//get the import type from import history table

var importHistoryEntity = await \_importHistory.GetEntityByConditionAsync(x => x.Id == importHistoryId);

var importType = importHistoryEntity.ImportType;

var model = new List<MappingModel>();

model = await \_fileService.getFileData(model,(ExcelImportType)importType);

var customers = await \_fileService.GetFileJson<Customer>((ulong)importHistoryId, "Customer.json", true);

var companies = await \_fileService.GetFileJson<Company>((ulong)importHistoryId, "Company.json", true);

var responseList = new List<CreateDataModel>();

for (var i=0; i<customers.Count(); i++)

{

var response = new CreateDataModel();

foreach (var col in model)

{

if (col.IsRequired == true)

{

if(col.TableName == "Customer")

{

var j = typeof(Customer);

AddData<Customer>(j, response, customers, col, i);

}

if (col.TableName == "Company")

{

var j = typeof(Company);

AddData<Company>(j, response, companies, col, i);

}

}

//when is required = false

else

{

if (col.TableName == "Customer")

{

var j = typeof(Customer);

var properties = j.GetProperties().Where(x => x.Name.ToLower() == col.ColumnName.ToString()).ToList();

foreach (var prop in properties)

{

if (prop != null)

{

var value = prop.GetValue(customers[i]);

if (value != null)

{

response.Data.Add(new DataToBeInserted() { ColumnName = col.ColumnName, Value = value });

}

}

}

}

if (col.TableName == "Company")

{

var j = typeof(Company);

var properties = j.GetProperties().Where(x => x.Name.ToLower() == col.ColumnName.ToString()).ToList();

foreach (var prop in properties)

{

if (prop != null)

{

var value = prop.GetValue(companies[i]);

if (value != null)

{

if (value.GetType().IsGenericType &&

value.GetType().GetGenericTypeDefinition() == typeof(List<>))

{

//don't know the logic to check count of list if 0 or >0

continue;

}else if(value is string && value !="")

{

response.Data.Add(new DataToBeInserted() { ColumnName = col.ColumnName, Value = value });

}else if (value.IsNumeric() && !value.Equals((object)0))

{

response.Data.Add(new DataToBeInserted() { ColumnName = col.ColumnName, Value = value });

}

}

}

}

}

}

}

responseList.Add(response);

}

if( responseList.Count() == 0 )

{

responseList[0].Error = "Error: Not able to get the data";

}

return responseList ;

}

Customer supplier LIST

using (MySqlDataReader reader = command.ExecuteReader())

{

// Check if the reader has rows

if (reader.HasRows)

{

// Iterate over the rows

while (reader.Read())

{

// Access the values from the result

int credit\_days = reader.GetInt32("credit\_days");

Console.WriteLine(credit\_days);

// Do something with the retrieved values

// ...

}

}

// Execute the stored procedure

//command.ExecuteNonQuery();

}

To check if null

int? creditDays = reader.IsDBNull(reader.GetOrdinal("credit\_days")) ? null : (int?)reader.GetInt32("credit\_days");

get customer list api

yasin sir’s code:

//var customers = (searchModel.Alternate)

// ? await \_customerRepository.GetAllEntitiesAsync(x => x.CustomerCompanyId == companyId)

// : await \_customerRepository.GetAllEntitiesAsync(x => x.UserCompanyId == companyId);

//var customerList = \_mapper.Map<List<CustomerListModel>>(customers.OrderByDescending(x => x.Id));

//foreach (var customer in customerList)

//{

// var company = (searchModel.Alternate)

// ? (await \_companyRepository.GetEntityByConditionAsync(x => x.Id == customer.UserCompanyId))

// : (await \_companyRepository.GetEntityByConditionAsync(x => x.Id == customer.CustomerCompanyId));

// if (company != null)

// customer.CompanyName = company.CompanyName;

// var invoices = await \_invoiceRepository.GetAllEntitiesAsync(x => x.SellerId == customer.UserCompanyId && x.BuyerId == customer.CustomerCompanyId

// && x.PaymentStatus == SaakhStaticsModel.PaymentStatusAccepted);

// if (invoices.Count > 0)

// {

// customer.OutstandingAmount = invoices.Sum(x => x.TotalInvoiceAmount);

// var payments = await \_paymentRepository.GetAllEntitiesAsync(x => invoices.Select(x => x.Id).Contains(x.InvoiceId) && x.PaymentStatus == SaakhStaticsModel.PaymentStatusAccepted);

// if (payments.Count > 0)

// customer.OutstandingAmount -= payments.Sum(x => x.ReceivedAmount);

// }

//}

//return customerList;

+

COALESCE((

SELECT SUM(i.total\_invoice\_amount)

FROM invoice i

WHERE i.buyer\_id = cs.customer\_company\_id

AND i.seller\_id = cs.user\_company\_id

AND i.payment\_status = \'accepted\'

AND i.deleted\_at is null

), 0)

- COALESCE((

SELECT SUM(p.received\_amount)

FROM payments p

WHERE p.buyer\_id = cs.customer\_company\_id

AND p.seller\_id = cs.user\_company\_id

AND p.payment\_status = \'accepted\'

AND p.deleted\_at is null

), 0) AS OutstandingAmount,

//changes

+

COALESCE((

SELECT SUM(i.total\_invoice\_amount)

FROM invoice i

WHERE i.buyer\_id = cs.customer\_company\_id

AND i.seller\_id = cs.user\_company\_id

AND i.payment\_status = \'accepted\'

AND i.deleted\_at is null

), 0)

- COALESCE((

SELECT SUM(p.received\_amount)

FROM payments p

JOIN invoice i ON i.id=p.invoice\_id

WHERE p.buyer\_id = cs.customer\_company\_id

AND p.seller\_id = cs.user\_company\_id

AND p.payment\_status = \'accepted\'

AND p.deleted\_at is null

), 0) AS OutstandingAmount,

Saakh saveErrorData API:

{

    "excelImportType": 1,

    "errorModel": [

        {

            "data": [

                {

                    "excelColumn": "Customer GSTN",

                    "value": "27APLPS0854P1ZW"

                },

                {

                    "excelColumn": "Customer Name",

                    "value": "Quality walls"

                },

                {

                    "excelColumn": "Credit days",

                    "value": "3"

                },

                {

                    "excelColumn": "Referral agent",

                    "value": 5

                }

            ],

            "error": "Error : Credit days should be numeric"

        }

    ],

    "correctData": [

        {

            "data": [

                {

                    "excelColumn": "Customer GSTN",

                    "value": "27ARLPS3839T1ZQ"

                },

                {

                    "excelColumn": "Customer Name",

                    "value": "Dell"

                },

                {

                    "excelColumn": "Credit days",

                    "value": 3

                }

            ]

        },

        {

            "data": [

                {

                    "excelColumn": "Customer GSTN",

                    "value": "27APLPS4738U1ZB"

                },

                {

                    "excelColumn": "Customer Name",

                    "value": "Samsung"

                },

                {

                    "excelColumn": "Credit days",

                    "value": 3

                }

            ]

        },

        {

            "data": [

                {

                    "excelColumn": "Customer GSTN",

                    "value": "27APLPD0635R1ZS"

                },

                {

                    "excelColumn": "Customer Name",

                    "value": "Breadman"

                },

                {

                    "excelColumn": "Credit days",

                    "value": 3

                },

                {

                    "excelColumn": "Referral agent",

                    "value": 21

                }

            ]

        }

    ],

    "createData": [],

    "filePaths": [

        "Company.json",

        "Customer.json",

        "CompanyDetail.json"

    ],

    "mappings": [

        {

            "excelColumn": "Customer GSTN",

            "excelAddrress": 1,

            "tableName": "Company",

            "columnName": "gstnumber",

            "visibleColumn": "GSTIN",

            "dataType": "String",

            "isRequired": **false**,

            "id": 0

        },

        {

            "excelColumn": "Customer Name",

            "excelAddrress": 2,

            "tableName": "Company",

            "columnName": "companyname",

            "visibleColumn": "Company Name",

            "dataType": "String",

            "isRequired": **false**,

            "id": 0

        },

        {

            "excelColumn": "Credit days",

            "excelAddrress": 3,

            "tableName": "Customer",

            "columnName": "creditdays",

            "visibleColumn": "Credit Days",

            "dataType": "Int32",

            "isRequired": **false**,

            "id": 0

        },

        {

            "excelColumn": "Referral agent",

            "excelAddrress": 4,

            "tableName": "Company",

            "columnName": "referralagent",

            "visibleColumn": "Agent Name",

            "dataType": "Int32",

            "isRequired": **false**,

            "id": 0

        }

    ],

    "requiredColumnName": [],

    "wrongMapping": **false**,

    "duplicateColumnName": [],

    "duplicateMapping": **false**

}