```csharp
using DevExpress.ClipboardSource.SpreadsheetML;
using DevExpress.DataAccess;
using DevExpress.DataAccess.ObjectBinding;
using DevExpress.Entity.ProjectModel;
using DevExpress.XtraPrinting.Native;
using DevExpress.XtraReports;
using DevExpress.XtraReports.UI;
using Meditech.App.CommonService.ViewModels;
using Meditech.Core.Common.DbContexts.Sale;
using Meditech.Core.Common.DBModels;
using Meditech.Core.Common.Repositories;
using Meditech.Core.Common.Services;
using Meditech.Lib.CommonShare;
using Meditech.Lib.Share.CommonClass;
using Microsoft.Data.SqlClient;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Options;
using System.Collections;
using System.Data;
using System.Reflection;
using System.Text.RegularExpressions;

namespace Meditech.App.CommonService.Services
{
    public interface IReportService
    {
        ReportPreviewModel GetReportPreview(Guid requestId);
        ReturnCode CreatePrinter(PrinterViewModel model, Guid UserId);
    }
    public class ReportService : IReportService
    {
        private readonly ILogger<ReportService> _logger;
        private readonly IPrintRequestRepository _printRequestRepository;
        private CommonUnitOfWork _unitOfWork;
        private readonly DbContextOptions<SaleDbContext> _dbOptionSale;
        private readonly IDBConnectServices _dBConnectServices;
        string _urlPrinter = "";
        private readonly IDonHangKhamSucKhoeRepository _donHangKhamSucKhoeRepository;
        private readonly IPhuongAnKinhDoanhKskRepository _phuongAnKinhDoanhKskRepository;
        private readonly IDonKSKData _donKSKData;

        public ReportService(ILogger<ReportService> logger,
            IPrintRequestRepository printRequestRepository,
            IConfiguration configuration,
            IOptions<HostConfig> options,
            CommonUnitOfWork unitOfWork,
            IDBConnectServices dBConnectServices,
            IDonHangKhamSucKhoeRepository donHangKhamSucKhoeRepository,
            IPhuongAnKinhDoanhKskRepository phuongAnKinhDoanhKskRepository,
            IDonKSKData donKSKData)
        {
            string dbConnSale = configuration.GetConnectionString("SaleDatabase") ?? "";
            _dbOptionSale = new DbContextOptionsBuilder<SaleDbContext>().UseSqlServer(dbConnSale).Options;
            _urlPrinter = options.Value.UrlPrinter;
            _logger = logger;
            _printRequestRepository = printRequestRepository;
            _unitOfWork = unitOfWork;
            _dBConnectServices = dBConnectServices;
            _donHangKhamSucKhoeRepository = donHangKhamSucKhoeRepository;
            _phuongAnKinhDoanhKskRepository = phuongAnKinhDoanhKskRepository;
            _donKSKData = donKSKData;
        }

        public ReportPreviewModel GetReportPreview(Guid requestId)
        {
            _logger.LogDebug($"GetReportPreview req={requestId}");
            ReportPreviewModel ret = new ReportPreviewModel();
            var req = _printRequestRepository.GetById(requestId);
            if (req == null)
            {
                _logger.LogWarning($"GetReportPreview req={requestId} not found");
                ret.IsSuccess = false;
                ret.ErrorMsg = "Yêu cầu không hợp lệ, xin vui lòng thao tác lại";
                return ret;
            }
            if (req.Hieulucden.HasValue &&
               req.Hieulucden.Value < DateTime.UtcNow)
            {
                _logger.LogWarning($"GetReportPreview req={requestId} expired");
                ret.IsSuccess = false;
                ret.ErrorMsg = "Yêu cầu đã quá thời gian cho phép, xin vui lòng thao tác lại";
                return ret;
            }

            ret.IsSuccess = true;
            ret.ReportModel = CreateXtraReport(req);

            if (ret.ReportModel == null)
            {
                _logger.LogWarning($"GetReportPreview req={requestId} not found");
                ret.IsSuccess = false;
                ret.ErrorMsg = "Không tìm thấy mẫu in";
                return ret;
            }
            _logger.LogInformation($"GetReportPreview req={requestId} success");
            return ret;
        }

        public XtraReport CreateXtraReport(PrintRequest mauin)
        {
            //Lấy ct
            int ct = 2;
            if (mauin.Tenmauin == null)
            {
                return null;
            }
    //Get bytes from XtraReport  to XML file
    if (mauin.DesignData == null)
    {
        var _tempRpt = Activator.CreateInstance(Type.GetType($"Meditech.App.CommonService.Reports.{mauin.Tenmauin}"));
        mauin.DesignData = GetByteXtraReport((XtraReport)_tempRpt);
    }

    //Get Xtrareport from XML file
    XtraReport _rptReport = XtraReport.FromXmlFile(CreateXmlFileFromBytes(mauin.Id.ToString(), mauin.DesignData));
    foreach (var param in _rptReport.Parameters) param.Visible = false;

    if (_rptReport.Parameters["soyte"] != null)
    {
        _rptReport.Parameters["soyte"].Value = "SỞ Y TẾ THANH HÓA";
    }
    if (_rptReport.Parameters["tendonvi"] != null)
    {
        _rptReport.Parameters["tendonvi"].Value = "Tập đoàn Medipha";
    }
    if (_rptReport.Parameters["ngayin"] != null)
    {
        _rptReport.Parameters["ngayin"].Value = DateTime.Now;
    }
    if (_rptReport.Parameters["ct"] != null)
    {
        _rptReport.Parameters["ct"].Value = ct;
    }

    //Get param data
    if (mauin.Tenmauin == "rptPhieuThu" || mauin.Tenmauin == "rptPhieuChi")
    {
        var data = getParamThuChi(mauin);

        foreach (PropertyInfo propertyInfo in data.GetType().GetProperties())
        {
            if (_rptReport.Parameters[propertyInfo.Name] != null)
                _rptReport.Parameters[propertyInfo.Name].Value = propertyInfo.GetValue(data);
        }
    }

    else
    {
        if (mauin.Param + "" != "")
        {
            List<ParamMauIn> lstParamMauIn = new List<ParamMauIn>();
            string[] arr1 = mauin.Param.Split(",");

            foreach (var item1 in arr1)
            {
                ParamMauIn paramMauIn = new ParamMauIn();
                // Biểu thức chính quy tìm kiếm tất cả các chuỗi nằm giữa dấu ngoặc kép
                var regex = new Regex("\"([^\"]*)\"");
                var matches = regex.Matches(item1);
                if (matches.Count == 2)
                {
                    paramMauIn.ParamName = matches[0].Groups[1].Value;
                    paramMauIn.ParamValue = matches[1].Groups[1].Value;
                }
                if (paramMauIn != null)
                {
                    lstParamMauIn.Add(paramMauIn);
                }
            }

            foreach (var item in _rptReport.Parameters)
            {
                if (item.Name + "" != "")
                {
                    ParamMauIn x1 = lstParamMauIn.Where(x => x.ParamName == item.Name).FirstOrDefault();
                    if (x1 != null)
                    {
                        item.Value = x1.ParamValue;
                    }
                }
            }

            if (mauin.Tenmauin == "DonKSK.rptDonKSK")
            {
                _rptReport.DataSource = _donKSKData.GetDonHang((Guid)_rptReport.Parameters[0].Value);
            }
        }
    }

    return _rptReport;
}
public static byte[] GetByteXtraReport(XtraReport report)
{
    using (MemoryStream stream = new MemoryStream())
    {
        report.SaveLayoutToXml(stream);
        return stream.ToArray();
    }
}
public static string CreateXmlFileFromBytes(string id, byte[]? reportData)
{
    if (reportData == null || reportData.Length < 1) return "";
    string saveFolder = Path.Combine(TempDirectory, "tmp", "report-design");
    if (!Directory.Exists(saveFolder))
    {
        Directory.CreateDirectory(saveFolder);
    }
    string tempFilePath = Path.Combine(saveFolder, id.ToString() + ".xml");
    if (File.Exists(tempFilePath)) File.Delete(tempFilePath);
    File.WriteAllBytes(tempFilePath, reportData);
    return tempFilePath;
}
public ReturnCode CreatePrinter(PrinterViewModel o, Guid UserId)
{
```

```
            ReturnCode ret = new ReturnCode();

            if (o == null || string.IsNullOrEmpty(o.TenMauIn))
            {
                ret.Code = ErrorCode.HasError;
                ret.Description = "Mẫu in không đúng";
                return ret;
            }

            var p = _printRequestRepository.GetAll().Where(w => w.Tenmauin == o.TenMauIn &&
                                                                w.Idphieu == o.IdPhieu &&
                                                                w.Nguoitaoid == UserId.ToString() &&
                                                                w.Loaiphieu == o.LoaiPhieu);

            if (p.Count() > 0)
            {
                var print = p.FirstOrDefault();

                if (print.Hieulucden > DateTime.UtcNow)
                {
                    ret.Description = $"{_urlPrinter}?reqId={print.Id.ToString().ToUpper()}";
                    return ret;
                }
                _printRequestRepository.DeleteMultiple(p.ToList());
            }

            Guid guid = Guid.NewGuid();
            _printRequestRepository.Insert(new PrintRequest()
            {
                Id = guid,
                Loaiphieu = o.LoaiPhieu,
                Tenmauin = o.TenMauIn,
                Sophieu = o.SoPhieu,
                Idphieu = o.IdPhieu,
                Thoigiantao = DateTime.UtcNow,
                Hieuluctu = DateTime.UtcNow,
                Hieulucden = DateTime.UtcNow.AddMinutes(5),
                Nguoitaoid = UserId.ToString(),
                Param = o.Param + "",
            });

            _unitOfWork.Save();

            ret.Description = $"{_urlPrinter}?reqId={guid}";
            return ret;

        }

        ParamThuChi getParamThuChi(PrintRequest mauin)
        {
            List<SqlParameter> ps = new List<SqlParameter>();
            ps.Add(new SqlParameter { ParameterName = "mathuchi", SqlDbType = SqlDbType.Int, Value = mauin.Idphieu });

            ps.Add(new SqlParameter { ParameterName = "maloi", SqlDbType = SqlDbType.Int, Value = 0, Size = 8, Direction = ParameterDirection.Output });
            ps.Add(new SqlParameter { ParameterName = "tenloi", SqlDbType = SqlDbType.NVarChar, Value = "", Size = 4000, Direction = ParameterDirection.Output });

            SqlParameter[] pr = ps.ToArray();
            ObjectReturnCode<DataTable> retDt = _dBConnectServices.DBGetDataTableStored(_dbOption: _dbOptionSale, storename: "[getPrintThuChi]", param: pr);

            var obj = new ParamThuChi();
            if (retDt.IsOk)
            {
                if (retDt.Data != null && retDt.Data.Rows.Count > 0)
                {
                    foreach (PropertyInfo propertyInfo in obj.GetType().GetProperties())
                    {
                        if (propertyInfo.PropertyType == typeof(System.Int32))
                        {
                            propertyInfo.SetValue(obj, System.Int32.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else if (propertyInfo.PropertyType == typeof(int))
                        {
                            propertyInfo.SetValue(obj, int.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else if (propertyInfo.PropertyType == typeof(decimal))
                        {
                            propertyInfo.SetValue(obj, decimal.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else if (propertyInfo.PropertyType == typeof(DateTime))
                        {
                            propertyInfo.SetValue(obj, DateTime.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else
                        {
                            propertyInfo.SetValue(obj, retDt.Data.Rows[0][propertyInfo.Name].ToString());
                        }
                    }
                }
            }


            return obj;
        }
        ParamThuChi getParamBaoCaoDoanhThu(PrintRequest mauin)
        {
            List<SqlParameter> ps = new List<SqlParameter>();
            ps.Add(new SqlParameter { ParameterName = "mathuchi", SqlDbType = SqlDbType.Int, Value = mauin.Idphieu });

            ps.Add(new SqlParameter { ParameterName = "maloi", SqlDbType = SqlDbType.Int, Value = 0, Size = 8, Direction = ParameterDirection.Output });
            ps.Add(new SqlParameter { ParameterName = "tenloi", SqlDbType = SqlDbType.NVarChar, Value = "", Size = 4000, Direction = ParameterDirection.Output });

            SqlParameter[] pr = ps.ToArray();
            ObjectReturnCode<DataTable> retDt = _dBConnectServices.DBGetDataTableStored(_dbOption: _dbOptionSale, storename: "[getPrintThuChi]", param: pr);

            var obj = new ParamThuChi();
            if (retDt.IsOk)
            {
                if (retDt.Data != null && retDt.Data.Rows.Count > 0)
                {
                    foreach (PropertyInfo propertyInfo in obj.GetType().GetProperties())
                    {
                        if (propertyInfo.PropertyType == typeof(System.Int32))
                        {
                            propertyInfo.SetValue(obj, System.Int32.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else if (propertyInfo.PropertyType == typeof(int))
                        {
                            propertyInfo.SetValue(obj, int.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else if (propertyInfo.PropertyType == typeof(decimal))
                        {
                            propertyInfo.SetValue(obj, decimal.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else if (propertyInfo.PropertyType == typeof(DateTime))
                        {
                            propertyInfo.SetValue(obj, DateTime.Parse(retDt.Data.Rows[0][propertyInfo.Name].ToString()));
                        }
                        else
                        {
                            propertyInfo.SetValue(obj, retDt.Data.Rows[0][propertyInfo.Name].ToString());
                        }
                    }
                }
            }


            return obj;
        }

        public static string TempDirectory
        {
            get
            {
                return Path.Combine(Path.GetTempPath(), "QuayThuocV2");
            }
        }

    }
    public class ParamThuChi
    {
        public int mathuchi { get; set; }
        public string sophieu { get; set; } = "1";
        public string ngay { get; set; } = "2";
        public string sohoadon { get; set; } = "";
        public decimal sotien { get; set; }
        public string noidung { get; set; } = "";
        public string hinhthuctt { get; set; } = "";
        public string lydotc { get; set; } = "";
        public string hoten { get; set; } = "";
        public string bangchu { get; set; } = "";
        public string ngayin { get; set; } = "";
    }
    public class ParamMauIn
    {
        public string ParamName { get; set; }
        public string ParamValue { get; set; }
    }
}
```