**Verslag Case Study Windows Forms**

Inhoud

[Voorwoord 3](#_Toc122299390)

[Het project 3](#_Toc122299391)

[Voorbereiding 3](#_Toc122299392)

[Testen met windows form 3](#_Toc122299393)

[De uitwerking van de app 4](#_Toc122299394)

[Aanmaken Github-repo 6](#_Toc122299395)

[Aanmaken van de SQlite-database 7](#_Toc122299396)

[Afwerken van de app 15](#_Toc122299397)

[Conclusie 19](#_Toc122299398)

[Links 19](#_Toc122299399)

[Bibliografie 19](#_Toc122299400)

# 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

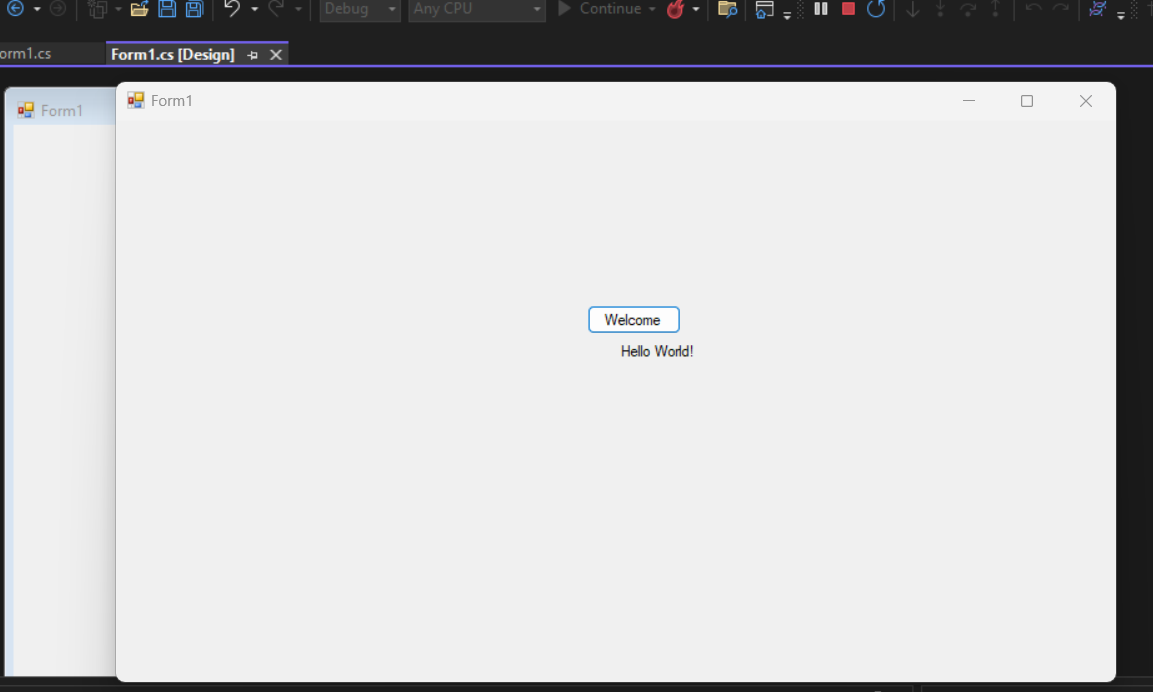
Als eerste ben ik begonnen met een klassiek hello-world experiment, om te testen. Hieronder de screenshots:

Afbeelding met tekst, schermafbeelding, monitor, scherm

Automatisch gegenereerde beschrijving

Afbeelding met tekst, schermafbeelding, monitor, scherm

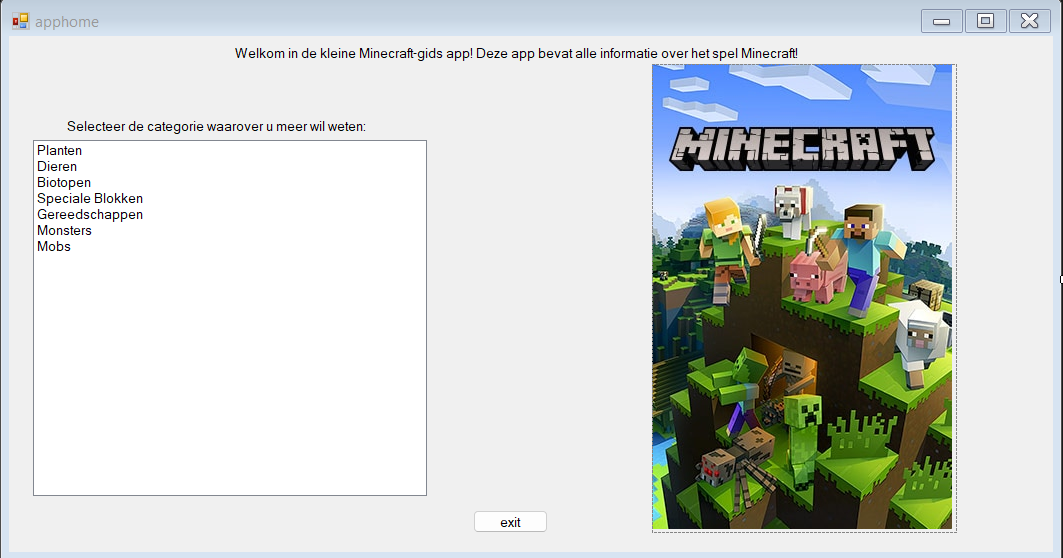
Automatisch gegenereerde beschrijving

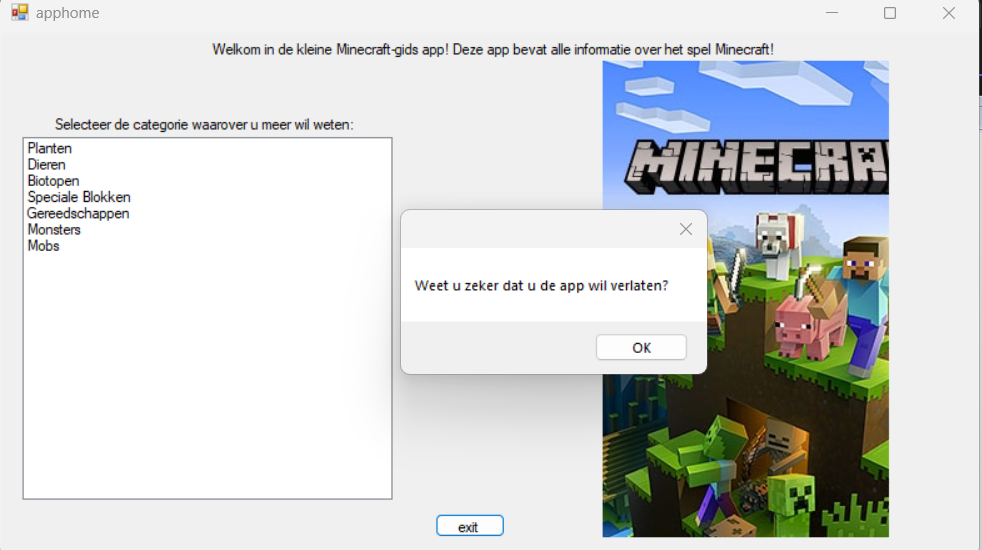


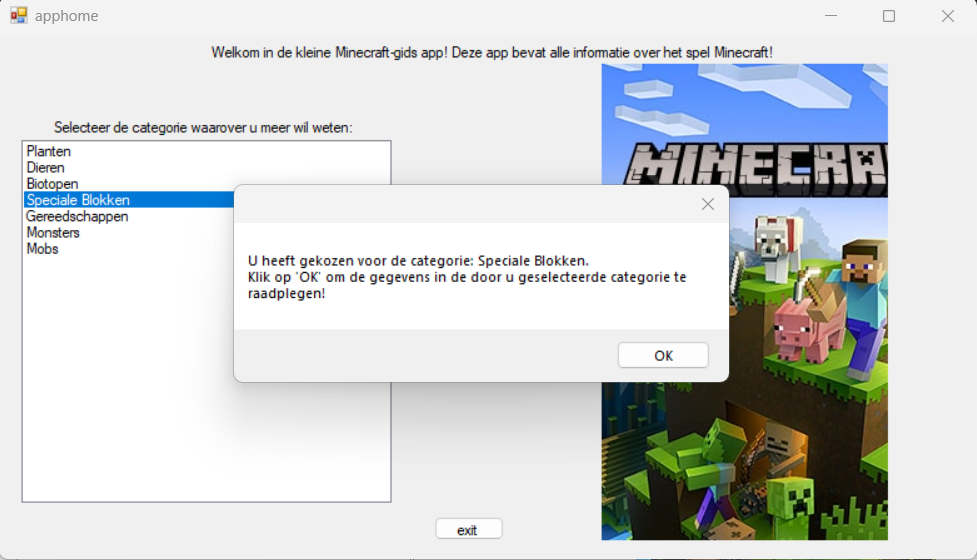
Een simpel experiment met een knop en een label. Als je op de knop drukt, verandert de tekst van het label naar “hello world!”.

## 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.



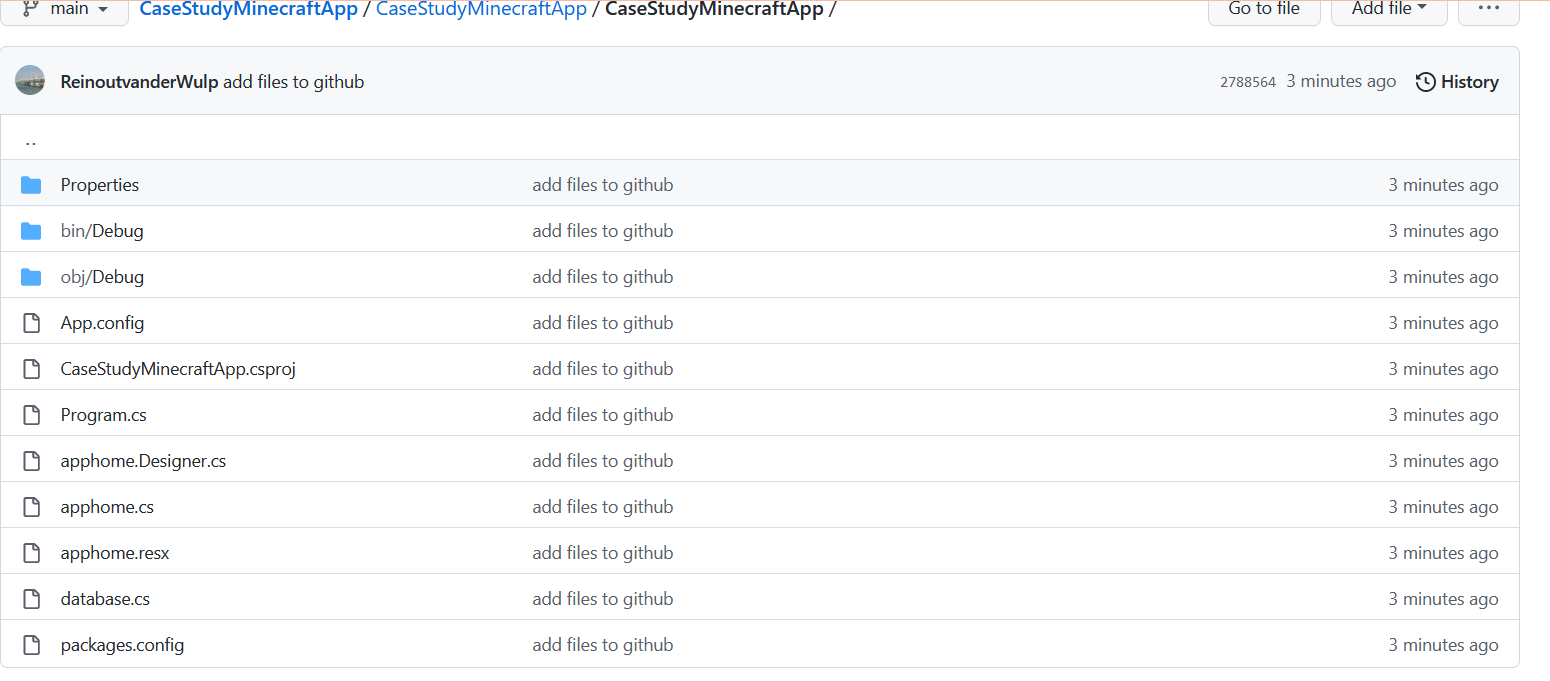




Als er op een van de categorieën geklikt word, is het de bedoeling dat nadat er op OK gedrukt word, de informatie opgehaald word uit de database. De database word in een volgende stap nog in orde gebracht. De knop exit onder in het scherm geeft een standaard melding om daarna de app te kunnen verlaten.

## Aanmaken Github-repo

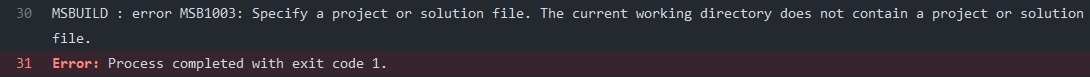
Nu de app er staat, is het tijd om de files op GitHub te plaatsen, en er een action aan toe te voegen:



Afbeelding met tekst

Automatisch gegenereerde beschrijving

De workflow geeft echter een error:



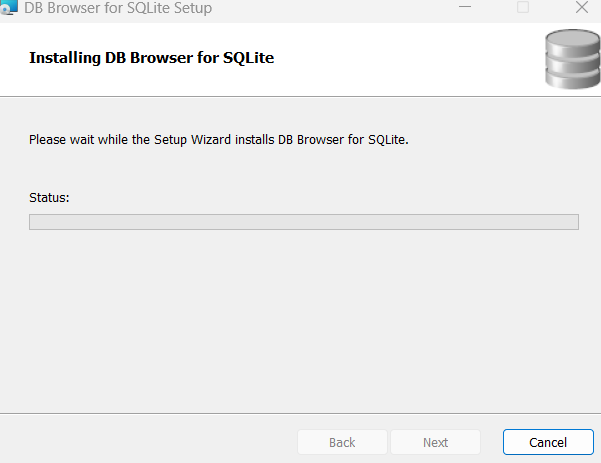
## Aanmaken van de SQlite-database

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

Afbeelding met tekst

Automatisch gegenereerde beschrijving

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:



Afbeelding met tekst

Automatisch gegenereerde beschrijving

Het aanmaken van de database met sqlite browser:

Afbeelding met tekst

Automatisch gegenereerde beschrijving

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

Afbeelding met tekst, schermafbeelding, monitor, scherm

Automatisch gegenereerde beschrijving

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

**Afbeelding met tafel

Automatisch gegenereerde beschrijving**

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?", "confirmation", MessageBoxButtons.OKCancel);

if (result == DialogResult.OK) {

Application.Exit();

}

}

catch (Exception ex) {

MessageBox.Show(ex.Message);

}

}

Afbeelding met tekst

Automatisch gegenereerde beschrijving

*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 += "\rExplosieweerstand: " + reader["explosieWeerstand"];

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);

}

}

Afbeelding met tekst

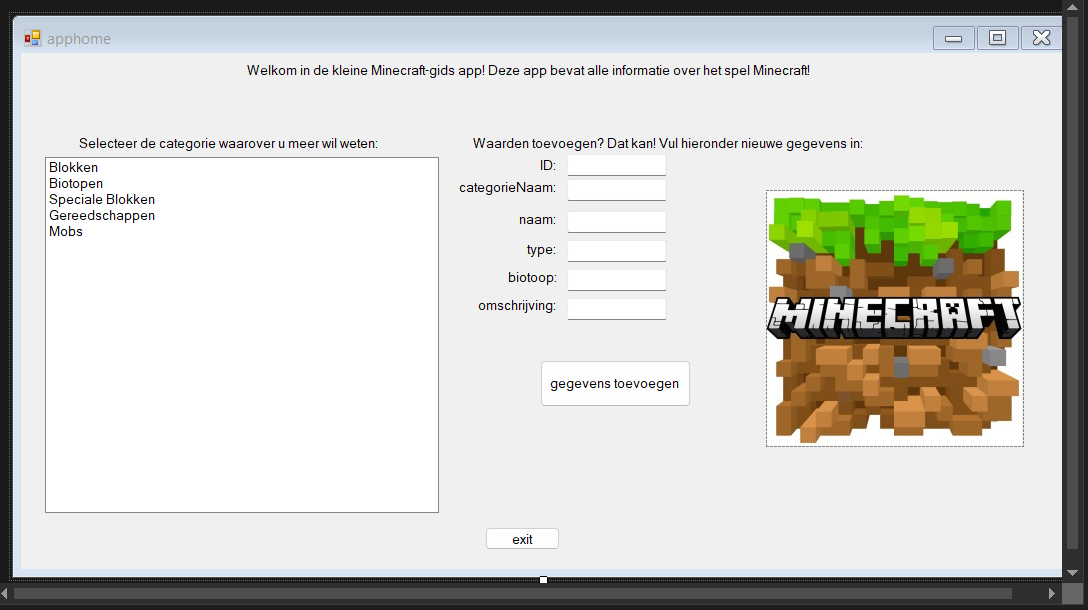
Automatisch gegenereerde beschrijving

Afbeelding met tafel

Automatisch gegenereerde beschrijving

*Design aanpassen:*

Ik pas het design zo aan dat er gegevens kunnen worden ingegeven. Er komen nieuwe labels en textboxen 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 categorienaam = categorienaamlabel.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, @categorienaam, @naam, @type, @biotoop, @omschrijving)");

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@categorienaam", categorienaam);

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);

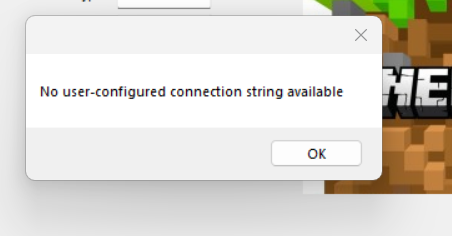
}

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.SQLite;

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 titletextlable\_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 += "\rExplosieweerstand: " + reader["explosieWeerstand"];

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 wil 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 categorienaam = categorienaamlabel.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, @categorienaam, @naam, @type, @biotoop, @omschrijving)");

command.Parameters.AddWithValue("@id", id);

command.Parameters.AddWithValue("@categorienaam", categorienaam);

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.cbtnuggets.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=VIlDni8-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>