

TRF

Tigerälskarnas Riksförbund

Rapport

Programmering 2 C#

NTI-Skolan

Johan Kämpe

<https://www.linkedin.com/in/johankampe>

<https://github.com/GoblinDynamiteer>

2017-07-17

Innehåll

1 Inledning.....	3
1.1 Syfte.....	3
1.2 Noteringar	3
1.3 Lösenord.....	3
1.4 Länkar	3
2 Genomförande och resultat	4
2.1 Använd programvara och litteratur.....	4
2.2 Avgränsningar och krav	5
2.2.1 Kravspecifikation	5
2.2.2 Redovisning	5
2.3 Metod	6
2.3.1 Arbete med Visual Studio	6
2.3.2 Kommentering av kod	9
2.4 Planering.....	10
2.4.1 Flödesschema	11
2.5 Programmets funktion	12
2.5.1 Inlogging och filer	12
2.5.2 Beskrivning av programmets huvudfönster	14
2.5.3 Lägg till en ny medlem.....	15
2.5.4 Lägg till en ny tiger	16
2.5.5 Sökfilter	17
2.5.6 Meny Fil	18
2.5.7 Meny Verktyg: Ändra lösenord	19
2.5.8 Meny Hjälp	20
2.6 Programmets kod	21
2.6.1 Klassdiagram.....	22
3 Diskussion och slutsats	24
3.1 Förslag på förbättringar.....	24

1 Inledning

1.1 Syfte

Syftet med uppgiften är att skapa ett program i programspråket C#, som underlag finns en projektspecifikation med vissa krav som måste uppfyllas.

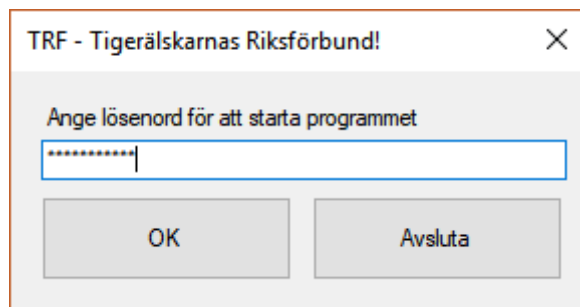
Utöver dessa krav får programmet skapas med fria händer.

1.2 Noteringar

Skärmdokument i rapporten visar ibland en äldre version av programmet, vissa skillnader kan förekomma i utseende och text i det inlämnade programmet.

1.3 Lösenord

Programmet startar med ett loginfönster, där ett lösenord måste skrivas in för att komma vidare.



Figur 1 loginfönster

Lösenordet sparas i en fil kallad *Login.pwd*. Om denna fil av någon anledning saknas i den katalog där programmets exekverbara fil ligger används ett standardlösenord.

Lösenord för att logga in i programmet	
TigrarÄger123!	Om filen login.pwd finns tillgänglig för programmet
123	Standardlösenord om login.pwd inte används

Användaren har möjlighet att själv byta lösenord i programmet, om så önskas.

1.4 Länkar

Uppgiften på GitHub

https://github.com/GoblinDynamiteer/P2CS/tree/master/course_exercises/project

Ikoner som används i programmet

https://www.iconfinder.com/icons/381599/error_icon

https://www.iconfinder.com/icons/285654/cat_icon

2 Genomförande och resultat

2.1 Använd programvara och litteratur

Programvara

- Microsoft Visual Studio Community 2017 v4.7.02045
- Microsoft Word 2016 v1701
- Draw.io: <https://www.draw.io>
- Trello, planeringsverktyg: <https://trello.com>

Litteratur

- Alishenas, T *Programmering 2 med C#* (ISBN 9789197420433)

2.2 Avgränsningar och krav

2.2.1 Kravspecifikation

Medlemmarna i TRF består av människor som på något sätt skaffat sig en tiger som husdjur. Ditt jobb här blir nu att skapa ett program som håller reda på namn och adress till medlemmarna och namn på tigern/tigrarna.

Du får helt fria händer, bortsett från nedanstående punkter:

- Programmet **ska** innehålla en databas, och det ska givetvis då gå att lägga till/ta bort medlemmar.
- Det **bör** någonstans finnas en länk till Tiger på Wikipedia, klickar man på den så ska webbsidan visas (<http://sv.wikipedia.org/wiki/Tiger>)

Tillägg för högre betyg:

- När man startar programmet så ska man komma till någon sorts inloggningsruta, och lösenord osv. ska i så fall sparas i en textfil på hårddisken (gärna krypterad).
- Kom ihåg att nämna inloggningsuppgifterna i rapporten

Mer att tänka på:

- Programmet bedöms till mycket stor del efter hur du löst allt (inklusive upplägg av databasen).
- Använd minst två klasser och försök att dela upp koden med metoder osv.
- Klassdiagram **ska** finnas med.
- Någon form av felhantering för att undvika körfel osv. **bör** finnas med.

2.2.2 Redovisning

Programmet

- Programmet ska fungera och uppfylla kravspecifikationen.
- Redovisa programmet genom att **skicka in källkodsfilerna** via din elevsida (komprimera ihop hela projektmappen).
- Programmet **ska** vara försett med **utförliga kommentarer** som förklarar de olika delarna i programmet.
- Programmet **ska** vara utan stilfel, dvs. med korrekta indrag och radbrytningar.
- För högre betyg **bör** programmet **utökas** med annan funktionalitet, utöver det som anges i beskrivningen, så länge detta inte strider mot den grundläggande beskrivningen, och uppfyller kraven.

Dokumentation

- **Projekt-rapport** ska skickas in tillsammans med källkoden en där du **beskriver** hur du gick till väga för att skapa programmet. Vilka avvägningar du gjort i programupplägget och motiveringar till dessa. Du **bör** också ge exempel på eventuella förbättringar.
- Uppgiften ska planeras med hjälp av pseudokod och/eller flödesschema som du sedan skickar in tillsammans med programmet.

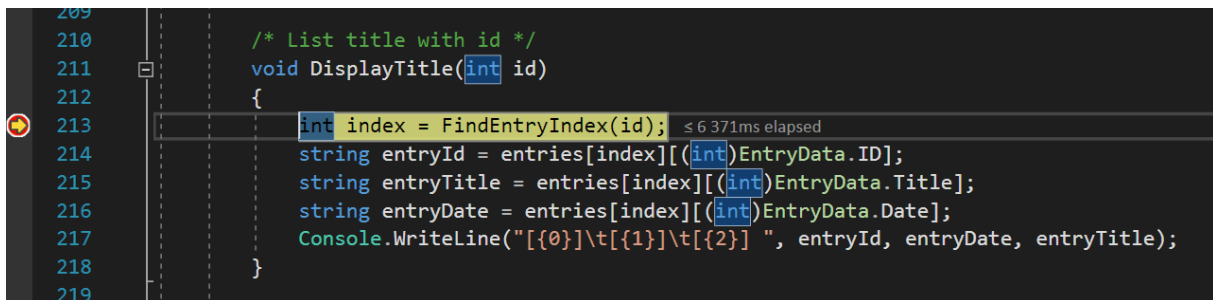
2.3 Metod

2.3.1 Arbete med Visual Studio

Programmets källkod skrevs uteslutande i utvecklingsmiljön *Visual Studio*. Som projektmall användes *Windows Forms App (.NET Framework)*.

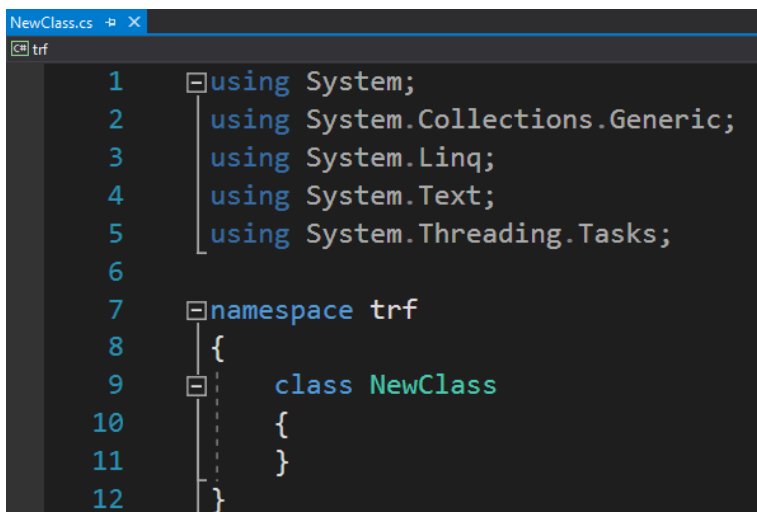
I Visual Studio finns ett hjälpverktyg kallat *IntelliSense*, som underlättar vid kodskrivning.

Visual Studio har också stöd för debugging med breakpoints, vilket användes för att felsöka programmet.



Figur 2 Kod i Visual Studio, debugläge med break point

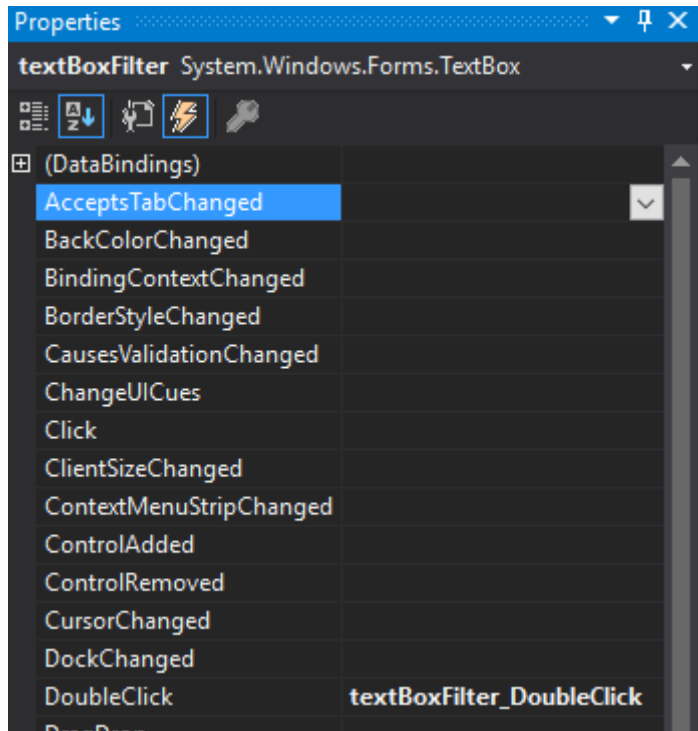
Egna klasser lades till i projektet genom att välja "Project -> Add Class..." i Visual Studios meny. Visual studio skapar då en ny cs-fil i projektet, med färdiggenererad kod för att skapa en ny klass.



Figur 3 Ny klassfil skapad i Visual Studio

På liknande sätt skapades nya Windows Forms till projektet, med "Project -> Add Windows Form..."

De flesta event-metoder som finns i projektet skapades genom att dubbelklicka på den kontroll som de tillhör, i Visual Studios Form Design-läge, eller genom att dubbelklicka på det tomma utrymmet till höger om det önskade eventet i kontrollens egenskaper.



Figur 4 Events för kontrollen textBoxFilter

Namnen som Visual Studio automatiskt genererar till metoderna behövs i de flesta fall.

```
private void exitToolStripMenuItem_Click(object sender, EventArgs e)
{
    UpdateDatabase();
    Program.QuitProgram();
}
```

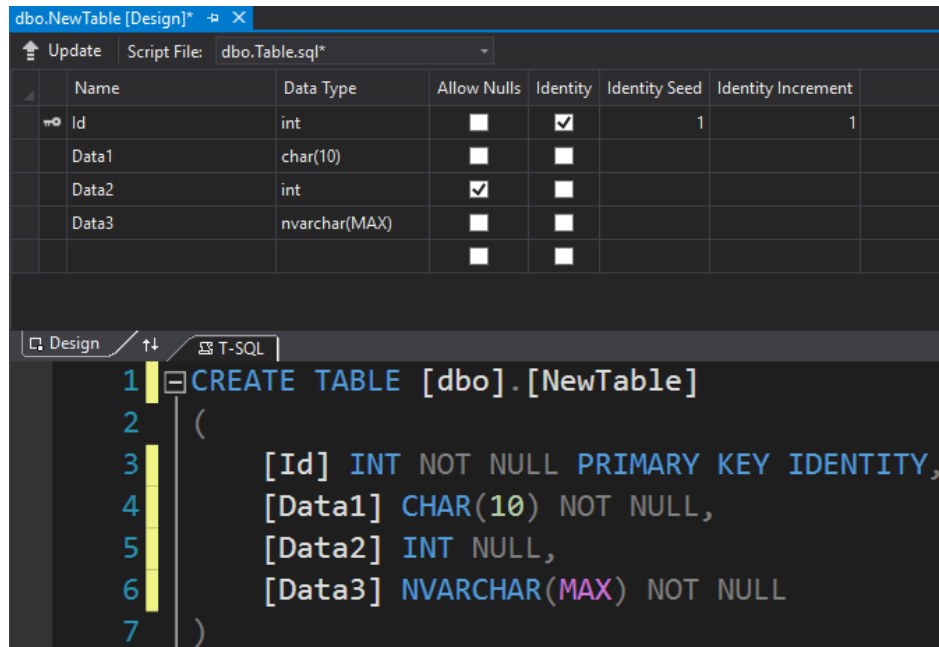
Figur 5 Automatiskt genererat metodnamn för event till kontroll

Databasen som används i programmet skapades och hanterades också i Visual Studio.

En ny databas läggs till i ett projekt med "Project -> Add New Item -> Service-based Database"

Kapitel 5 i läroboken *Programmering 2 med C#* användes i stor grad som hjälpmedel vid arbetet med databasen.

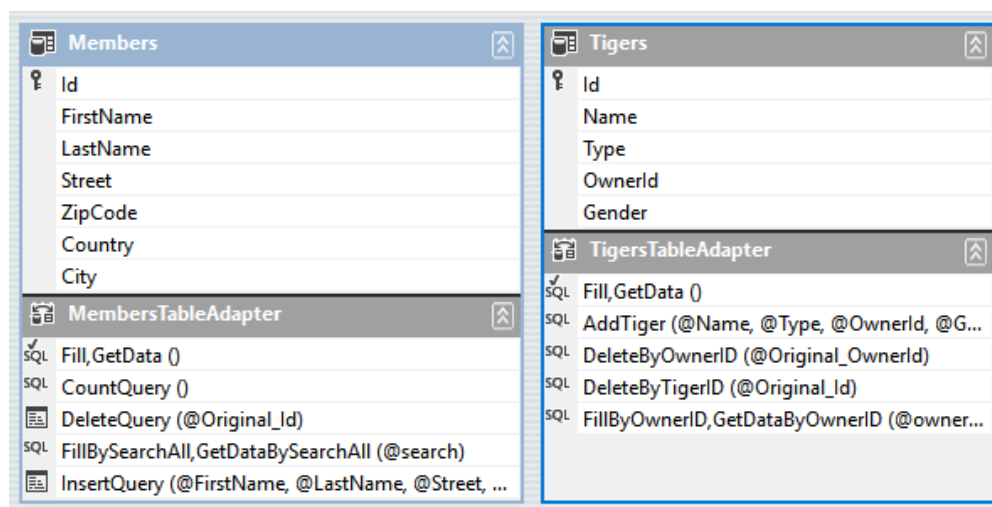
Databasens tabeller skapades i Visual Studio med hjälp av dess "Table Designer".



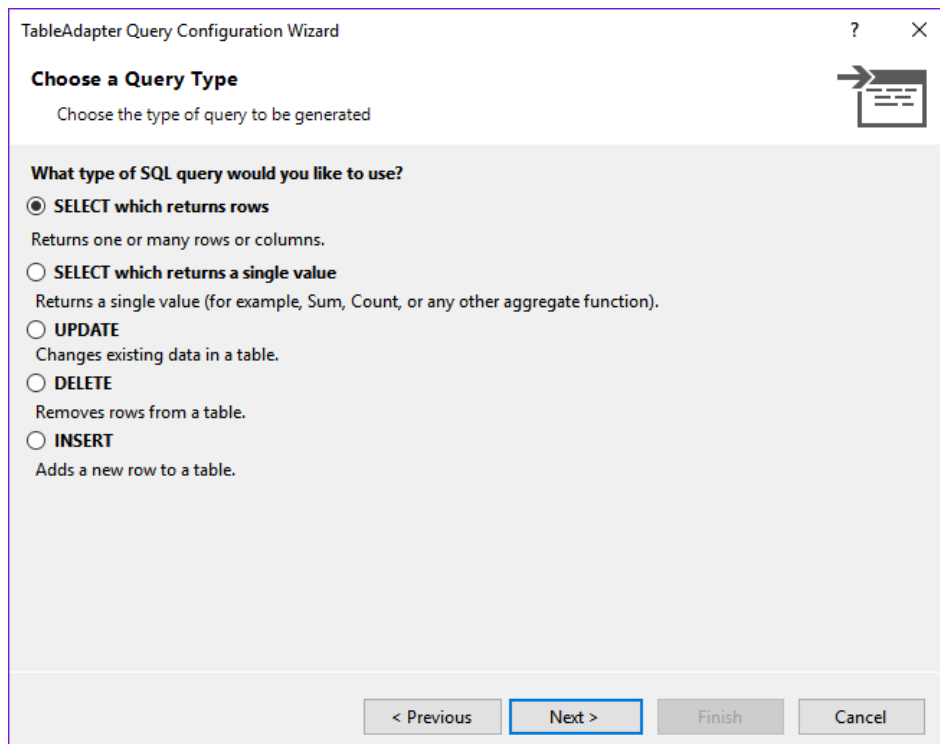
Figur 6 Visual Studios Table Designer

Visual Studio använder sig av så kallade "Table Adapters" för att sköta kommunikation mellan databaser och projektets kod. Till dessa adapters är det möjligt att skapa SQL-kommandon, som sparas som metoder till adapter-klassen.

Till projektet skapades flertalet SQL-kommandon (query), för att exempelvis söka bland medlemmar i TRF eller lägga till eller ta bort medlemmar. SQL-kommandona kan skapas med hjälp av guider i programmet.



Figur 7 Table Adapters med SQL-queries



Figur 8 SQL-guide i Visual Studio

2.3.2 Kommentering av kod

Programmets kod är kommenterad, där koden inte är automatiskt genererad av Visual Studio.

Flerradskommentarer (/* KOMMENTAR */) beskriver den kod som finns på raden under medans enradskommentarer (// KOMMENTAR) beskriver den kod som finns till vänster om kommentaren.

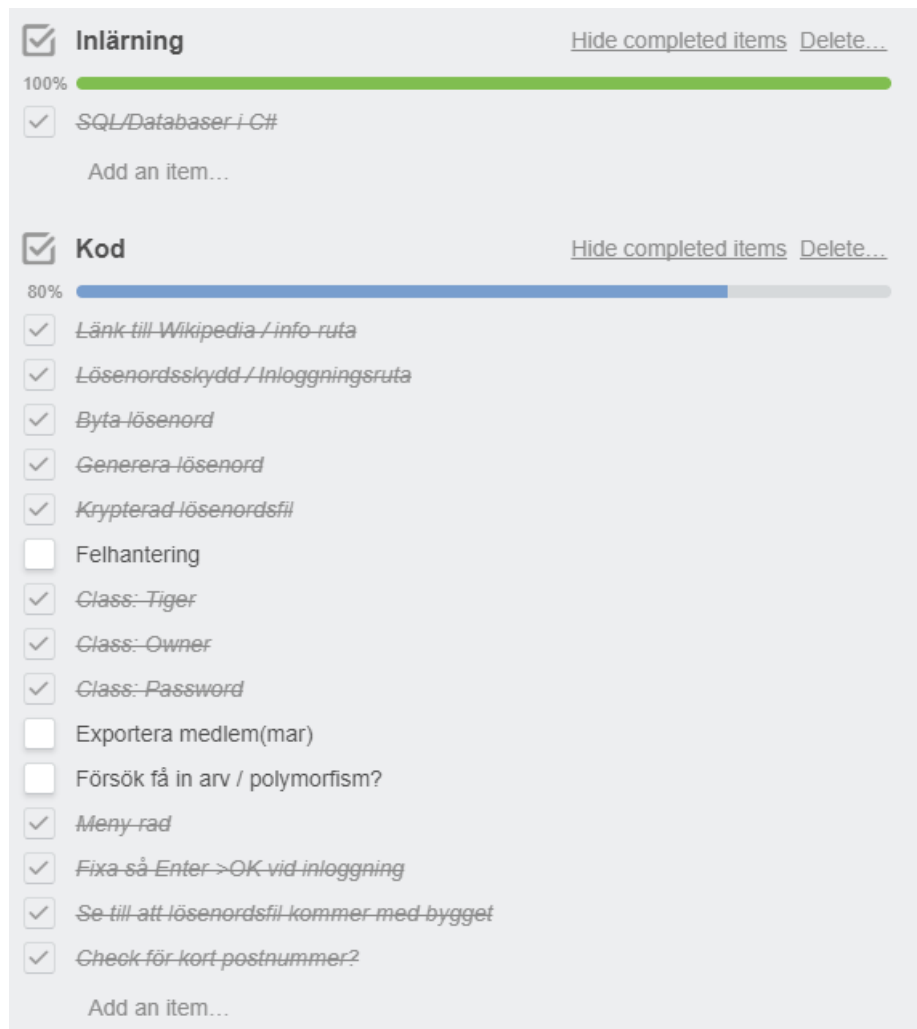
```
/* Anropas när användaren dubbelklickar på sökfilter-rutan */
private void textBoxFilter_DoubleClick(object sender, EventArgs e)
{
    textBoxFilter.Text = ""; // Rensa filter
}
```

Figur 9 Kommentarer i källkoden

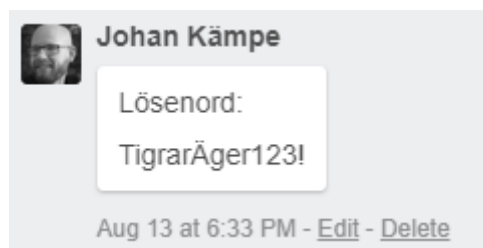
2.4 Planering

Projektet planerades genom att skapa ett flödesschema med webverktyget draw.io, endast programmets grundläggande funktion, samt inloggningsruta lades till.

Det webbaserade planeringsverktyget Trello användes som checklista och som en sorts "dagbok" för att skriva ned tankar och kommentarer kring utvecklingen av programmet.

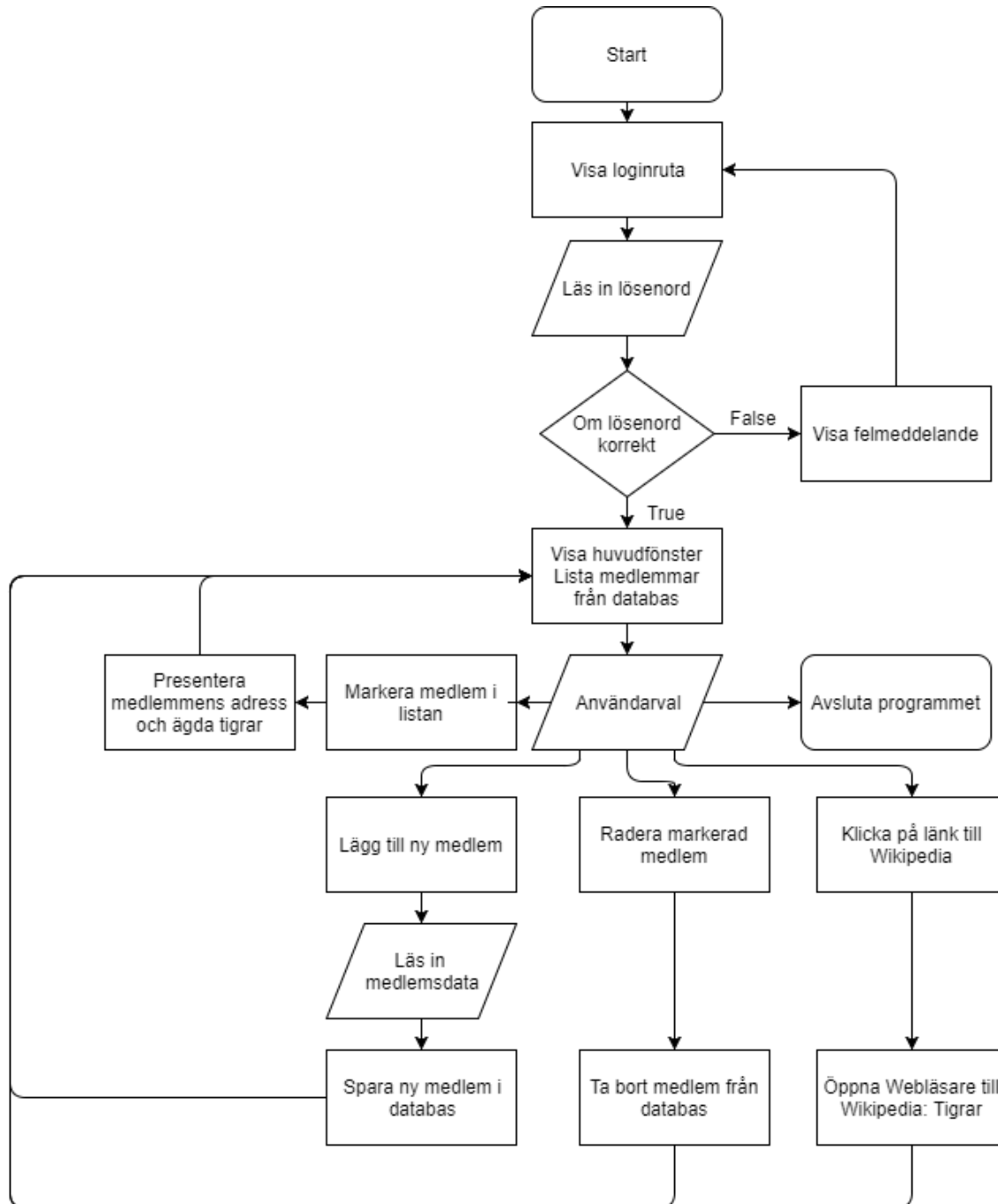


Figur 10 Checklistor i Trello



Figur 11 Kommentar i Trello

2.4.1 Flödesschema



Figur 12 Flödesschema

2.5 Programmets funktion

2.5.1 Inloggning och filer

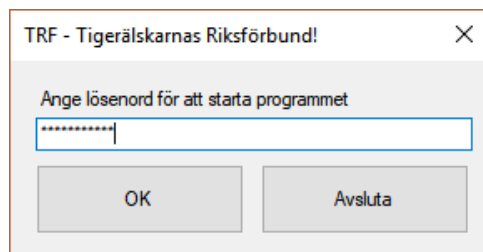
Programmet startas med den exekverbara filen *trf.exe*. För att programmet ska fungera korrekt behövs också databasfilen *Members.mdf* i samma katalog.

En tredje fil, *Login.pwd* tillhör också programmet, denna fil innehåller ett sparad lösenord för att logga in. Denna fil är dock inte nödvändig för programmets funktion.

#	Name ^	Size	Modified	Comment
1	login.pwd	35 bytes	3 hrs	
2	Members.mdf	8,00 MB	23 secs	
3	trf.exe	344,50 KB	2 mins	

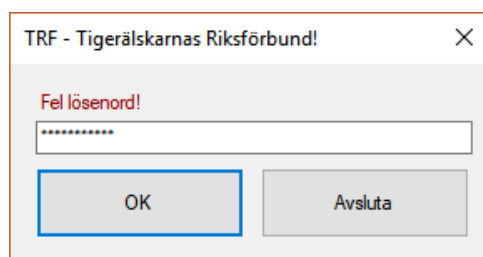
Figur 13 Programmets filer

Vid uppstart visas en inloggningsruta, där användaren uppmanas att skriva in ett lösenord för att fortsätta.



Figur 14 Inloggningsruta

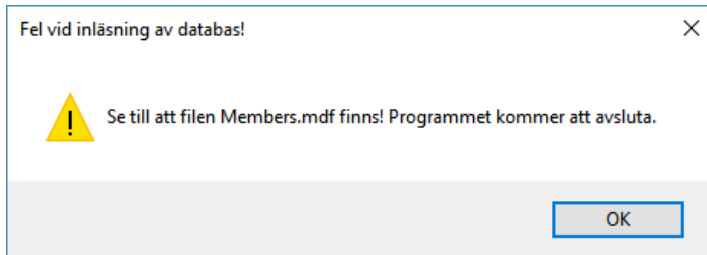
Om ett felaktigt lösenord skrivs in och användaren trycker på *OK*-knappen visas ett felmeddelande. Programmet avslutas om användaren trycker på "krysset" eller *Avsluta*-knappen.



Figur 15 Felmeddelande om fel lösenord

Om filen *Login.pwd* fanns vid programmets uppstart är lösenordet **TigrarÄger123!**, annars är lösenordet **123**. Om användaren har valt ett annat lösenord vid ett tidigare tillfälle så är det det aktuella lösenordet.

Om databasfilen *Members.mdf* saknas kommer ett felmeddelande att visas efter lyckad inloggning, och programmet avslutas.

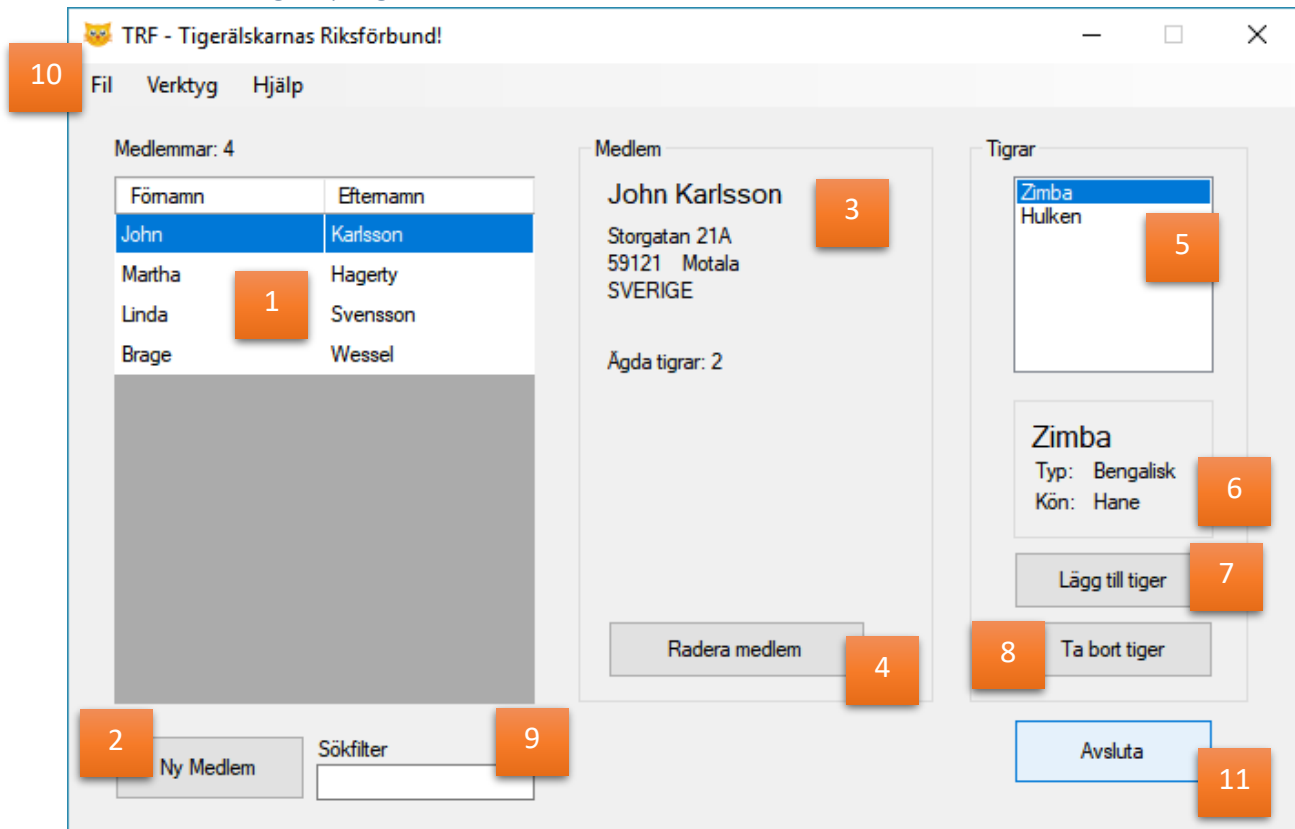


Figur 16 Felmeddelande om databas

Filerna *Members.mdf* och *Login.pwd* kopieras alltid automatiskt av Visual Studio till det kompilerade programmets katalog vid varje kompilering. Det innebär att ändringar (databasändringar eller lösenordbyte) som har utförts vid körning av programmet via Visual Studio kommer att återställas vid varje ny körning via Visual Studio.

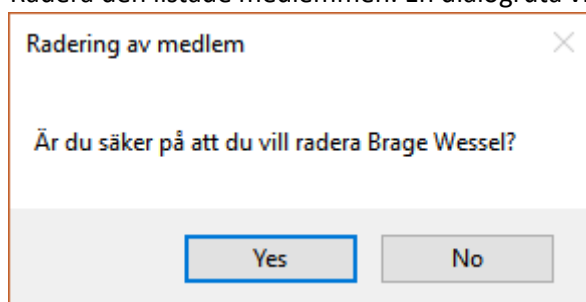
Om databasfilen finns, och den laddas in av programmet korrekt, så presenteras programmets huvudfönster.

2.5.2 Beskrivning av programmets huvudfönster



Figur 17 Huvudfönster med beskrivning

- 1 Medlemslista, användaren klickar på ett namn för att välja en annan medlem. Listan kan sorteras genom att klicka på rubrikerna "Förnamn" eller "Efternamn".
- 2 Lägger till en ny medlem, ett nytt fönster öppnas.
- 3 Information om den markerade medlemmen.
- 4 Radera den listade medlemmen. En dialogruta visas för att bekräfta.



Figur 18 Radering av medlem

- 5 Den markerade medlemmens tigrar.
- 6 Information om den markerade tigrern.
- 7 Lägg till en ny tiger till den markerade medlemmen. Ett nytt fönster öppnas.
- 8 Ta bort markerad tiger från den markerade medlemmen. En dialogruta visas för att bekräfta.
- 9 Filtrera medlemslistan med sökterm.
- 10 Menyrad.
- 11 Avsluta programmet.

2.5.3 Lägg till en ny medlem

När användaren trycker på knappen *Ny MedLem* öppnas ett nytt fönster där användaren ombeds fylla i den nya medlemmens namn och adress.

The screenshot shows a Windows-style dialog box titled "Lägg till ny medlem". It contains a form with the following fields and values:

- Fönamn: Ronald
- Efternamn: McDonald
- Gata: 31st Street
- Postnummer: 60523
- Ort: Oak Brook
- Land: USA

At the bottom of the dialog are two buttons: "Lägg till" (highlighted with a blue border) and "Avbryt".

Figur 19 Fönstret Lägg till ny medlem

Om formuläret inte är korrekt ifyllt när användaren trycker på *Lägg till*-knappen visas felikoner vid de felaktiga inmatningarna.

This screenshot shows the same dialog box as Figure 19, but with validation errors. The 'Gata' and 'Postnummer' fields are now empty, and each has a red exclamation mark icon to its right, indicating they are required fields. The 'Lägg till' button remains highlighted with a blue border.

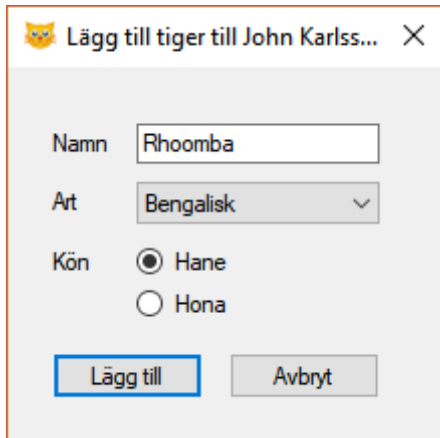
Figur 20 Felaktig inmatning av ny medlem

Om formuläret är korrekt ifyllt läggs den nya medlemmen till i databasen. Användaren kan också välja att avbryta inmatningen genom att trycka på *Avbryt*-knappen.

2.5.4 Lägg till en ny tiger

När användaren trycker på knappen *Lägg till tiger* öppnas ett nytt fönster där användaren ombeds fylla i den nya tigers information.

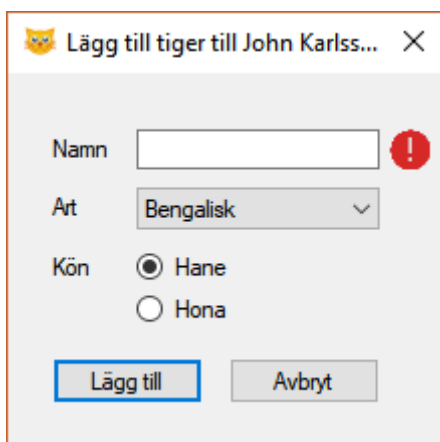
Tigern kommer läggas till till den medlem som är markerad. Detta reflekteras också i fönstrets titel.



Figur 21 Fönstret Lägg till tiger

Förutom namn väljs också tigers arttillhörighet i en lista, och dess kön väljs med radioknappar. Endast ett kön kan väljas.

Om namnet inte är ifyllt när användaren trycker på *Lägg till*-knappen visas en felikon vid namninmatningen.



Figur 22 Felaktig inmatning av ny tiger

Om namnet är korrekt ifyllt läggs den nya tigern till i databasen. Användaren kan också välja att avbryta inmatningen genom att trycka på *Avbryt*-knappen.

2.5.5 Sökfilter

Med sökfiltret kan användaren filtrera medlemslistan genom att skriva in en sökterm.

The screenshot shows a web application window titled "TRF - Tigerälskarnas Riksförbund!". The interface includes a menu bar with "Fil", "Verktyg", and "Hjälp". The main content area is divided into three sections:

- Medlemmar: 2 (Filter)**: A table with two columns, "Förnamn" and "Efternamn". The first row shows "John" and "Karlsson". The second row, which is highlighted in blue, shows "Linda" and "Svensson". Below the table is a large grey rectangular area.
- Medlem**: A section displaying the details of the selected member, "Linda Svensson". It includes the address "Lindgatan 12", "48291 Lund", and "SVERIGE". Below this, it says "Ägda tigrar: 0". At the bottom of this section is a button labeled "Radera medlem".
- Tigrar**: A section for managing tigers. It contains a large empty box for a tiger's photo, followed by labels for "Typ:" and "Kön:". Below these are two buttons: "Lägg till tiger" and "Ta bort tiger". At the bottom of this section is a button labeled "Avsluta".

At the bottom left of the interface, there is a button labeled "Ny Medlem" and a search filter section with the label "Sökfilter" and a text input field containing the text "erig".

Figur 23 Filtrering av medlemslistan med söktermen "erig" för Sverige

Medlemslistan uppdateras automatiskt varje gång texten i rutan ändras.

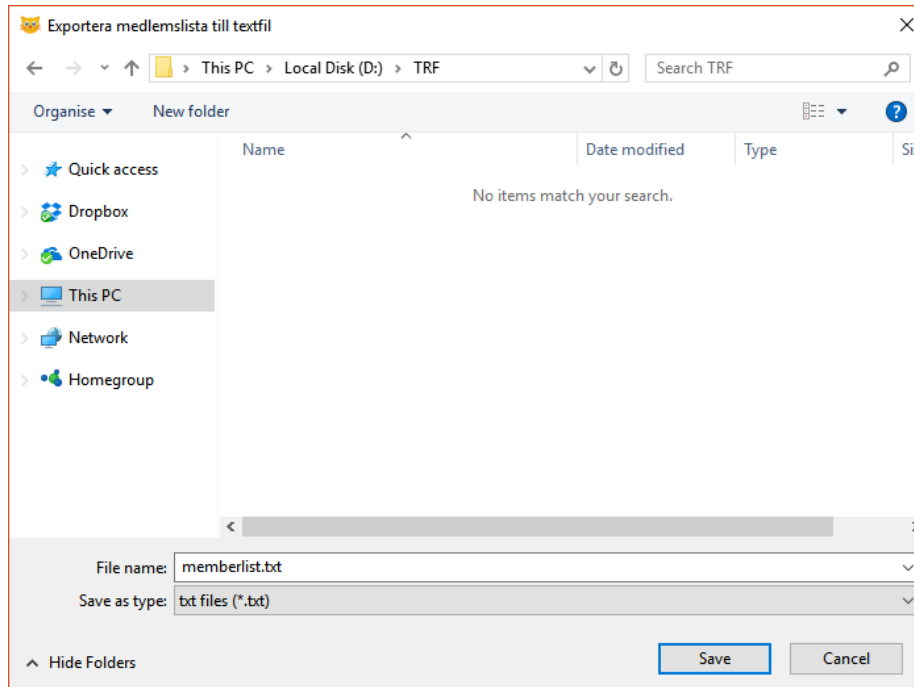
Samtliga namn och adressuppgifter används i sökningen. Dock visas enbart postnummer vid exakta sökningar. Exempelvis skulle en medlem med postnummer "59132" enbart visas om hela numret 59132 är inskriven i sökfiltret.

Tigerdata används inte i sökresultatet.

2.5.6 Meny Fil

Under *Fil* i menyraden finns *Exportera* och *Avsluta*.

Med *Exportera*-alternativet kan användaren exportera medlemsdata till en textfil. Användaren uppmanas att välja en plats där textfilen ska sparas, och namn på textfilen.



Figur 24 Sparadialog för exportering av medlemmar till textfil.

Textfilen listar samtliga medlemmarna i TRF, med adressuppgifter. Ägda tigrar listas ej.



Figur 25 Exporterad textfil med medlemmar

2.5.7 Meny Verkttyg: Ändra lösenord

Under *Verkttyg* i menyraden finns *Ändra Lösenord*, där användaren har möjlighet att välja ett nytt lösenord som används för att starta programmet.

Figur 26 Fönstret Ändra lösenord.

Användaren kan välja att visa lösenordet i klartext genom att kryssa i rutan *Visa Lösenord*. Det finns också möjlighet att generera ett slumpmässigt lösenord som skrivs in i rutan genom att trycka på knappen *Generera*.

Figur 27 Nytt slumpmässigt genererat lösenord, visat i klartext.

När användaren trycker på *OK*-knappen sparas det nya lösenordet i filen *Login.pwd*.

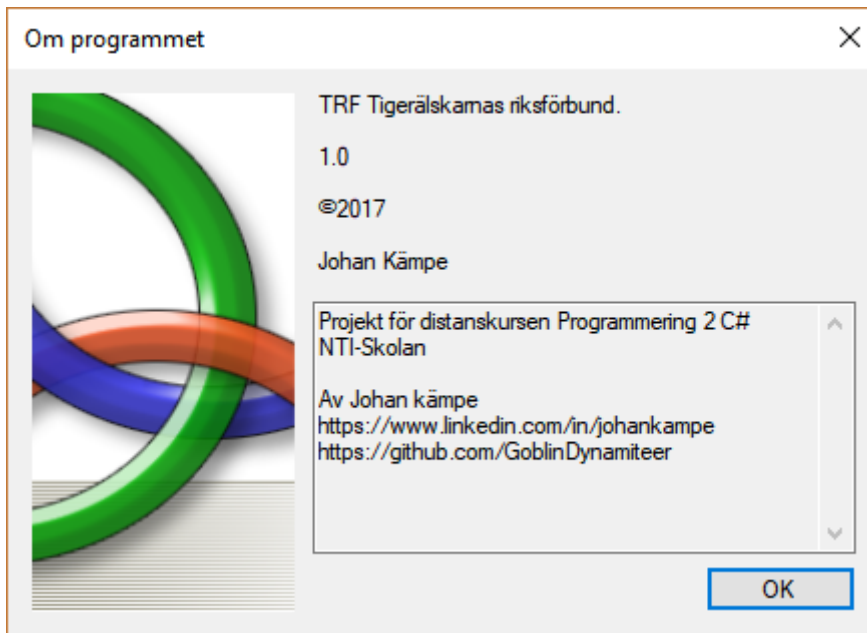
Tomma lösenordet får inte användas, ett felmeddelande visas om användaren trycker på *OK*-knappen utan att ha angivit ett lösenord.

Figur 28 Felmeddelande om tomt nytt lösenord.

2.5.8 Meny Hjälp

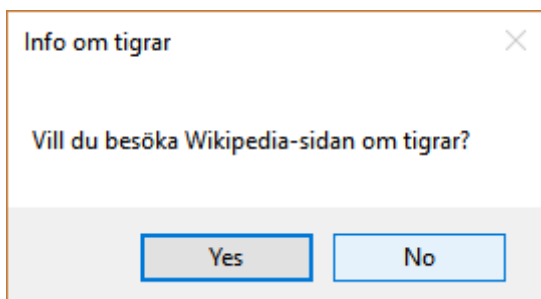
Under *Hjälp* i menyraden finns *Om programmet* och *Info om tigrar*.

Med valet *Om programmet* visas en ruta med information om programmet.



Figur 29 Om programmet

När användaren trycker på *Info om tigrar* visas en fråga om användaren vill besöka Wikipedia-sidan om tigrar. Om *Ja*-alternativet öppnas websidan <http://sv.wikipedia.org/wiki/Tiger> i en webbläsare.



Figur 30 Fråga om att gå till Wikipedia-sidan om Tigrar.

2.6 Programmets kod

Källkoden beskrivs enbart övergripande i denna rapport, i den inlämnade källkodsfilen finns ytterligare kommentarer som beskriver koden.

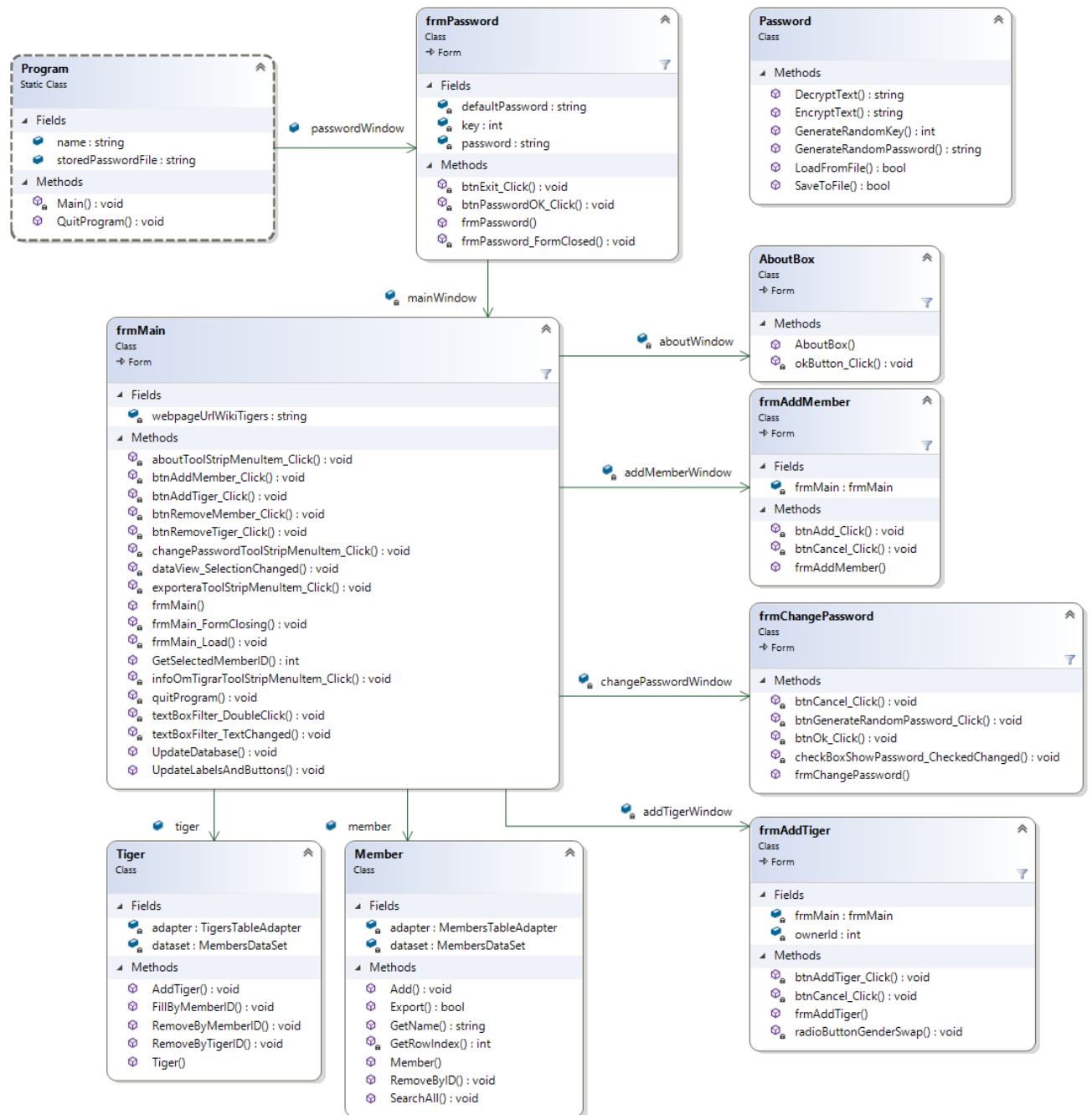
Kod som är automatiskt genererad av Visual Studio beskrivs eller kommenteras ej.

Programmet består av klasser som Visual Studio har genererat samt flera egenskapade klasser.

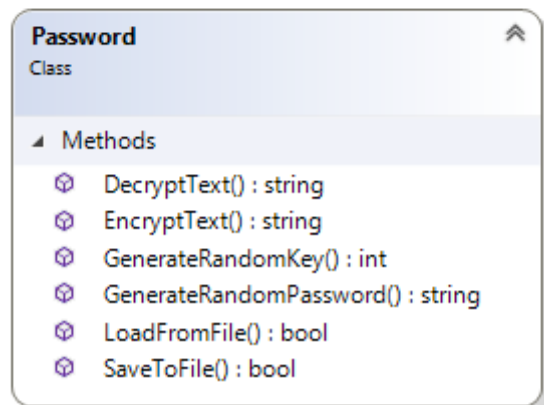
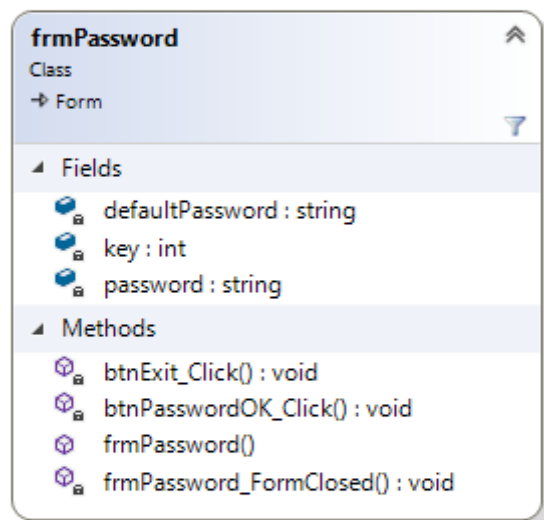
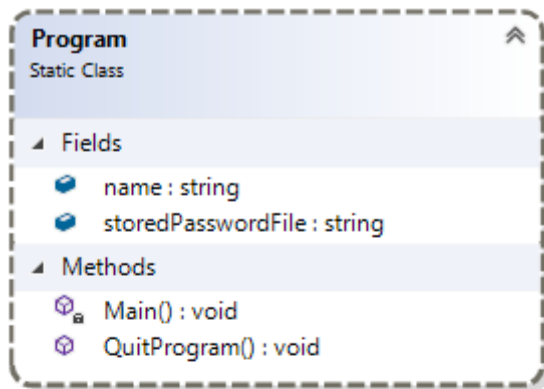
Form-klasser	
Namn	Beskrivning
<i>frmMain</i>	Form för programmets huvudfönster
<i>frmAddMember</i>	Form för "Lägg till medlem"
<i>frmAddTiger</i>	Form för "Lägg till tiger"
<i>frmPassword</i>	Form för Login-ruta
<i>frmChangePassword</i>	Form för "Ändra lösenord"
<i>AboutBox</i>	Form för "Om programmet"

Andra klasser	
Namn	Beskrivning
<i>Member</i>	Hanterar medlemmar i TRF
<i>Password</i>	Hanterar lösenord
<i>Tiger</i>	Hanterar tigrar i TRF
<i>Program</i>	Program-klassen

2.6.1 Klassdiagram



Figur 31 Klassdiagram



Figur 32 Klassdiagram för Program, frmPassword och Password

Klassdiagrammen kan också ses i filen *ClassDiagram.cd* som tillhör projektet.

3 Diskussion och slutsats

3.1 Förslag på förbättringar

Gömda texttrutor för som håller Medlem-ID och Tiger-ID

I programmet används två texttrutor i formen *frmMain* som håller ID för den aktuella markerade medlemmen eller tigern. Dessa texttrutor används i koden för att hämta detta ID. Rutorna göms med egenskapen *visible = false*.

Detta upplevs som en dålig lösning och skulle antagligen kunna lösas på ett snyggare sätt.

Sökfiltret

Sökfiltret skulle kunna utökas att inkludera även de ägda tigrarna. Det skulle också kunna förbättras med matchning av postnummer.

Postnummer

Postnummer lagras som heltal i tabellen medlemmar i databasen. Dock finns det länder som har text i sina postnummer. Detta skulle kunna ändras.

Exportering

Ingen filtrering eller sortering kan göras vid exporteringen av medlemmar till textfil. Detta skulle kunna läggas till. Också tigrarna skulle kunna inkluderas i exporten.

Sortering av medlemslistan

Extra sorteringfunktionalitet skulle kunna läggas till, exempelvis sortera medlemmar efter antal ägda tigrar.

Landlista

En lista med världens samtliga länder skulle kunna användas när en ny medlem ska läggas till. Eventuellt går detta att ordna automatiskt med .NET.

Lösenord

Lösenordet sparas tillsammans med en "krypteringsnyckel" i en textfil. Nyckeln är de antal steg som varje tecken har ökats med för att "kryptera" lösenordet så att det inte går att läsa.

Detta känns väldigt osäkert och skulle kunna förbättras. Också så använder programmet ett "standardlösenord" om textfilen inte finns tillgänglig. Också detta är ingen bra lösning.

Vid byte till nytt lösenord av användaren skulle det vara bra om det gamla lösenordet behövde skrivas in innan byte blev möjligt.

Tabellrelationer

Ingen relation mellan tabellerna *Tigers* och *Members* finns i databasen. En relation hade kunnat skapats mellan *OwnerId* och *memberId*.