	2022-01-08
	12	250	241.2	1723	1000	100	83	1670	2022-01-08
	12	250	241.2	1724	1000	100	83	1670	2022-01-08
	12	250	241.2	1723	1000	100	83	1670	2022-01-08
	12	250	241.2	1723	1000	100	83	1670	2022-01-08

# 3) 프로젝트 수행 절차

```
/*===== 차트 1 =====  
참조 1개  
private void comboBox3_SelectedIndexChanged(object sender, EventArgs e)  
{  
    try  
    {  
        ConnectDB();  
        SqlCommand cmd = new SqlCommand();  
        cmd.Connection = conn;  
  
        // 모두 보기에서 차트 보기 추가  
        if (comboBox1.Text == "All")  
        {  
            cmd.CommandText = $"select {comboBox2.Text} from ProductData where PageNo = '{comboBox3.Text}' order by WorkingTime";  
        }  
        else  
        {  
            cmd.CommandText = $"select {comboBox2.Text} from ProductData where convert(date, WorkingTime) = '{comboBox1.Text}' and PageNo = '{comboBox3.Text}'";  
        }  
        SqlDataAdapter da = new SqlDataAdapter(cmd);  
        DataSet ds = new DataSet();  
        da.Fill(ds, "mytest");  
  
        chart1.Series[0].Points.Clear();  
        string[] ch1;  
        if (ds.Tables.Count > 0)  
        {  
            for (int i = 0; i < ds.Tables[0].Rows.Count; i++)  
            {  
                DataRow dr = ds.Tables[0].Rows[i];  
                ch1 = dr.ItemArray.Select(o => o == null ? string.Empty : o.ToString()).ToArray();  
  
                foreach (var item in ch1)  
                {  
                    chart1.Series[0].Points.AddXY("", item);  
                }  
            }  
        }  
    }  
    catch (Exception ex)  
    {  
        MessageBox.Show("오류 발생: " + ex.Message);  
    }  
    finally  
    {  
        conn.Close();  
    }  
}
```

속성 : RealPower

PageNo 6

WorkingTime (X)	RealPower (Y)
0	1700
20	1700
40	1700
45	1200
48	1000
50	1200
52	1000
54	1200
56	1000
58	1200
60	1700

# 3) 프로젝트 수행 절차

```
// 불량품이 없을 때에는 양품만 보여주는 파이차트 코드
참조 1개
private void Chart_Click(object sender, EventArgs e)
{
    try
    {
        ConnectDB();
        chart1.Series[0].Points.Clear();

        //양품 데이터
        SqlCommand cmdGood = new SqlCommand();
        cmdGood.Connection = conn;

        if (comboBox1.Text == "All")
        {
            cmdGood.CommandText = "SELECT COUNT(*) AS TotalGoodCount FROM ProductData WHERE (RealPower BETWEEN 1650 AND 1750 OR RealPower BETWEEN 670 AND 720)";
        }
        else
        {
            cmdGood.CommandText = "SELECT COUNT(*) AS GoodCount FROM ProductData " +
                "WHERE CONVERT(date, WorkingTime) = @SelectedDate " +
                "AND (RealPower BETWEEN 1650 AND 1750 OR RealPower BETWEEN 670 AND 720)";

            cmdGood.Parameters.AddWithValue("@SelectedDate", comboBox1.SelectedItem.ToString());
        }
        int GoodCount = Convert.ToInt32(cmdGood.ExecuteScalar()); //ToInt32 - 문자형식을 숫자형식으로 변환

        //불량품 데이터
        SqlCommand cmdDefective = new SqlCommand();
        cmdDefective.Connection = conn;

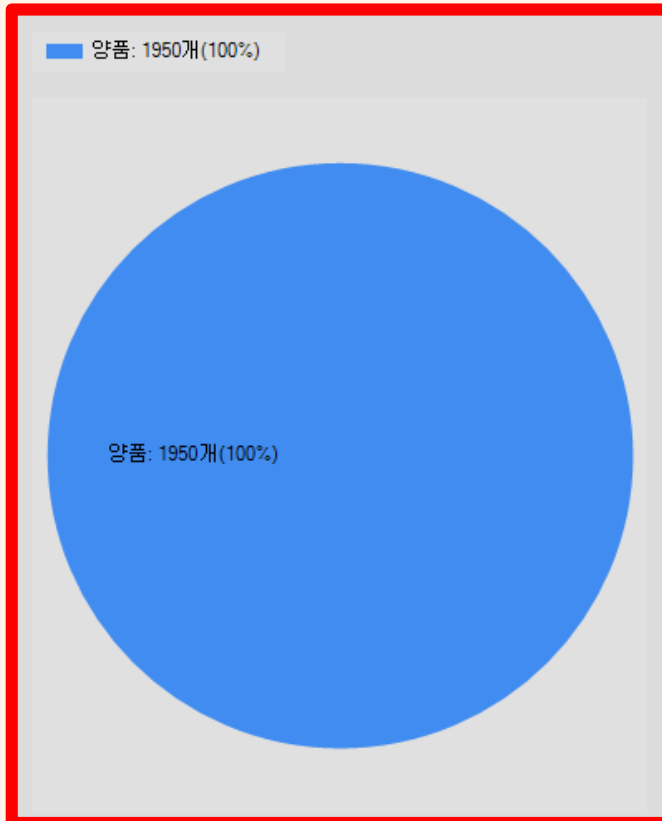
        if (comboBox1.Text == "All")
        {
            cmdDefective.CommandText = "SELECT COUNT(*) AS TotalDefectiveCount FROM ProductData WHERE ([RealPower] < 1650 OR [RealPower] > 1750) AND ([RealPower] < 670 OR [RealPower] > 720)";
        }
        else
        {
            cmdDefective.CommandText = "SELECT COUNT(*) AS DefectiveCount FROM ProductData " +
                "WHERE CONVERT(date, WorkingTime) = @SelectedDate " +
                "AND ([RealPower] < 1650 OR [RealPower] > 1750) AND ([RealPower] < 670 OR [RealPower] > 720)";

            cmdDefective.Parameters.AddWithValue("@SelectedDate", comboBox1.SelectedItem.ToString());
        }
        int DefectiveCount = Convert.ToInt32(cmdDefective.ExecuteScalar());
    }
}
```

RealPower(출력)의  
정상범위와 그렇지  
않은것으로 SQL문 작성

# 3) 프로젝트 수행 절차

불량품이 없는 경우



```
//전체 데이터
SqlCommand cmdTotal = new SqlCommand();
cmdTotal.Connection = conn;

if (comboBox1.Text == "All")
{
    cmdTotal.CommandText = "SELECT COUNT(*) AS TotalAllCount FROM ProductData";
}
else
{
    cmdTotal.CommandText = "SELECT COUNT(*) AS TotalCount FROM ProductData " +
        "WHERE CONVERT(date, WorkingTime) = @SelectedDate";

    cmdTotal.Parameters.AddWithValue("@SelectedDate", comboBox1.SelectedItem.ToString());

    int TotalCount = Convert.ToInt32(cmdTotal.ExecuteScalar());

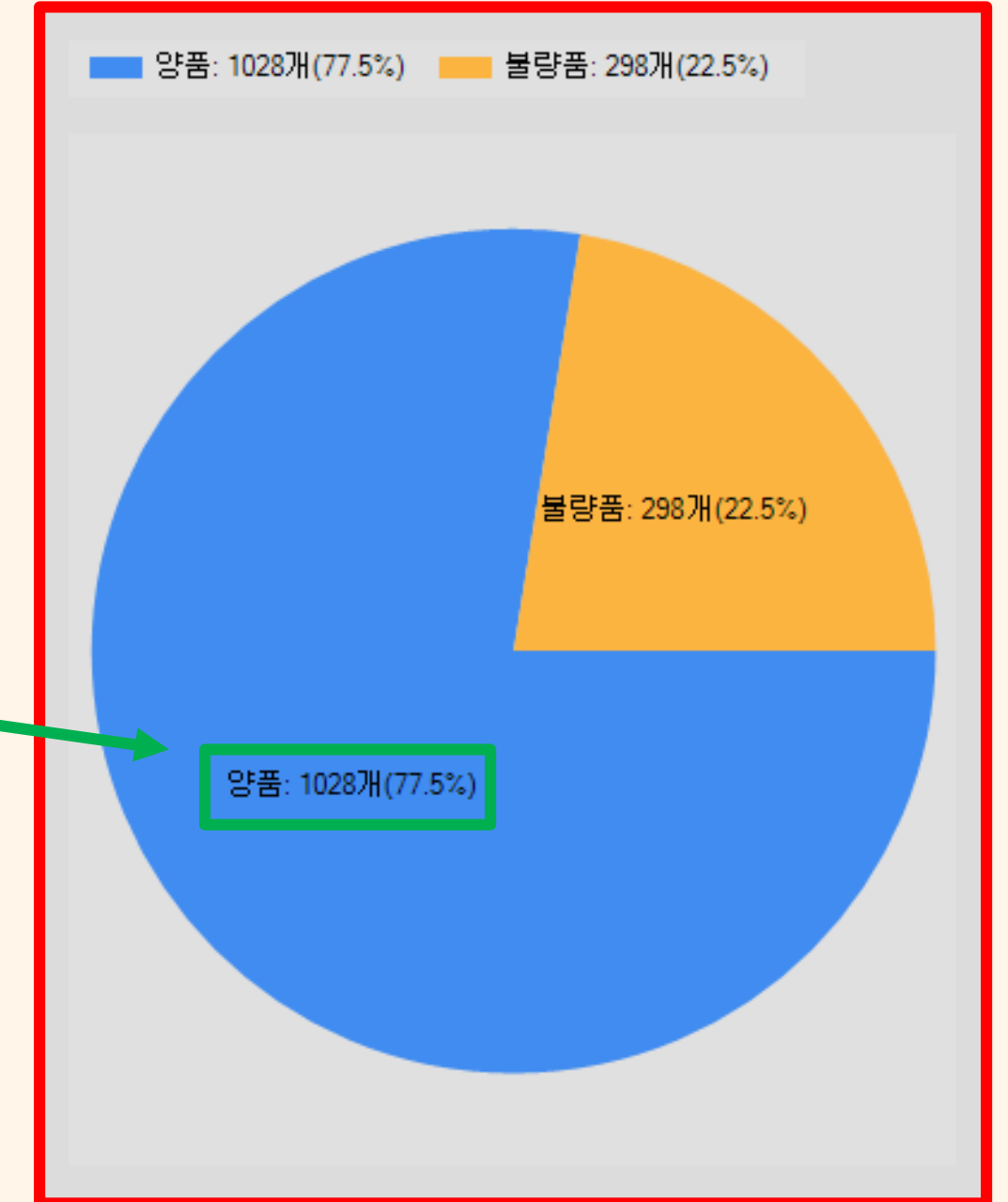
    //양품, 불량품 퍼센트
    double GoodData = ((double)GoodCount / TotalCount) * 100.0;
    GoodData = Math.Round(GoodData, 1);
    double DefectiveData = ((double)DefectiveCount / TotalCount) * 100.0;
    DefectiveData = Math.Round(DefectiveData, 1);

    //양품, 불량품 개수
    int GoodDataCount = Convert.ToInt32(cmdGood.ExecuteScalar());
    int DefectiveDataCount = Convert.ToInt32(cmdDefective.ExecuteScalar());

    chart1.Series[0].Points.AddXY("양품: " + GoodDataCount + "개(" + GoodData + "%)", GoodCount);

    // 불량품이 있는 경우에만 파이차트에 추가
    if (DefectiveCount > 0)
    {
        chart1.Series[0].Points.AddXY("불량품: " + DefectiveDataCount + "개(" + DefectiveData + "%)", DefectiveCount);
    }
}
catch (Exception ex)
{
    MessageBox.Show("날짜를 선택해 주세요.");
}
finally
{
    conn.Close();
}
```

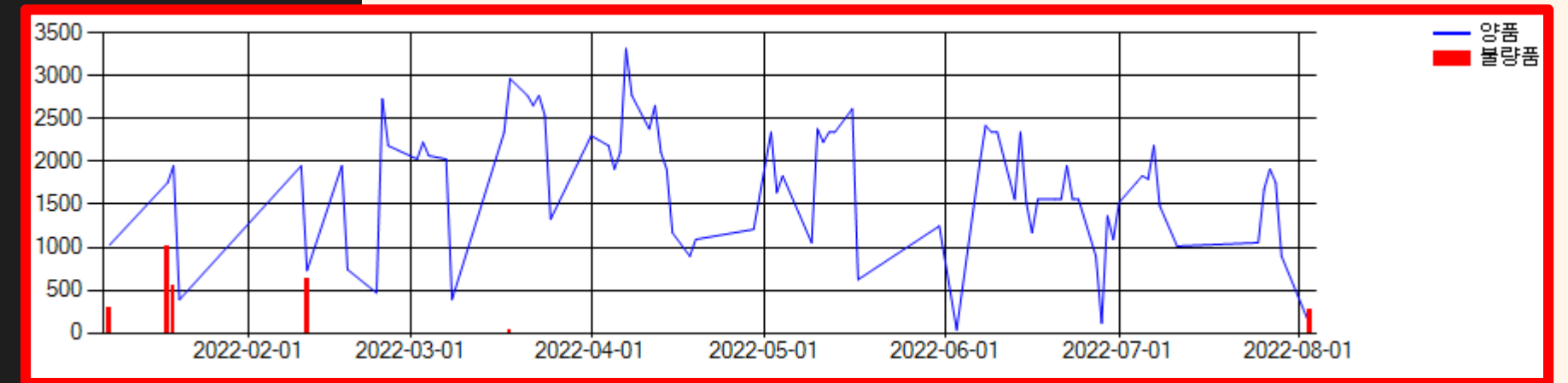
불량품이 있는 경우  
파이차트에 표시되도록 함



양품과 불량품의 비율을  
구하기위해 GoodCount,  
DefectiveCount, TotalCount  
변수를 사용해 계산

# 3) 프로젝트 수행 절차

```
37 try
38 {
39     ConnectDB();
40     SqlCommand cmd = new SqlCommand();
41     cmd.Connection = conn;
42     string sql = "";
43     string sql2 = "";
44     List<string> date = new List<string>();
45
46     sql = "select count(*) as count, workingtime from productdata " +
47         "where(realpower between 1650 and 1750 or realpower between 670 and 720) " +
48         "group by workingtime order by workingtime"; // 양품
49     cmd.CommandText = sql;
50     SqlDataReader dr = cmd.ExecuteReader();
51
52
53     while (dr.Read())
54     {
55         date.Add(dr[1].ToString());
56         chart2.Series[0].Points.AddXY(DateTime.Parse(dr[1].ToString()), int.Parse(dr[0].ToString()));
57     }
58     dr.Close();
59
60     sql2 = "select count(*) as count, workingtime from productdata " +
61         "WHERE([RealPower] < 1650 OR [RealPower] > 1750) AND([RealPower] < 670 OR [RealPower] > 720) " +
62         "group by workingtime order by workingtime"; // 불량품
63     cmd.CommandText = sql2;
64     SqlDataReader dr2 = cmd.ExecuteReader();
65     while (dr2.Read())
66     {
67         chart2.Series[1].Points.AddXY(DateTime.Parse(dr2[1].ToString()), int.Parse(dr2[0].ToString()));
68     }
69     dr2.Close();
70 }
71 catch (Exception ex)
72 {
73     MessageBox.Show(ex.StackTrace);
74     MessageBox.Show(ex.Message);
75 }
76 finally { conn.Close(); }
```



양품은 선 그래프  
불량품은 막대그래프로 표시

# 3) 프로젝트 수행 절차

```
//DB(데이터베이스)에 있는 양품 데이터만 보여주는 버튼  
참조 1개  
private void GoodData_button_Click(object sender, EventArgs e)  
{  
    try  
    {  
        ConnectDB();  
        SqlCommand cmd = new SqlCommand();  
        cmd.Connection = conn;  
  
        if (comboBox1.Text == "All")  
        {  
            cmd.CommandText = "SELECT TRIM(PageNo) AS PageNo, TRIM(Speed) AS Speed, TRIM(Length) AS Length, TRIM(RealPower) AS RealPower, TRIM(SetFrequency) AS SetFrequency,"  
                                + " TRIM(SetDuty) AS SetDuty, TRIM(SetPower) AS SetPower, TRIM(GateOnTime) AS GateOnTime, TRIM(WorkingTime) AS WorkingTime FROM ProductData"  
                                + " WHERE (RealPower BETWEEN 1650 AND 1750 OR RealPower BETWEEN 670 AND 720) order by WorkingTime";  
        }  
        else  
        {  
            cmd.CommandText = "SELECT TRIM(PageNo) AS PageNo, TRIM(Speed) AS Speed, TRIM(Length) AS Length, TRIM(RealPower) AS RealPower, TRIM(SetFrequency) AS SetFrequency, " +  
                                "TRIM(SetDuty) AS SetDuty, TRIM(SetPower) AS SetPower, TRIM(GateOnTime) AS GateOnTime, TRIM(WorkingTime) AS WorkingTime " +  
                                "FROM ProductData " +  
                                "WHERE CONVERT(date, WorkingTime) = @SelectedDate " +  
                                "AND (RealPower BETWEEN 1650 AND 1750 OR RealPower BETWEEN 670 AND 720)";  
            cmd.Parameters.AddWithValue("@selectedDate", comboBox1.SelectedItem.ToString());  
        }  
  
        SqlDataAdapter da = new SqlDataAdapter(cmd);  
        DataSet ds = new DataSet();  
        da.Fill(ds, "mytest");  
  
        dataGridView2.DataSource = ds.Tables["mytest"];  
    }  
    catch (Exception ex)  
    {  
        MessageBox.Show("날짜를 선택해 주세요");  
    }  
    finally  
    {  
        conn.Close();  
    }  
}
```

양품과 불량품을  
DataGridView에 띄우기  
위한 코드

양품

불량품

	PageNo	Speed	Length	RealPower	
▶	21	250	241,1	1721	1
	22	250	241,2	1700	1
	23	250	241,1	1723	1
	24	250	241,2	1703	1
	25	250	241,1	1721	1
	26	250	241,2	1702	1
	27	250	241,1	1724	1
	28	250	241,2	1704	1
	29	250	241,1	1726	1

양품

불량품

	PageNo	Speed	Length	RealPower	
▶	1	250	241,1	1810	
	2	250	241,2	1900	
	3	250	241,1	1810	
	4	250	241,2	1900	
	5	250	241,1	1810	
	6	250	241,2	1900	
	7	250	241,1	1800	
	8	250	241,2	1900	
	9	250	241,1	1810	



# 3) 프로젝트 수행 절차

```
333 private void dataGridView2_CellClick(object sender, DataGridViewCellEventArgs e)
334 {
335     string selectedData = "";
336     Form fc = Application.OpenForms["Reports"];
337     if (report.IsDisposed)
338         report = new Reports(this);
339     if (fc != null)
340     {
341         if (e.RowIndex >= 0)
342         {
343             selectedData = "Wn";
344             DataGridViewRow selectedRow = dataGridView2.Rows[e.RowIndex];
345             for (int i = 0; i < 9; i++)
346             {
347                 selectedData += selectedRow.Cells[i].Value.ToString() + " ";
348             }
349             report.Show(selectedData, true);
350         }
351         if (fc == null)
352         {
353             if (e.RowIndex >= 0)
354             {
355                 selectedData = "PageNO Speed Length RealPower SetFrequency SetDuty SetPower GateOnTime WorkingTimeWn";
356                 DataGridViewRow selectedRow = dataGridView2.Rows[e.RowIndex];
357                 for (int i = 0; i < 9; i++)
358                 {
359                     selectedData += selectedRow.Cells[i].Value.ToString() + " ";
360                 }
361                 report.Show(selectedData, false);
362             }
363         }
364     }
365 }
```

품질 현황

PageNO	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime		
WorkingTime	22	250	241,2	1699	1000	100	82	1154	2022-01-08

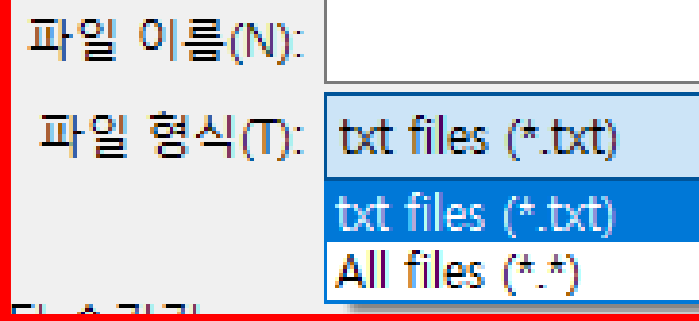
문제점 및 개선 방향

```
public void Show(string selectedData, bool append)
{
    if (append)
        richTextBox2.Text += selectedData;
    else
        richTextBox2.Text = selectedData;
    Show();
}
```

보고서 창이 있을때와  
없을때 그리드뷰의 셀  
클릭시 불러오는 데이터를  
다르게 함

# 3) 프로젝트 수행 절차

```
/*===== 파일 저장 =====  
참조 1개  
private void button1_Click(object sender, EventArgs e)  
{  
    try  
    {  
        Stream myStream;  
        SaveFileDialog saveFileDialog1 = new SaveFileDialog();  
  
        saveFileDialog1.Filter = "txt files (*.txt)|*.txt|All files (*.*)|*.*";  
        saveFileDialog1.FilterIndex = 1;  
        saveFileDialog1.RestoreDirectory = true;  
  
        if (saveFileDialog1.ShowDialog() == DialogResult.OK)  
        {  
            String txtSave = textBox1.Text + "\n\n작성일 : " + label8.Text + "\n\n작성자 : " + textBox2.Text + "\n\n개요\n" +  
richTextBox1.Text + "\n\n품질 현황\n" + richTextBox2.Text + "\n\n문제점 및 개선 방향\n" +  
richTextBox3.Text + "\n\n조치 및 결과\n" + richTextBox4.Text;  
  
            this.Text = saveFileDialog1.FileName;  
            using (StreamWriter sw = new StreamWriter(saveFileDialog1.FileName))  
            {  
                sw.Write(txtSave);  
            }  
        }  
    }  
    catch (Exception ex)  
    {  
        MessageBox.Show("오류 발생: " + ex.Message);  
    }  
    finally  
    {  
        Close();  
    }  
}
```



필터를 사용하여 파일형식 지정

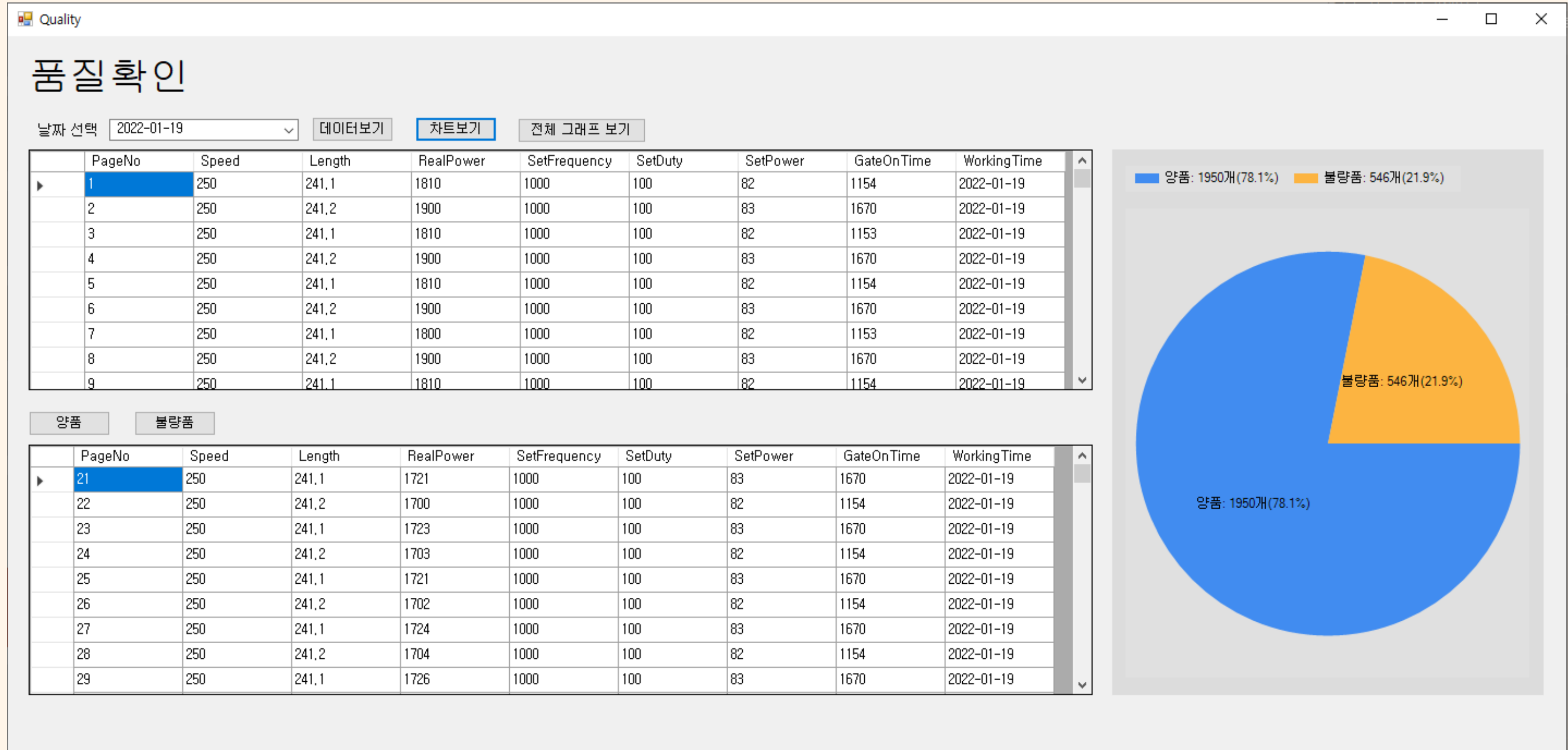
저장 경로는 이전에 쓴 경로

보고서에 저장된 Text를 저장



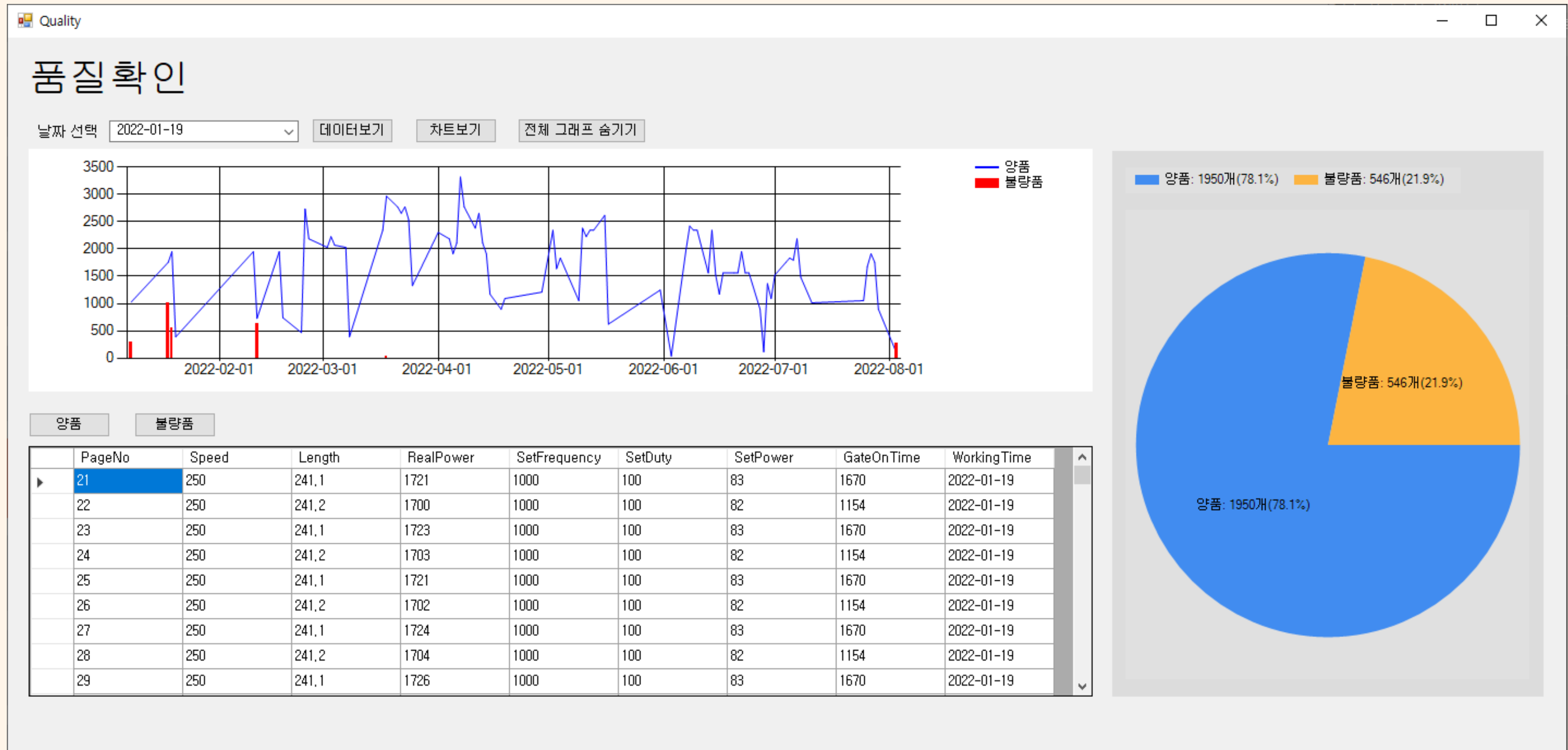
# 4) 프로젝트 수행 결과

## 실행 화면



# 4) 프로젝트 수행 결과

## 실행 화면



# 4) 프로젝트 수행 결과

## 실행 화면

Quality

### 품질 확인

날짜 선택: 2022-01-19    데이터보기    차트보기    전체 그래프 보기

	PageNo	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime	WorkingTime
▶	1	250	241.1	1810	1000	100	82	1154	2022-01-19
	2	250	241.2	1900	1000	100	83	1670	2022-01-19
	3	250	241.1	1810	1000	100	82	1153	2022-01-19
	4	250	241.2	1900	1000	100	83	1670	2022-01-19
	5	250	241.1	1810	1000	100	82	1154	2022-01-19
	6	250	241.2	1900	1000	100	83	1670	2022-01-19
	7	250	241.1	1800	1000	100	82	1153	2022-01-19
	8	250	241.2	1900	1000	100	83	1670	2022-01-19
	9	250	241.1	1810	1000	100	82	1154	2022-01-19

양품    불량품

	PageNo	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime	WorkingTime
▶	21	250	241.1	1721	1000	100	83	1670	2022-01-19
	22	250	241.2	1700	1000	100	82	1154	2022-01-19
	23	250	241.1	1723	1000	100	83	1670	2022-01-19
	24	250	241.2	1703	1000	100	82	1154	2022-01-19
	25	250	241.1	1721	1000	100	83	1670	2022-01-19
	26	250	241.2	1702	1000	100	82	1154	2022-01-19
	27	250	241.1	1724	1000	100	83	1670	2022-01-19
	28	250	241.2	1704	1000	100	82	1154	2022-01-19
	29	250	241.1	1726	1000	100	83	1670	2022-01-19

양품: 1950개 (78.1%)

셀을 클릭하면 클릭한  
데이터 행의 값들이  
품질 현황에 뜬다

보고서

### 보고서 제목

저장

작성일 : 2024년 4월 11일 목요일 오전 11:46

작성자 :

개요

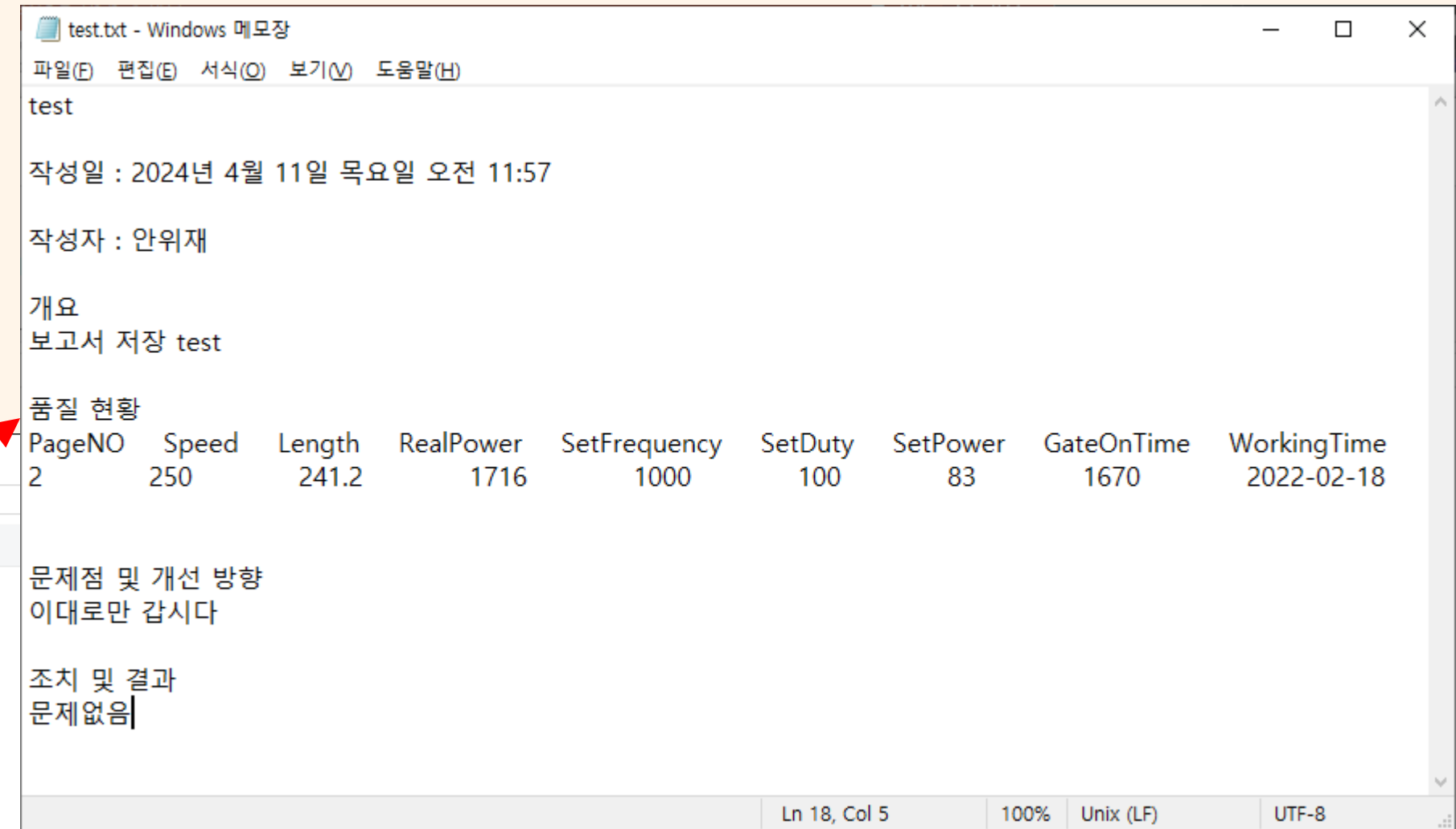
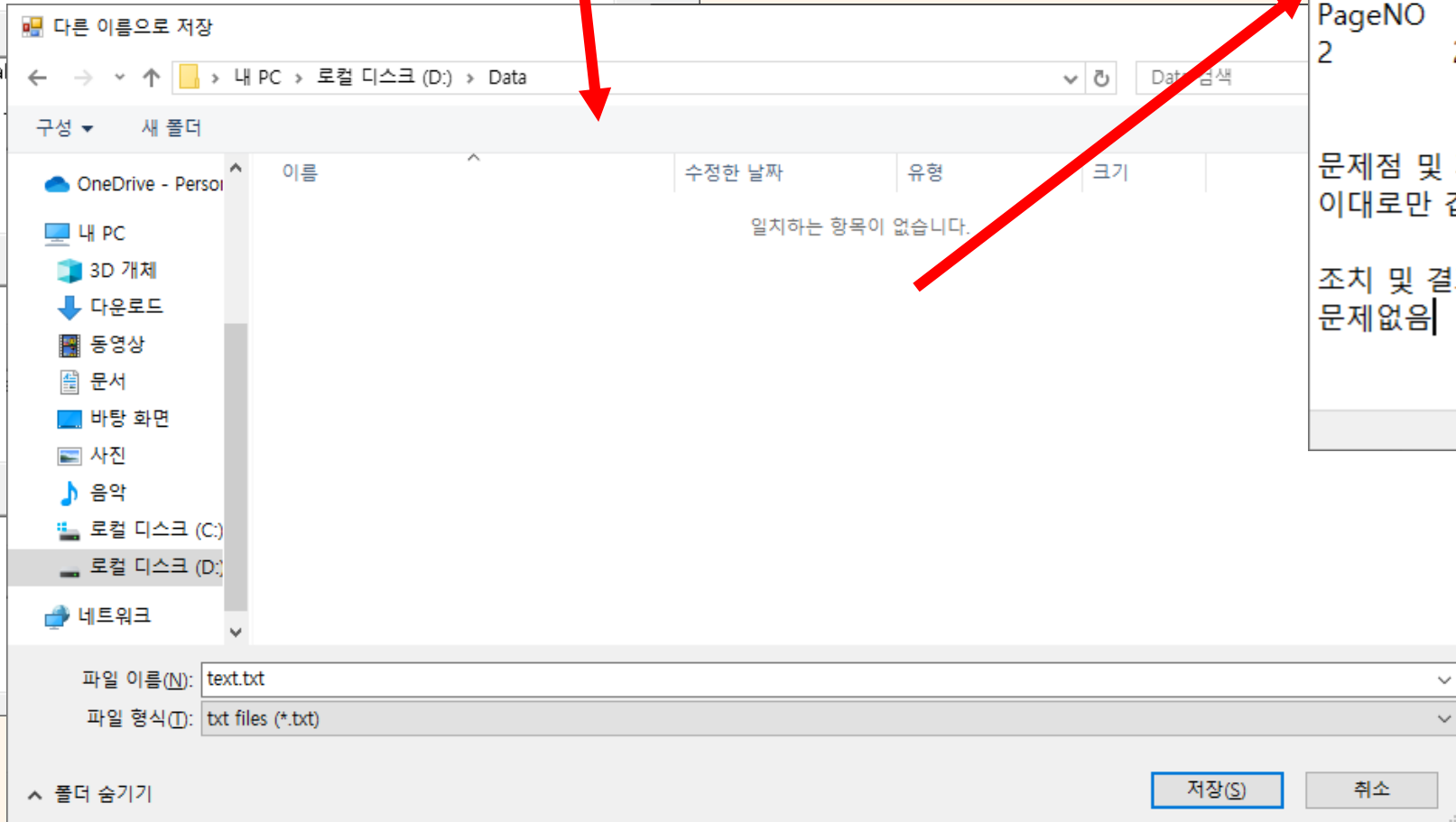
품질 현황

PageNO	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime
WorkingTime							
22	250	241.2	1699	1000	100	82	1154
							2022-01-08

문제점 및 개선 방향

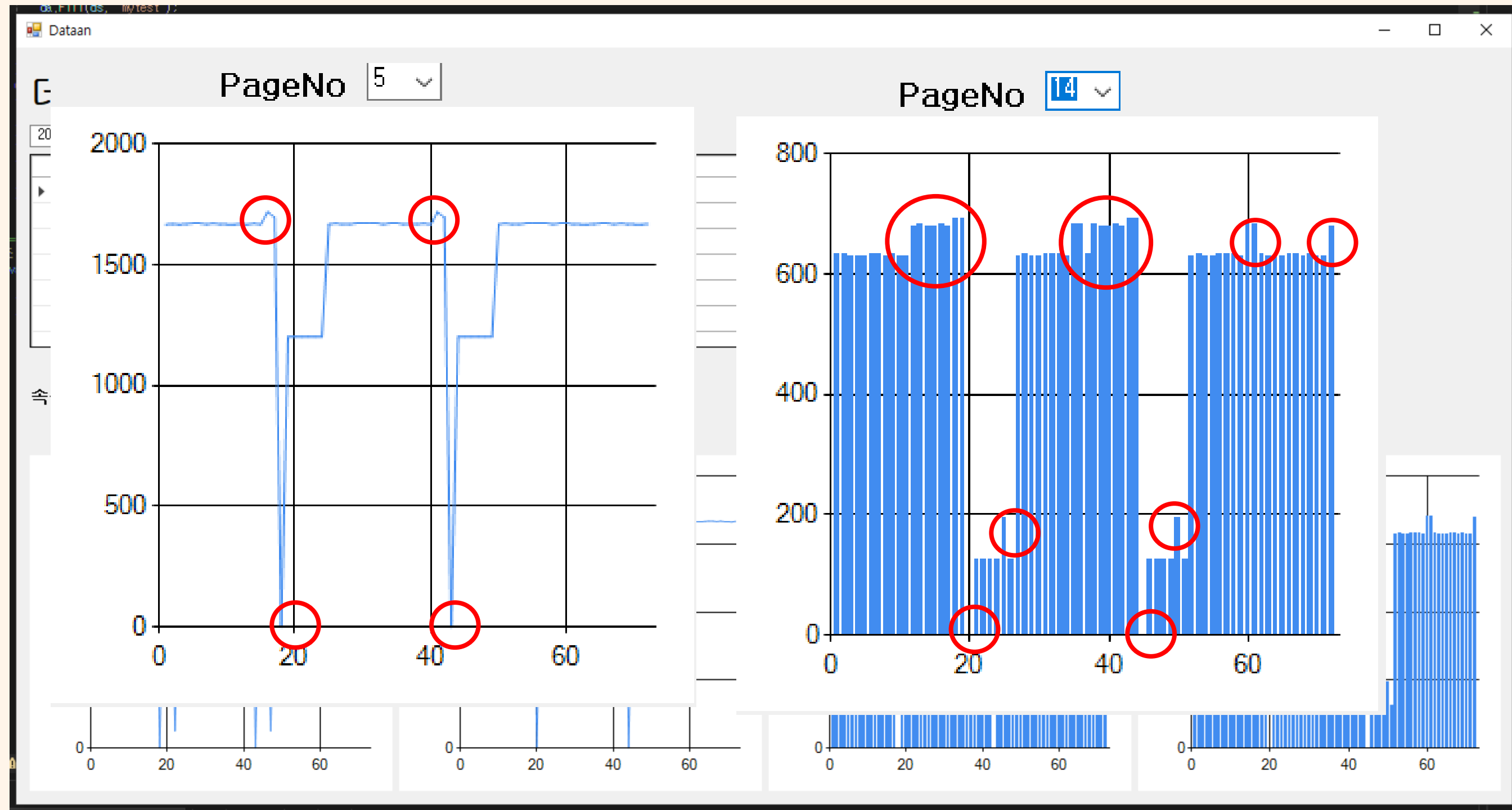
조치 및 결과

# 실행 화면



# 4) 프로젝트 수행 결과

불량품이 생산될 때 제조 환경  
데이터를 시각적으로 판단 할 수 있습니다.



# 4) 프로젝트 수행 결과

## 실행 화면

파일 불러오기

C:\Users\KB\Downloads\Dataset\_전자부품(배터리팩) 예지보전 AI 데이터셋\data\preprocessed\test\WeldingTest\_02\_OK\_Label.csv

열기 저장 데이터보기

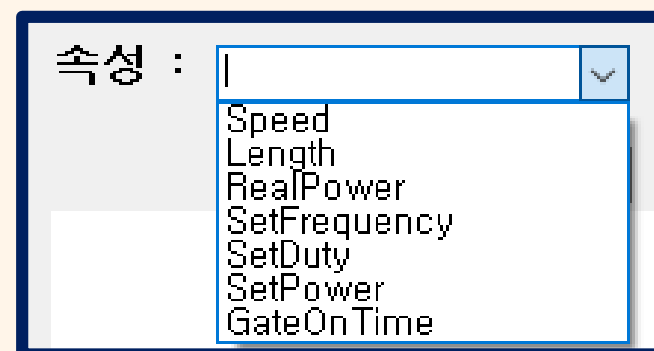
	PageNo	Speed	Length	RealPower	SetFrequency	SetDuty	SetPower	GateOnTime	WorkingTime
▶	1	250	241.1	1660	1000	100	82	1154	2022-02-11
	2	250	241.2	1685	1000	100	83	1670	2022-02-11
	3	250	241.1	1666	1000	100	82	1153	2022-02-11
	4	250	241.2	1690	1000	100	83	1670	2022-02-11
	5	250	241.1	1670	1000	100	82	1154	2022-02-11
	6	250	241.2	1693	1000	100	83	1670	2022-02-11
	7	250	241.1	1669	1000	100	82	1153	2022-02-11
	8	250	241.2	1694	1000	100	83	1670	2022-02-11
	9	250	241.1	1673	1000	100	82	1154	2022-02-11
	10	250	241.2	1695	1000	100	83	1670	2022-02-11
	11	250	241.1	1675	1000	100	82	1154	2022-02-11
	12	250	241.2	1698	1000	100	83	1670	2022-02-11
	13	30	19.4	682	1000	100	38	650	2022-02-11
	14	30	19.4	680	1000	100	38	650	2022-02-11
	15	30	19.4	682	1000	100	38	650	2022-02-11
	16	30	19.4	680	1000	100	38	650	2022-02-11
	17	30	19.4	680	1000	100	38	650	2022-02-11
	18	30	19.4	680	1000	100	38	650	2022-02-11



# 4) 프로젝트 수행 결과

## 결과 분석

- 같은 날에 작업한 모든 공정 중 같은 용접 순서에서 문제가 발생 > 해당 공정에 대한 문제인식 및 조치 가능
- 전체적인 불량/양품의 비율을 쉽게 파악하여 품질 관리 가능
- 이상이 있는 항목에 대한 보고서를 작성하여 파일로 보관하여 문제가 있는 데이터들의 관리 가능
- 데이터 분석 폼을 이용하여 이후 각 속성의 값을 다르게 주어 작업을 할 때에도 해당하는 값에 대해 분석이 가능하여 어떤 작업이 이뤄지든 품질 관리가 가능      Ex)



# 5) 자체 평가

## 안위재

대용량 데이터를 활용하여 데이터  
그리드 뷰 또는 그래프로 시각화 하기  
위해 데이터에 대한 이해와 차트와  
그리드뷰에 관련된 메소드의 이해가  
중요하다는 것을 배웠습니다.

## 이유탉

Git hub를 통해서 협업을 하는 것의  
효율성을 경험해봤습니다.  
비록 이번 프로젝트에서 한번 날렸지만  
앞으로는 조심할 수 있을 것 같습니다.  
일찍 경험해봐서 다행이라 생각했습니다.

## 신선호

데이터그리드뷰, SQL 쿼리문, 데이터 차트 시각화 등  
수업시간에 배웠던 내용들 중 부족했던 부분들을  
프로젝트를 하면서 다시금 복습할 수 있었던 시간이었고,  
팀원들이 짜놓은 코드를 이해하면서 C#에 대한 전반적인  
공부를 할 수 있었던 시간이었습니다.

## 신중하

처음 C#으로 주소 데이터프로젝트를  
해보았는데 방대한 데이터를 다루어야  
하는 만큼 SQL문법의 중요성을 알게 되었고  
프로그램을 구현하기 전 업무를  
숙지하여야지만 정확한 프로그램을 짤 수  
있다는 것을 깨닫게 되었습니다.

# 감사합니다

THANK YOU FOR WATCHING MY PRESENTATION

데이터셋 출처 - 중소벤처기업부, Korea AI Manufacturing Platform(KAMP), 전자부품(배터리팩) 예지보전 AI 데이터셋,  
스마트제조혁신추진단((주)인터엑스, 네스트필드(주)), 2022.12.23., [www.kamp-ai.kr](http://www.kamp-ai.kr)