National College of Ireland

Computing Project

Cybersecurity

Academic Year 2022/2023

Gvidas Virsilas

X18417984

x18417984@student.ncirl.ie

MemPass – Memorable Password Manager

Technical Report – Work in Progress

Contents

Executive Summary 2
1.0 Introduction 2
1.1. Background 2
1.2. Aims 2
1.3. Technology 3
1.4. Structure 3
2.0 System 3
2.1. Requirements 3
2.1.1. Functional Requirements 4
2.1.1.1. Use Case Diagrams 4
2.1.1.2. Description & Priority 4
2.1.1.3. Use Case 4
2.1.2. Data Requirements 13
2.1.3. User Requirements 13
2.1.4. Environmental Requirements 13
2.1.5. Usability Requirements 13
2.2. Design & Architecture 14
2.3. Implementation 16
2.4. Graphical User Interface (GUI) 24
2.5. Testing 28
2.6. Evaluation 38
3.0 Conclusions 38
4.0 Further Development or Research 39
5.0 References 39
6.0 Appendices 39
6.1. Project Proposal 40
6.2. Reflective Journals 47

Executive Summary

This paper seeks to describe the development procedure and justification for my project's password manager called MemPass. MemPass was developed to protect a user's credentials by offering the opportunity for a user to generate their password that would be unique to them and be able to store them locally to their machine and account. It aims to counter real life problems that would compromise a user's credentials such as database breaches, phishing attacks, company leaks as well as being a convenient way of helping the person remember their details by themselves. I decided to create this application due to the frustration we all experience when having too many passwords for certain sites and the inability for managers to create an actual memorable password that a user can recall without the reliance of a manager. Having this in mind I got an idea of making a "memorable" password generator and manager that would create passwords from user's input that specifically they could relate to the keywords to be used in the generation. In the report, I describe in depth the application's development, functions, testing and the results of the intended purpose of the application.

1.0 Introduction

1.1. Background

I decided to undertake this project because I was looking for a challenge in an area that I have not worked on in software development. I had gotten the idea to work on a password manager application that correlates with my specialisation by taking inspiration from the previous 4th year student project uploads on the NCI site. Passwords are almost a daily thing in people's lives, and everyone uses many passwords for different platforms, but one flaw in this is that remembering these can be a struggle especially when you have many. When I was tasked in thinking up of an idea for my Final project, I used this flaw as an objective my software will accomplish and decided to go for a password generator/manager that will create passwords unique to the person.

1.2. Aims

My project will aim to achieve the goal of generating a memorable password for the user and having the ability to save that password in the application or locally. It will be designed to be secure and fast to the user and customizable to the user's liking by answering certain questions to be included in the password. The password will be 100% randomly generated using the details provided by the user and can only be accessed with a user account specifically to that person. The user may also add extra layers of security by allowing for randomly generated special characters and alphanumeric characters to their password if they wish to add. The user will be able to finalise their password and add additions before it is updated to the database, they may also update any current stored questions, passwords and tags in the vault.

1.3. Technology
The main technologies that are used in the project currently are the I am working with for my project:

Microsoft SQL Management Studio – This is the software I am using to connect my user database to a local database to store user data.

C#– This is the coding language I used to create my project with

Visual Studio 2022– This is the IDE I used to create my project with that allows for efficient connection between all aspects of the application and access to the libraries that are integrated in the IDE

Windows Forms - This is the framework I used to create the GUI of the application and allow coding to the functions of the GUI

Microsoft Excel - Microsoft Excel is used to generate a user's vault to be able to store locally on the machine without a database

1.4. Structure
In this document, I will be discussing and demonstrating how I developed this project and break down all the functionalities of the application with the use of use cases and code snippets to show the architecture of the app. Each key element that makes up MemPass will be covered in-depth with a description, along with its GUI and all other features of the application. This report will also provide a thorough explanation of the testing procedure, the outcomes of the testing, and the presentation of the results. A conclusion that will also summarize the project's advantages and disadvantages, and its potential for the future will also be included.

2.0 System

2.1. Requirements
The main requirements that my system will have are:

- To provide the user of a way to create and login to an account to have access to the password manager
- To provide the user with a list of all the saved passwords in the account
- Delete/Update and download saved credentials
- Allow the user to generate a new password
- Allow user to save password
- Allow user to change their manager master password
- Allow user to log out

2.1.1. Functional Requirements

The functional requirements for this project are arranged from most to least important, with the most important requirements coming highest. The rating is based on how important user integrity, application functionality, security, confidentiality.

2.1.2. Use Case Diagram

Requirement 1: **User Registration**

Description & Priority:

This requirement outlines the procedure for registering into MemPass. The user has the option to create an account with their email address if they don't already have one. The reason why this is needed is because it is most crucial that the user must be logged in before they can access any of the application's capabilities but needs to have an account.

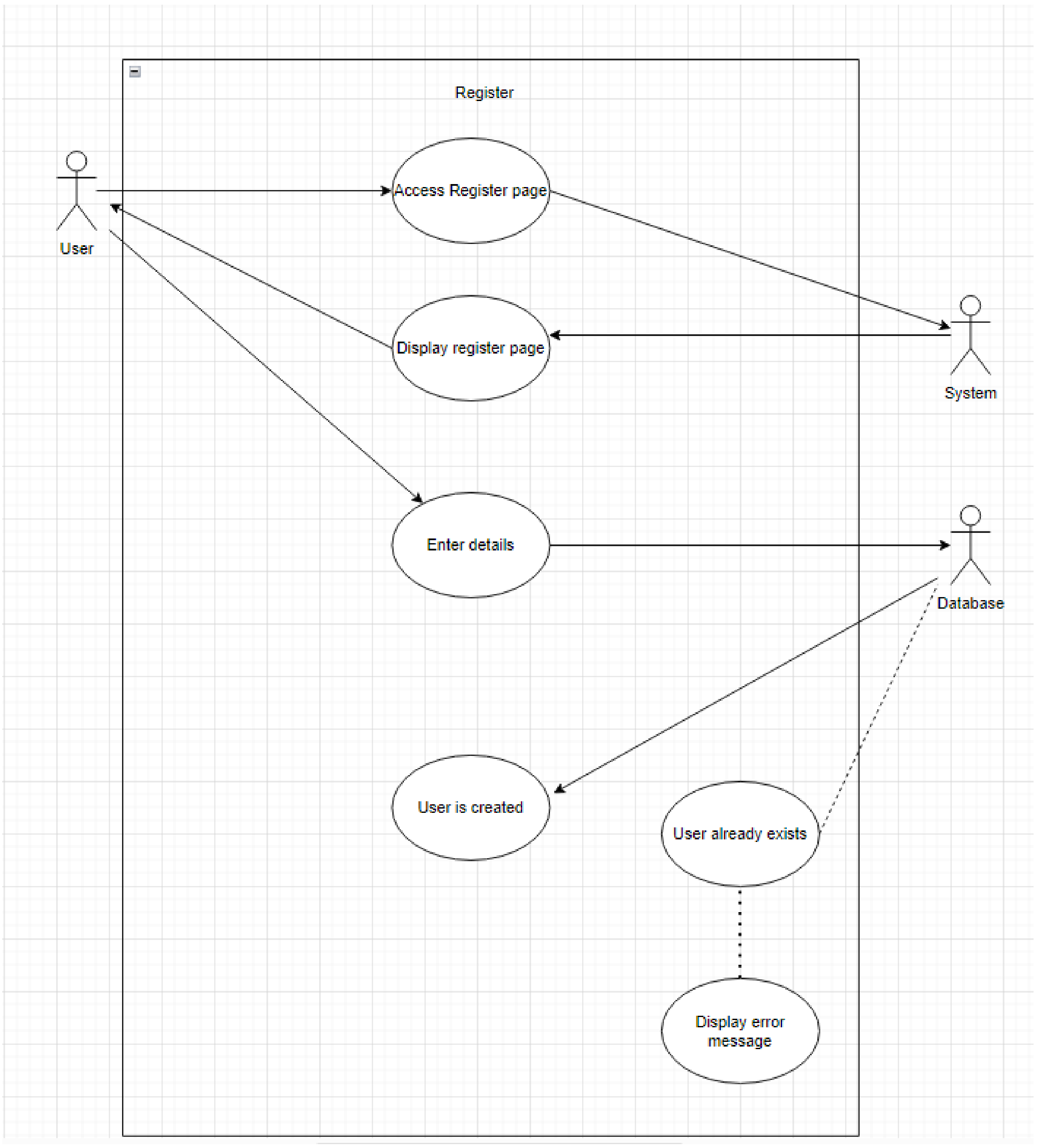
Scope

The scope of this use case is to register the user

Description

This use case describes the process of registering

Use Case Diagram



Precondition

Application and database must be connected.

Activation

This use case begins when the user accesses the application and clicks register.

Main flow

1. The user accesses the app.
2. The user navigates to register page
3. The app will display the user the register page
4. The user enters their credentials and submits via sign up button
5. System will check for user in database
6. Application will confirm user creation.

Alternate flow

None

Exceptional flow

E1: No database connection

1. Connection cuts out during any of the steps of the flow
2. The system will not proceed with any actions
3. When connection returns, the user is brought back to step 1 of the main flow

E2: The user tries to create an already existing account with same credentials

1. User enters an already existing user details.
2. System displays an error message.
3. The user is brought back to step 3 of the main flow

Termination

The use case is terminated when the user is successfully registered into the database and is greeted with a message.

Post condition

Database monitors every action inputted from the user

Requirement 2: **User login**

Description & Priority:

This requirement outlines the procedure for logging into MemPass. The user has the option to login to their account with their email address and password if they don't already have one. The reason why this is needed is because it is most crucial that the user must be logged in before they can access any of the application's capabilities.

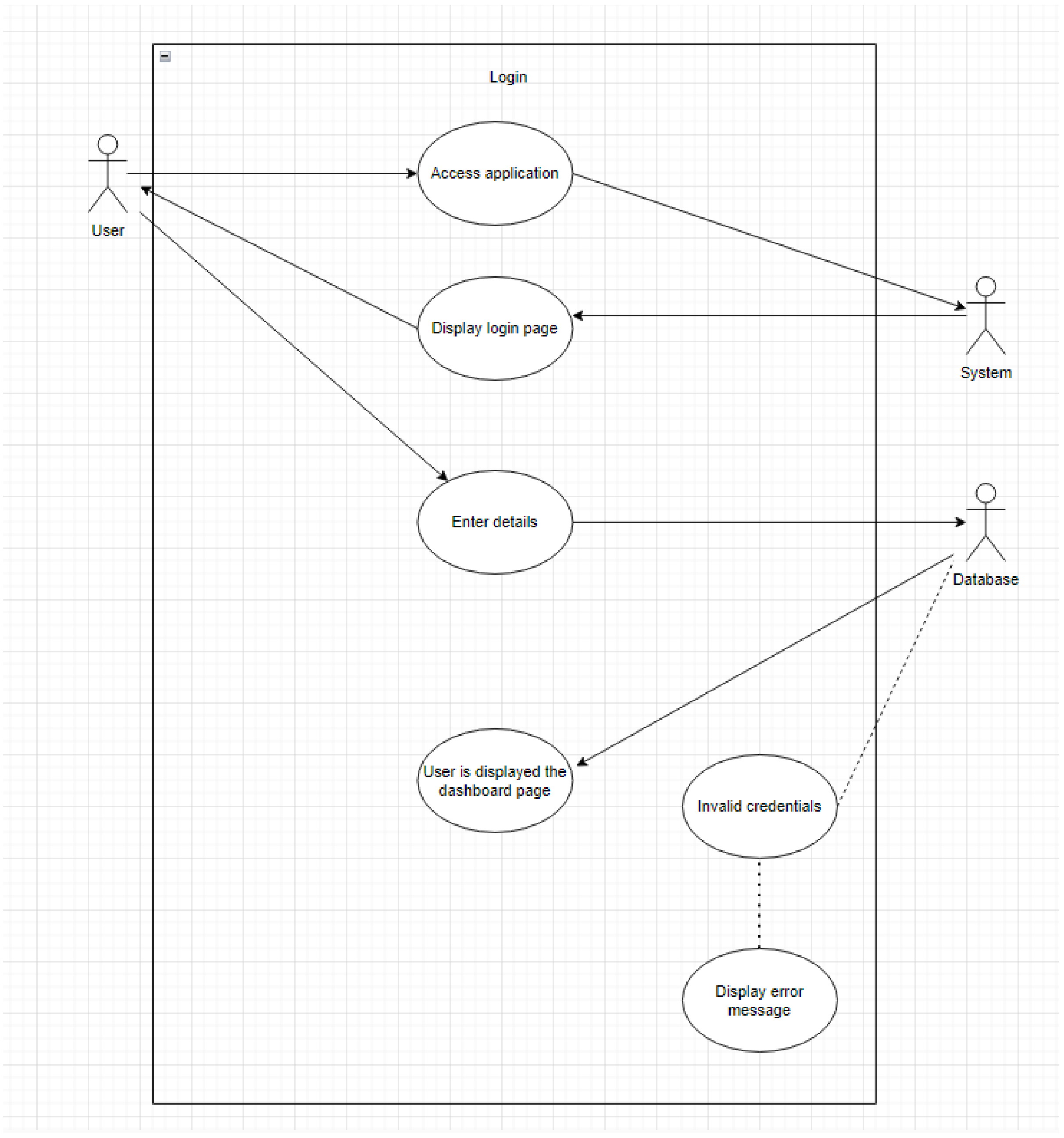
Scope

The scope of this use case is to log in the user for access

Description

This use case describes the process of logging in

Use Case Diagram



Precondition

User must have an account created.

Activation

This use case begins when the user accesses the application.

Main flow

1. The user accesses the application.
2. The system displays the login page.
3. User enters their credentials and clicks “Login”
4. System verifies in database user exists
5. System displays the dashboard page

Alternate flow

A1. User does not have an account

1. User clicks sign up button
2. Application displays sign up page
3. User creates an account with their credentials
4. The use case continues at position 1 of the main flow

Exceptional flow

E1: No database connection

1. Connection cuts out during any of the steps of the flow
2. The system will not proceed with any actions
3. When connection returns, the user is brought back to step 1 of the main flow

E2: The user tries to login without signing up,

1. User enters invalid user credentials.
2. System displays an error message.
3. The user is brought back to step 1 of the main flow

Termination

The use case is terminated when the user is successfully logged into the application and is greeted with a message.

Post condition

Database monitors every action inputted from the user

Requirement 3: **Password manager**

Description & Priority:

This requirement outlines the procedure for generating and adding a password into the user's MemPass account by answering 3 security questions. The user has the option to change their master password to their MemPass account only when logged in. The reason why this is because it acts as a security feature so your password cannot be recovered by a malicious user. This page is also where the user will access the main features of the application such as generating their password and exporting it as an Excel file and managing their vault. It is most crucial that the user must be logged in before they can access any of the application's capabilities and only the user can access their own passwords.

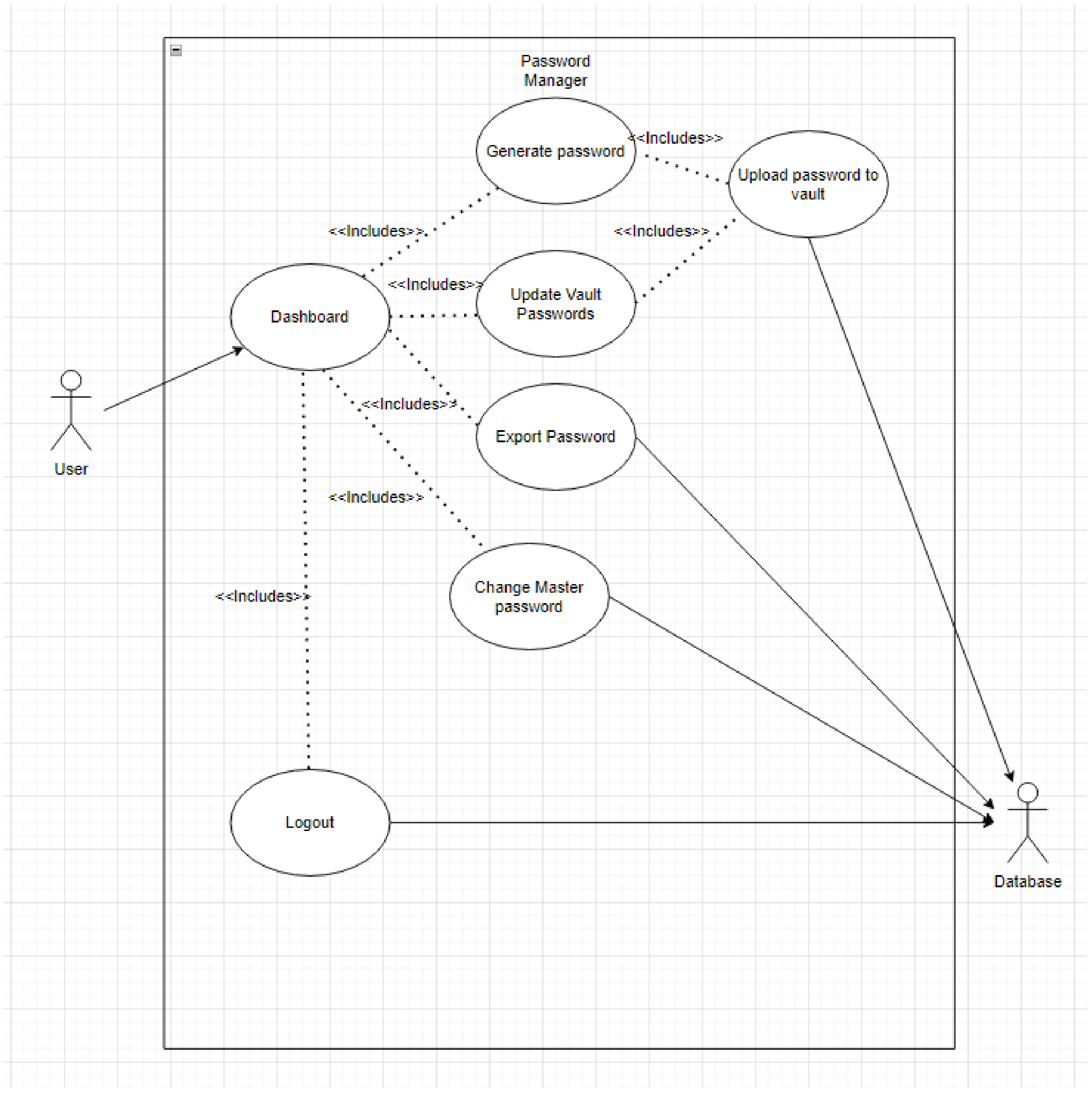
Scope

The scope of this use case is to show a user how to operate the password manager and become familiar with its functionalities. It also explains how the manager stores their credentials into the database.

Description

This use case describes the process of the password manager, database, and functionalities with the user. The use case also demonstrates how the password manager stores the data in the database.

Use Case Diagram



Precondition

Users must be logged in with a database connection.

Activation

This use case begins when the user signs into the application.

Main flow

1. The user accesses the dashboard.
2. User chooses their feature in the application, by inputting their details.
3. Users update their details with corresponding buttons.
4. System verifies changes and uploads to the database.

Alternate flow

A1. User does not enter details correctly in corresponding fields.

1. User doesn't fill the fields for the desired feature.
2. Application displays an error message.
3. The use case continues at position 2 of the main flow.

Exceptional flow

E1: No database connection

1. Connection cuts out during any of the steps of the flow
2. The system will not proceed with any actions
3. When connection returns, the user is brought back to step 1 of the main flow

Termination

The use case is terminated when the user is satisfied with their interaction with the application and closes the application or logs out.

Post condition

Database monitors every action inputted from the user

2.1.2. Data Requirements

In addition to allowing complete control over passwords, MemPass should adhere to the tightest industry requirements for data confidentiality, integrity, and availability. The password manager requires information from the user such as their 3 security question answers, password and email and inputs for check boxes. This is required to generate a password securely and with ease.

2.1.3. User Requirements

The user should create a strong and memorable master password to access the application. It is important to be unique, creative, lengthy, and unpredictable when creating your master password to avoid attacks such as brute forcing and dictionary attacks. The user should follow safe password keeping practices such as regular changes and avoiding sharing with other people.

The project also provides the user:

- Security
- Confidentiality
- Integrity
- Usability
- Consistency
- Reliability

2.1.4. Environmental Requirements

Since there are no audio features in this application, the user experience won't be impacted by outside noises or disruptions.

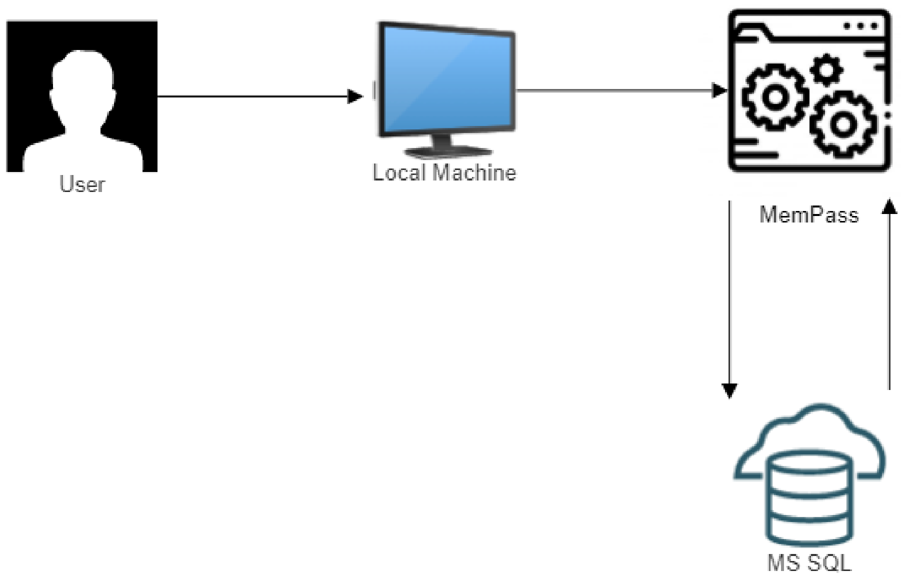
2.1.5. Usability Requirements

The interface includes features that make learning and recalling how to use it easier. There isn’t any necessity for constant manual reference by users. Pop up messages and labels are provided for user convenience and only necessary features are implemented to prevent confusion for the user.

The usability requirements in the project include:

- **Efficiency of use:** Objectives may be attained fast and with little to no user mistake.
- **Intuitiveness:** The interface is straightforward to use and navigate; the buttons, headers, and error and help messages are clear.
- **Low perceived workload:** Rather than being intimidating, hard, or annoying, the interface is simple to use.

2.2. Design & Architecture



The system architecture is depicted in the above figure. The user can utilize their machine to access the program. MemPass is a C# application so it has multi-platform compatibility, and it can be used on many devices no matter the operating system, as long as it supports the C# application and database.

The program operates in a local, offline, and secure environment, a user does not need Internet access.

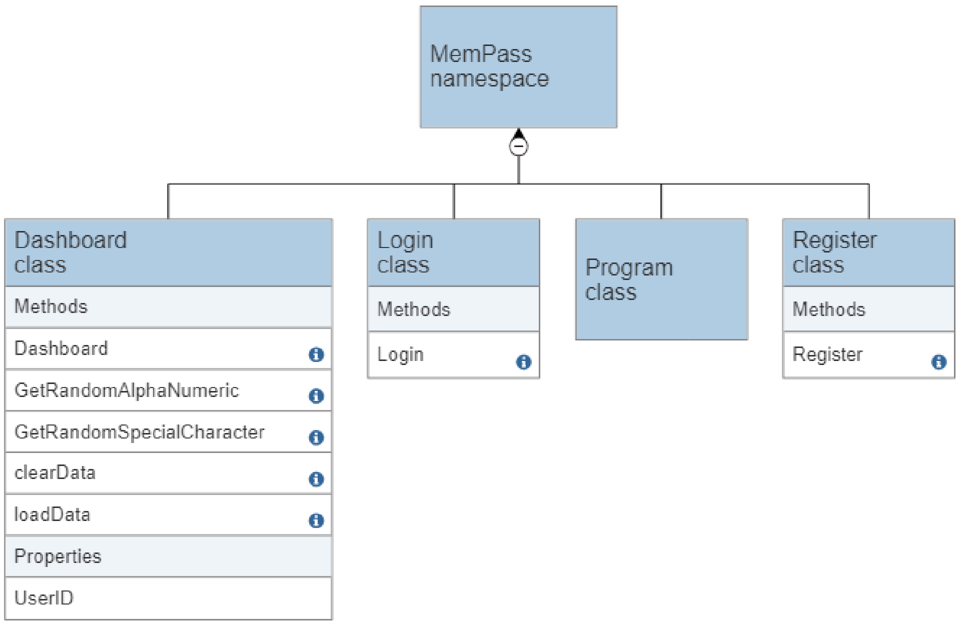
The system is formed in the front-end using my knowledge in Windows Forms. Windows Forms is the UI framework integrated as part of Visual Studio's that was used to create the GUI and add individual functions to the buttons and layout of the application.

Using Windows Forms I was able to create the layout of my whole application to make it aesthetically pleasing and easy to navigate. I had also decided to use Windows Forms as it allows data binding between its controls and the database and gives easy data exchange between the two. Windows Forms also allows for validation and makes sure the user enters the specific criteria and is able to display error messages if the incorrect information is provided.

The application passes all of the data to the database which is also on the user's machine which the application is running on. The application communicates with the database by storing the user's inputs in their respective accounts. Each user is unique and can only see their own stored passwords and no one else.

The design of the project is purposefully made simple and easy to navigate. I designed it on a single page and the user can access all of the features in one place without any hassle.

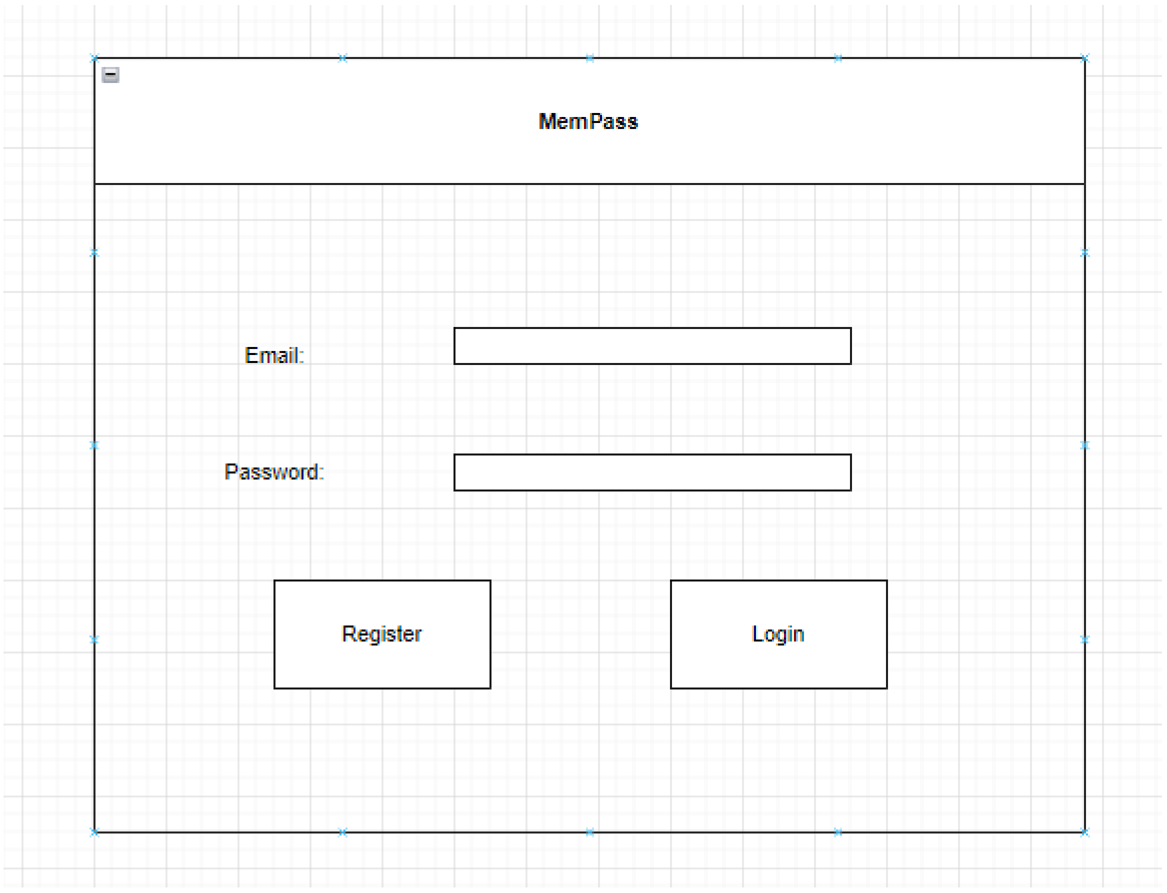
Class Diagram:



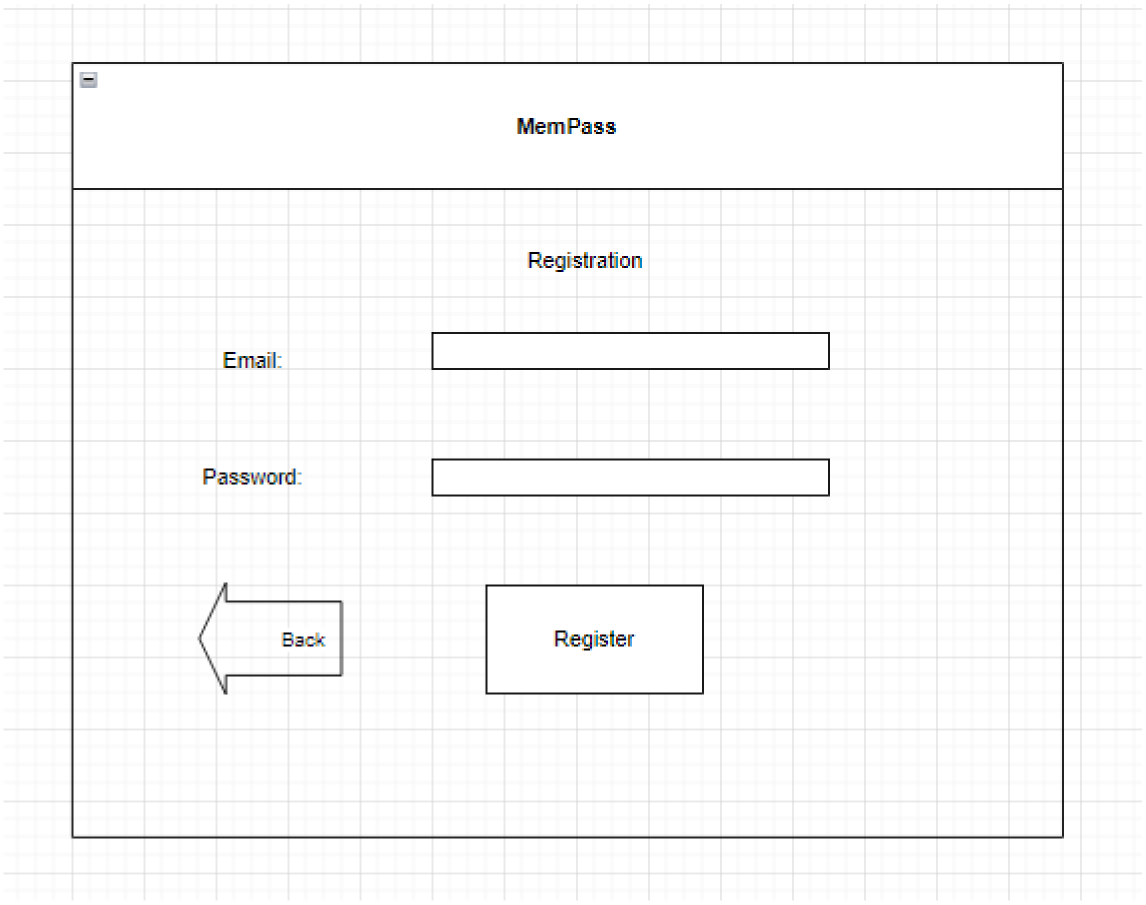
2.3. Implementation

Concept Wireframes:

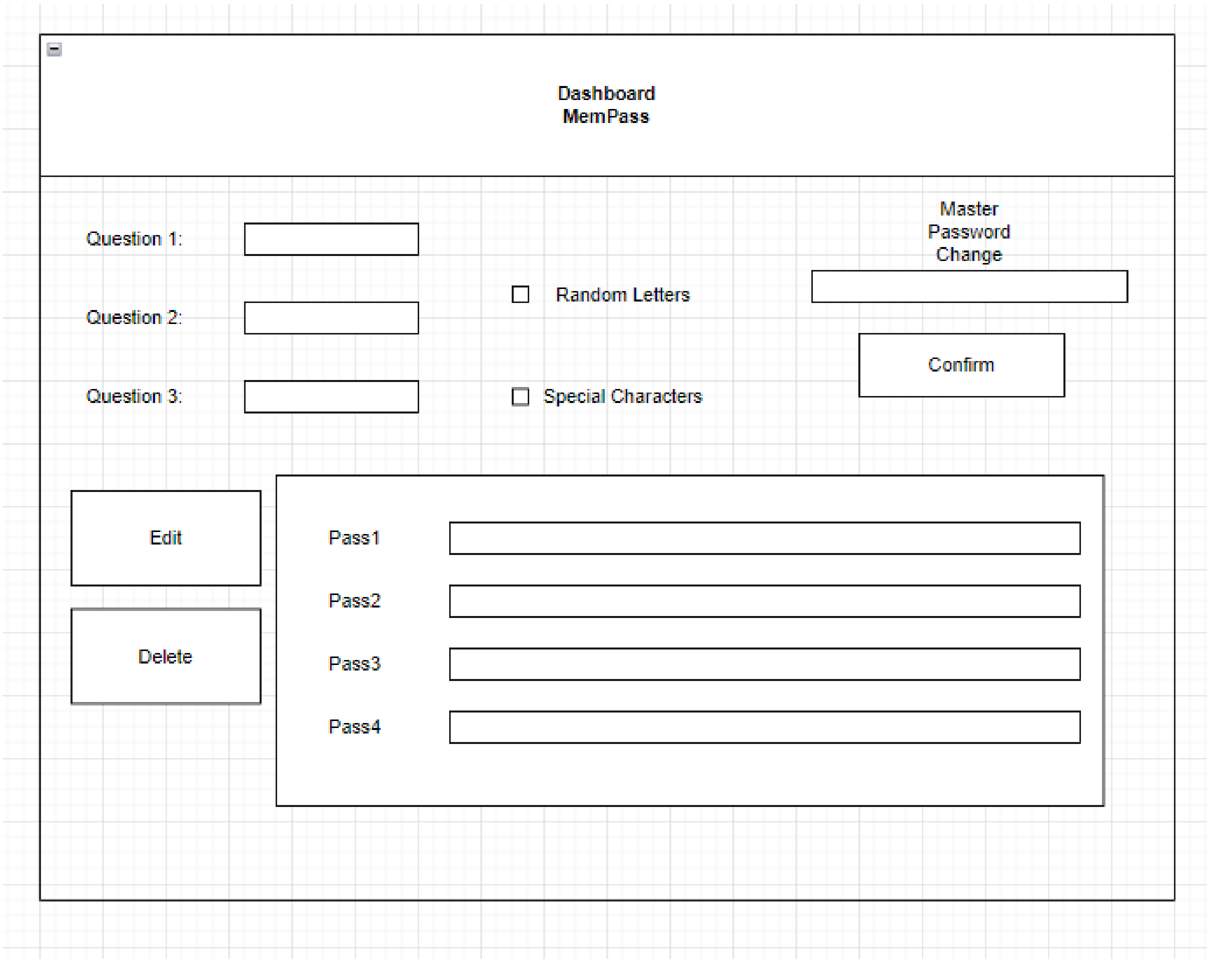
Login:



Sign-up:

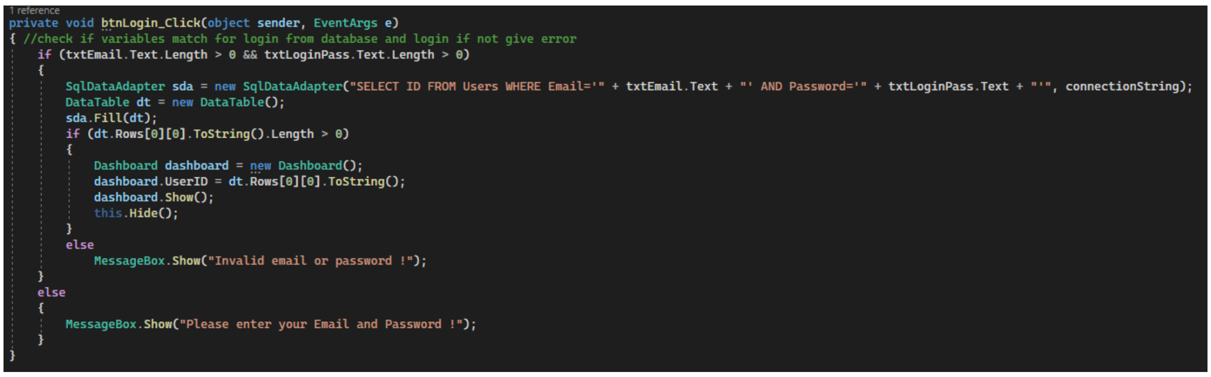


Dashboard:


Dashboard
MemPass
Question 1:
Question 2:
Question 3:
Random Letters
Special Characters
Master Password Change
Confirm
Edit
Delete
Pass1
Pass2
Pass3
Pass4

Login and registration is one of the top priorities in creating the application. It enables authorized users to access their data by identifying themselves and can be used to protect their sensitive information.

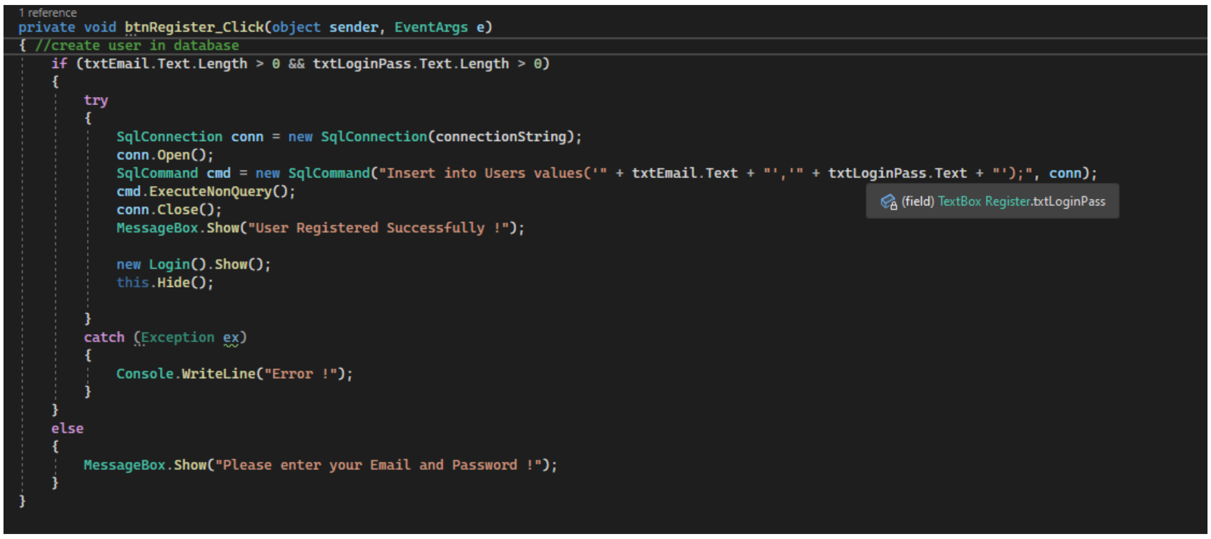
Login:



```
1 reference
private void btnLogin_Click(object sender, EventArgs e)
{ //check if variables match for login from database and login if not give error
    if (txtEmail.Text.Length > 0 && txtLoginPass.Text.Length > 0)
    {
        SqlDataAdapter sda = new SqlDataAdapter("SELECT ID FROM Users WHERE Email='" + txtEmail.Text + "' AND Password='" + txtLoginPass.Text + "'", connectionString);
        DataTable dt = new DataTable();
        sda.Fill(dt);
        if (dt.Rows[0][0].ToString().Length > 0)
        {
            Dashboard dashboard = new Dashboard();
            dashboard.UserID = dt.Rows[0][0].ToString();
            dashboard.Show();
            this.Hide();
        }
        else
            MessageBox.Show("Invalid email or password !");
    }
    else
    {
        MessageBox.Show("Please enter your Email and Password !");
    }
}
```

The logic in the attached code snippet checks to see if the login text fields have been filled in. If they do, the SQL code is called to search the database for an email and password combination that matches. The User ID linked to the supplied credentials is then fetched from the database. The program obtains the user's information from the database and launches the dashboard page after successfully validating the email and password.

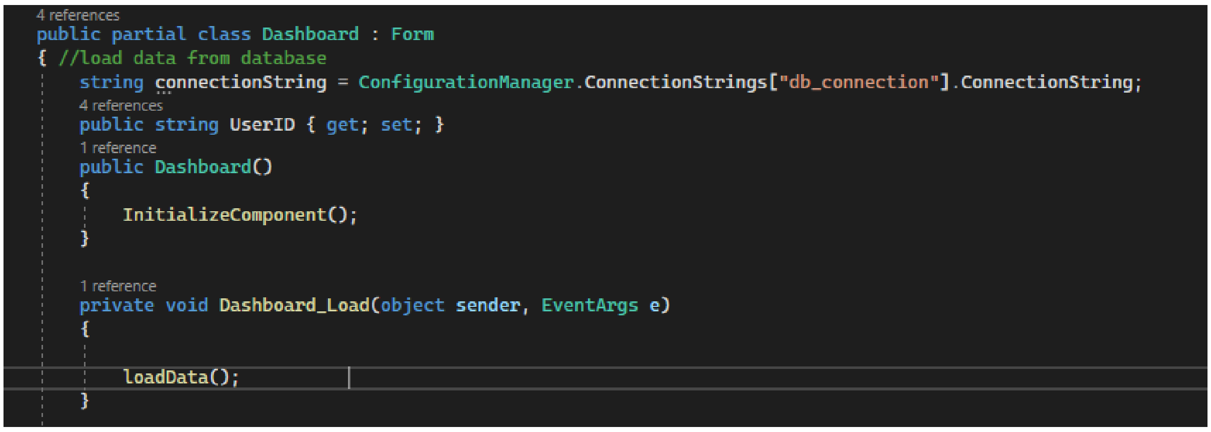
Registration:



```
1 reference
private void btnRegister_Click(object sender, EventArgs e)
{ //create user in database
    if (txtEmail.Text.Length > 0 && txtLoginPass.Text.Length > 0)
    {
        try
        {
            SqlConnection conn = new SqlConnection(connectionString);
            conn.Open();
            SqlCommand cmd = new SqlCommand("Insert into Users values('" + txtEmail.Text + "','" + txtLoginPass.Text + "');", conn);
            cmd.ExecuteNonQuery();
            conn.Close();
            MessageBox.Show("User Registered Successfully !");

            new Login().Show();
            this.Hide();

        }
        catch (Exception ex)
        {
            Console.WriteLine("Error !");
        }
    }
    else
    {
        MessageBox.Show("Please enter your Email and Password !");
    }
}
```

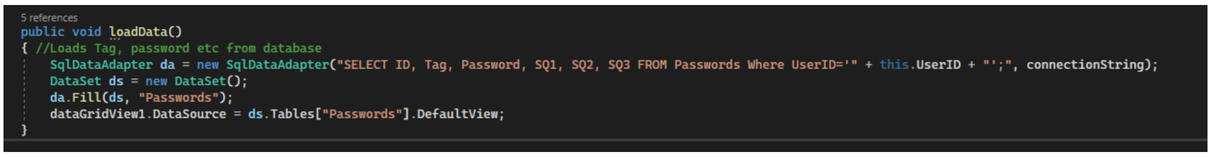
Similar to how the login page works, the registration page creates a new user account linked to the email and password the user enters into the appropriate fields without first retrieving the user's information from the database.

Dashboard:



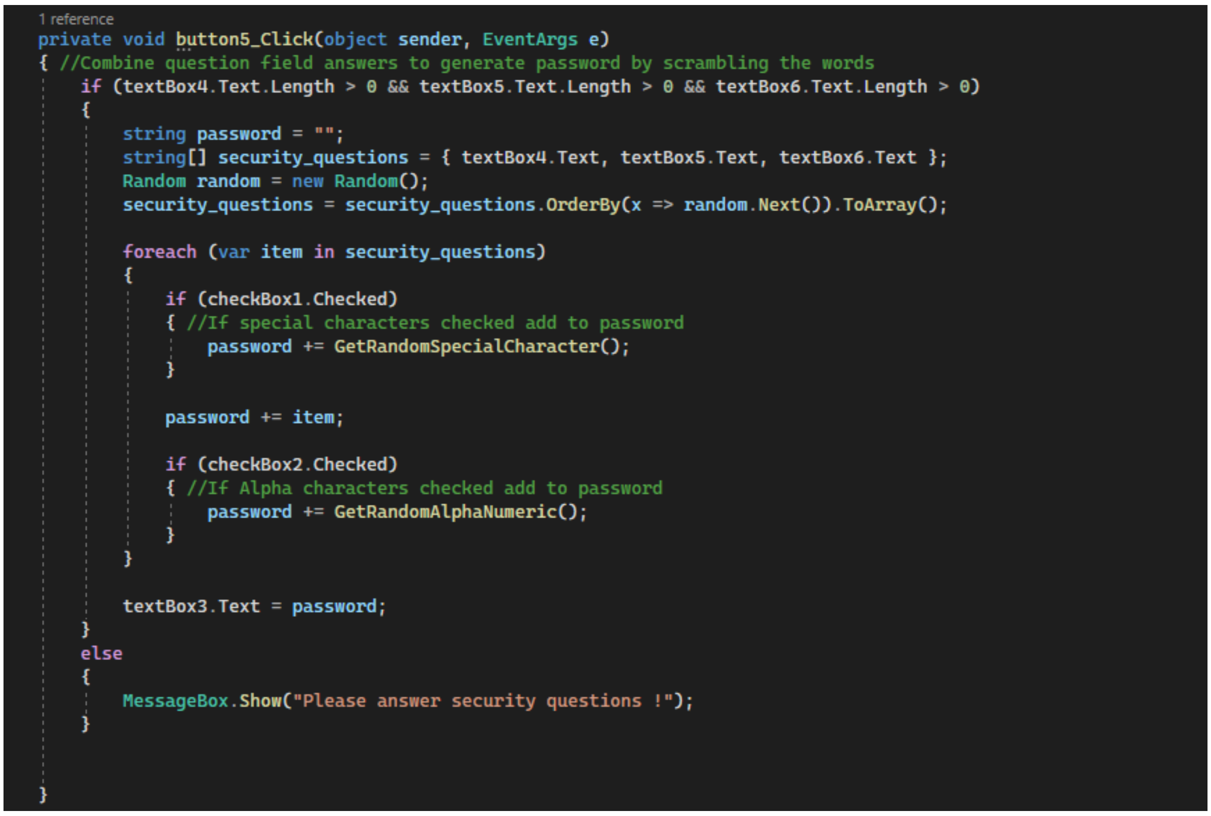
```
4 references
public partial class Dashboard : Form
{ //load data from database
    string connectionString = ConfigurationManager.ConnectionStrings["db_connection"].ConnectionString;
    4 references
    public string UserID { get; set; }
    1 reference
    public Dashboard()
    {
        InitializeComponent();
    }

    1 reference
    private void Dashboard_Load(object sender, EventArgs e)
    {

        loadData();
    }
```

The following code snippet shown loads the related data to be presented in the dashboard's vault section after receiving the User ID from the database as soon as the dashboard is loaded.



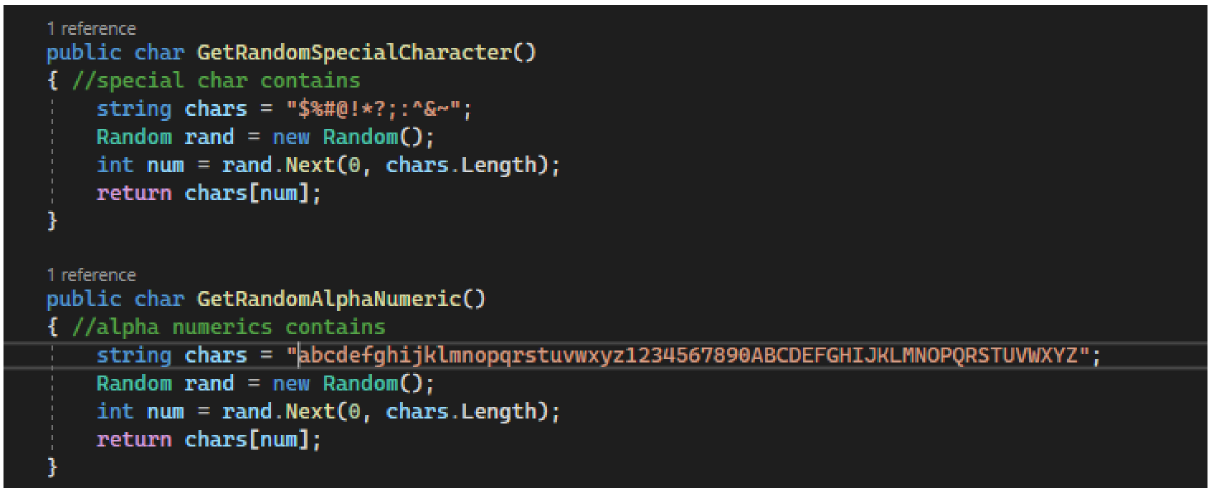
```
5 references
public void loadData()
{ //Loads Tag, password etc from database
    SqlDataAdapter da = new SqlDataAdapter("SELECT ID, Tag, Password, SQ1, SQ2, SQ3 FROM Passwords Where UserID='" + this.UserID + "';", connectionString);
    DataSet ds = new DataSet();
    da.Fill(ds, "Passwords");
    dataGridView1.DataSource = ds.Tables["Passwords"].DefaultView;
}
```

The following code snippet instructs it to populate the GridView using the password ID, which has been constructed at the bottom of the vault section, in a row-oriented fashion, with the appropriate data to be fetched from the database. A password ID contains the tag, password, the three secret questions and User ID.



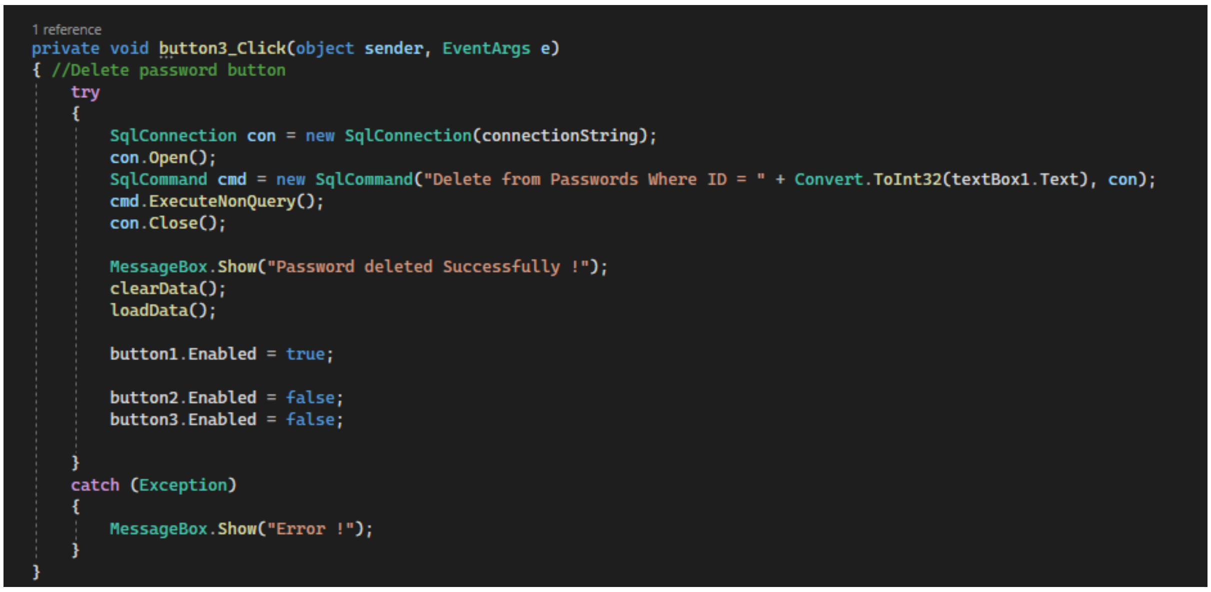
```
1 reference
private void button5_Click(object sender, EventArgs e)
{ //Combine question field answers to generate password by scrambling the words
    if (textBox4.Text.Length > 0 && textBox5.Text.Length > 0 && textBox6.Text.Length > 0)
    {
        string password = "";
        string[] security_questions = { textBox4.Text, textBox5.Text, textBox6.Text };
        Random random = new Random();
        security_questions = security_questions.OrderBy(x => random.Next()).ToArray();

        foreach (var item in security_questions)
        {
            if (checkBox1.Checked)
            { //If special characters checked add to password
                password += GetRandomSpecialCharacter();
            }

            password += item;

            if (checkBox2.Checked)
            { //If Alpha characters checked add to password
                password += GetRandomAlphaNumeric();
            }
        }

        textBox3.Text = password;
    }
    else
    {
        MessageBox.Show("Please answer security questions !");
    }


}
```

This code snippet illustrates how a password string may be generated depending on user input. It checks the user's inputs in each of the three text boxes for the secret question first. It gathers all of these distinct inputs and saves them in an array for later randomization, which reorders them. The password variable is created after the scrambling procedure. Additionally, the user can add additional characters to the password string by using the Special Characters and/or AlphaNumeric checkboxes. The generated password is shown in a different textbox. The user can keep clicking the same button for different combinations of a password the user would like.



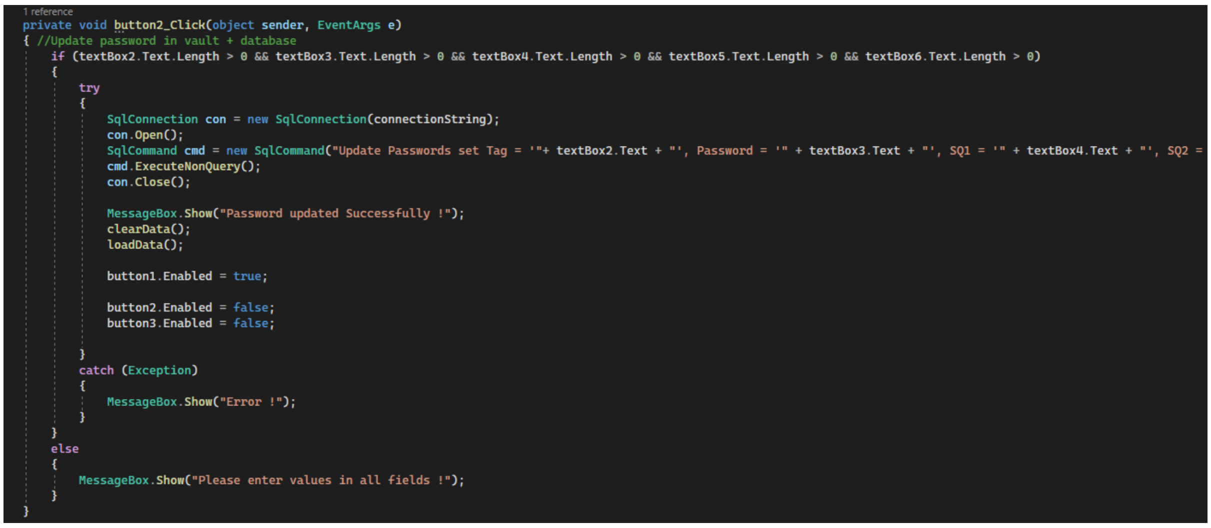
```
1 reference
public char GetRandomSpecialCharacter()
{ //special char contains
    string chars = "$%#@!*?;:^&~";
    Random rand = new Random();
    int num = rand.Next(0, chars.Length);
    return chars[num];
}

1 reference
public char GetRandomAlphaNumeric()
{ //alpha numerics contains
    string chars = "abcdefghijklmnopqrstuvwxyz1234567890ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    Random rand = new Random();
    int num = rand.Next(0, chars.Length);
    return chars[num];
}
```

This code snippet shows the character bank and to pick one for the generation in the figure before this.



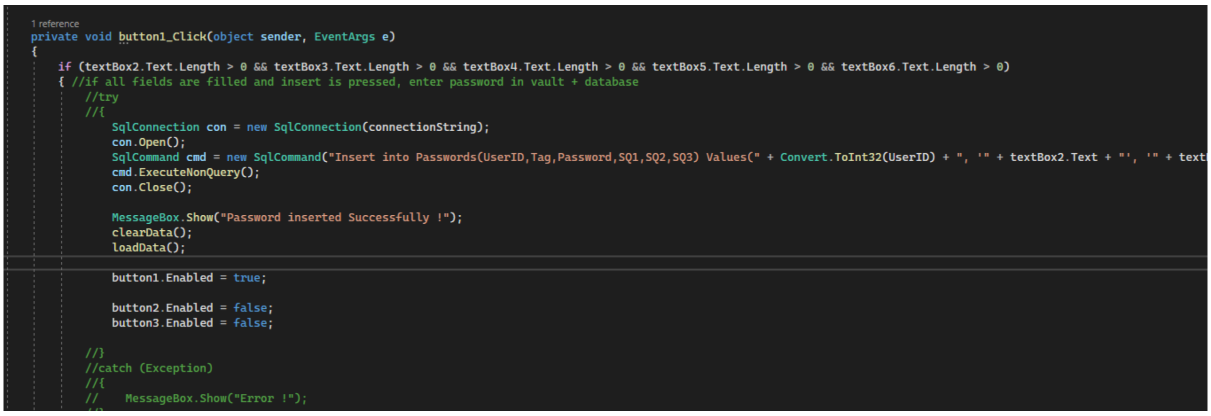
```
1 reference
private void button3_Click(object sender, EventArgs e)
{ //Delete password button
    try
    {
        SqlConnection con = new SqlConnection(connectionString);
        con.Open();
        SqlCommand cmd = new SqlCommand("Delete from Passwords Where ID = " + Convert.ToInt32(textBox1.Text), con);
        cmd.ExecuteNonQuery();
        con.Close();

        MessageBox.Show("Password deleted Successfully !");
        clearData();
        loadData();

        button1.Enabled = true;

        button2.Enabled = false;
        button3.Enabled = false;

    }
    catch (Exception)
    {
        MessageBox.Show("Error !");
    }
}
```

The above code sample shows how things work when a user selects a password from the vault and clicks the delete button to start the deletion process. The code then launches a database call to delete the relevant material connected to the distinct ID created for the chosen password within the vault. Following the display of a confirmation notice that the password has been successfully deleted, the screen is refreshed to provide the most recent list of passwords that are still active in the vault.

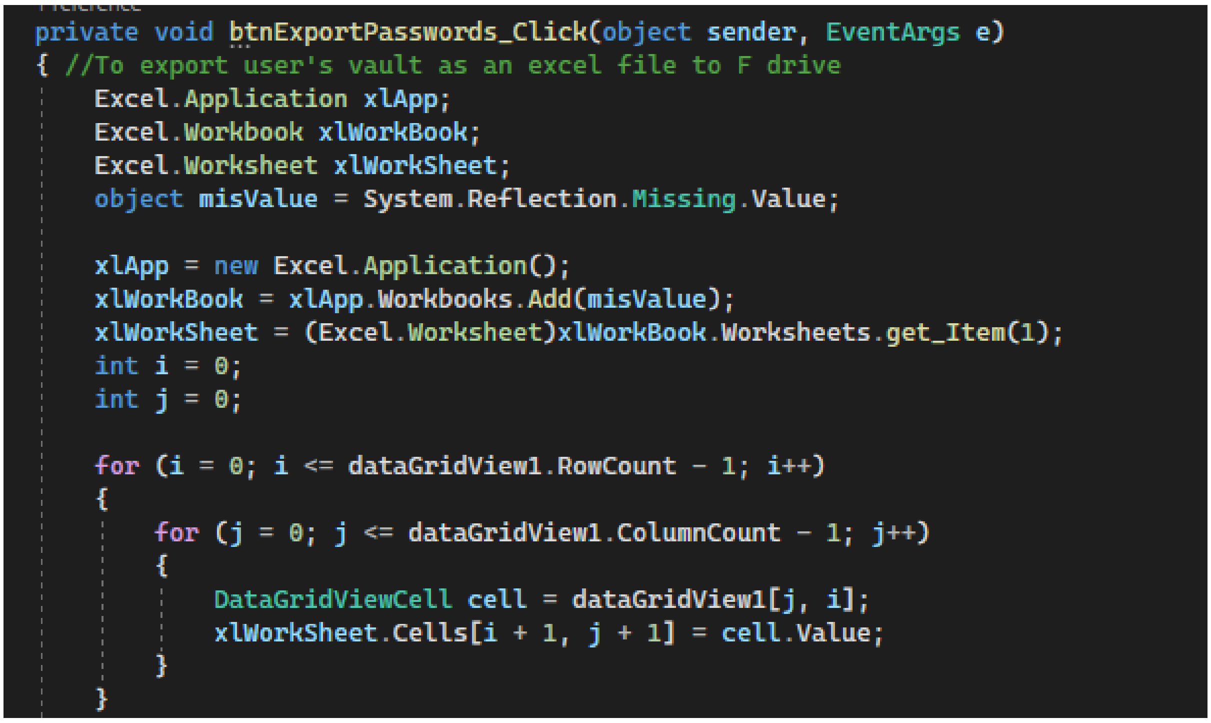


```
1 reference
private void button2_Click(object sender, EventArgs e)
{ //Update password in vault + database
    if (textBox2.Text.Length > 0 && textBox3.Text.Length > 0 && textBox4.Text.Length > 0 && textBox5.Text.Length > 0 && textBox6.Text.Length > 0)
    {
        try
        {
            SqlConnection con = new SqlConnection(connectionString);
            con.Open();
            SqlCommand cmd = new SqlCommand("Update Passwords set Tag = '"+ textBox2.Text + "', Password = '" + textBox3.Text + "', SQ1 = '" + textBox4.Text + "', SQ2 =
            cmd.ExecuteNonQuery();
            con.Close();

            MessageBox.Show("Password updated Successfully !");
            clearData();
            loadData();

            button1.Enabled = true;

            button2.Enabled = false;
            button3.Enabled = false;

        }
        catch (Exception)
        {
            MessageBox.Show("Error !");
        }
    }
    else
    {
        MessageBox.Show("Please enter values in all fields !");
    }
}
```

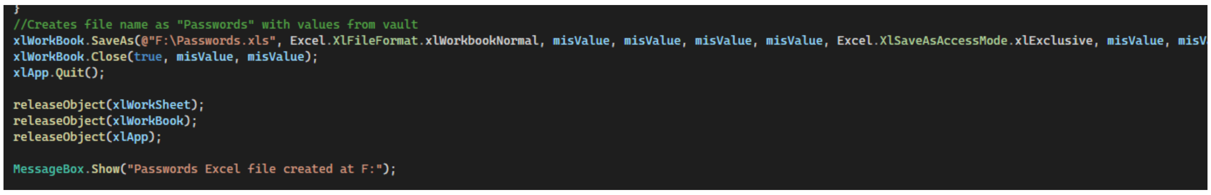
A similar procedure for updating a password is shown in the accompanying code snippet. When a user activates the update password button, the code tells the database to update the password information with the freshly produced information collected from the text fields. This makes sure that any modifications made by the user are reflected in the database's password information.


```
1 reference
private void button1_Click(object sender, EventArgs e)
{
    if (textBox2.Text.Length > 0 && textBox3.Text.Length > 0 && textBox4.Text.Length > 0 && textBox5.Text.Length > 0 && textBox6.Text.Length > 0)
    { //if all fields are filled and insert is pressed, enter password in vault + database
        //try
        //{
            SqlConnection con = new SqlConnection(connectionString);
            con.Open();
            SqlCommand cmd = new SqlCommand("Insert into Passwords(UserID,Tag,Password,SQ1,SQ2,SQ3) Values(" + Convert.ToInt32(UserID) + ", '" + textBox2.Text + "', '" + text
            cmd.ExecuteNonQuery();
            con.Close();

            MessageBox.Show("Password inserted Successfully !");
            clearData();
            loadData();

            button1.Enabled = true;

            button2.Enabled = false;
            button3.Enabled = false;

        //}
        //catch (Exception)
        //{
        //    MessageBox.Show("Error !");
```

This code snippet illustrates the behaviour that occurs when the insert button is hit after the text fields have been filled with data. When the button is pressed, the code extracts all the string values from the fields and gives each one a special ID. The program then uses this ID as a reference to access the associated data from the database. This ID creates a connection between the user and the data submitted, enabling later modifications or deletions of the connected material in response to user requests.


```
private void btnExportPasswords_Click(object sender, EventArgs e)
{ //To export user's vault as an excel file to F drive
    Excel.Application xlApp;
    Excel.Workbook xlWorkBook;
    Excel.Worksheet xlWorkSheet;
    object misValue = System.Reflection.Missing.Value;

    xlApp = new Excel.Application();
    xlWorkBook = xlApp.Workbooks.Add(misValue);
    xlWorkSheet = (Excel.Worksheet)xlWorkBook.Worksheets.get_Item(1);
    int i = 0;
    int j = 0;

    for (i = 0; i <= dataGridView1.RowCount - 1; i++)
    {
        for (j = 0; j <= dataGridView1.ColumnCount - 1; j++)
        {
            DataGridViewCell cell = dataGridView1[j, i];
            xlWorkSheet.Cells[i + 1, j + 1] = cell.Value;
        }
    }
```

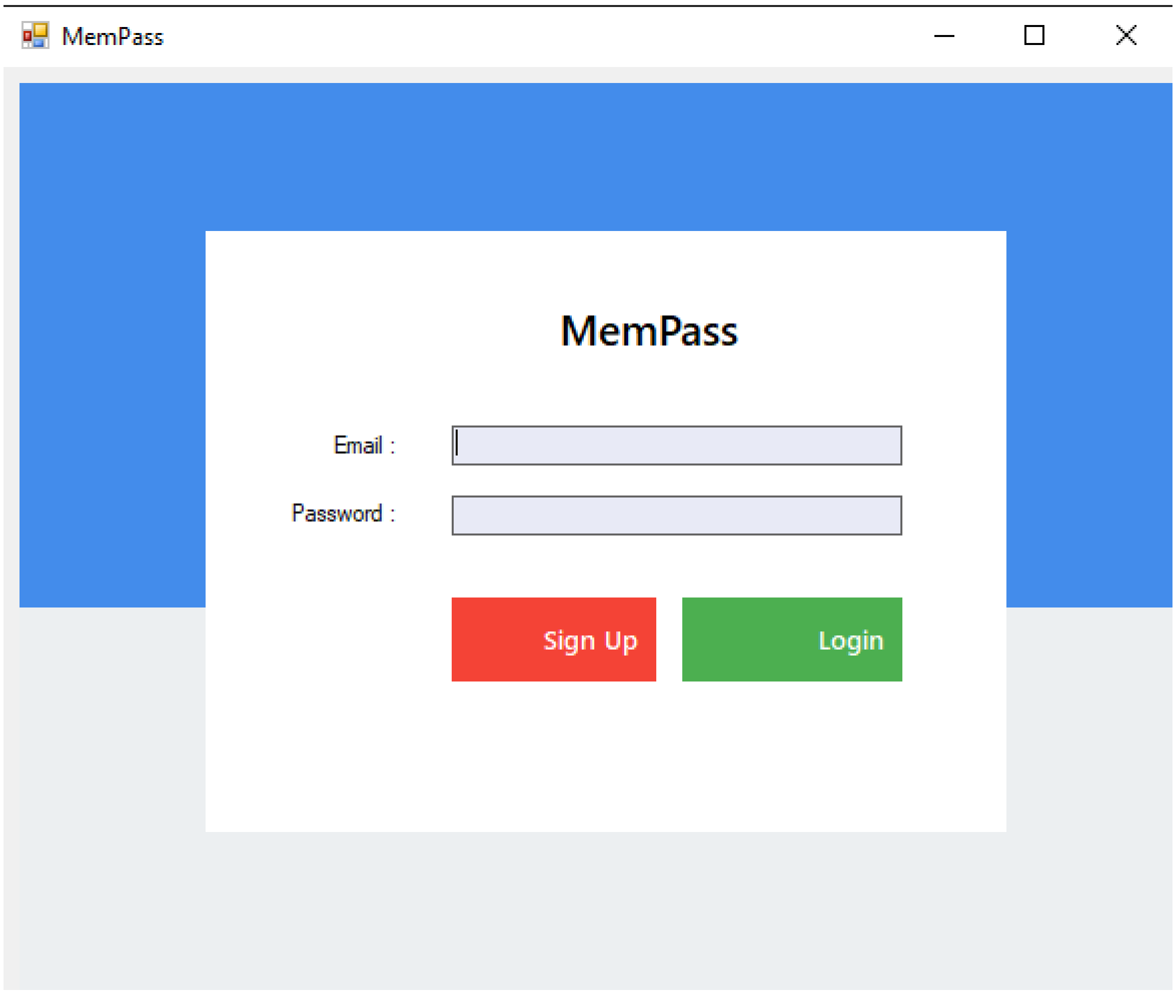
This code snippet demonstrates the action for the export password button. It is telling the excel library to pull the values from the GridView and export them in the same format present in the vault.


```
}
//Creates file name as "Passwords" with values from vault
xlWorkBook.SaveAs(@"F:\Passwords.xls", Excel.XlFileFormat.xlWorkbookNormal, misValue, misValue, misValue, misValue, Excel.XlSaveAsAccessMode.xlExclusive, misValue, misV
xlWorkBook.Close(true, misValue, misValue);
xlApp.Quit();

releaseObject(xlWorkSheet);
releaseObject(xlWorkBook);
releaseObject(xlApp);

MessageBox.Show("Passwords Excel file created at F:");
```

Once it is generated it will be named “Passwords” and be saved in the user’s F: drive locally on their machine as an excel file.

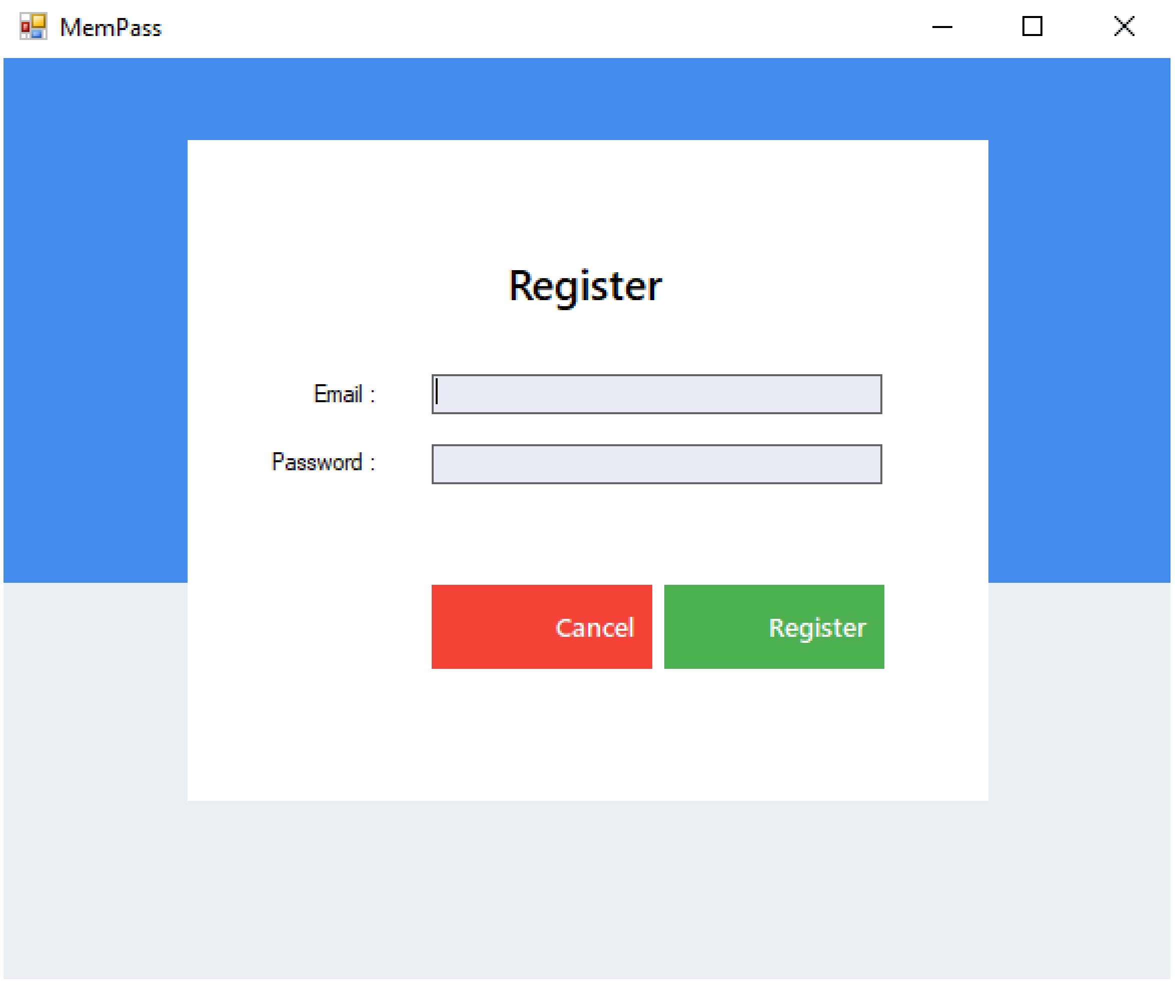
2.4. Graphical User Interface (GUI)

Login page is first shown when the application is launched:

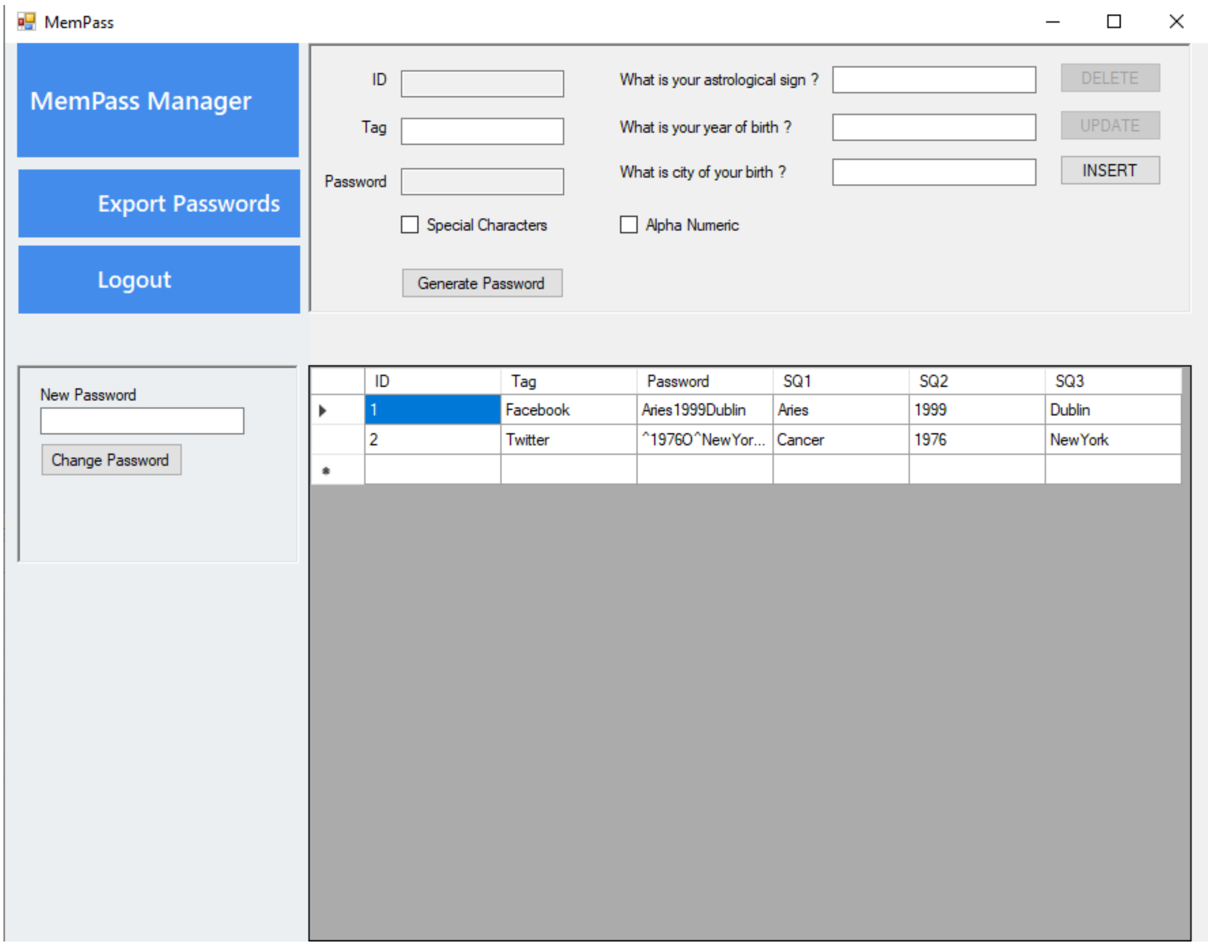


User can enter their account details and click login or press “Sign up” to be redirected to the sign up page

Registration page:

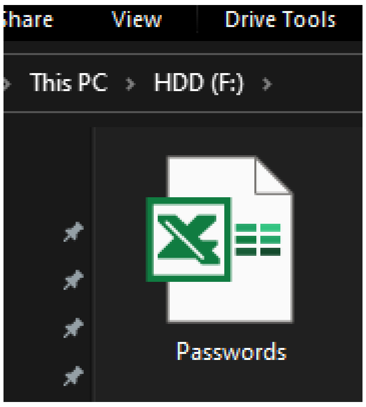


Dashboard:

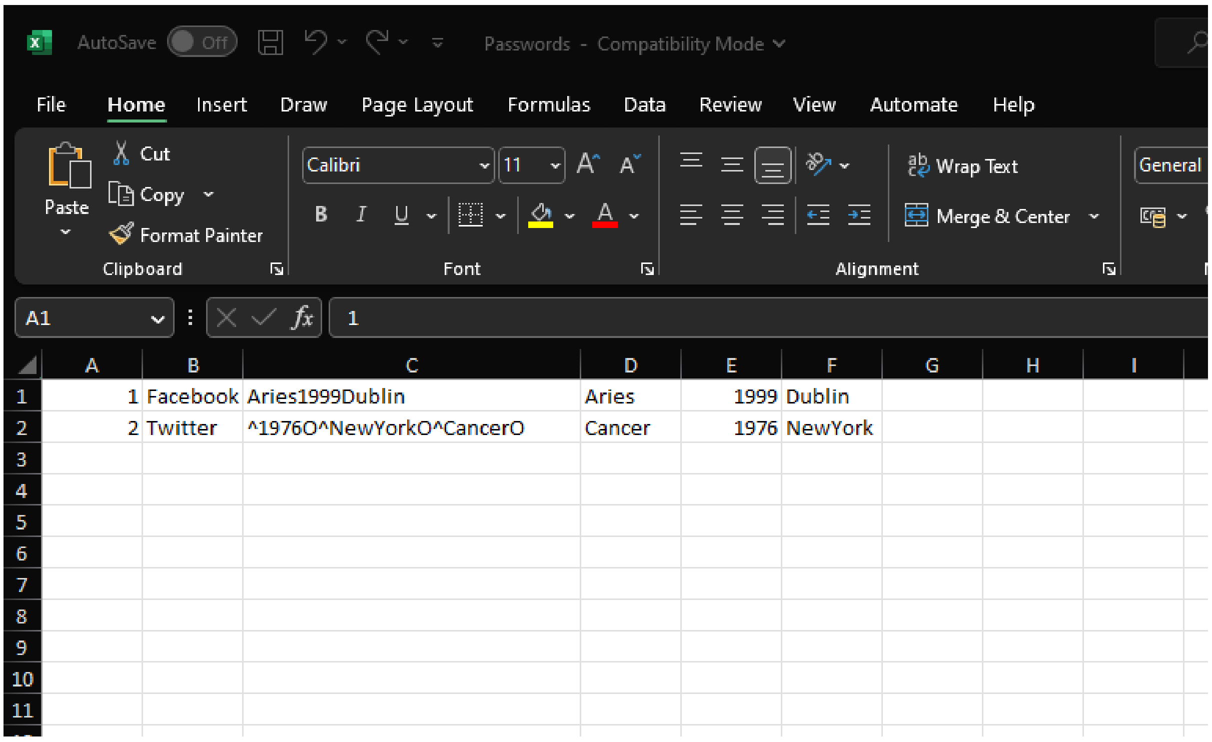


	ID	Tag	Password	SQ1	SQ2	SQ3
▸	1	Facebook	Aries1999Dublin	Aries	1999	Dublin
	2	Twitter	^1976O^NewYor...	Cancer	1976	NewYork
*						

Main application page, where the user accesses all of the functions of the application.



Example of passwords being exported to F: drive



	A	B	C	D	E	F	G	H	I
1	1	Facebook	Aries1999Dublin	Aries	1999	Dublin			
2	2	Twitter	^1976O^NewYorkO^CancerO	Cancer	1976	NewYork			
3									
4									
5									
6									
7									
8									
9									
10									
11									

2.5. Testing

Unit testing:

Test Case ID	T1
Test Component	Register
Test Description	Test the ability for a user registration for the database
Test Priority	Very Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Comments**
Step 1: Click the “Sign up” button.	Click	The Registration page successfully loads.	The Registration page successfully loads.	There were no issues during this step.
Step 2: Fill in the required fields.	Text	The user can type in the requested data.	The user was able to type in the requested data.	There were no issues during this step.
Step 3: Click the “Register” button.	Click	The user’s account will be created and logged in to the main page	The user successfully created the account and is logged in	There were no issues during this step.

Test Case ID	T2
Teste Component	Login
Test Description	Test the ability for a user login to the MemPass application
Test Priority	Very Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Comments**
Step 1: Click the “Login” button.	Click	The Login page gives error to input correct details	The Login page gives error message	There were no issues during this step.
Step 2: Fill in the required fields.	Text	The user can type in the requested data.	The user was able to type in the requested data.	There were no issues during this step.
Step 3: Click the “Login” button.	Click	The user is logged in and greeted to the main page	The user successfully logged in and greeted with main page	There were no issues during this step.

Test Case ID	T3
Teste Component	Generate password
Test Description	Test the ability for a user to generate a password using Secret Questions.
Test Priority	Very Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Comments**
Step 1: Click the “Generate Password” button.	Click	The Dashboard page gives error to input correct details	The Dashboard page gives error message	There were no issues during this step.
Step 2: Fill in the required fields.	Text	The user can type in the requested data.	The user was able to type in the requested data.	There were no issues during this step.
Step 3: Tick Checkboxes	Click	The user can check the special char or alpha num checkboxes	The user was able to check the boxes	There were no issues during this step.
Step 4: Click the “Generate” button.	Click	The user successfully generated password and is displayed in textfield	The user successfully generated the password and appears in the correct text field.	There were no issues during this step.

Test Case ID	T4
Teste Component	Insert password in vault
Test Description	Test the ability for a user to store password and questions in the vault.
Test Priority	Very Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Comments**
Step 1: Click the "Insert" button.	Click	The Dashboard page gives error to input correct details	The Dashboard page gives error message	There were no issues during this step.
Step 2: User presses "Insert" after filling in the details.	Click	Will insert new row in the vault with details	The application uploads the details in the vault successfully	There were no issues during this step.
Step 3: ID is generated for the row	Click	The user can see ID of the password	The user was able to see the ID only when clicked in the vault.	The password ID textfield does not show anything, but works when the user selects from the vault.

Test Case ID	T5
Teste Component	Delete/Update password from vault
Test Description	Test the ability for a user to update their vault.
Test Priority	Moderate

Action	**Inputs**	**Expected Result**	**Actual Output**	**Comments**
Step 1: User clicks the row to update/delete	Click	The Dashboard page retrieves relevant data	The Dashboard page outputs the data in the fields	There were no issues during this step.
Step 2: User updates fields and presses generate and “Insert” or “Delete”	Click/Text	Dashboard will give confirmation message and update	The application uploads the new details in the vault successfully.	There were no issues during this step.

Test Case ID	T6
Test Component	Change Master password
Test Description	Test the ability for a user to update their MemPass password.
Test Priority	Moderate

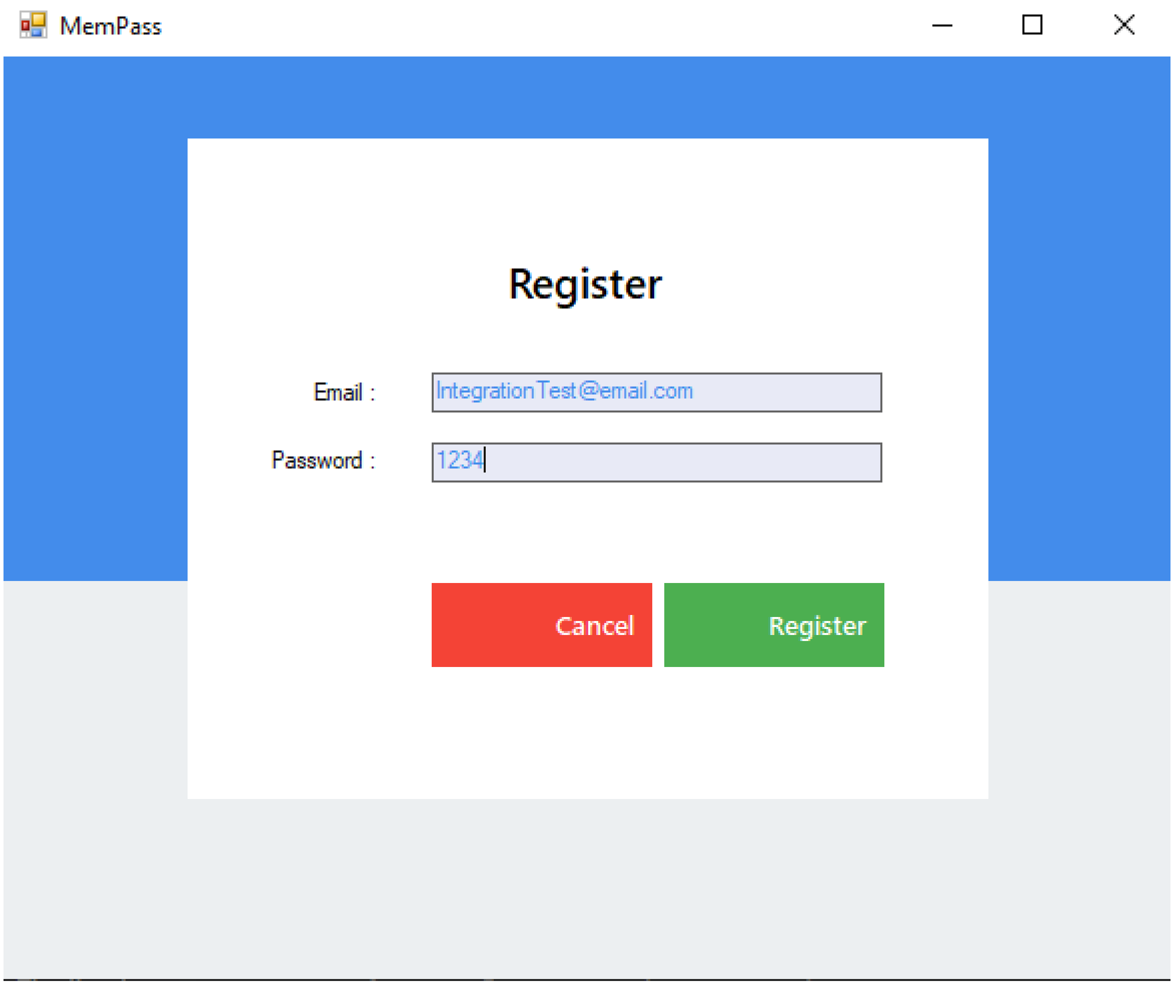
Action	**Inputs**	**Expected Result**	**Actual Output**	**Comments**
Step 1: Fill in the required fields.	Text	The user can type in the requested data.	The user was able to type in the requested data.	There were no issues during this step.
Step 2: User updates fields and presses “Change Password”	Click	Dashboard will give confirmati on message and update	The application uploads the new details in the database successfully.	There were no issues during this step.

Test Case ID	T7
Test Component	Export Passwords
Test Description	Test the ability for a user to export their MemPass vault.
Test Priority	Important

Action	**Inputs**	**Expected Result**	**Actual Output**	**Comments**
Step 1: Click the "Export Passwords" button.	Click	Password vault will be exported to F:Drive	Excel File was created will user's details	There were no issues during this step. Although it will try to name it and replace old file if user requests another export
Step 2: User can view details in excel file	Click/Read	Passwords and questions appear as they should	The application successfully exports the passwords and security questions in same row format.	There were no issues during this step.

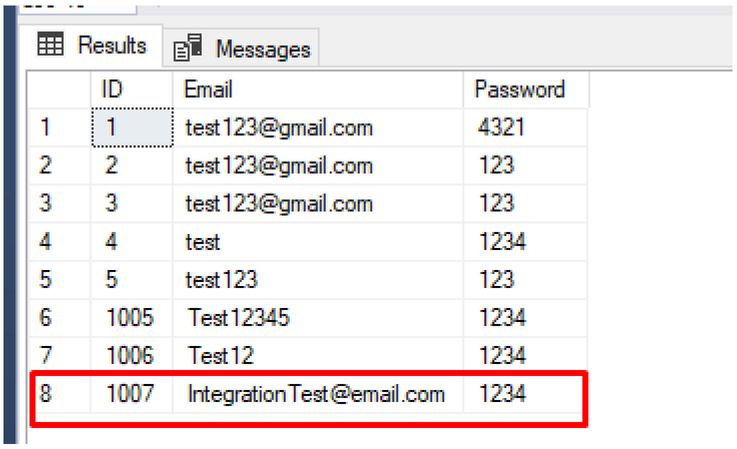
Integration Testing:

The following findings were obtained after unit testing was done on the key application functionality and all the components were combined:



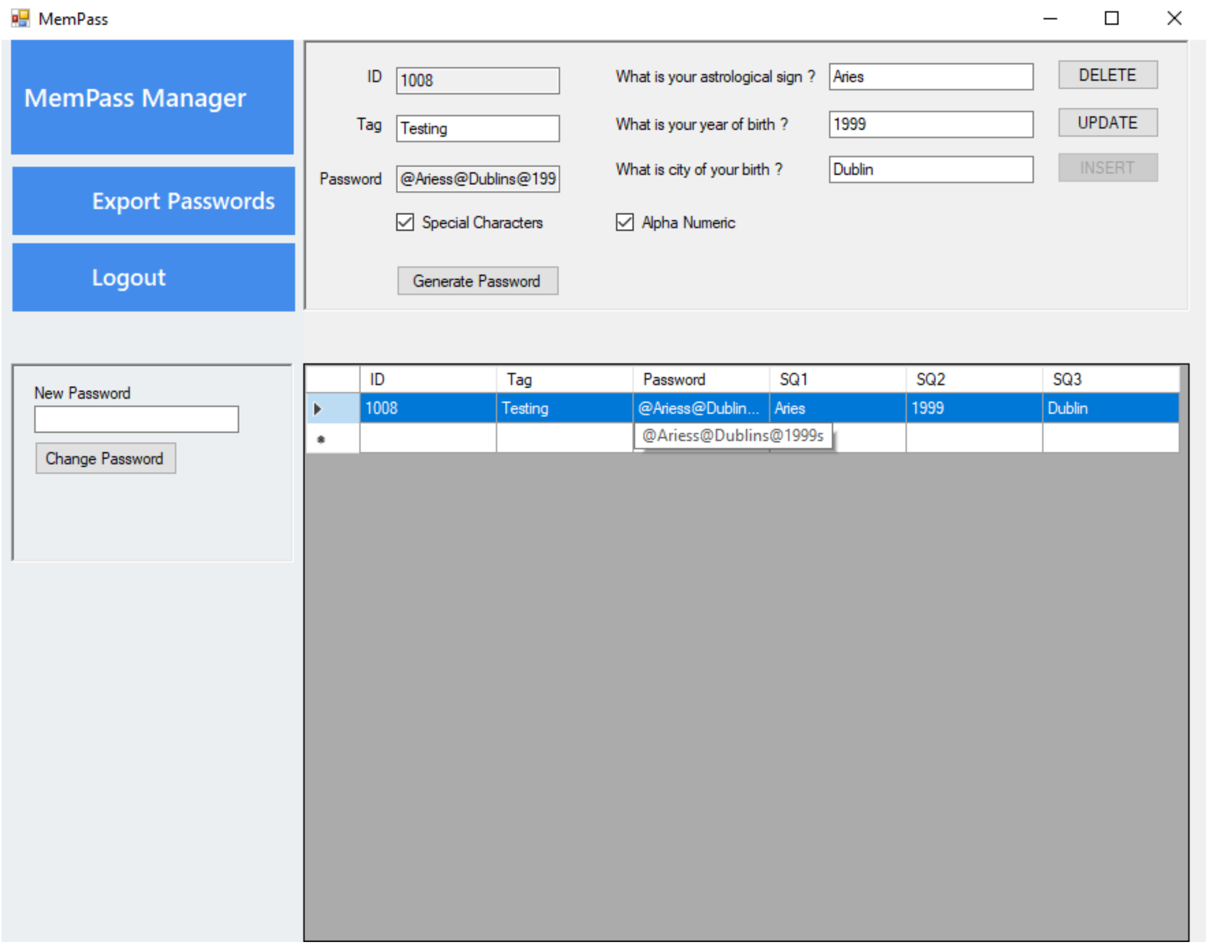
Email : IntegrationTest@email.com

Password : 1234



	ID	Email	Password
1	1	test123@gmail.com	4321
2	2	test123@gmail.com	123
3	3	test123@gmail.com	123
4	4	test	1234
5	5	test123	123
6	1005	Test12345	1234
7	1006	Test12	1234
8	1007	IntegrationTest@email.com	1234

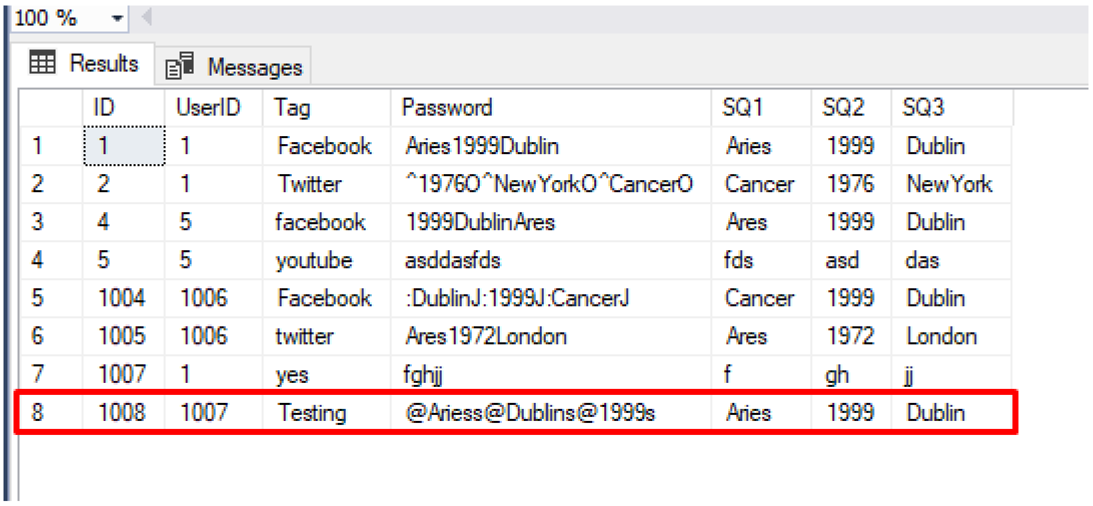
I started by creating a new user, and the database query that followed shows the precise information that was submitted into each field throughout the registration process. The information that was returned from the database accurately matched what the user entered, demonstrating that the registration procedure was effective in storing the given information in the database for later use and reference.



ID: 1008
Tag: Testing
Password: @Ariess@Dublins@199
What is your astrological sign ?: Aries
What is your year of birth ?: 1999
What is city of your birth ?: Dublin
☑ Special Characters
☑ Alpha Numeric
New Password:

ID	Tag	Password	SQ1	SQ2	SQ3
1008	Testing	@Ariess@Dublin...	Aries	1999	Dublin

@Ariess@Dublins@1999s



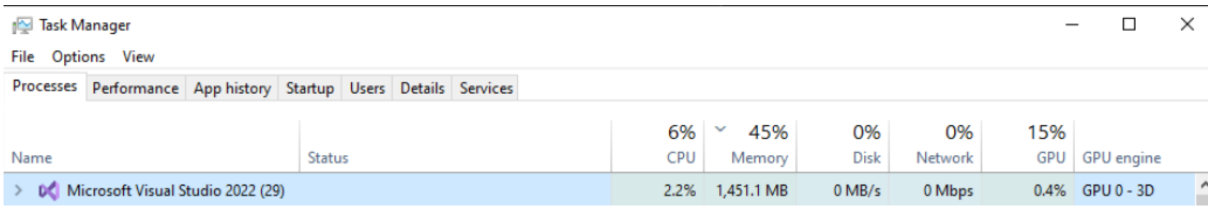
100 %

Results | Messages

	ID	UserID	Tag	Password	SQ1	SQ2	SQ3
1	1	1	Facebook	Aries1999Dublin	Aries	1999	Dublin
2	2	1	Twitter	^1976O^NewYorkO^CancerO	Cancer	1976	NewYork
3	4	5	facebook	1999DublinAres	Ares	1999	Dublin
4	5	5	youtube	asddasfds	fds	asd	das
5	1004	1006	Facebook	:DublinJ:1999J:CancerJ	Cancer	1999	Dublin
6	1005	1006	twitter	Ares1972London	Ares	1972	London
7	1007	1	yes	fghjj	f	gh	jj
8	1008	1007	Testing	@Ariess@Dublins@1999s	Aries	1999	Dublin

The accompanying screenshots shows how a new password was successfully generated and stored in the user's vault and in the database. The process of creating a password is shown in the first screenshot, where a randomly generated password is presented and then saved in the user's vault. The recall of the saved password from the database is shown in the second screenshot, illustrating how the application and database management system are integrated. This guarantees that the user's vault continually reflects the passwords that have been stored, ensuring data correctness and consistency.

Acceptance Testing:

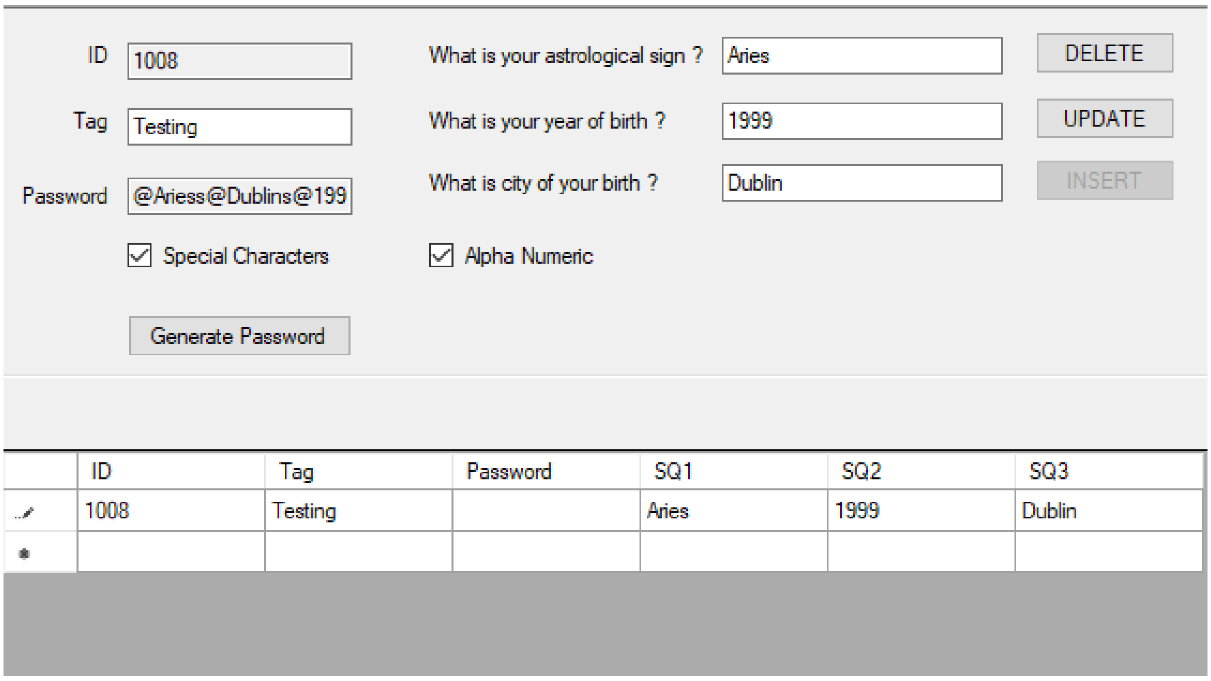


Name	Status	6% CPU	45% Memory	0% Disk	0% Network	15% GPU	GPU engine
Microsoft Visual Studio 2022 (29)		2.2%	1,451.1 MB	0 MB/s	0 Mbps	0.4%	GPU 0 - 3D

According to the evidence given, it shows that integration testing and unit testing were successful on MemPass. The application is comprehensive, accurate, and performs as intended, according to the test findings. Any mistakes found while developing the product were immediately corrected and implemented gradually.

Additionally, MemPass has modest resource needs, which can be seen positively according to performance like CPU utilization, memory consumption, and power usage. In modern systems with higher memory capacities, average memory usage may not be a significant concern for most machines.

Based on these results, it can be said that MemPass is a safe platform for protecting and changing passwords. Some small flaws were discovered during the testing. Users can manually delete fields from the vault, but doing so only impacts the visual representation and not the data itself. This is one problem. Another flaw is that, if the "X" button is used to quit the application window, the program keeps running in the background and is therefore unreachable unless explicitly terminated with Visual Studio.


ID 1008

Tag Testing

Password @Ariess@Dublins@199

☑ Special Characters

☑ Alpha Numeric

What is your astrological sign ? Aries

What is your year of birth ? 1999

What is city of your birth ? Dublin

	ID	Tag	Password	SQ1	SQ2	SQ3
	1008	Testing		Aries	1999	Dublin

2.6 Evaluation

A user-friendly product that smoothly provides all the features described in the aforementioned document parts will be offered to the end user. Initial testing has revealed that the program is already incredibly quick and responsive. The program is expected to retain its present level of performance and responsiveness with continuing development, guaranteeing a seamless and effective user experience.

3. Conclusions

I first used a variety of technologies and frameworks, such as Identity and a Web application architecture, when creating my Mid-point prototype. But I soon came to the conclusion that mastering these technologies would take a lot of time and work beyond my skill-set in terms of coding. I came to the conclusion that using these technologies further would make it more difficult for me to accomplish the project effectively after taking the limitations and difficulties into account.

I choose to create a desktop application utilizing MS SQL Studio and Visual Studio technologies after conducting assessments and research. The design of the application is simple and attractive. Its major strength comes in efficiently preventing data breaches, password leaks, and phishing scams by securely storing user credentials locally. Although the execution and methods employed were different from what I would have chosen, the core concept and functionality are still there, and I am still pleased with the outcome.

The program is a good prototype even if it isn't meant for commercial usage and might require some work. It has the potential to develop into a more reliable and secure program by adding new features, improving the user interface's engagement, and putting extra security measures in place.

4. Further Development or Research

With additional time and resources, I would plan to create a more sophisticated system with integration of hashing and more password generation options, such as adding password requirements, more and randomised questions. Early into my project I was very inexperienced with the new technologies that I must use for my application which resulted with time being exhausted in development.

With further development and research, one of the possible things I would incorporate by implementing strong encryption and decryption algorithms. My application currently lacks a form of hashing which I would've loved to include in my project, but due to my skill set in coding and knowledge in approaching this, I excluded it in my project.

An automated password change reminder function and password leak database checks might also be incorporated into the project as future improvements. Users would receive automatic reminders from this feature to update their passwords on a regular basis for greater security. To further reinforce data protection safeguards, the program would also have the capacity to cross-reference passwords against known database breaches.

5. References

codewithmosh.com. (Mosh Hamedani). All-Access Subscription. [online] Available at: https://codewithmosh.com/p/all-access [Accessed 14 May 2023].

www.youtube.com. (dotnet24). How to Add - Edit And Delete button in Grid view C# Windows Application. [online] Available at: https://www.youtube.com/watch?v=kad3u7JMjzQ [Accessed 14 May 2023].

www.youtube.com. (Amit Andipara). Insert Update Delete View and search data from database in C#.net. [online] Available at: https://youtu.be/TIAOr2S6-SY [Accessed 14 May 2023].

www.youtube.com. (SwIT Clog). Login Form in C# windows form application with SQL database in Visual Studios. [online] Available at: https://youtu.be/_S-LacqE7OM.

Microsoft (n.d.). Databases - SQL Server. [online] learn.microsoft.com. Available at: https://learn.microsoft.com/en-us/sql/relational-databases/databases/databases?view=sql-server-ver16.

6. Appendices

Project Proposal

National College of Ireland

Project Proposal

Password Generator

30/10/2022

BSc (Honours) in Computing

Cybersecurity

2022/2023

Gvidas Virsilas

X18417984

X18417984@student.ncirl.ie

Contents

1.0 Objectives 41
2.0 Background 42
3.0 State of the Art 42
4.0 Technical Approach 43
5.0 Technical Details 44
6.0 Special Resources Required 44
7.0 Project Plan 44
8.0 Testing 46

Objectives

My idea for this project will be a Password Generator and Manager. The unique twist to this is that it will be able to generate passwords that are memorable to the user based on their interests or things that are particularly important to them.

The objective of this project is essentially to make passwords less of a hassle for a user to remember and be less dependable on software to manage all their passwords. Almost everybody today uses some type of password manager that autofills their details automatically without having to type anything at all. This poses a great risk to your confidential login details if an unknown user gains access to the owner's machine or master password to the manager. Along with security risks these password managers can essentially prevent you from logging in if you are on a new machine without the software installed, unless you can remember the random string of characters the manager makes and remembers for you.

The Web application will be able to ask a user a set of random questions and would use these answers to be implemented to the user's new generated password and be likely to be more familiar with their password. The password should be easier to remember, rather than a series of random strings of letters, characters, and numbers to remember as most modern password managers generate. The app can also save the newly made password to be saved on the machine and be recalled for later use.

Background

As part of my specialisation in Cybersecurity, my requirement was to think of an idea revolving around online security. I decided to go with a password generator/manager as it is something I use myself daily on multiple devices such as my smartphone and computer. After looking into doing a password manager I realized how dependent we are on applications to remember all our passwords on websites. One good example of an issue with password managers I often come across personally, is when I access a new machine. I would often not know the password to my account for a certain site and would have to look up the password on my manager just to see what it is, which is an annoying process especially when my smartphone is not nearby. With recurring database breaches happening online often, a hacker could find out your master password if it is used somewhere else. If someone were to gain access to the main password to your manager it would be a huge security risk, as all your passwords would be exposed.

To meet my objectives I set out, I decided that I would code an application that is able to generate a memorable password that can give the option to locally save it on the machine if the user wishes to. The application will not store any information online from the user and would only store passwords when the user chooses to. During my 4th year, I intend to follow my milestone-milestone plan rigidly as well as doing weekly check-ups with my supervisor. I will make sure to give my supervisor insights on the progress of my app as well as receiving any advice that I need to follow from them.

State of the Art

There are many applications that exist today that can generate and store your passwords in one place. What makes my proposal stand out from the rest is that it makes you less dependent on the password managers to fill in your details. I want my application to be used as sort of an assistant in creating the perfect password for the user personally, with the sole idea of making my application more efficient in terms of security and convenience in the long term for the user instead of having a feature that automates an action that is only ideal for a short term.

Some examples I found through my research that are like my project include:

- LastPass

  LastPass is a browser extension that is available for most mainstream internet browsers such as Google Chrome, Firefox, Edge, and Safari. The extension can be used between browsers, and it uses a master password and email to login to your password "vault". LastPass can generate or remember your own password to their vault for that specific website you wish to save your details for. It can automatically fill out your details without even pressing or typing anything. As well as login details, LastPass can store payment details on their vault that works the same way as the password vault with autofill options.

Although LastPass can be very convenient, it poses a great security risk if someone gains access to your “master password”. The malicious user has access to all your passwords, card details and emails which can be devastating. LastPass also focuses too much on automation for websites that you, as the user could never even see your password that it generates making you dependent on the extension. Also, this extension only generates random strings of characters for your passwords, which is unrealistic to remember for a user.

- Dashlane

Dashlane is also a browser extension that works very similarly to LastPass. It can sync to multiple devices such as your smartphone and computer. The unique aspect to Dashlane is that it actively scans the Dark Web for database leaks to check if your password has been compromised. It also has a feature that rates the security of your passwords from 0-100% with 0% being very insecure, with tips if it should be changed.

Like LastPass, the same security flaws exist in Dashlane as most password managers that work with a “master password”. I personally think that password managers overlook the flaws and outweigh their features over security which is a concern. This is what will make my application stand out from not being able to store any data in a “vault” to prevent a security breach.

Technical Approach

I choose to approach my development by setting out my activities, steps, and tasks on a platform I have been using in the past for projects called Asana. Asana will let me establish my requirements into tasks that can be followed in a certain order after I identify how the project shall flow. By using Asana, I will be able to keep track of things that I will have to get done and prioritise certain tasks that need to be completed first.

To firstly identify my requirements, I will need to research what exactly I will need to create the features I intend to implement. This research will consist of looking at potential coding languages, libraries, or other types of technologies that I could be not aware of. I will be looking at online tutorials on the different approaches I can take in coding my application so I could get an insight on how I can incorporate my own idea. The research will be very important for me to partake in as I never took up a big project like this by myself and I have not created something similar before. I will have to be careful of plagiarism and if I do incorporate some code, I need to put my own contribution and reference the author.

The prioritisation feature of Asana can be scaled from High, Medium, and Low priority for its tasks. By using prioritisation, I can identify what are the major important requirements in my application to make it work as a password generator which would be a high priority and less important features can be things such as cosmetic looks to the app or any extra features I might think to implement as the project goes along which could be considered low priority. Using Asana regularly, I can form a roadmap to help me stay focused and be always aware of what will need to be completed.

Along with my development process of my application, I will also factor in the regular submissions that are required for Software Project such as the monthly journal, supervisor meetings. I will need to keep track of all the due dates and will note all important dates into my Asana task list to make sure no task is missed.

Technical Details

I will intend to use C# as my main programming language for the project. C# is the most familiar language to me, and I have been using the most for past projects. I will be making use of the .NET framework to be able to use its own libraries. I will also make use of Angular as it is an excellent extension of libraries in developing the front-end of the application.

The main text editor I will use for the creation of the application will be Visual Studio as it is free and includes many features that makes the completion of code a smoother process with things such as debugging, syntax highlighting and embedded Git.

I still must research exactly how I will use these technologies when I establish my final design and review online tutorials as it is hard to establish the core technologies when I don't have it finalized. Some of these libraries are still very new to me especially Angular and I will have to learn as I progress through my project and see which are suitable or if I will add extra technologies to polish my project even more.

Some of these technologies will also require me to do testing and trial and error to see which will fit my project the best and to my capabilities of coding this.

Special Resources Required

At the time of writing this proposal, I will not be needing any extra special resources to complete my project as I already have access to everything I will be needing. I will, however, have to organize meetings with my supervisor that will help me throughout my project and guide me through.

Project Plan

As the final submission date for Software Project will be sometime in late May, I intend to space out all my work throughout the year with keeping in respect other submissions too such as the Midpoint and my other modules. As of writing this proposal, currently my project is still in the research phase. I am still considering what things I intend to implement and how I can go about doing it, and things I can't make work as expected and to look for ways to work around.

My first supervisor meeting will be after the reading week, during the meeting I can declare my idea to my supervisor and hopefully I could get a word of advice on what would be my first few steps for this project. I already intend to create basic use cases of my project to display the primary function of the password generator and how I can form subtasks to be added in Asana from there. I intend to show these tasks to my supervisor in the upcoming weeks.

From there I will start to design how the application will look like before any type of coding is written. From there I will begin to create the main page of the application that users will see when they launch the application. The main page will consist of the options the user wants to have in their password, so I will code checkboxes, fields, and a design. This will bring me to the other pages of the application such as a page for the random questions the app will ask the user, a review page of the information provided and a results page etc.

Once the final concept of how the design will look is finished, I will categorise the features in milestones. Each milestone will have their own subtasks and their priorities to it. The coding of each of these milestones will begin in the order I will set out in Asana. The core functions will be the top priority which are to create an application that can generate a password from questions, then other functions that are less major such as saving the password to the machine or making the UI. Although making the less major components also important, I want to make sure the app is working as intended firstly then move onto qualities that include extra content to work along the major functions. This is so if I encounter obstacles along the way and would make me run out of time for my deadlines, I will still have the functions that would make the application work for its “completeness”.

Once of course the coding is all done for each task, testing will take place and fix any issues to make sure everything is all working properly.

This is a rough structure of deadlines that will have things added and moved around that I will follow until May. These dates could be changed around because of the consideration to other assignments for other modules due for submission.

Design and look of application – 7th November

Main page Skeleton – 10st December

Main page Questions – 20th December

Midpoint documentation and Slides – 20th December

Extra pages skeletons – 15th January

Add functions to the skeleton pages – 30th January

Generator – 28th February

Non major functions – 25th March

Testing – 30th March

Polish application – 10st April

Fix issues – May 8th

Testing – May 8th

Documentation+ Video – May 17th

Testing

The way I plan on evaluating my system is by using a couple types of testing which are Black box testing, Unit testing, End to end Testing, and Acceptance Testing.

Testing of my application will incorporate both Black Box and End to end testing for the first initial tests to see and find if any sort of errors or bugs occur. This lets me see from the eyes of a user when they go to use my application. Each possible option of my application will be tested and noted down if the actions were successful or not if the actions were not successful, details explaining what was pressed to cause the error to happen will be considered. Once all possible options are explored, I would begin to analyse the features that were not working correctly by looking over what it is supposed to do and what it is doing. I will have to look over code and spot the error in the code that is causing it and change it appropriately to make it work for its intent. After that I would run the tests again to make sure everything is functioning with the new fixed code.

After I am satisfied with my fixes, I will perform an acceptance test by informally requesting colleagues and friends to try out my application. I will be conducting this informally to eliminate the process of going through the ethics form process as it would provide me with the same exact results regardless of formality and the audience since my application will not have a specific target audience. Getting the opinion of my colleagues and friends can give an honest review of my application of what they think is good or might need a change. This also lets me know if they would find an error in the application in case I have missed myself while conducting my own testing. Once I gather the opinions and criticism (if any) of my colleagues and friends, I will go back to my coding and initiate the examination of it again. After I test the system by Black box and end to end testing on my own again and make the changes, I will request my colleagues and friends again if I will need to or else, I will be happy as is.

Reflective Journals

Supervision & Reflection Template

Student Name	Gvidas Virsilas
Student Number	X18417984
Course	BSHCYB4
Supervisor	Keith Maycock

Month: October

What?

During the month of October, we were introduced to the Final Project and what will be expected to be done during our 4th year. We were tasked with creating a unique idea that we will be working towards until May. This idea must consist of building something innovative and related to our specialisation. The key is that it must be modern and useful to people nowadays.

On the 9th of October I submitted my project pitch idea of a memorable Password generator.

So What?

With having a set idea, I had investigated the technologies already existing that are like my idea and be able to get an even better insight of what my idea can become. I was also able to research the technologies and decide what I can use towards building my project. I am still a little bit worried about how exactly how I will go ahead and code my idea as I never created something this complicated before which will be a challenge for me.

Now What?

Due to my inexperience in coding such a big project, I will need to do a lot more research and code revision to get me confident in tackling this task by myself. I will also have to discuss my project progress with my

supervisor on what will be the best way to approach it and what changes I will need to do along the way to make sure it reaches the final project criteria.	
Student Signature	Gvidas Virsilas

Supervision & Reflection Template

Student Name	Gvidas Virsilas
Student Number	X18417984
Course	Cybersecurity
Supervisor	Keith Maycock

Month: November

What? In this month I decided there will be slight changes to my proposal than what I had originally planned. After my first meeting with Keith he had suggested that I should add a bit more complexity to my project. To add more complexity, I have decided I will add a password system and include a database for the stored passwords.
So What? Having the meeting with Keith helped me establish a more concrete set plan of what my application will include and what I can start working towards for the midpoint presentation. Some challenges still remain on things such as what technologies I might use for the database related stuff for my application.
Now What? I will have to do a furthermore research in the next month and communicate with my supervisor in meetings on the progress and development of my project

Student Signature	Gvidas Virsilas

Supervision & Reflection Template

Student Name	Gvidas Virsilas
Student Number	X18417984
Course	**BSHCYB4**
Supervisor	Keith Maycock

Month: December

What?

During this month, I had made considerable progress and a start in my project. On the 20th of December I also had to submit my Midpoint submission. I created a login/registration system and progressed my technical report.

So What?

Starting my project with the login system allowed me to create one of the major parts of the project and a baseline to work on and gradually add new things as I progress my project. Creating the login and registration system was very successful and works without any issues with a backend server. The challenges that remain is to create the actual password generation and a way to store user's passwords in the database. I will also have to add design to the application.

Now What?

What can you do to address outstanding challenges?

To overcome these challenges, I will have to research a lot in technologies that can be implemented in my project and receive guidance from my supervisor and see what steps I could take. I will also have to learn how to efficiently use the backend for storage for user data

Student Signature	Gvidas Virsilas

Supervision & Reflection Template

Student Name	Gvidas Virsilas
Student Number	X18417984
Course	BSHCYB4
Supervisor	Keith Maycock

Month: January

What?

This month I had not made a lot of progress due to a project and a TABA due for 2 different modules in the first 2.5 weeks in January. Due to these deadlines, I only managed minor progress in researching on the password generation that I can implement into my project.

So What?

Although the progress is minor, I manage to find references that I can look for inspiration to put in my project from the research. Challenges remain of coding the main purpose of the application.

Now What?

Since my semester 1 projects are out of the way, February will be the perfect time to focus and dedicate more time towards the coding of my project. I am hoping to create a more sophisticated login system and begin the password generation tech of the project.

Student Signature	Gvidas Virsilas

Supervision & Reflection Template

Student Name	Gvidas Virsilas
Student Number	X18417984
Course	BSHCYB4
Supervisor	Keith Maycock

Month: February

What?

During this month I spent a lot of time getting used to applying my research in practice for my application. I had created test projects and successfully created code for random password generation for a user and another test app by having the user supply the application keywords and outputting a password in different order than supplied.

So What?

This is a notable achievement for me as it is the baseline I have to start with in creating my project idea into a reality. With the code I currently have I can now add things and tweak the code to my project concept. I had successfully made code in C# that will ask a user for an input and generate a password for the user. Although it still needs an algorithm that will ask the user a random set of questions and make a secure password from the inputs.

Now What?

The current challenges that still exist are creating the intended purpose of password generation by making it memorable and secure. Implement coding into the main application and create a database and styling for the application. I plan on tackling these challenges by researching more in depth in these sections and creating testing plans as I move forward with the experimentation, and then eventually implementing into my main app.

Student Signature	Gvidas Virsilas

Supervision & Reflection Template

Student Name	Gvidas Virsilas
Student Number	X18417984
Course	BSHCYB4
Supervisor	Keith Maycock

Month: March

What?

Far more development of my application has taken place during the month of March. I have specifically succeeded in developing the application's core feature, which allows users to enter a question answer and generate a corresponding password. Moreover, decorative elements have been included to improve the application's homepage and other pages. The database has the capability to securely store the generated passwords in the user's account.

So What?

As a result, I will soon be able to complete my project application and begin the paperwork necessary to complete my project as a whole. All I need to do now to finish my application's key features is to add smaller technologies and polish it to a high quality.

My application has made great progress, but there are still a few small issues. They include fixing application flaws, such as the occasional creation of unusual or difficult to remember passwords. I will also need to add suitable aesthetics and perhaps changing the database to a hosted solution, I also want to enhance the overall user experience.

Now What?

I must put in a large amount of work as I near the end of my assignment, and to ensure if I want to receive a high grade I must follow the grading rubric and implement it correctly. I'll be researching and extensively testing my program in the following days, as well as starting the documentation.

Student Signature	Gvidas Virsilas

Supervision & Reflection Template

Student Name	Gvidas Virsilas
Student Number	X18417984
Course	BSHCYB4
Supervisor	Keith Maycock

Month: April

What?

This month I finalized my project application and added the finishing touches as I planned. I have also begun continuing my documentation and overall wrapping up my final submission.

So What?

Overall, I am happy how I developed my project. I managed to code my password manager "MemPass" with passwords being able to be generated from the user's input from the application's questions. Although I would've liked to include more randomisation to generation and more security features such as password specifications if I had more time.

Now What? At this stage I think I have done to the best of my ability, and I am happy to leave the application as is and considering the amount of time I have left, I won’t risk adding new things in case the application becomes unresponsive or does not work as intended. These final two weeks will be spent on documentation and eventually submitting my project overall on May 14th.

Student Signature	Gvidas Virsilas