

Verslag Case Study Windows Forms

Inhoud

Voorwoord	3
Het project.....	3
Vorbereiding.....	3
Testen met windows form	3
De uitwerking van de app.....	4
Aanmaken Github-repo.....	6
Aanmaken van de SQLite-database	7
Afwerken van de app.....	15
Conclusie	19
Links.....	19
Bibliografie	19

Voorwoord

In dit project ga ik een app maken in Windows Forms met C#. Als onderwerp heb ik minecraft gekozen. De app zal ook bestaan uit een database gevuld met gegevens over minecraft. Ik zal gebruikmaken van een ORM en overerving. Aan de slag!

Het project

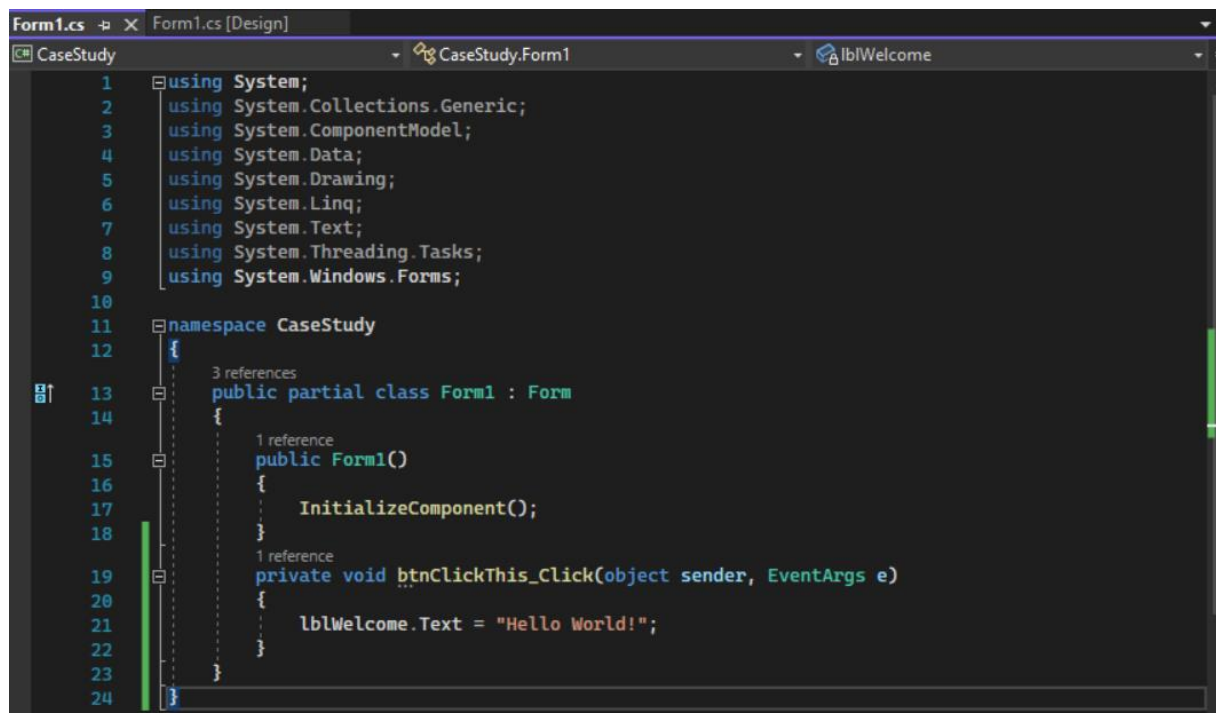
Mijn idee voor dit project is om een handboek-appje voor minecraft te maken. Dan is het bijvoorbeeld mogelijk om info over verschillende monsters te raadplegen, die dan uit een database opgehaald word.

Voorbereiding

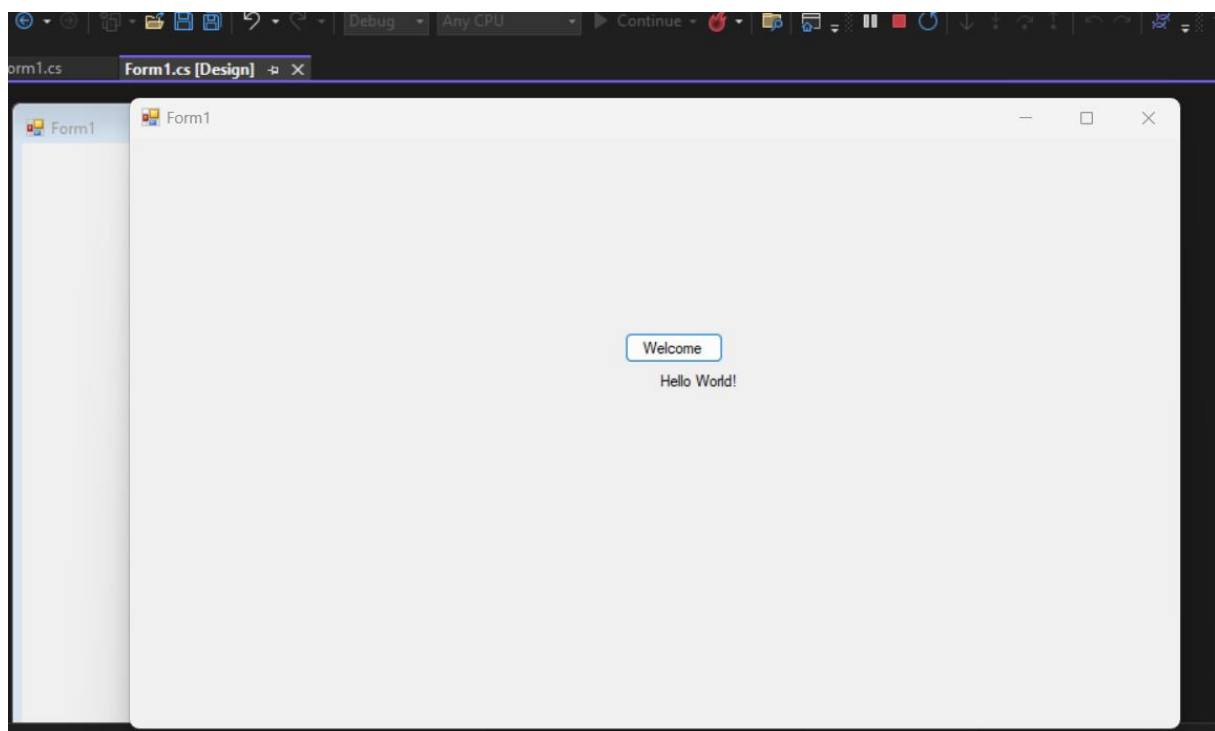
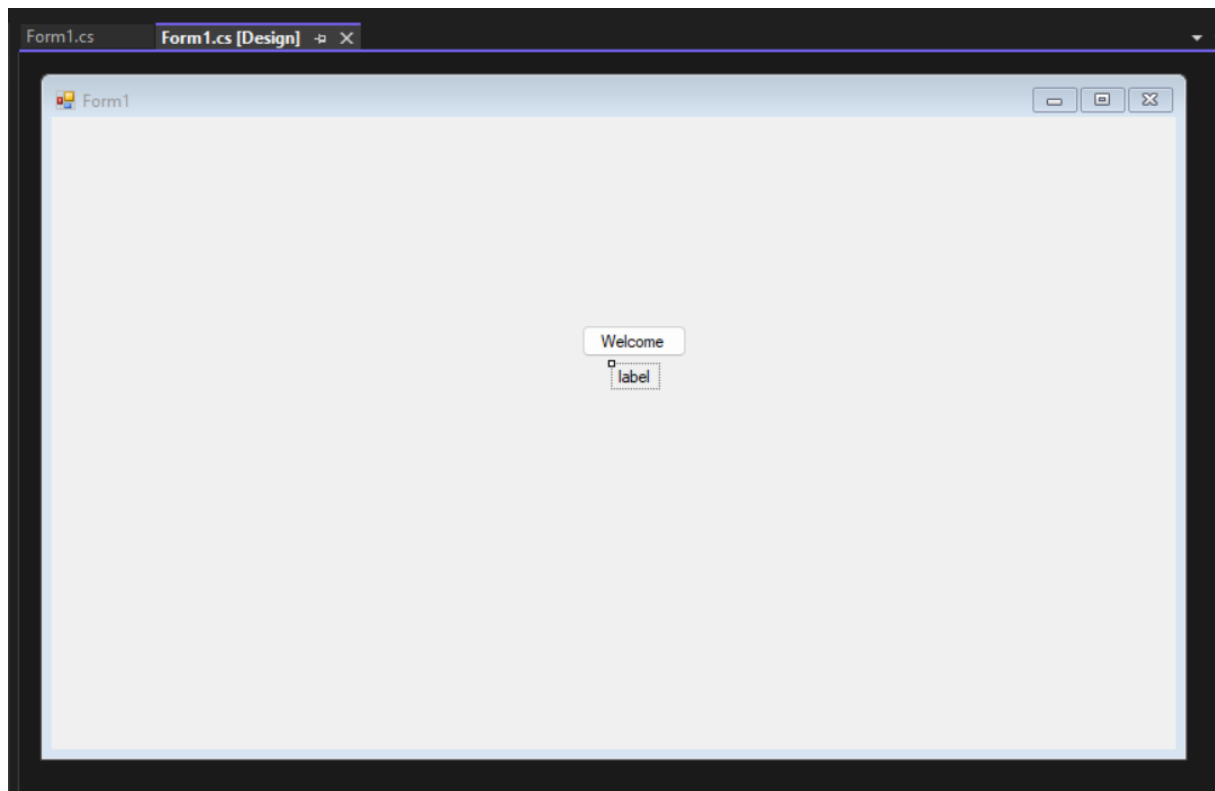
De eerste stap die ik heb ondernomen is uitgebreid research doen. De bronnen die ik daarbij gevonden en geraadpleegd heb, zijn in de bibliografie terug te vinden.

Testen met windows form

Als eerste ben ik begonnen met een klassiek hello-world experiment, om te testen. Hieronder de screenshots:



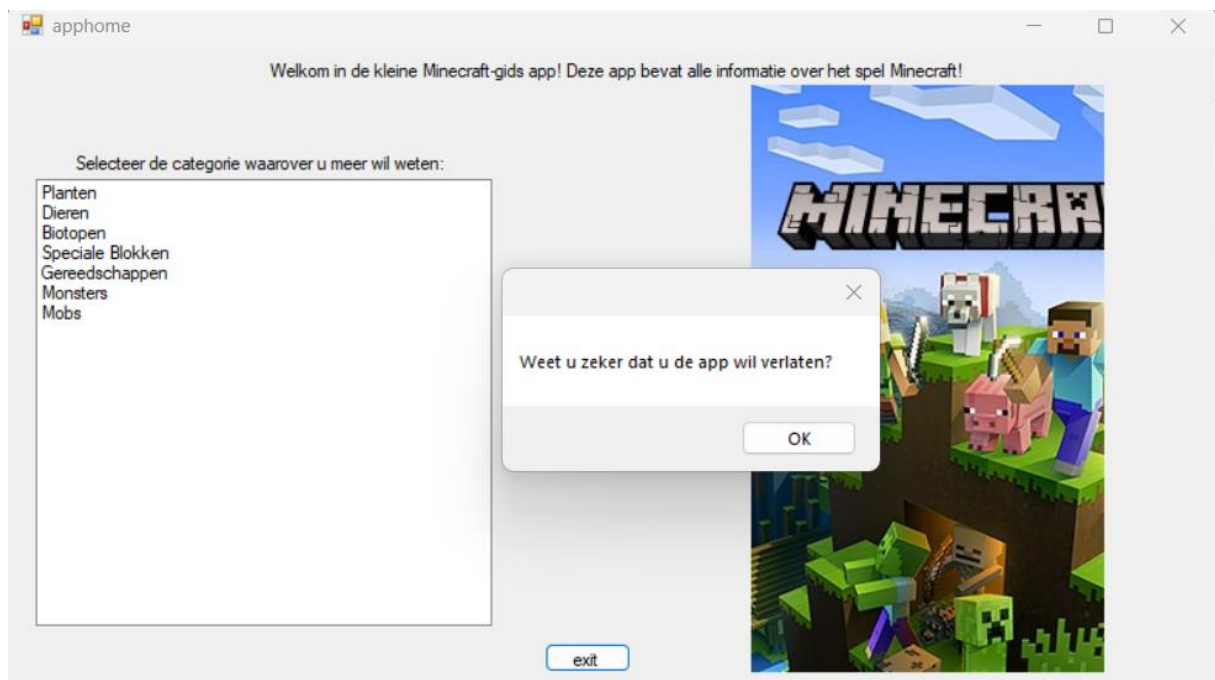
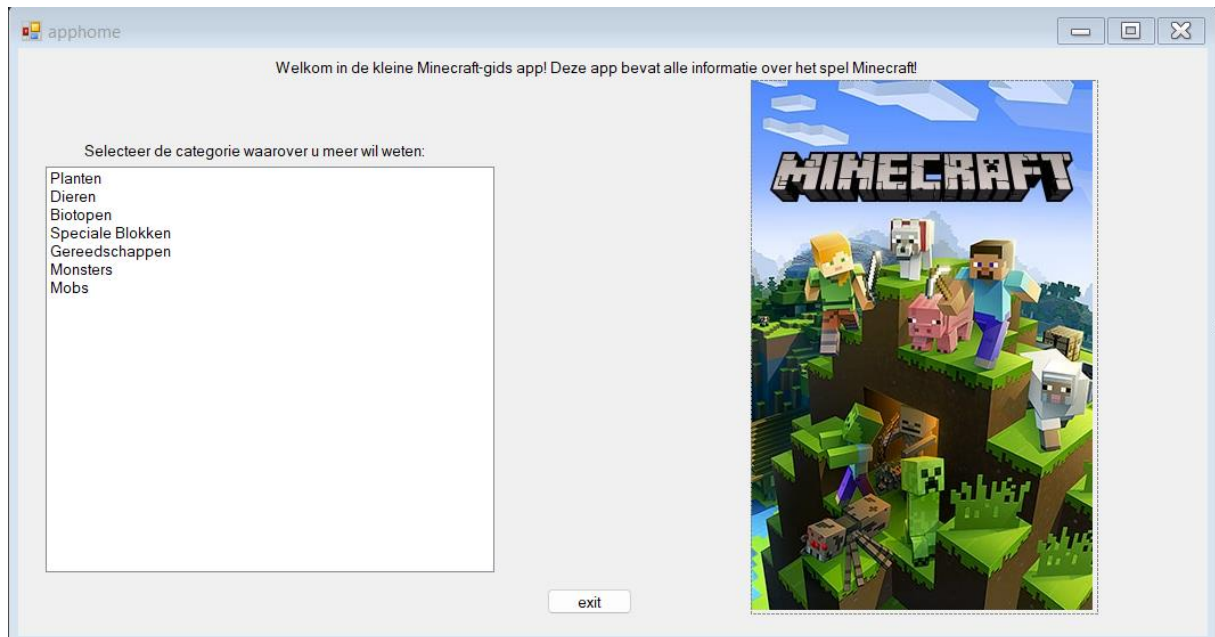
```
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10
11 namespace CaseStudy
12 {
13     public partial class Form1 : Form
14     {
15         public Form1()
16         {
17             InitializeComponent();
18         }
19         private void btnClickThis_Click(object sender, EventArgs e)
20         {
21             lblWelcome.Text = "Hello World!";
22         }
23     }
24 }
```

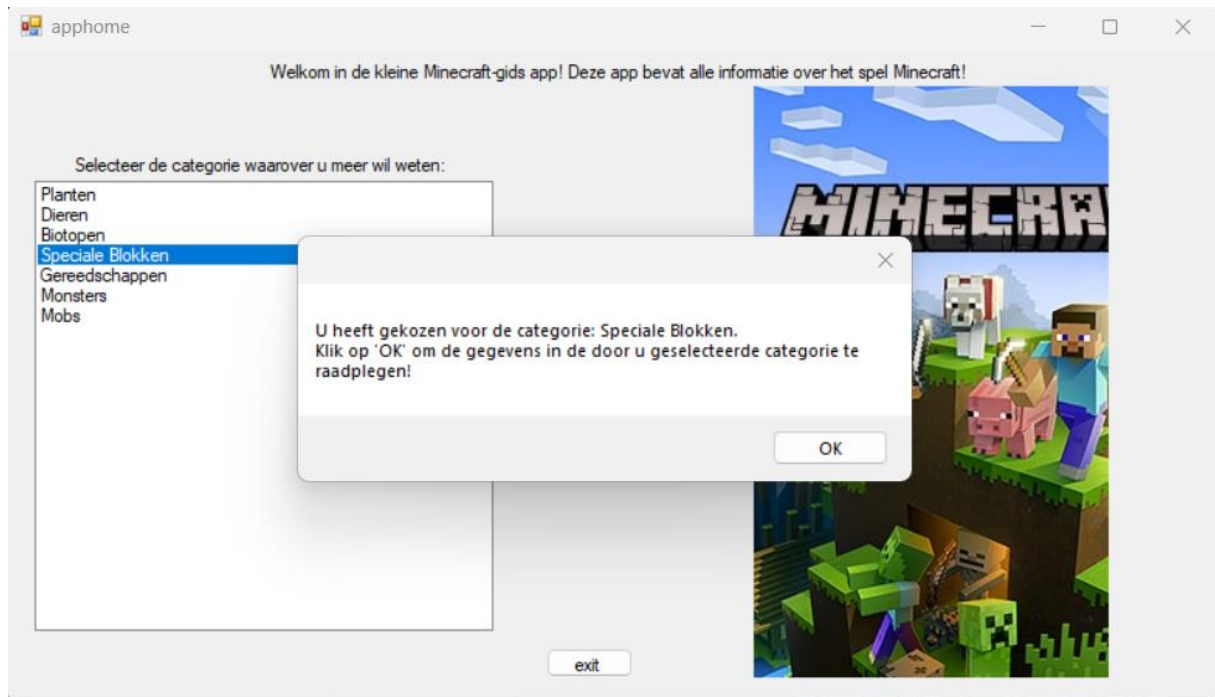


Een simpel experiment met een knop en een label. Als je op de knop drukt, verandert de tekst van het label naar "hello world!".

De uitwerking van de app

Nu tijd voor de echte app. Die bestaat in eerste instantie uit een simpel scherm met boodschappen en afbeelding, en een selectiescherm waar de informatie per categorie uit de database kan worden opgehaald.

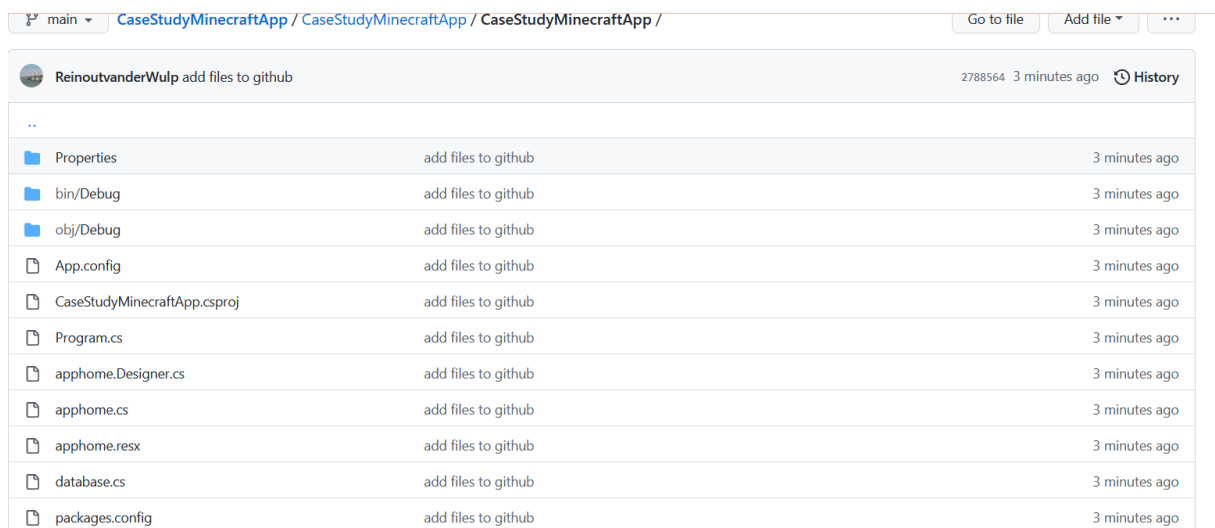




Als er op een van de categorieën geklikt word, is het de bedoeling dat nadat er op OK gedrukt word, de informatie opgehaald word uit de database. De database word in een volgende stap nog in orde gebracht. De knop exit onder in het scherm geeft een standaard melding om daarna de app te kunnen verlaten.

Aanmaken Github-repo

Nu de app er staat, is het tijd om de files op GitHub te plaatsen, en er een action aan toe te voegen:



```
build (Release)
cancelled 3 minutes ago in 42s

> Checkout 20s

Install .NET Core 14s

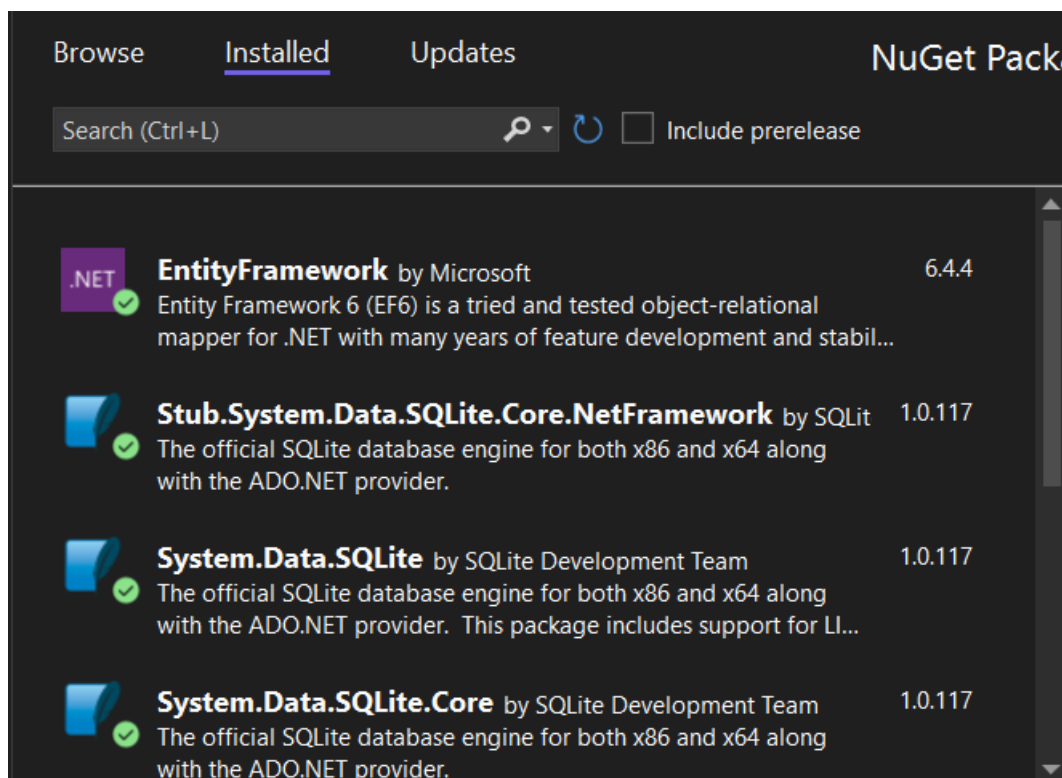
1 ▶ Run actions/setup-dotnet@v3
9 "C:\Program Files\PowerShell\7\pwsh.exe" -NoLogo -Sta -NoProfile -NonInteractive -ExecutionPolicy Unrestricted -Command &
'D:\a\_actions\actions\setup-dotnet\v3\externals\install-dotnet.ps1' -Channel 6.0
10 dotnet-install: Note that the intended use of this script is for Continuous Integration (CI) scenarios, where:
11 dotnet-install: - The SDK needs to be installed without user interaction and without admin rights.
12 dotnet-install: - The SDK installation doesn't need to persist across multiple CI runs.
13 dotnet-install: To set up a development environment or to run apps, use installers rather than this script. Visit
https://dotnet.microsoft.com/download to get the installer.
14
15 Error: The operation was canceled.
```

De workflow geeft echter een error:

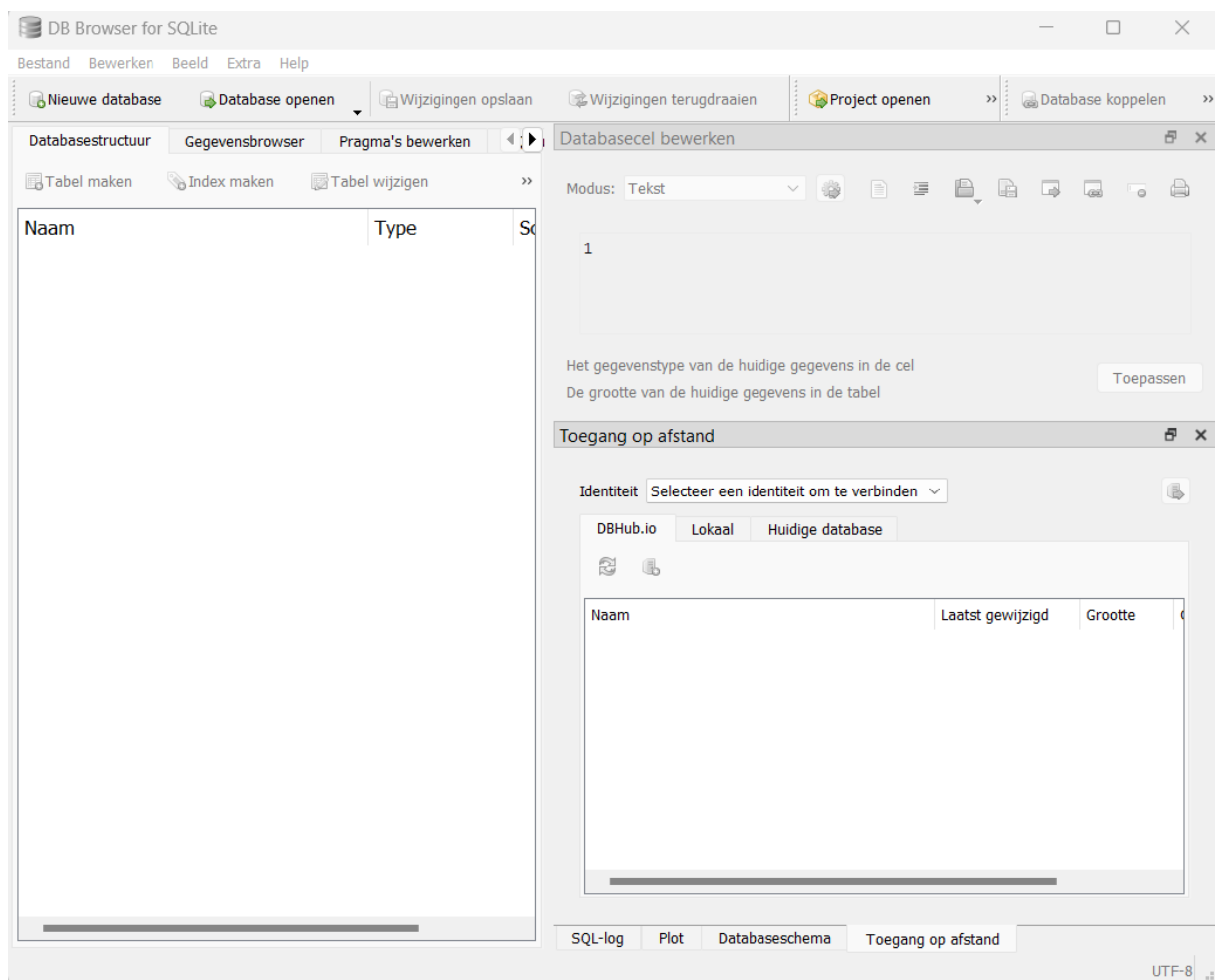
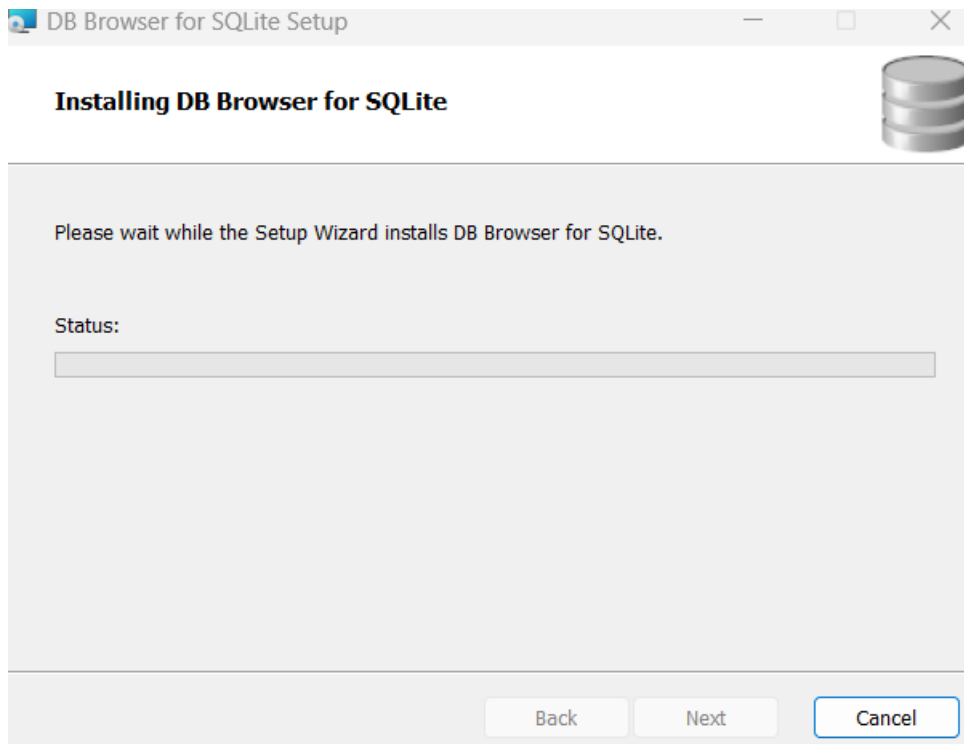
```
30 MSBUILD : error MSB1003: Specify a project or solution file. The current working directory does not contain a project or solution
file.
31 Error: Process completed with exit code 1.
```

Aanmaken van de SQLite-database

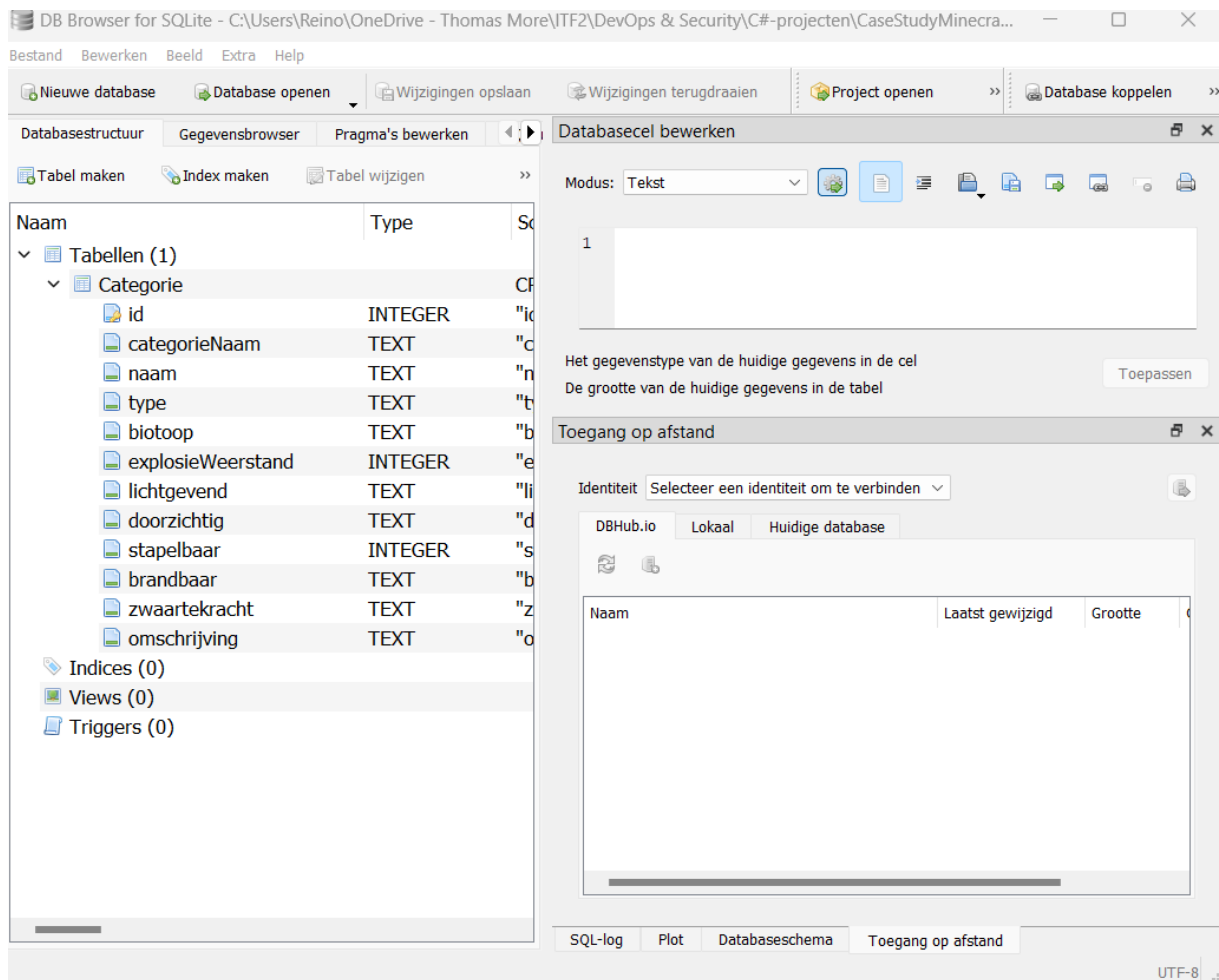
Nu is het tijd om de SQLite-database aan het project toe te voegen! Eerst de benodigde packages downloaden:



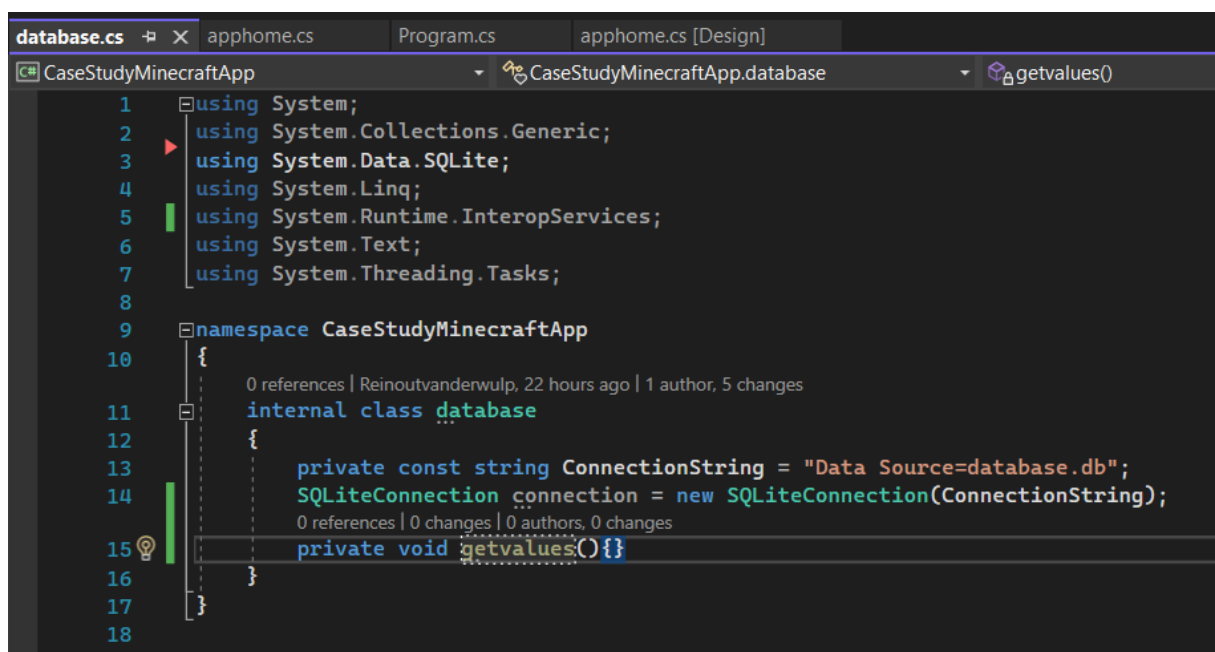
De volgende stap is de code schrijven om met de database in interactie te gaan, maar eerst maak ik de database aan met een tool die ik in een tutorial gevonden heb, Db browser voor SQLite. De screenshots:



Het aanmaken van de database met sqlite browser:



Ik maak nu een C#-klasse voor de connectie, en plaats er een methode in om de waarden te raadplegen:



Volgende stap is de database vullen en integreren in de app. De gevulde DB:

Databasestructuur Gegevensbrowser Pragma's bewerken SQL uitvoeren												
Tabel:	Categorie											
	id	categorieNaam	naam	type	biotoop	explosieWeerstand	lichtgevend	doorzichtig	stapelbaar	brandbaar	zwaartekracht	omschrij
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1 blokken	grasblok	blok	bijna alle		3	nee	nee	64	nee	nee	Het grasblok bedekt d
2	2 blokken	steen	blok	bijna alle		30	nee	nee	64	nee	nee	steen is ondergronds
3	3 blokken	aarde	blok	bijna alle		2.5	nee	nee	64	nee	nee	aarde komt voor tusse
4	4 blokken	keisteen	blok	bijna alle		30	nee	nee	64	nee	nee	keisteen zit in de gest
5	5 blokken	bode...	blok	bodem ...		18000000	nee	nee	64	nee	nee	bodemsteen vormd de
6	6 blokken	zand	blok	woestij...		2.5	nee	nee	64	nee	ja	zand bedekt de woest
7	7 blokken	grind	blok	bijna alle		3	nee	nee	64	nee	ja	grind komt voor in de
8	8 blokken	zands...	blok	woestij...		4	nee	nee	64	nee	nee	zandsteen is een stee
9	9 blokken	spinn...	blok	verlate...		20	nee	nee	64	nee	nee	spinnenweb is een blc
10	10 blokken	obsidi...	blok	the en...		6000	nee	nee	64	nee	nee	obsidiaan komt voor i
11	11 biotoop	ocean...	biot...	ocean...								diepe oceaan met kan
12	12 biotoop	geber...	biot...	geberg...								hooggebergte met pie
13	13 biotoop	woest...	biot...	woestij...								woestijn met cactus
14	14 biotoop	vlakte	biot...	vlaktes...								vlakte met veel plante
15	15 biotoop	mesa...	biot...	mesavl...								kleurrijke vlakte met r
16	16 speciale blokken	dispe...	blok	alle		17.5	nee	nee	64	nee	nee	schiet het materiaal a
17	17 speciale blokken	nootbl...	blok	alle		4	nee	nee	64	nee	nee	geeft een muzieknoot
18	18 speciale blokken	zuiger	blok	alle		2.5	nee	nee	64	nee	nee	duwt het blok boven,
19	19 speciale blokken	kleefz...	blok	alle		2.5	nee	nee	64	nee	nee	duwt het blok naast z
20	20 speciale blokken	deur	blok	alle		15.25	nee	nee	64	nee	nee	geeft een huiselijke sf
21	21 gereedschappen	fakkel	blok	alle		nee 14	nee	nee	64	nee	nee	verlicht mijnen en anc
22	22 gereedschappen	kist	blok	alle		12.5	nee	nee	64	nee	nee	opbergruimte voor alk
23	23 gereedschappen	kiet m	blok	alle		12.5	nee	nee	64	nee	nee	opbergruimte, maar n

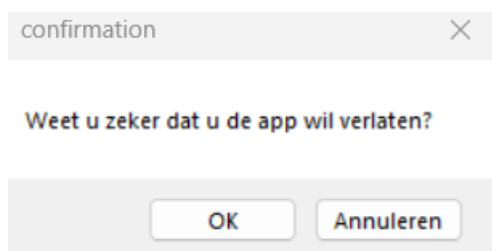
Nu ga ik methodes voorzien waarmee de app de database kan raadplegen, en kan aanpassen. Hiervoor moet ik soms het appdesign aanpassen, en code veranderen. Alle verschillende functies staan hieronder gedocumenteerd.

App afsluiten:

Voor deze functie hoef ik de layout niet aan te passen. De knop is ook al aanwezig, ik hoef alleen met de hulp van een tutorial de code zo aan te passen dat de app daadwerkelijk afgesloten word.

Screenshots van de code en de werking:

```
private void exitbutton_Click(object sender, EventArgs e)
{
    try
    {
        DialogResult result = MessageBox.Show("Weet u zeker dat u de app wil verlaten?", "confirmati", MessageBoxButtons.OKCancel);
        if (result == DialogResult.OK) {
            Application.Exit();
        }
    }
    catch (Exception ex) {
        MessageBox.Show(ex.Message);
    }
}
```



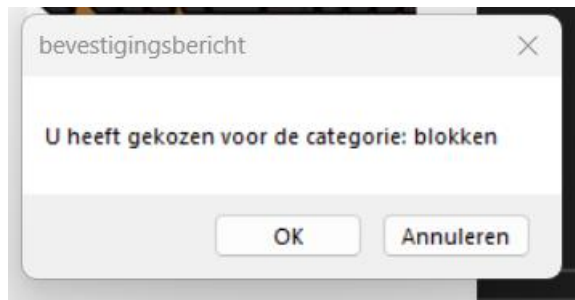
Gegevens uit de database halen per categorie:

De volgende functie haalt data uit de db via de categorieNaam. De data wordt vervolgens gedisplayed.

Screenshots en code van de gegevens die uit de db worden gehaald:

```
private void listBox_SelectedIndexChanged(object sender, EventArgs e)
{
    try
    {
        string text = listBox.GetItemText(listBox.SelectedItem);
        DialogResult result = MessageBox.Show("U heeft gekozen voor de categorie: " + text, "bevestigingsbericht", MessageBoxButtons.OKCancel);
        if (result == DialogResult.OK) {
            string connectionString = "Data Source=database.db";
            SQLiteConnection connection = new SQLiteConnection(connectionString);
            connection.Open();
            string sql = "SELECT * FROM Categorie WHERE categorieNaam = @text";

            SQLiteCommand command = new SQLiteCommand(sql, connection);
            command.Parameters.AddWithValue("@text", text);
            SQLiteDataReader reader = command.ExecuteReader();
            string dataresult = "";
            if (reader.HasRows)
            {
                while (reader.Read()) {
                    dataresult += "\rid: " + reader["id"];
                    dataresult += "\rCategorieNaam: " +
reader["categorieNaam"];
                    dataresult += "\rNaam: " + reader["naam"];
                    dataresult += "\rtype: " + reader["type"];
                    dataresult += "\rBiotoop: " + reader["biotoop"];
                    dataresult += "\rExploratieWeerstand: " +
reader["exploratieWeerstand"];
                    dataresult += "\rLichtgevend: " +
reader["lichtgevend"];
                    dataresult += "\rDoorzichtig: " +
reader["doorzichtig"];
                    dataresult += "\rStapelbaar: " +
reader["stapelbaar"];
                    dataresult += "\rBrandbaar: " + reader["brandbaar"];
                    dataresult += "\rZwaartekracht: " +
reader["zwaartekracht"];
                    dataresult += "\rOmschrijving: " +
reader["omschrijving"];
                }
            }
            MessageBox.Show(dataresult);
            connection.Close();
        }
    }
    catch (Exception ex) {
        MessageBox.Show(ex.Message);
    }
}
```



id: 1
 Categoriennaam: blokken
 Naam: grasblok
 Type: blok
 Biotoop: bijna alle
 Explosieweerstand: 3
 Lichtgevend: nee
 Doorzichtig: nee
 Stapelbaar: 64
 Brandbaar: nee
 Zwaartekracht: nee
 Omschrijving: Het grasblok bedekt de bovenwereld in vele kleuren

id: 2
 Categoriennaam: blokken
 Naam: steen
 Type: blok
 Biotoop: bijna alle
 Explosieweerstand: 30
 Lichtgevend: nee
 Doorzichtig: nee
 Stapelbaar: 64
 Brandbaar: nee
 Zwaartekracht: nee
 Omschrijving: steen is ondergronds het meest voorkomende blok in de bovenwereld

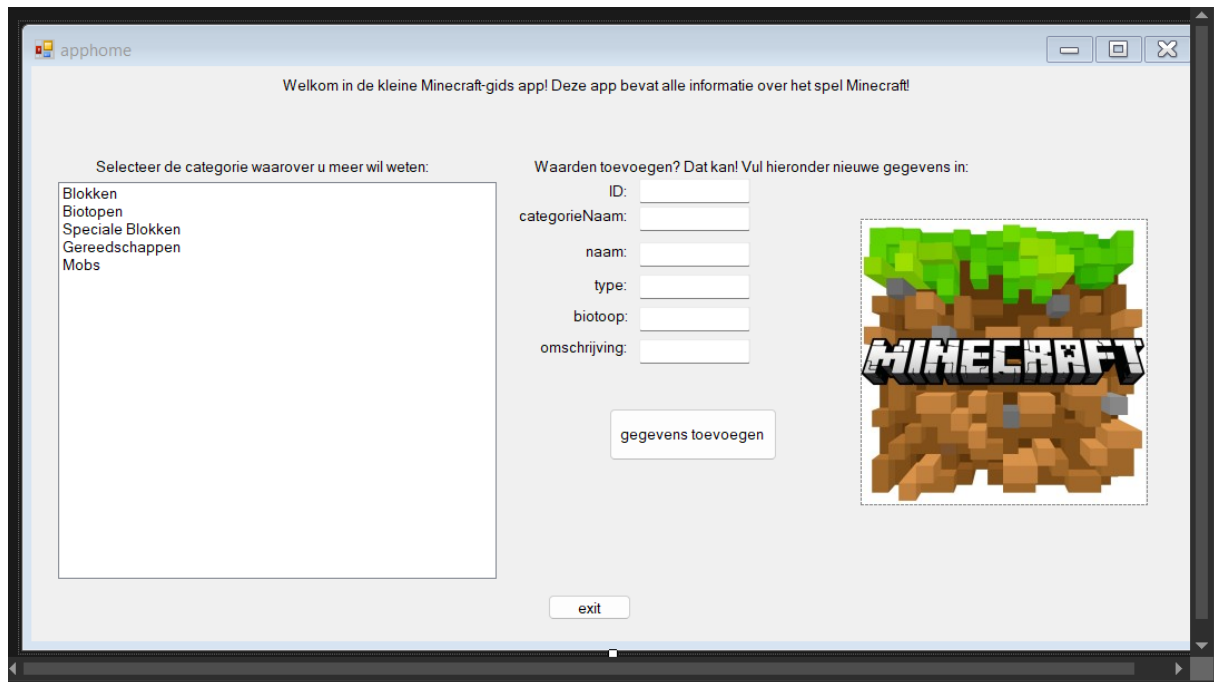
id: 3
 Categoriennaam: blokken
 Naam: aarde
 Type: blok
 Biotoop: bijna alle
 Explosieweerstand: 2
 Lichtgevend: nee
 Doorzichtig: nee
 Stapelbaar: 64
 Brandbaar: nee
 Zwaartekracht: nee
 Omschrijving: aarde komt voor tussen het grasblok en de gesteente-blokken

id: 4
 Categoriennaam: blokken
 Naam: keisteen
 Type: blok
 Biotoop: bijna alle
 Explosieweerstand: 30
 Lichtgevend: nee
 Doorzichtig: nee
 Stapelbaar: 64
 Brandbaar: nee
 Zwaartekracht: nee
 Omschrijving: keisteen zit in de gesteentelagen in de bovenwereld en kan zelfgemaakt worden

id: 5
 Categoriennaam: blokken
 Naam: bodemsteen
 Type: blok
 Biotoop: bodem bovenwereld
 Explosieweerstand: 18000000
 Lichtgevend: nee
 Doorzichtig: nee
 Stapelbaar: 64

Design aanpassen:

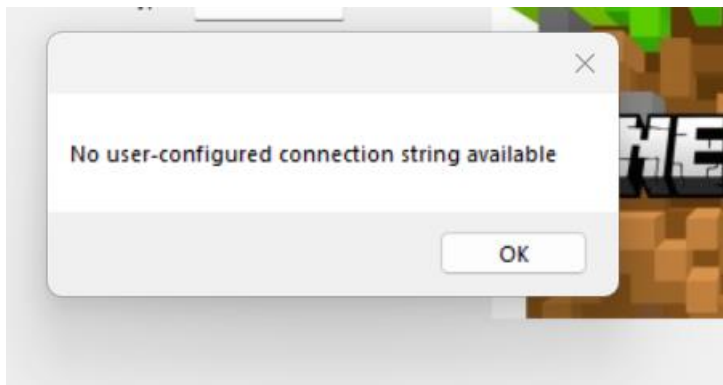
Ik pas het design zo aan dat er gegevens kunnen worden ingegeven. Er komen nieuwe labels en textboxes plus een button bij:



Gegevens toevoegen aan de database:

De volgende functie werkt helaas niet. De bedoeling was om een functie te bouwen met een ORM om data toe te voegen. Screenshots en code:

```
private void voegtoebutton_Click(object sender, EventArgs e)
{
    try {
        string id = idLabel.Text;
        string categoriaam = categoriaamLabel.Text;
        string naam = naamLabel.Text;
        string type = typeLabel.Text;
        string biotoop = biotoopLabel.Text;
        string omschrijving = omschrijvingLabel.Text;
        string connectionString = "Data Source:
database.db; Provider=System.Data.SQLite";
        database db = new database(connectionString);
        dynamic command = db.Single("INSERT INTO Categorie VALUES(@id,
@categoriaam, @naam, @type, @biotoop, @omschrijving)");
        command.Parameters.AddWithValue("@id", id);
        command.Parameters.AddWithValue("@categoriaam", categoriaam);
        command.Parameters.AddWithValue("@naam", naam);
        command.Parameters.AddWithValue("@type", type);
        command.Parameters.AddWithValue("@biotoop", biotoop);
        command.Parameters.AddWithValue("@omschrijving", omschrijving);
        MessageBox.Show("De waarden werden succesvol toegevoegd",
"Bevestigingsbericht", MessageBoxButtons.OKCancel);
    }
    catch (Exception ex) {
        MessageBox.Show(ex.Message);
    }
}
```



Afwerken van de app

In dit deel van mijn verslag is alle code te vinden, opgesplitst per file.

Database.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Mighty;

namespace CaseStudyMinecraftApp
{
    internal class database : MightyOrm
    {
        public database(string connection)
        {
        }
    }
}
```

Apphome.cs:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Diagnostics;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static System.Net.Mime.MediaTypeNames;
using Application = System.Windows.Forms.Application;

namespace CaseStudyMinecraftApp
{
    public partial class apphome : Form
    {
        public apphome()
        {
            InitializeComponent();
        }

        private void titleLabel_Click(object sender, EventArgs e)
        {
        }
    }
}
```

```

    }

    private void listBox_SelectedIndexChanged(object sender, EventArgs e)
    {
        try
        {
            string text = listBox.GetItemText(listBox.SelectedItem);
            DialogResult result = MessageBox.Show("U heeft gekozen voor de categorie: " + text, "bevestigingsbericht", MessageBoxButtons.OKCancel);
            if (result == DialogResult.OK) {
                string connectionString = "Data Source=database.db";
                SQLiteConnection connection = new SQLiteConnection(connectionString);
                connection.Open();
                string sql = "SELECT * FROM Categorie WHERE categorieNaam = @text";

                SQLiteCommand command = new SQLiteCommand(sql, connection);
                command.Parameters.AddWithValue("@text", text);
                SQLiteDataReader reader = command.ExecuteReader();
                string dataresult = "";
                if (reader.HasRows)
                {
                    while (reader.Read()) {
                        dataresult += "\rid: " + reader["id"];
                        dataresult += "\rCategorieNaam: " +
reader["categorieNaam"];
                        dataresult += "\rNaam: " + reader["naam"];
                        dataresult += "\rtype: " + reader["type"];
                        dataresult += "\rBiotoop: " + reader["biotoop"];
                        dataresult += "\rExploratieWeerstand: " +
reader["exploratieWeerstand"];
                        dataresult += "\rLichtgevend: " +
reader["lichtgevend"];
                        dataresult += "\rDoorzichtig: " +
reader["doorzichtig"];
                        dataresult += "\rStapelbaar: " +
reader["stapelbaar"];
                        dataresult += "\rBrandbaar: " + reader["brandbaar"];
                        dataresult += "\rZwaartekracht: " +
reader["zwaartekracht"];
                        dataresult += "\rOmschrijving: " +
reader["omschrijving"];
                    }
                }
                MessageBox.Show(dataresult);
                connection.Close();
            }
        }
        catch (Exception ex) {
            MessageBox.Show(ex.Message);
        }
    }

    private void exitbutton_Click(object sender, EventArgs e)
    {
        try
        {
            DialogResult result = MessageBox.Show("Weet u zeker dat u de app wilt verlaten?", "bevestigingsbericht", MessageBoxButtons.OKCancel);
            if (result == DialogResult.OK) {
                Application.Exit();
            }
        }
    }

```



```

    }
    catch (Exception ex) {
        MessageBox.Show(ex.Message);
    }
}

private void textBox1_TextChanged(object sender, EventArgs e)
{

}

private void verwijderenLabel_Click(object sender, EventArgs e)
{

}

private void toevoegenbutton_Click(object sender, EventArgs e)
{

}

private void deletebutton_Click(object sender, EventArgs e)
{

}

private void idLabel_Click(object sender, EventArgs e)
{

}

private void idtextBox_TextChanged(object sender, EventArgs e)
{

}

private void voegtoebutton_Click(object sender, EventArgs e)
{
    try {
        string id = idLabel.Text;
        string categoriaNaam = categoriaNaamLabel.Text;
        string naam = naamLabel.Text;
        string type = typeLabel.Text;
        string biotoop = biotoopLabel.Text;
        string omschrijving = omschrijvingLabel.Text;
        string connectionString = "Data Source:
database.db; Provider=System.Data.SQLite";
        database db = new database(connectionString);
        dynamic command = db.Single("INSERT INTO Categorie VALUES(@id,
@categoriaNaam, @naam, @type, @biotoop, @omschrijving)");
        command.Parameters.AddWithValue("@id", id);
        command.Parameters.AddWithValue("@categoriaNaam", categoriaNaam);
        command.Parameters.AddWithValue("@naam", naam);
        command.Parameters.AddWithValue("@type", type);
        command.Parameters.AddWithValue("@biotoop", biotoop);
        command.Parameters.AddWithValue("@omschrijving", omschrijving);
        MessageBox.Show("De waarden werden succesvol toegevoegd",
"Bevestigingsbericht", MessageBoxButtons.OKCancel);
    }
    catch (Exception ex) {
        MessageBox.Show(ex.Message);
    }
}

```

```

    }
}
Program.cs:

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace CaseStudyMinecraftApp
{
    internal static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new apphome());
        }
    }
}

```

De laatste stap die ik neem is het maken van de .exe file van de app.

Conclusie

Ik vond het een verassend project om aan te werken. Windows form is mij goed bevallen, alleen is de ORM en SQLite toch iets lastiger dan gedacht geweest. SQLite heb ik echter in de laatste fase beter mee kunnen werken dankzij SQLitebrowser. Ik denk dat ik nog meer met windows form ga spelen. Ik kijk terug op een mooi en leerzaam project om aan te werken.

Links

Link Github repository:

<https://github.com/ReinoutvanderWulp/CaseStudyMinecraftApp>

Link youtube video:

<https://youtu.be/6NxMz454W90>

Bibliografie

Hieronder de bronnen die ik gebruikt heb:

Windows form:

<https://learn.microsoft.com/en-us/visualstudio/ide/create-csharp-winform-visual-studio?view=vs-2022>

<https://learn.microsoft.com/en-us/visualstudio/get-started/csharp/tutorial-windows-forms-picture-viewer-layout?view=vs-2022>

<https://www.guru99.com/c-sharp-windows-forms-application.html>

<https://www.youtube.com/watch?v=8ioUk4RajpM>

<https://www.c-sharpcorner.com/UploadFile/c713c3/how-to-exit-in-C-Sharp/>

<https://stackoverflow.com/questions/26459103/how-to-make-exit-confirmation-yes-or-no-on-control-button-using-c-sharp-window-f>

<https://www.makeuseof.com/winforms-input-dialog-box-create-display/>

<https://learn.microsoft.com/en-us/dotnet/api/system.windows.forms.form.dialogresult?view=windowsdesktop-6.0>

Setting up GitHub pipeline:

<https://www.cb nuggets.com/blog/certifications/microsoft/setting-up-a-ci-pipeline-with-github-actions-in-c-with-examples>

<https://github.blog/2022-02-02-build-ci-cd-pipeline-github-actions-four-steps/>

<https://stackoverflow.com/questions/61188284/github-action-dotnetcore-sln-and-project-in-same-folder>

<https://www.youtube.com/watch?v=VIIDni8-iWM>

<https://learn.microsoft.com/en-us/dotnet/devops/create-dotnet-github-action>

SQLite:

<https://www.c-sharpcorner.com/UploadFile/5d065a/how-to-use-and-connect-sqlite-in-a-window-application/>

<https://www.codeguru.com/dotnet/using-sqlite-in-a-c-application/>

<https://github.com/liviucotfas/ase-windows-applications-programming/blob/master/10%20-%20WinForms%20-%20Databases%20-%20SQLite.md>

<https://learn.microsoft.com/en-us/ef/core/get-started/winforms>

<https://zetcode.com/csharp/sqlite/>

<https://sqlitebrowser.org/>

Overerving:

<https://learn.microsoft.com/en-us/dotnet/desktop/winforms/advanced/how-to-inherit-windows-forms?view=netframeworkdesktop-4.8>

<https://learn.microsoft.com/en-us/dotnet/desktop/winforms/advanced/windows-forms-visual-inheritance?view=netframeworkdesktop-4.8>

<https://stackoverflow.com/questions/2094401/how-to-inherit-system-data-sqlite-in-c-sharp>

<https://medium.com/swlh/introducing-mighty-a-new-net-9bcc7bdd4814>

<https://www.geeksforgeeks.org/c-sharp-inheritance-in-constructors/>

ORM:

<https://stackoverflow.com/questions/3629761/best-orm-for-net-windows-forms-applications>

<https://foxlearn.com/windows-forms/data-access-sql-database-in-csharp-23.html>

<https://supportcenter.devexpress.com/ticket/details/q448226/how-do-i-use-orm-data-model-in-windows-forms>

<http://persianprogrammer.com/Education/Index/ORMs-for-NET-Core>

<https://mightyorm.github.io/Mighty/docs/getting-started>