**38924846-L Sandi**

DEMONSTRATION DOCUMENT



**September 24, 2024**

# TABLE OF CONTENTS

Contents

[TABLE OF CONTENTS 1](#_Toc144995991)

[PHYSICAL DATA MODEL 2](#_Toc144995992)

[DATABASE SCHEMA 3](#_Toc144995993)

[PHYSICAL DATA MODEL 4](#_Toc144995994)

[SQL CODE 5](#_Toc144995995)

[CREATING TABLES 5](#_Toc144995996)

[INSERTING REPORTED ITEMS INTO DATABASE 6](#_Toc144995997)

[QUERIES 7](#_Toc144995998)

[UPDATING USER DETAILS(ACCOUNT DETAILS) 7](#_Toc144995999)

[DELETING USER ACCOUNT 7](#_Toc144996000)

[USER MANUAL FOR THE NORTH WEST UNIVERSITY LOST AND FOUND 10](#_Toc144996001)

[Computer specifications 10](#_Toc144996002)

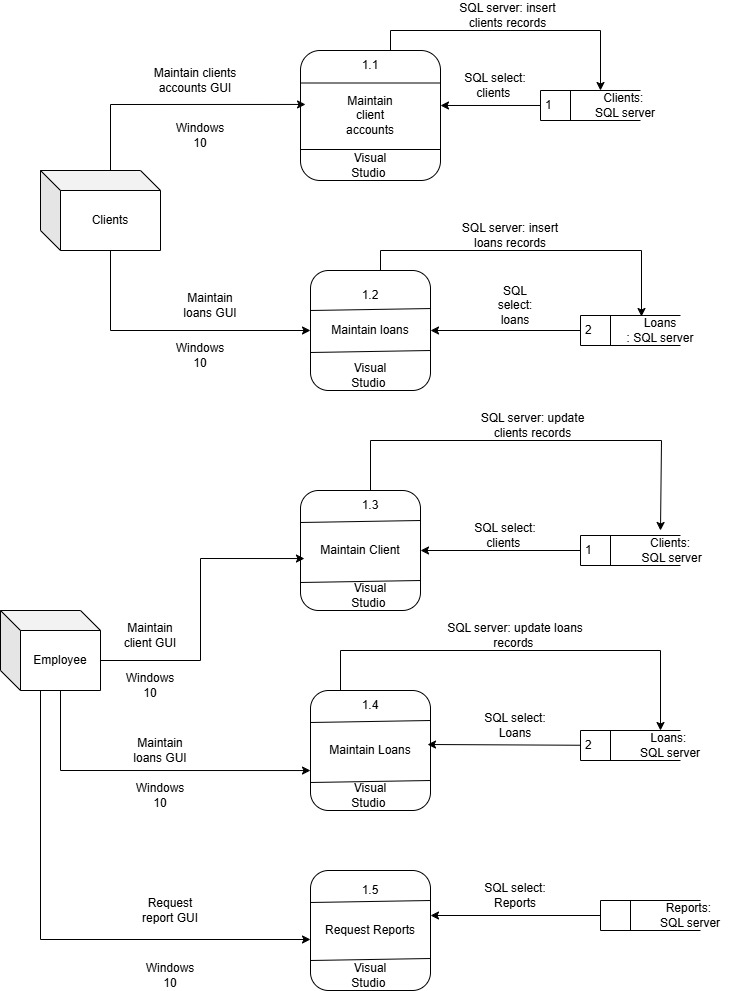
[Installation of Visual Studio 2019 and package(workload) 10](#_Toc144996003)

[Running of the file 11](#_Toc144996004)

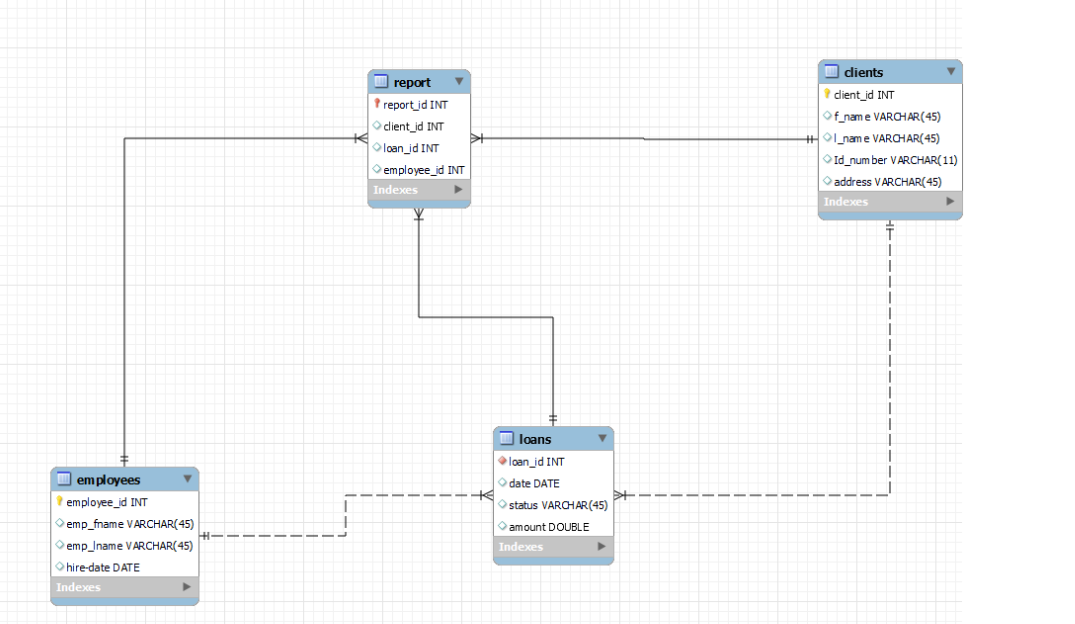
[USER INFERFACE 12](#_Toc144996005)

[Detailed Dairy 17](#_Toc144996006)

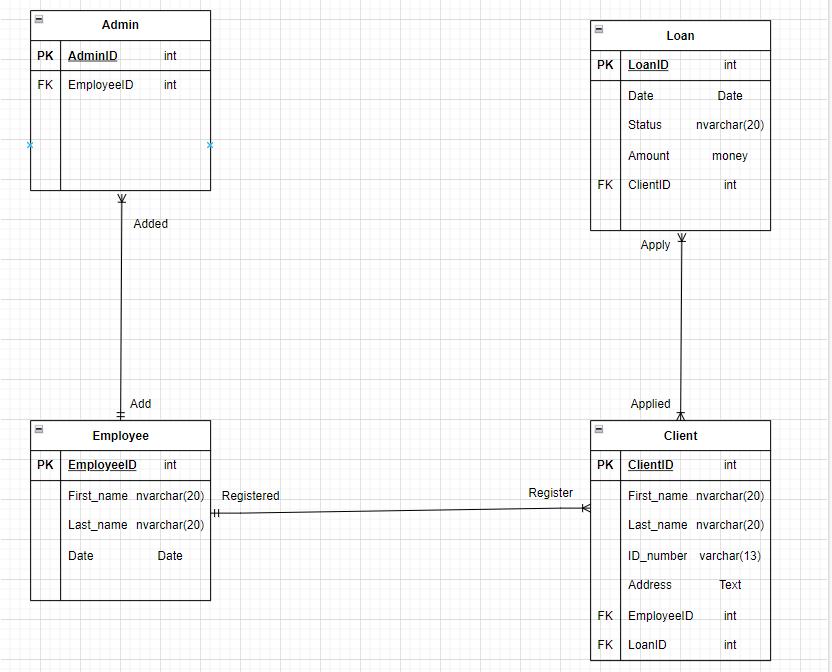
# PHYSICAL PROCESS MODEL



# DATABASE SCHEMA



# PHYSICAL DATA MODEL



# SQL CODE

## CREATING TABLES

CREATE TABLE [dbo].[Admin] (

[adminID] INT NOT NULL,

[admin\_Name] NVARCHAR (45) NULL,

[admin\_Surname] NVARCHAR (45) NULL,

[EmployeeID] INT NULL,

PRIMARY KEY CLUSTERED ([adminID] ASC)

);

CREATE TABLE [dbo].[Loans] (

[LoanID] INT IDENTITY (1 1) NOT NULL,

[Status] VARCHAR (45) NULL,

[Amount] INT NULL,

[Client\_ID] INT NULL,

PRIMARY KEY CLUSTERED ([LoanID] ASC)

);

CREATE TABLE [dbo].[Clients] (

[ClientID] INT IDENTITY(1,1) NOT NULL,

[FirstName] NVARCHAR(45) NULL,

[LastName] NVARCHAR(45) NULL,

[Email] NVARCHAR(100) NULL,

PRIMARY KEY CLUSTERED ([ClientID] ASC)

);

CREATE TABLE [dbo].[Employee] (

[Employee\_ID] INT IDENTITY (1,1) NOT NULL,

[First\_Name] VARCHAR (45) NULL,

[Last\_Name] VARCHAR (45) NULL,

[Date] DATE NULL,

PRIMARY KEY CLUSTERED ([Employee\_ID] ASC)

);

## INSERTING NEW ClIENT INTO DATABASE

command = new SqlCommand("SELECT \* FROM ", connection);

reader = command.ExecuteReader();

// Define the update query to change the client's information

string updateQuery = "UPDATE Clients SET FirstName = @firstName, LastName = @lastName, Email = @email WHERE ClientID = @clientID";

// Create a new SqlCommand

SqlCommand command = new SqlCommand(updateQuery, connection);

command.Parameters.AddWithValue("@firstName", txtFirstName.Text);

command.Parameters.AddWithValue("@lastName", txtLastName.Text);

command.Parameters.AddWithValue("@email", txtEmail.Text);

command.Parameters.AddWithValue("@clientID", txtClientID.Text); // Assuming clientID is passed from a textbox or variable

command.ExecuteNonQuery();

## QUERIES

string query = "SELECT \* FROM tblAdministrator WHERE administratorEmail = @adminEmail";

SqlCommand cmd = new SqlCommand(query, con);

cmd.Parameters.AddWithValue("@adminEmail", txtAdminEmail.Text);

string query = "SELECT COUNT(\*) FROM tblUsers WHERE userEmail = @userEmail";

SqlCommand cmd = new SqlCommand(query, con);

cmd.Parameters.AddWithValue("@userEmail", txtLoginEmail.Text);

## UPDATING CLIENTS DETAILS(ACCOUNT DETAILS)

// Define the update query for the Clients table

string updateQuery = "UPDATE Clients SET FirstName = @firstName, LastName = @lastName, Email = @email WHERE ClientID = @clientID";

// Create a new SqlCommand

SqlCommand command = new SqlCommand(updateQuery, connection);

// Add parameters to prevent SQL injection

command.Parameters.AddWithValue("@firstName", txtFirstName.Text); // User input for first name command.Parameters.AddWithValue("@lastName", txtLastName.Text); // User input for last name command.Parameters.AddWithValue("@email", txtEmail.Text); // User input for email command.Parameters.AddWithValue("@clientID", txtClientID.Text);;

adapter.UpdateCommand.ExecuteNonQuery();

## DELETING FROM CLIENTS ACCOUNT

string deleteQuery = "DELETE FROM Clients WHERE ClientID = @clientID";

// Create a new SqlCommand

SqlCommand command = new SqlCommand(deleteQuery, connection);

command.Parameters.AddWithValue("@clientID", txtClientID.Text)

command.ExecuteNonQuery();

MessageBox.Showw("Client deleted successfully."); adapter.DeleteCommand.ExecuteNonQuery()

## DELETING FROM LOANS ACCOUNT

string deleteQuery = "DELETE FROM Loans WHERE LoanID = @loanID";

SqlCommand command = new SqlCommand(deleteQuery, connection);

command.Parameters.AddWithValue("@loanID", txtLoanID.Text); // Assuming loanID is passed from a textbox or variable

command.ExecuteNonQuery();

MessageBox.Show("Loan deleted successfully.");

## INSERTING INTO THE CLIENTS AND LOANS

try {

connection.Open(); // Begin the transaction

transaction = connection.BeginTransaction(); // Step 1: Insert into Clients table string insertClientQuery = "INSERT INTO Clients (FirstName, LastName, Email) OUTPUT INSERTED.ClientID VALUES (@firstName, @lastName, @email)";

SqlCommand clientCommand = new SqlCommand(insertClientQuery, connection, transaction); clientCommand.Parameters.AddWithValue("@firstName", txtFirstName.Text); // Assuming user input clientCommand.Parameters.AddWithValue("@lastName", txtLastName.Text); // Assuming user input clientCommand.Parameters.AddWithValue("@email", txtEmail.Text);

int newClientID = (int)clientCommand.ExecuteScalar();

string insertLoanQuery = "INSERT INTO Loans (Status, Amount, Client\_ID) VALUES (@status, @amount, @clientID)";

SqlCommand loanCommand = new SqlCommand(insertLoanQuery, connection, transaction); loanCommand.Parameters.AddWithValue("@status", txtLoanStatus.Text); // Assuming user input loanCommand.Parameters.AddWithValue("@amount", txtLoanAmount.Text); // Assuming user input loanCommand.Parameters.AddWithValue("@clientID", newClientID);

loanCommand.ExecuteNonQuery();

transaction.Commit();

MessageBox.Show("Client and Loan records inserted successfully.");

} catch (Exception ex) { if (transaction != null) {

transaction.Rollback();

} MessageBox.Show("An error occurred: "); }

loanCommand.ExecuteNonQuery();

## UPDATING LOANS ACCOUNT

command = new SqlCommand("SELECT \* tblReports", con);

string updateQuery = "UPDATE Loans SET Status = @status, Amount = @amount WHERE LoanID = @loanID";

// Create a new SqlCommand

SqlCommand command = new SqlCommand(updateQuery, connection);

// Add parameters to prevent SQL injection

command.Parameters.AddWithValue("@status", txtStatus.Text); // User input for loan status command.Parameters.AddWithValue("@amount", txtAmount.Text); // User input for loan amount command.Parameters.AddWithValue("@loanID", txtLoanID.Text); adapter.UpdateCommand.ExecuteNonQuery();

## REQUESTING REPORT

string selectQuery = "SELECT reportID, adminID, locationID, itemID FROM tblReport WHERE adminID = @adminID OR locationID = @locationID OR itemID = @itemID";

// Create a new SqlCommand

SqlCommand command = new SqlCommand(selectQuery, connection);

// Add parameters for filtering

command.Parameters.AddWithValue("@adminID", string.IsNullOrEmpty(txtAdminID.Text);

command.Parameters.AddWithValue("@locationID", string.IsNullOrEmpty(txtLocationID.Text);

command.Parameters.AddWithValue("@itemID", string.IsNullOrEmpty(txtItemID.Text) ;

GridViewReport.DataBind()

# USER MANUAL FOR IVY LEAGUE MICROFINANCE SYSTEM

## Computer specifications

Minimum System Requirements for Visual Studio 2019:

Operating System:

Windows 10 version 1507 or higher (64-bit)

Windows Server 2016 or higher (64-bit)

Processor:

1.8 GHz or faster dual-core processor

Quad-core processor recommended for better performance.

Memory (RAM):

2 GB of RAM minimum (4 GB or more recommended)

Additional RAM may be required for larger projects.

Hard Disk Space:

Minimum 20 GB of free space on the system drive (SSD recommended for better performance)

Graphics:

Minimum DirectX 10 compatible graphics card with a resolution of 1024x768 or higher

Internet:

Internet connection is required for some features and services.

## Installation of Visual Studio 2022and package

Download Visual Studio:

Visit the official Visual Studio website (https://visualstudio.microsoft.com/downloads/) and download the version of Visual Studio you want to install. Choose the edition (e.g., Visual Studio Community, Professional, Enterprise) that best suits your needs.

Run the Installer:

Once the installer is downloaded, run it by double-clicking the downloaded file. This will launch the Visual Studio Installer.

Choose a Workload:

During the installation process, you'll be asked to select a workload that best suits your development needs. For ASP.NET Web Application (.NET Framework) using C#, consider choosing the "ASP.NET and web development" workload. This workload includes the necessary tools and components for web development using ASP.NET with C#.

To find this workload:

a. Click on the "Workloads" tab on the left side of the installer.

b. Scroll down or use the search bar to locate "ASP.NET and web development."

c. Check the checkbox next to "ASP.NET and web development."

Continue with Installation:

After selecting the ASP.NET and web development workload and any necessary components, click the "Install" button to proceed with the installation.

Installation Progress:

Wait for the installation to complete. Visual Studio will download and install the selected components. The duration of the installation may vary depending on your internet speed and the components chosen.

Launch Visual Studio:

Once the installation is finished, you can launch Visual Studio by clicking the "Launch" button within the installer or by finding it in your Start menu or desktop shortcut.

## Running of the file

You need to download the zip file of the project.

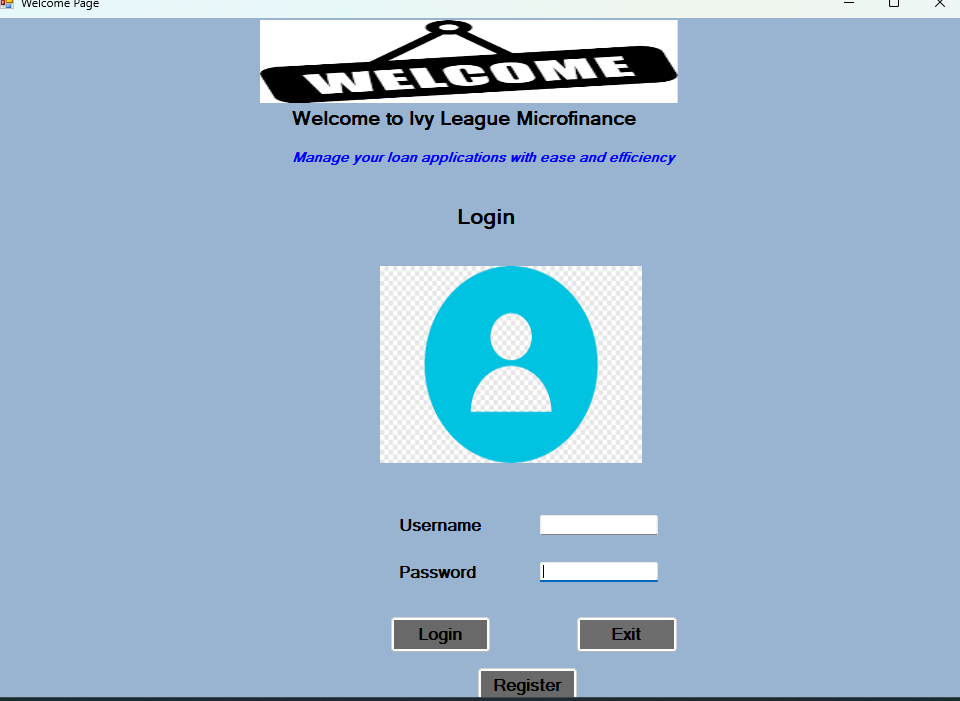
Then locate the file path of where the project is downloaded.

Extract or unzip the project file.

Open Visual Studio the click on open existing project then locate your project then click on open

Your interface will appear then click on the play button to run your project

# USER INFERFACE



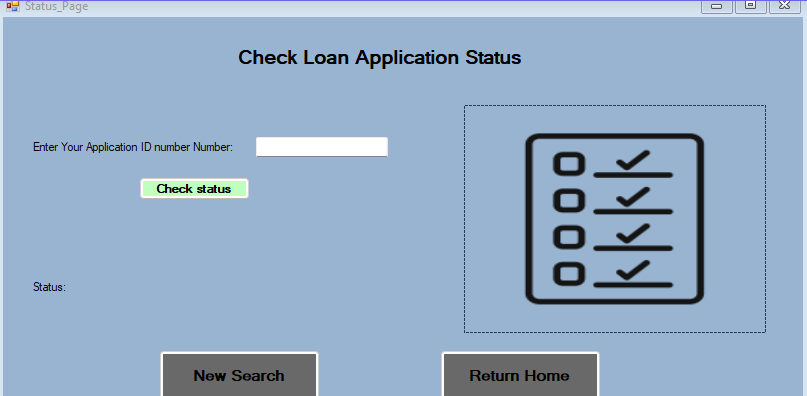
A screenshot of a computer

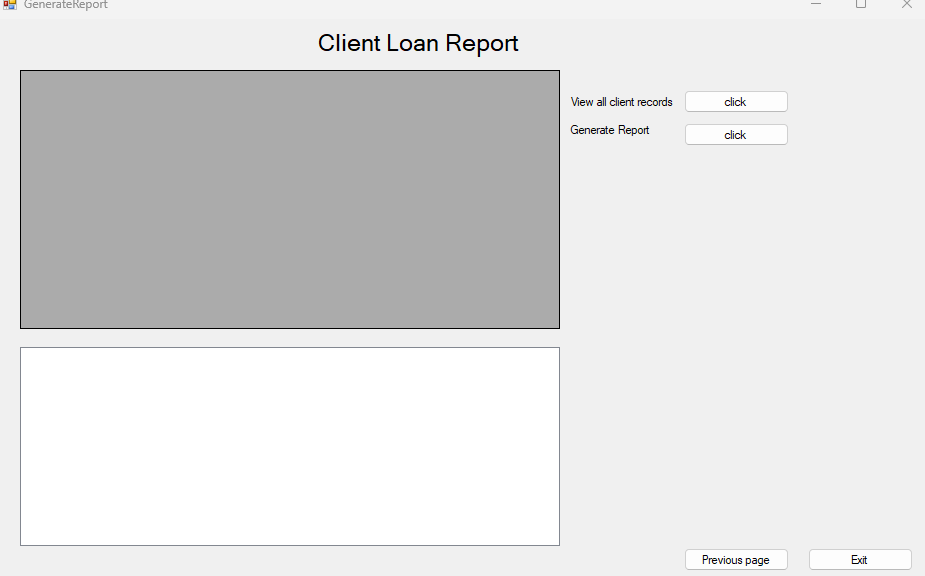
Description automatically generated

A screenshot of a computer

Description automatically generatedA screen shot of a computer

Description automatically generated





# DETAILED DAIRY

We first had our meeting where we discussed on how the interface should look. started with the user interface of the project. T Van Wyk had experienced problems uploading the project on GitHub thereafter L Sandi uploaded the project om GitHub there after we started coding and implementing changes to the project. Everyone got a fair chance on coding the project. Throughout the coding projects we recognised mistakes we made and alter the project accordingly.

This is the link to our project ‘s repository stored on github: [WestKamo/Ivy\_League\_Microfinance (github.com)](https://github.com/WestKamo/Ivy_League_Microfinance)

