|  |  |  |
| --- | --- | --- |
| Three Counties Database System | Comp4 | |
| Zachary Britton – 3974 Godalming College – 64395 | |  |

Contents

[Investigation and Analysis 7](#_Toc383612227)

[The Organization 7](#_Toc383612228)

[The Problem 7](#_Toc383612229)

[Methods of Research 8](#_Toc383612230)

[Interview 8](#_Toc383612231)

[Questionnaire 8](#_Toc383612232)

[Record Inspection 8](#_Toc383612233)

[Observation 8](#_Toc383612234)

[Interview 9](#_Toc383612235)

[Questions 9](#_Toc383612236)

[Interview Transcript 10](#_Toc383612237)

[Interview Analysis 12](#_Toc383612238)

[Interview Conclusion 13](#_Toc383612239)

[Questionnaire 14](#_Toc383612240)

[Questionnaire Analysis 16](#_Toc383612241)

[Record Inspection 20](#_Toc383612242)

[Observation 21](#_Toc383612243)

[Prospective Users 21](#_Toc383612244)

[Analysis 22](#_Toc383612245)

[Data Dictionary 24](#_Toc383612246)

[Current System 24](#_Toc383612247)

[Systems Flow 24](#_Toc383612248)

[Systems flow table 26](#_Toc383612249)

[Current data flow diagram 26](#_Toc383612250)

[Scope 27](#_Toc383612251)

[Numbered Objectives 27](#_Toc383612252)

[Potential Solutions 29](#_Toc383612253)

[Chosen Solution 29](#_Toc383612254)

[Design 30](#_Toc383612255)

[Proposed flow diagrams 30](#_Toc383612256)

[Editing Data 30](#_Toc383612257)

[Contact 32](#_Toc383612258)

[Contact Book 34](#_Toc383612259)

[Data Flow Diagram 35](#_Toc383612260)

[Data Requirements 36](#_Toc383612261)

[Data Dictionary 36](#_Toc383612262)

[Validation 37](#_Toc383612263)

[Data Storage Design 38](#_Toc383612264)

[Entity Relationship Diagram 38](#_Toc383612265)

[Normalisation 39](#_Toc383612266)

[DDL Statements 40](#_Toc383612267)

[SQL Statements 40](#_Toc383612268)

[Pseudo Code 41](#_Toc383612269)

[Quick Sort 41](#_Toc383612270)

[UI Designs 42](#_Toc383612271)

[Login and Loading Screens 42](#_Toc383612272)

[Adding, Editing and Deleting Members 42](#_Toc383612273)

[Contact Screen Designs 45](#_Toc383612274)

[Security Measures 46](#_Toc383612275)

[Password Protection and Encryption 46](#_Toc383612276)

[Testing Strategy 47](#_Toc383612277)

[Data set for member entry 47](#_Toc383612278)

[System Testing 48](#_Toc383612279)

[System’s Maintenance 52](#_Toc383612280)

[System Overview 52](#_Toc383612281)

[Forms 52](#_Toc383612282)

[Form layout 56](#_Toc383612283)

[Sub Routines 57](#_Toc383612284)

[Variable lists 65](#_Toc383612285)

[Login 65](#_Toc383612286)

[Change Password 65](#_Toc383612287)

[Main Menu 65](#_Toc383612288)

[Add Family 66](#_Toc383612289)

[Add Family Members 66](#_Toc383612290)

[Edit Family Members 67](#_Toc383612291)

[Delete Family Member 67](#_Toc383612292)

[Add group Type 68](#_Toc383612293)

[Add Group 68](#_Toc383612294)

[Contact 68](#_Toc383612295)

[Database Maintenance 69](#_Toc383612296)

[DDL 70](#_Toc383612297)

[User Manual 71](#_Toc383612298)

[Contents Page 72](#_Toc383612299)

[Overview 73](#_Toc383612300)

[Technical Specifications 74](#_Toc383612301)

[Installation Instructions 75](#_Toc383612302)

[System Usage Instructions 77](#_Toc383612303)

[First time running 77](#_Toc383612304)

[Adding Families 78](#_Toc383612305)

[Adding Family Members 79](#_Toc383612306)

[Editing Family Members 80](#_Toc383612307)

[Deleting Members 81](#_Toc383612308)

[Adding Group Types 82](#_Toc383612309)

[Adding Groups 83](#_Toc383612310)

[Adding members to groups 84](#_Toc383612311)

[Contacting church members 85](#_Toc383612312)

[Trouble Shooting 86](#_Toc383612313)

[Error Messages 86](#_Toc383612314)

[“Database Closed, or Already Set Up” 86](#_Toc383612315)

[“Database Error” 86](#_Toc383612316)

[“Login Failed” 87](#_Toc383612317)

[“Data Connection Failed” 87](#_Toc383612318)

[“Invalid data, please adjust” 88](#_Toc383612319)

[Appraisal 89](#_Toc383612320)

[Comparison Against Numbered Objectives 89](#_Toc383612321)

[User Feedback 92](#_Toc383612322)

[User Feedback Analysis 95](#_Toc383612323)

[Possible Extensions 96](#_Toc383612324)

[Improved Group System 96](#_Toc383612325)

[Online Functionality 96](#_Toc383612326)

[Improved Security 96](#_Toc383612327)

[Appendix 97](#_Toc383612328)

[Screen Shots – Testing 97](#_Toc383612329)

[1 97](#_Toc383612330)

[2 97](#_Toc383612331)

[3 98](#_Toc383612332)

[4 99](#_Toc383612333)

[5 100](#_Toc383612334)

[6 100](#_Toc383612335)

[7 101](#_Toc383612336)

[8 101](#_Toc383612337)

[9 102](#_Toc383612338)

[10 102](#_Toc383612339)

[11 103](#_Toc383612340)

[12 104](#_Toc383612341)

[13 105](#_Toc383612342)

[14 105](#_Toc383612343)

[15 106](#_Toc383612344)

[16 106](#_Toc383612345)

[17 107](#_Toc383612346)

[18 108](#_Toc383612347)

[19 109](#_Toc383612348)

[20 109](#_Toc383612349)

[Code 110](#_Toc383612350)

[frmLogin.vb 110](#_Toc383612351)

[frmChangePassword.vb 111](#_Toc383612352)

[frmMainMenu.vb 113](#_Toc383612353)

[frmAddFamily.vb 116](#_Toc383612354)

[frmAddFamilyMembers.vb 120](#_Toc383612355)

[frmDeleteMember.vb 125](#_Toc383612356)

[frmEditFamilyMembers.vb 129](#_Toc383612357)

[frmAddGroupType.vb 136](#_Toc383612358)

[frmAddGroup.vb 138](#_Toc383612359)

[frmMemberGroup.vb 143](#_Toc383612360)

[frmContact.vb 146](#_Toc383612361)

# Investigation and Analysis

## The Organization

Three Counties Church is a local Christian organization in Haslemere, Surrey. They are based in two locations, Hammer Church and Kings Road. They employ around 10 people plus many volunteers, who are involved in admin, pastoral care, leadership and youth work. The church has around 250 members and has regular weekly services and events.

**Locations:**

Hammer Church

Linchmere Road

Hammer

West Sussex

GU27 3QW

Three Counties Church

Kings Road

Haslemere

Surrey

GU27 2QA

**Contact:**

Admin Centre – 01428 653011

Email – [office@3countieschurch.org](mailto:office@3countieschurch.org)

Web – [www.3countieschurch.org](http://www.3countieschurch.org)

## The Problem

The church uses a combination of a very simple excel based spread sheet and unorganised and un used access database. Both of these have not been set up with any web or email integration. The two database conflict and some things are updated on and not the other. There is no system in place dictating what should be done and therefore lots of things get done manually

# Methods of Research

## Interview

An interview involves talking face to face with someone involved in the organisation; it is used to get solid, specific and detailed information about the problem and what the organization requires about the solution. There are some issues with interviews as people can lie, but face to face this is rare, also finding time to interview people can be an issue. However the pros massively outweigh the cons, with an interview you can get the interviewee to elaborate on the question and to give you really detailed points on certain parts of your investigation. From an interview you get qualitative data.

I am going to be interviewing the admin manager of Three Counties Church – Sarah Noble. She heads up the administration at the Church’s office at Kings Road. She is the best person to interview as from her I can get the bulk of my qualitative data, and the best information on what the program needs to do.

## Questionnaire

A questionnaire is the collection of quantitative data using lists of questions and distributing them to a large number of people, they can then be collated into graphs and tables. The problem with questionnaires is that people generally don’t want to fill them in and will just throw them out, also people have the tendency to lie or not be completely truthful saying what you think they want. Although the benefit is that you can reach a large number of people with a questionnaire and get a good representation of a sample of people.

I am going to be conducting a questionnaire within the church congregation to collect data on what the members of the church want such as their favoured mode of contact, what groups they are involved in. Also to collect important logistical data about internet access and security preferences.

## Record Inspection

This involves taking a look at and analysing the files and records that they currently use and what it is used for and the extent and volume of the data they collect. The information collected from a record inspection is very useful as it shows what exact data your program will need to hold and process, and the size that your database needs to be. This method has no risk of bias or false data.

I will collect some files from the database and copies of documents they send out to the congregation. This will show me what my program will be handling and the extent of data to be held.

# ­­­Observation

Another way to collect data is to observe the company at work and see how they currently run things; from this you can see where the current shortfalls of the system are and how they can be improved. Also from this you can see what documents are used and the extent of their usage. As you do this yourself there is no risk of bias or false data.

# Interview

## Questions

1. What is your current system for personal information storage, sorting and filing?

From this question I will be able discern what the old system was and how it was used, this will also give me an idea of the kind of data my program will be manipulating and the tasks it will have to undertake.

1. How do you use this current system to email or send letters to people, are they integrated?

This will again show me in a more complex sense what the old system was capable of doing and how/if it was integrated manually or via computer. I should also get from this question a bit of scope on.

1. What problems do you have with this system, or what could be improved?

This question will give me tips on what the problems the user wants to eradicate with the new system, and what parts they want improved over the old one.

1. What works well with this current system?

This will tell me what functionality they like and want to keep the same for the new system, and what procedures and methods they want me to use.

1. How often do you send out emails/letters/texts etc, and what do they usually consist of?

This will give me a rough idea of how much contact with the congregation is happening and in what form, allowing me to gear my system toward integration of these contact methods.

1. Approximately how many people are in the church and of those how many are part of families and groups?

This will show me how much data my system will have to hold and what kind of grouping functionality there will need to be. It will also show how much of the church is made up of families and I can use this to enhance the email integration extra emails don’t need to be sent out to one family.

1. If you were to have a new fully computerized system what would you want it to do, and ideally how would you like it too operate?

This question will give me lots of highly valuable data on what the user wants for their new system. What it should be capable of, different functions and tasks for it to carry out. It will also show me how they want it to operate, e.g. simple, modular etc

1. What extra features would you like a new system to have e.g. email integration, registering and sorting?

This will show me what extra or unique features they want the system to have, for example more church orientated function, admin validation and other tailor made features.

1. What privacy and data protection do you have in place and how well does this work, could it be improved?

This will allow me to see and improve on current data protection procedures, to ensure that my system follows the Data Protection Act

## Interview Transcript

1:00pm - Saturday 29th July

Interviewer: Zac Britton

Interviewee: Sarah Noble – Admin Manager

1. What is your current system for personal information storage, sorting and filing?

* Very simple access database with light mail integration but with lots of niggles and bugs.
* All stored on a PC and not accessible online.
* Contact book has to be produced separately and is not integrated
* No real sorting at all, unless searching by surname
* Also a few non related Excel spreadsheets which confuse everything

1. How do you use this current system to email or send letters to people, are they integrated?

* Simple integration but can’t do groups or other complex things
* Capable of doing separate people but not family or other groups

1. What problems do you have with this system, or what could be improved?

* Keeps old data without replacing new data
* Various niggles, crashing and broken functions
* Bad email integration

1. What works well with this current system?

* Not much all in need of a good clean up
* Simple to search through

1. How often do you send out emails/letters/texts etc, and what do they usually consist of?

* No text sent/functionality
* A lot of emails and some posts on website
* Text functionality would be cool

1. What hardware do you have in your current system, e.g. computers, servers etc?

* Multiple PC’s some always on
* NAS

1. Approximately how many people are in the church and of those how many are part of families and groups?

* 247 church members
* 125 families
* 17 small groups
* 3 Sunday kids groups
* 5 non-Sunday groups

1. If you were to have a new fully computerized system what would you want it to do, and ideally how would you like it too operate?

* Easy to contact people
* Get information and contact quickly
* Group emails and text
* Ability to print contact book from database in different forms
* Accessible to church members on the web
* Able to input data such as when child protection need renewing – notifies admin

1. What extra features would you like a new system to have e.g. email integration, registering and sorting?

* Record of small groups – who attends, times etc text notification near time
* Contact for different organisations
* Lots of grouping available
* Easy to use and access
* Allow them to add own data online but not change on database until admin approves, allow them to add and adjust child data

1. What privacy and data protection do you have in place and how well does this work, could it be improved?

* Password protection on admin PC
* Option to show what data people want to be made available online/in book
* Password change regularly
* Online sign in secure website

## Interview Analysis

From the interview I have gathered a lot of useful data for my project especially vital information on what the end user wants from the new system.

The first question gave me the bulk data about their current system, they use a simple yet buggy access database with very light email integration that is tricky to use. There are also a few excel spreadsheets of group data but they cause confusion as they don’t get updated when the database is. All the data is stored on one PC which is always on but is not accessible online to anyone. The contact book that they produce from it has to be produced separately as the database does not have this function built in. The system being very simple has no sorting or grouping at all, unless you search for surname to get families. This has given me a lot to improve on and solid information on what they are used to at the moment.

The second question follows on from the first and gives me more detail on their email (or other forms of contact) integration. The current system has very simple email integration that is capable of doing separate people but no groups or other complex actions. This can be greatly improved in the new system and a fully functioning and complex integration system can be created.

The third question shows me what problems there are with the current system and how I can improve on it. The current system had a habit of keeping old data even when it was updated or sometimes not even replacing it. The system has various niggles, broken functions and a tendency to crash or break. The email integration is also bad and my interviewee really wanted improvement on this.

The fourth question is the opposite of the previous and allows me to see what parts of the current system they like and potentially want to keep the same/similar. Unfortunately there wasn’t much that they like from the current system other that the simplicity of parts it. This shows that there is a lot to be improved on in the new system.

The fifth question gives me information on their methods of correspondence, currently they do not send any texts but they do email regularly so email functionality is a necessity. Every now and then they update their website but not that often, and they thought text functionality would be cool.

The sixth question is vital as it shows what the computer hardware and capabilities are for the church. They currently have around 6 mid range PC’s some of which are always on. They also have a small Network Attached Storage device.

Question seven is too give me an idea of the size of the church and therefore the rough size of the database. The church currently consists of 247 people, made up of 125 families. There are 17 small groups within the church that normally meet on Wednesdays. 3 Sunday kids groups and 5 non-Sunday groups e.g. healing hour, debt counselling

The eighth question is vital and gives me lots of information on what the end user wants in the new system. They would like a system that makes it easy to contact church members and groups, and allows them to get information fast and accurately. They would like the ability to send group emails and texts and the ability to print the contact book in various forms. The end user would also like church members to be able to access data on the web and also the ability to input data such as child protection information that allows them to get notified when it needs updating.

Question nine follows on from the previous and gives me more detail on what they want from the new system. Ideally they would like the ability to record small groups with information on who attends and with notifications. The ability to add other organisations contact details and do lots of grouping. They want it to be simple and easy to use. They would specifically like the ability to add data online but that has to be verified by an admin before it goes through, with the added ability to allow parents to add children.

Most importantly question ten tells me what privacy and data protection features they want integrated in the system. They have currently and would like to keep password protection on the PC but also on the system too. They would like the option to restrict what data certain people want available online or in the contact book. The ability to have to change password regularly and a secure online sign in would be appreciated.

## Interview Conclusion

Overall I have collected a lot of data here to use to base my system on and to take into account whilst planning and programming. Some features will have to be further considered after taking a questionnaire of the church congregation and seeing what they want. But in general I have most of the base information here that I will use for the system.

It is obvious from this that they want a simple to use yet highly functional system, with good grouping and contact facilities all integrated into a well rounded database package with added web functionality.

# Questionnaire

|  |
| --- |
| Q1. Which category below includes your age? (Circle one) |
| 17 or younger 18-20 21-29 30-39 40-49 50-59 60 or older |
| This question will show me a few things, one it will ensure I have a good sample of ages so that my data is valid across the whole church. Two I can also use it to found out whether certain things are more prominent with certain ages. |
| Q2. Do you have access to the internet? (Circle one) |
| Yes No |
| This will help me decide whether an online application is viable and if so what kind of volume of people will be using it. It will also show me what percentage could emails and web data if it was a feature, and how many would still need contact by post or text. |
| Q3. How confident are you with using a computer? (Circle one) |
| Not at all Not really Mildly Very |
| This will show me how good the church is with computers, thus showing me how simple it should be and how many should be able to use it. It will also show me if it is viable for a complex online system to be created. |
| Q4. Do your family members also attend Three Counties Church? (Circle one) |
| Yes No |
| This will allow me to see what proportion of the church is made up of families to help me see how well a family grouping system will work. It will also help me create sorting functions. |
| Q5. How many groups (Children’s, Small, Youth) do you attend? (Circle one) |
| 1 2 3 4 5 6 |
| This will also assist me with the grouping functions and show how many different groups should be allowed on the database. |
| Q6. What are your preferred methods of receiving information from the church? (Circle all applicable) |
| Email Text Web Post |
| This will show me what the default method of contact should be within the system and whether all are viable. |
| Q7. Would receiving notification of events by text or email benefit you? (Circle one) |
| Yes No |
| This will show me whether I should add an automatic notification system into the programme, and whether it would be well used/appreciated. |
| Q8. If you are in a family do you prefer receiving separate emails each or to one main email address (Circle one) |
| Separate One |
| This will show whether a family email function is viable or whether most will prefer to receive an email each. |
| Q9. Do you own a Smartphone? (Circle one) |
| Yes No |
| This will give me an idea of what majority if the congregation own smart/web enabled phones that would be able to fill in database updates online. If the number is high I will have to ensure compatibility on mobile browsers. |
| Q10. How often do you use the login section of the church website? (Circle one) |
| Everyday Couple of times a week Once a week Once a month Hardly Never |
| This will show me whether people currently use the login section of the church website, which is currently only for downloading sermons. This will allow me to see whether the web service will be well used or whether it will have to be advertised more in church. |

# Questionnaire Analysis

Q1. Age

I tried to collect a good range of ages from the church congregation however as you can see from the graph the church has a lot of young people and a lot of middle aged to older people but not many in between (probably due to the lack of a local university).

This gives me a good idea of who will be using the system, the majority of whom will most probably be adults in the 40-49 age group, or potentially the higher end of the youth. Obviously the technical ability will most likely decrease as the age gets older, so I will have to take this into account.

Q2. Internet connection

This question was very definitive 100% of those questioned have an internet connection, which means that a web based application is very much viable and accessible for all. It also means that the majority should be able to receive emails and web updates.

Q3. Computer Literacy

The majority of those asked said that they were very confident with basic computer use and thus would easily be able to access the church website and the contact system and information system on there. Only 20% claimed they were only mildly confident but will still be able to go online and access the website and app as long as it was not too complicated.

Q4. Family Attendance

Over half of the sample questioned has family members at the church, there are a lot of families at the church and especially lots of children. This means that having good family sorting facilities in the system is a must, not just by surname as there are some families with the same surname. It also means that a family email grouping facilities will also be useful so that kid’s emails will be sent to parents etc.

Q5. Group Attendance

All of those questioned attend at least one group and most attended two. This means that a grouping functionality based on groups attended will work well, especially if integrated into an email or text facility. This could send out emails or texts a day before the group or maybe week before a big even, but only be sent out automatically to those who want it.

Q6. Preferred contact methods

The majority’s favoured contact method is email, with text coming a close second. This means that the bulk of correspondence will be sent out through email so the system needs to be optimized for this. The system should probably also be able to send out texts and web updates as well as these are both also well liked, the ability to create address stickers would also be good but is not desperately necessary as it is the least preferred with only 6% selecting this.

Q7. Text/Email notifications

The vast majority of the congregation said they would benefit from an automated text or email notifications, with only 5% saying no. Therefore putting an integrated automatic notification feature in the system would be well used and a good use of programming time. Also this mirrored onto the website in a more general form will be a good feature.

Q8. Family emails

The majority of those in families at Three Counties would rather have separate emails, especially the teenagers who get information about their own youth events. A few wanted them just to one email specifically those with kids. This means that both options should be available and people should be able to choose which they want online.

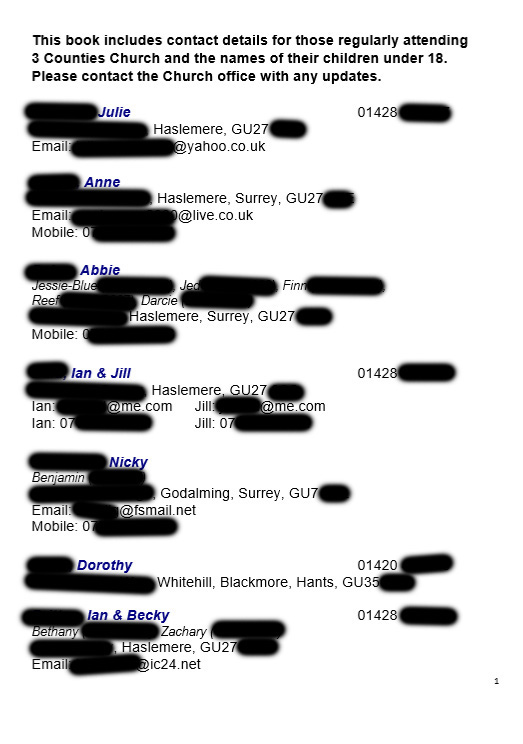
Q9. Smartphone ownership

The number of Smartphone users in the church is actually quite well balanced with a 60-40 sway to ownership. This means that some people could access the church website on the move. Therefore the web app needs to be mobile browser friendly so people can use the contact details to contact people on the move.

Q10. Web login

The majority of people do not use the church login service as currently it is only for accessing previous recorded sermons, however people do know it’s there. Hopefully the number of users will increase when greater functionality and its subsequent advertisement.

# Record Inspection

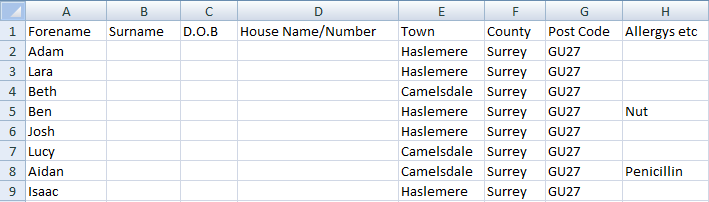
1. Church Contact Book

This is a copy of page one of the church contact book made in Microsoft word. The important details have been blurred for privacy reasons.

This shows what kind of data is kept and what data is available to the church. The data here Name, address, email is the bare bones data that subject to permission is displayed in the contact book. Some numbers are unavailable as people don’t want them published.

Obviously more data is kept such as ages and genders but this is not published and I was not allowed data from the actual database.

1. Excel spreadsheet Kidzone



This is one of many extra separate spreadsheets that are kept on the Admin computer (Again with sensitive data removed). This is what should be integrated into the new system without having to have separate data kept for groups like this. The data contained should be inputted into the new system and the groups attended attached as attributes so they can be sorted or printed if need be.

This highlights one of the many setbacks of the current system, the fact that multiple databases collide and do not work on sync. If one was updated the others would not and this needs to be addressed in the new system.

# Observation

I observed the system at the church office for a few hours to see how things were run and what kind of operations they have to do on a daily basis. The majority of procedures carried out are contact and organization based such as noting creating and advertising events, and contacting the church members. New members are added now and then and edited with children, marriages and location changes. During the observation a church member requested a new contact book which was subsequently found (as a premade document) and printed. From my observation I noticed that the overall way of running things was rather complicated and took a long time to do something simple, there were no search functions or sorting.

The church website was updated once with a new sermon and a new event advert but no database integration was in place. Emails where sent without finding email addresses from the database but finding them on the outlook contact list, this seemed a bit long winded and awkward. Other tasks like editing and copying the previous Sundays sermon where carried out and various call to and from other people.

# Prospective Users

Head of Admin – Sarah Noble – Most admin related jobs, database work, contact

Admin – Monica Spyznyck – Smaller admin tasks, database work, contact

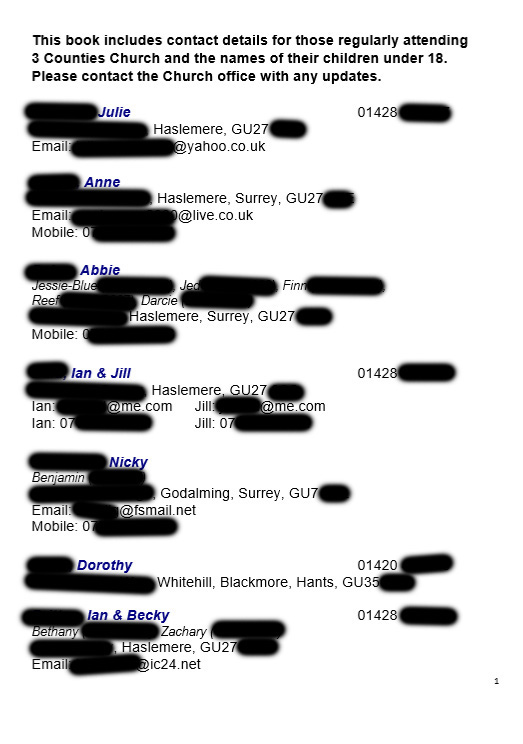
Church Leader – Dick Durrant – Oversees operation and does the odd bit of admin

# Analysis

Contact Book – Document Specification

Document Specification Sheet

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Volumetrics** | | | | | | | | | | | | | | |
| Document description | | | | System | | | | Document | | | | Name | | Sheet |
| Church Contact Book | | | | Member contact system | | | | 1 | | | | Zachary Britton | | 1 of 16 |
| Stationery ref. | | | | Size | | | | Number of parts | | | | Method of preparation | | |
| N/A | | | | A5 Book | | | | 250 | | | | Typed | | |
| Filing sequence | | | | | | Medium | | | | | | Prepared by | | |
| Newest Only | | | | | | File | | | | | | Admin | | |
| Frequency of preparation | | | | | | Retention period | | | | | | Location of file | | |
| Once every 2 years | | | | | | Till out of date | | | | | | Admin office | | |
| Volume | Minimum | | | | Maximum | | Av/Abs | | | Growth rate/fluctuations | | | | |
| 250 | | | | 500 | | 350 | | | (find out) | | | | |
| Users/receipts | | Purpose | | | | | | | | | | | Frequency of use | |
| Church Members  Admin | | To be able to contact those in church  To be able to contact church members | | | | | | | | | | | Daily  Daily | |
| **Data Dictionary** | | | | | | | | | | | | | | |
| Ref | Name | | Data Type | | | | | | Length | | Occurrence | | Source of data | |
| 1 | Intro Text | | X | | | | | |  | | 1 | | Admin | |
| 2 | Name | | X | | | | | | 10 -35 | | 7 | | Database | |
| 3 | Number | | 9(5)9(6) | | | | | | 11 | | 4 | | Database | |
| 4 | House Name/Number | | X | | | | | | 1-20 | | 7 | | Database | |
| 5 | Street Name | | X | | | | | | 10-20 | | 7 | | Database | |
| 6 | Town | | X | | | | | | 5-20 | | 7 | | Database | |
| 7 | County | | X | | | | | | 6-11 | | 4 | | Database | |
| 8 | Post Code | | A(2)9(1-2)X(1)9(1)A(2) | | | | | | 7-8 | | 7 | | Database | |
| 9 | Email Text | | X | | | | | | 6 | | 5 | | Admin | |
| 10 | Email | | X | | | | | | 10-40 | | 5 | | Database | |
| 11 | Mobile Text | | X | | | | | | 7 | | 3 | | Admin | |
| 12 | Mobile No. | | 9(5)9(6) | | | | | | 11 | | 3 | | Database | |
| 13 | Children | | X | | | | | | 15-100 | | 3 | | Admin | |



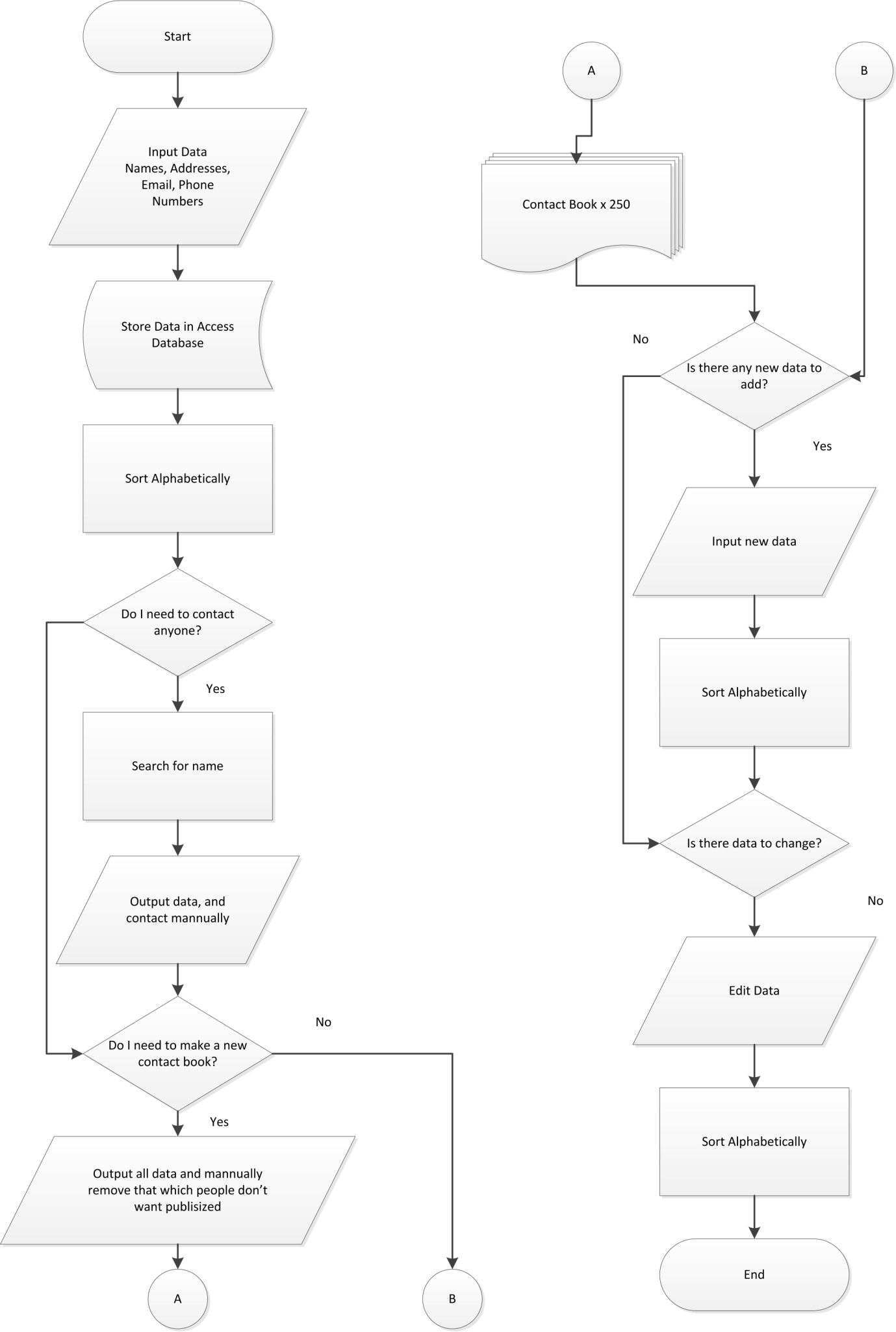
# Data Dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Field Purpose | Field Type | Field Size | Example Data | Validation |
| First name | Stores member’s forename | String | 2-15 | Zachary | Text and –‘ |
| Surname | Stores member’s surname | String | 2-15 | Britton | Text |
| D.O.B | Stores member’s date of birth | Date | 10 | 12/06/1996 | DD/MM/YYYY (date) |
| House Name/ Number | Stores member’s house name or number | String/Integer | 1-30 | Fountain Cottage | Text or numbers |
| Road Name | Stores member’s road name | String | 4-30 | Marleycombe Road | Text and Spaces |
| Town | Stores member’s town of residence | String | 4-30 | Haslemere | Text and –‘ |
| County | Stores member’s county of residence | String | 4-30 | Surrey | Text and –‘ |
| Post Code | Stores member’s post code | String | 6-8 | GU27 1NW | Post code (from gov) |
| Email | Stores member’s email address | String | 5-40 | Zac.britton@example.com | Email |
| House Phone | Stores member’s landline phone no. | Integer | 11 | 01428666666 | Numbers |
| Mobile Phone | Stores member’s mobile phone no. | Integer | 11 | 07983647281 | 07 and 9 numbers |

# Current System

## Systems Flow

This flowchart was created with the data from my investigation, it is quite brief and limited but shows how the system works and the order in which processes are carried out on the database. Only the database and details related processes are shown as all other processes are manual and don’t always happen consistently or in the same way, so they are not easily charted



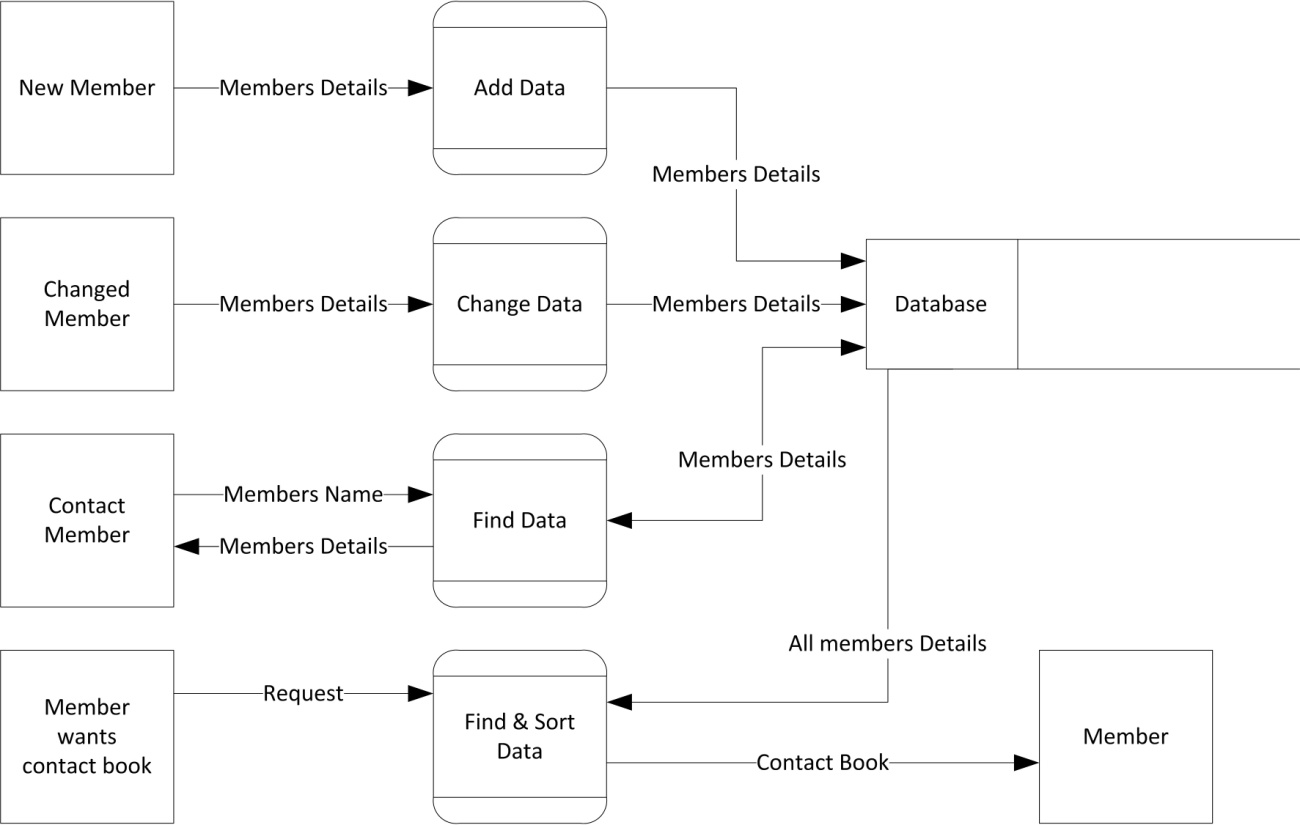
## Systems flow table

This table has been created from the systems flowchart and shows the processes and inputs in a simpler way.

|  |  |
| --- | --- |
| **Input** | **Process** |
| Data (Name, Addresses, Email, Phone) | Sort Data Alphabetically |
|  | Search For Name |
| Data in simple access | Data for contact |
|  | Contact Book x250 |
| **Storage** | **Output** |

## Current data flow diagram

This data flow diagram again was created with the data from my investigation; it is also quite brief and limited and shows how data flows around the system. Only the database and details related processes are shown as all other processes are manual and don’t always happen consistently or in the same way.



# Scope

The system will have to be capable of certain tasks as requested by the end-user, to aid the design of the system it shall be created based on numbered objectives set out below.

## Numbered Objectives

1. The system will be able to display the details stored in the database quickly and simply
   1. The data should be displayed in under 5 seconds
   2. To display the data a maximum of 6 button presses will be used
   3. The display of the data will be in a simple table
2. The system will be based around a solid and secure server based database
   1. The software will be password protected
   2. The database will be stored remotely and be accessible from multiple computers
3. The system will have adequate security and privacy features and abide with the data protection act
   1. The software will request a password reset every month
   2. The software will allow editing and removal of data as to abide with the data protection act
   3. The software will automatically log out after 10 minutes of inactivity
4. The system will allow the admin to send out automated and custom emails and texts
   1. It will allow emails and texts to be sent on set dates and times automatically
   2. The system will be capable of efficiently sending emails and texts in under a minute
   3. The software will allow single or multiple recipients from custom or set groups
5. The system will allow quick and functional sorting by means of surname age and other attributes
   1. The system will allow the searching of and within groups
   2. The system will allow simple and complex searching by tick box attributes or by name, age or other physical detail
   3. The search will give results in under 10 seconds
6. The system will allow flexible grouping with custom groups with automated contact and notification systems
   1. The system will allow over 30 different groups
   2. Groups selectable and editable in data input window
   3. Automated emailing for group activities and events can be set
   4. The system will allow for the input of meeting and event time and dates for groups
7. The system will allow church members to access and edit their own details online and see others too
   1. The web app should be accessible via the church website login section
   2. It will show only data that the members want to be visible online as to abide with privacy laws
   3. The editing process should be simple and only take a couple of clicks to accomplish
   4. The web app should allow members to add data of their family members
8. The system will allow members to select whether they want their data published or not
   1. They should be able to select whether they want it available in the contact book and/or online
   2. This should be accomplished by a simple tick box operation either online or by the administrator
9. The system will allow the production of the contact book in various forms and types
   1. The book will be creatable in .pdf or .docx formats
   2. It should be created by the system in under a minute
10. The system will have functional automatic validation and manual validation for member submitted data
    1. The system will use RegEX validation to ensure all data entered is in the right format and can be processed correctly by the system
    2. The system will validate instantly, and upon saving data

## Potential Solutions

Python coded program based on a raspberry pi for small size and convenience within church. Simple and functional interface using an SQLite database, stored on the Pi with a small web based application, to allow for member data storage. Use of simple algorithms to sort and search data within program, exports files to Raspbian compatible formats. Simple python web app to allow members to enter own data online, with function within main program to validate data.

Vb.Net coded program for a standard windows PC as this would integrate well with the current system and equipment within the church. Again with a simple yet functional complex interface using MySQL Server from Oracle for storing and saving data. The server could be based locally or remotely allowing for easy and secure access. Using VB will allow me to carry out more complex functions and use a higher resolution cleaner interface. Paired with a Web Express app it will allow easy integration with the program and allow online functionality available to church members.

Vb.Net coded program with a MSSQL server and a Web Express app online. This again is similar to the solution above but using Microsoft’s SQL server instead, this will mean different syntax and mildly different functionality but other than this not much change.

Android application based on a tablet with a server based database and android app. This option has the added convenience of being portable so admin work can not only be done within the office but also at home or around other locations on the go. However this would involve learning another language and bring in many more security issues.

## Chosen Solution

I have chosen to do a VB.Net program with a MySQL server and a Web Express online application. There are many reasons for choosing this one, it has many advantages over coding on the Raspberry Pi and Android most notably speed, the Pi is lacking in processing power and so are most tablets. Also the benefit of coding a Desktop based application is that the admin within the church is all done in one place so it does not need to be possible and working long hours is much more preferable on a PC than a tablet or Raspbian based program. As it is computer based and more powerful than the two other languages it gives me greater flexibility in my coding and programming process. There are also more security issues with Android as the member app would have to have very strict security and connection encoding so that peoples information is not hacked or stolen.

I have chosen MySQL over MSSQL as it is easier to set up and access and will allow greater flexibility with server location and security. It will also make it easier to troubleshoot and fix any problems that come up during the programming process.

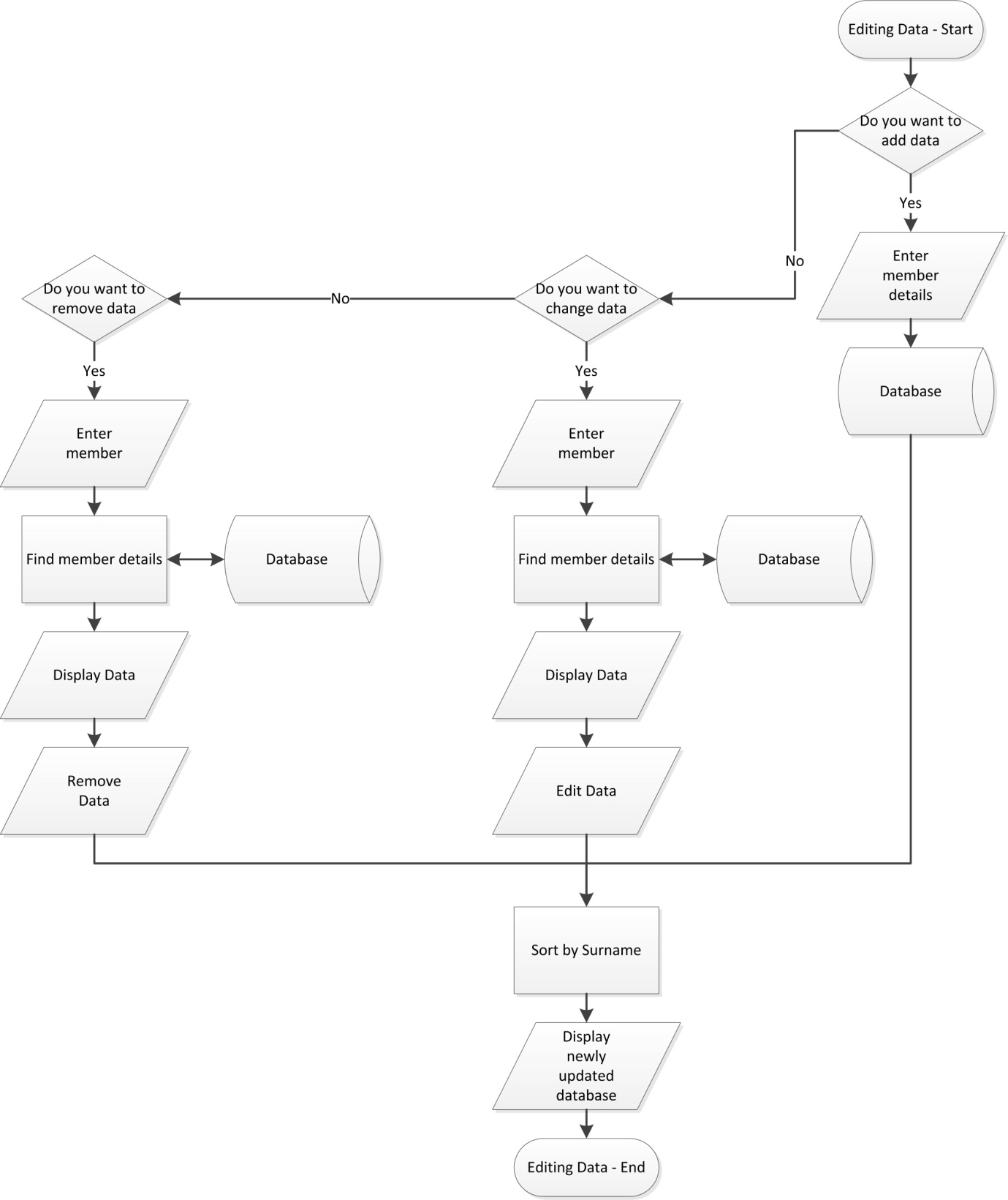
As for using VB.Net over other powerful languages such as C++ and Java, I already have solid experience coding with VB which will make the process of program creation much faster and will allow me to create a simple to use yet functional program. Also the use of Web Developer will allow me to create a secure and simple web app with my knowledge of VB putting me in good stead as Web Developer uses very similar syntax.

# Design

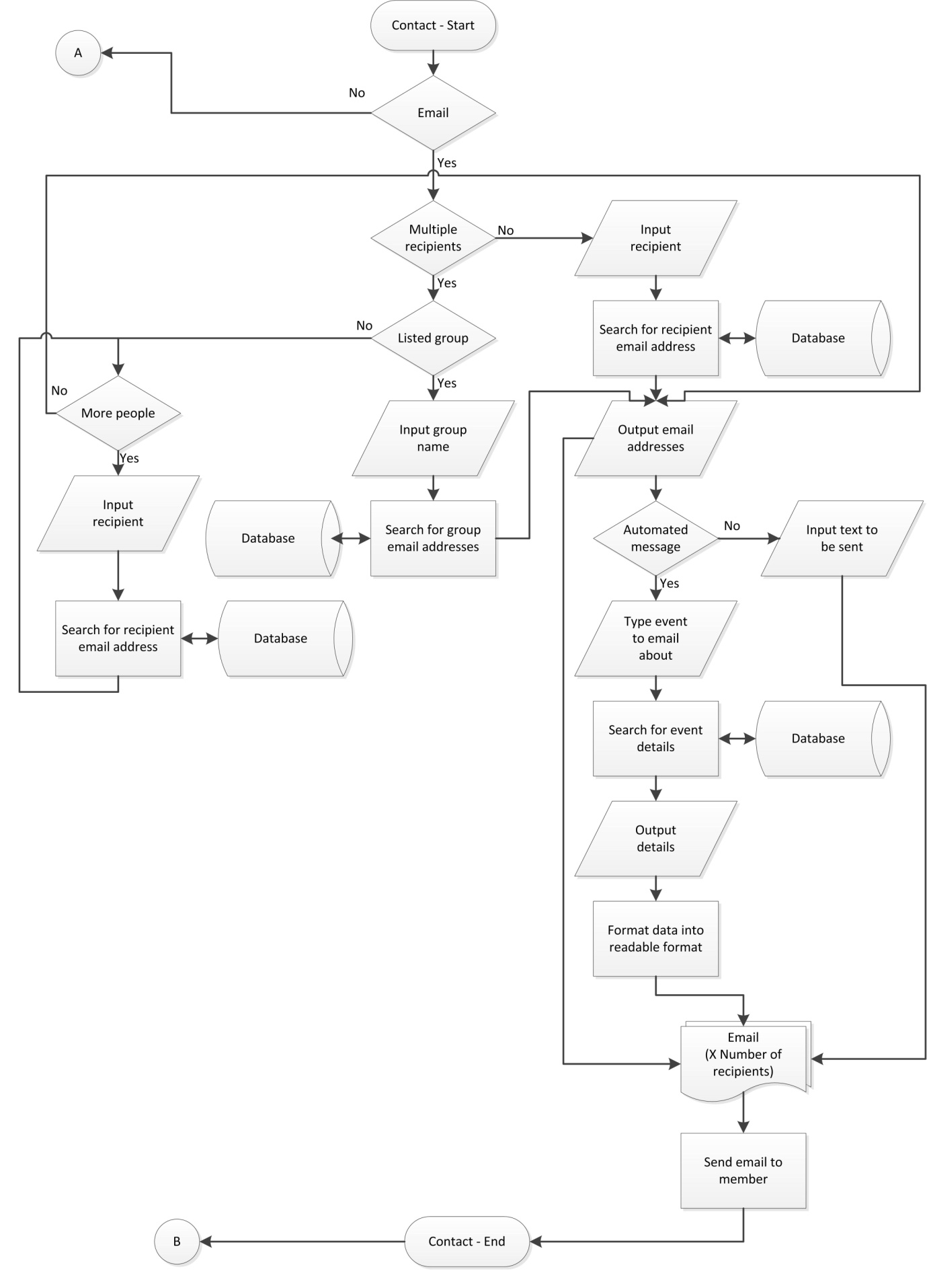
# Proposed flow diagrams

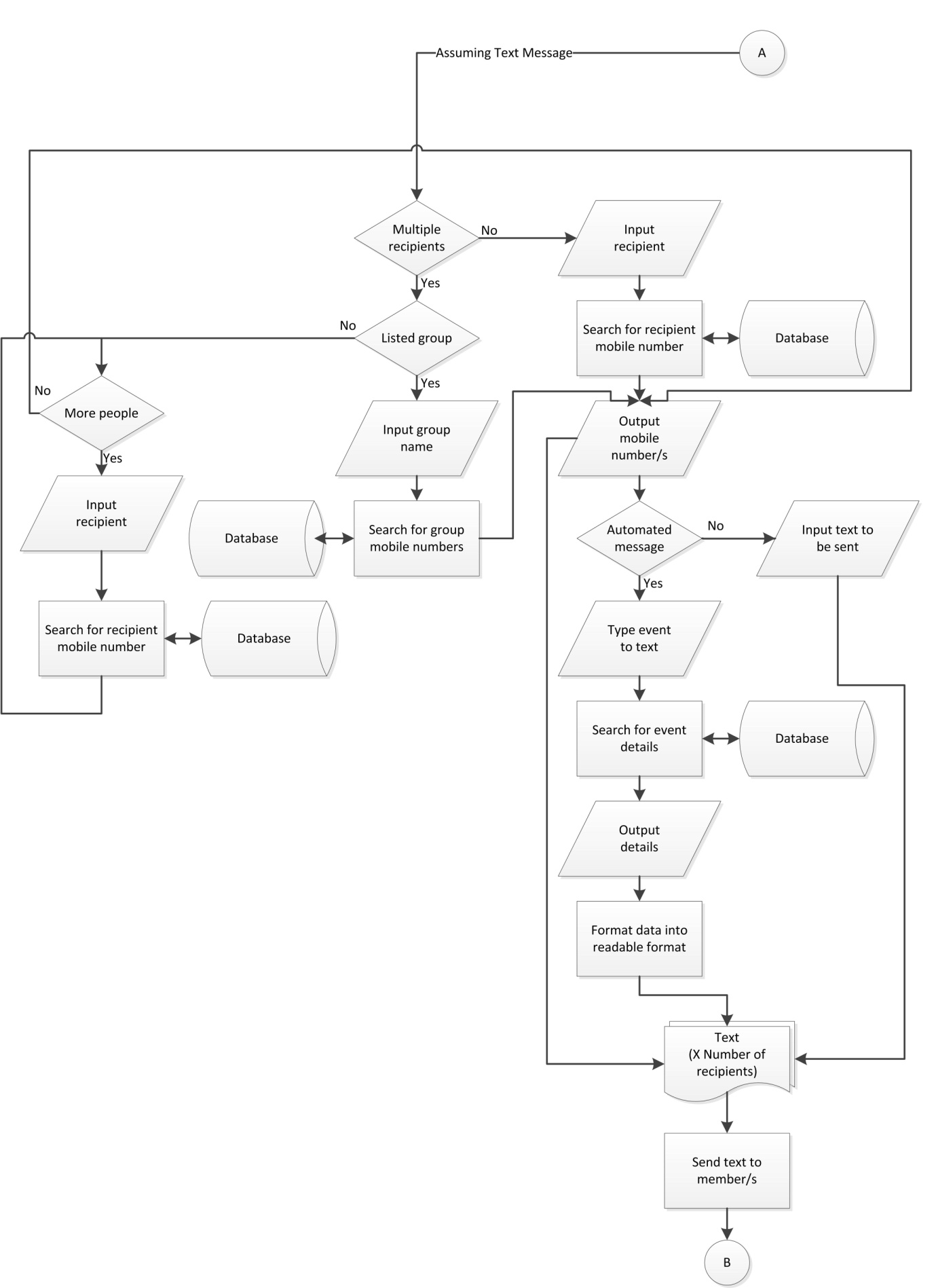
## Editing Data

This systems flowchart shows how the adding, editing and removing part of the proposed system will work, it shows how the option is chosen and how that process is carried out. This will help when coding the program.

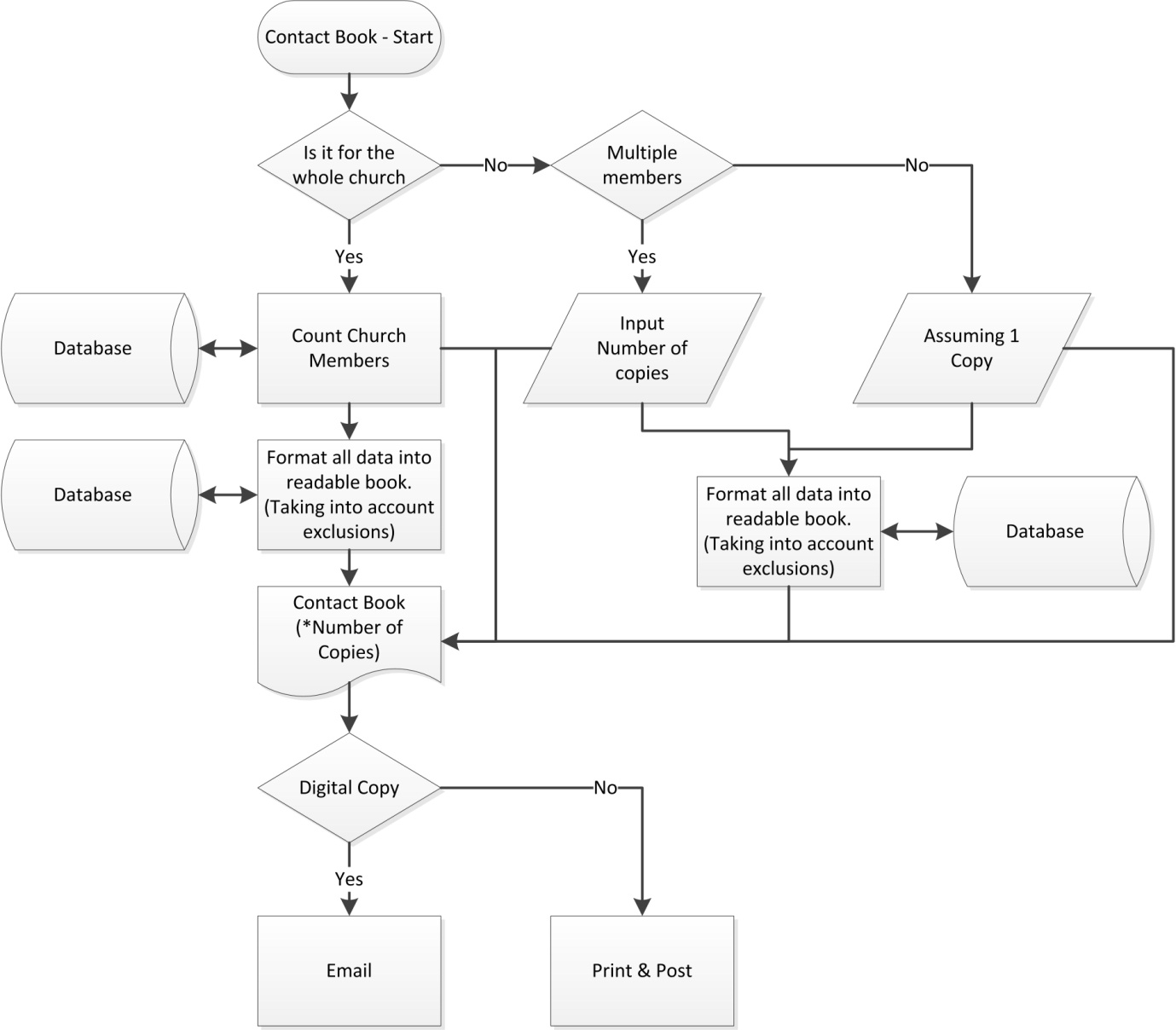
****

Contact

This slightly larger systems flow chart is for the contact (automated and manual) part of the proposed system. It follows the process of deciding on text or email and whether said message is to single or multiple recipients and whether it is an automated group or event email. This flowchart will be vital in the development and creation of the program. 

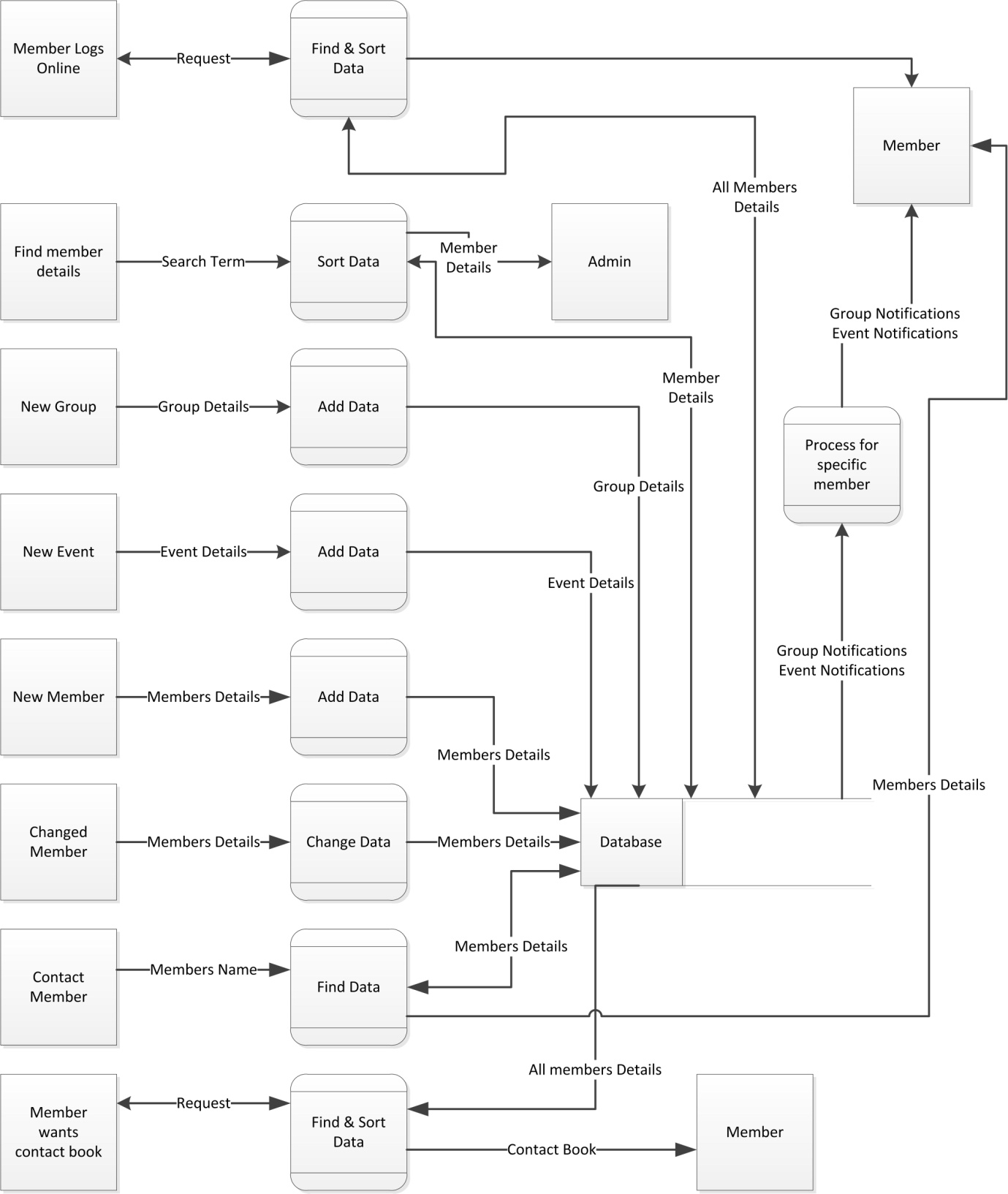


## Contact Book

This flowchart shows how the automated creation of the contact book will work, taking data from the database and turning it into a well formatted book. It shows the process for creating single and multiple copies and creating a full church batch. 

# Data Flow Diagram

This shows the flow of data around the proposed system all working around the central database structure. It shows the various processes that need certain data and why.



# Data Requirements

## Data Dictionary

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field Name | Field Purpose | Field Type | Field Size | Example Data | Validation |
| First name | Stores member’s forename | String | 2-15 | Zachary | Text and –‘ |
| Surname | Stores member’s surname | String | 2-15 | Britton | Text |
| D.O.B | Stores member’s date of birth | Date | 10 | 12/06/1996 | DD/MM/YYYY (date) |
| House Name/ Number | Stores member’s house name or number | String/Integer | 1-30 | Fountain Cottage | Text or numbers |
| Road Name | Stores member’s road name | String | 4-30 | Marleycombe Road | Text and Spaces |
| Town | Stores member’s town of residence | String | 4-30 | Haslemere | Text and –‘ |
| County | Stores member’s county of residence | String | 4-30 | Surrey | Text and –‘ |
| Post Code | Stores member’s post code | String | 6-8 | GU27 1NW | Post code (from gov) |
| Email | Stores member’s email address | String | 5-40 | Zac.britton@example.com | Email |
| House Phone | Stores member’s landline phone no. | Integer | 11 | 01428666666 | Numbers |
| Mobile Phone | Stores member’s mobile phone no. | Integer | 11 | 07983647281 | 07 and 9 numbers |
| Group Name | Stores name of group | String | 3-50 | Healing Hour | Text and spaces – ‘ |
| Group Day | Stores the day of the week the group meets on | Date | 6-9 | Tuesday | Day |
| Group Time | Stores the time that the group meets | Time | 4 | 1500 | Time |

# Validation

Using Regular Expressions (RegEX)

|  |  |
| --- | --- |
| Input | RegEx |
| Name | ^[-‘ A-z]+$ |
| Surname | ^[-‘ A-z]+$ |
| D.O.B. | ^(((0[1-9]|[12]\d|3[01])\/(0[13578]|1[02])\/((19|[2-9]\d)\d{2}))|((0[1-9]|[12]\d|30)\/(0[13456789]|1[012])\/((19|[2-9]\d)\d{2}))|((0[1-9]|1\d|2[0-8])\/02\/((19|[2-9]\d)\d{2}))|(29\/02\/((1[6-9]|[2-9]\d)(0[48]|[2468][048]|[13579][26])|((16|[2468][048]|[3579][26])00))))$ |
| Mobile Number | ^0[0-9]{10}$ |
| Landline Number | ^0[0-9]{10}$ |
| House Name/Number | ^[- A-z\d]+$ |
| Road Name | ^[- A-z]+$ |
| City | ^[- A-z]+$ |
| County | ^[- A-z]+$ |
| Post Code | ^(GIR ?0AA|[A-PR-UWYZ]([0-9]{1,2}|([A-HK-Y][0-9]([0-9ABEHMNPRV-Y])?)|[0-9][A-HJKPS-UW]) ?[0-9][ABD-HJLNP-UW-Z]{2})$ |
| Email | ^[A-z0-9.\_%+-]+@[A-z0-9.-]+\.[A-z]{2,4}$ |

# Data Storage Design

## Entity Relationship Diagram

Family

Contact Data

Member

Church

This diagram shows the relationship between the parts and types of the data in the database. The relationship between most of the parts of the data is one to many which is ok for putting in the database, however a few are many to many which means an extra table will be needed to assign groups to group types and groups to members otherwise it would not be possible to normalise the data.

Groups

## Normalisation

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **FirNam** |  | **FirNam** |  | **ID** |  | **ID** |
| **SurNam** |  | **SurNam** |  | **FirNam** |  | **FirNam** |
| **DOB** |  | **DOB** |  | **SurNam** |  | **SurNam** |
| **House Num/Nam** |  |  |  | **DOB** |  | **DOB** |
| **Town** |  | **House Num/Nam** |  |  |  | **FamilyID** |
| **County** |  | **Town** |  | **FamilyID** |  | **PrefContID** |
| **PostCode** |  | **County** |  | **House Num/Nam** |  |  |
| **ContactData** |  | **PostCode** |  | **Town** |  | **FamilyID** |
| **GrName** |  |  |  | **County** |  | **House Num/Nam** |
| **MeetingDay** |  | **ContactData** |  | **PostCode** |  | **Town** |
| **StartTime** |  |  |  |  |  | **County** |
| **EndTime** |  | **GrName** |  | **ContactID** |  | **PostCode** |
| **Location** |  | **MeetingDay** |  | **Data** |  |  |
|  |  | **StartTime** |  |  |  | **ContactID** |
|  |  | **EndTime** |  | **GroupID** |  | **MemberID** |
|  |  | **Location** |  | **GrName** |  | **ContactTypeID** |
|  |  |  |  | **MeetingDay** |  | **Data** |
|  |  |  |  | **StartTime** |  |  |
|  |  |  |  | **EndTime** |  | **ContactTypeID** |
|  |  |  |  | **Location** |  |  |
|  |  |  |  |  |  | **GroupTypeID** |
| Key |  |  |  |  |  |  |
| Primary Key |  |  |  |  |  | **GroupID** |
| Foreign Key |  |  |  |  |  | **GrName** |
|  |  |  |  |  |  | **GroupTypeID** |
|  |  |  |  |  |  | **MeetingDay** |
|  |  |  |  |  |  | **StartTime** |
|  |  |  |  |  |  | **EndTime** |
|  |  |  |  |  |  | **Location** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  | **GroupID** |
|  |  |  |  |  |  | **MemberID** |

## DDL Statements

CREATE TABLE IF NOT EXISTS Member (MemberID Int(4) PRIMARY KEY AUTO\_INCREMENT, FirNam Varchar(40), SurNam Varchar(40), DOB Varchar(10), FamilyID Varchar(40), PrefContID Int(4));

CREATE TABLE IF NOT EXISTS Family (FamilyID Varchar(40) PRIMARY KEY, HouseNam Varchar(40), Street Varchar(40), Town Varchar(40), County Varchar(40), PostCode Varchar(12));

CREATE TABLE IF NOT EXISTS Contact (ContactID Int(5) PRIMARY KEY AUTO\_INCREMENT, MemberID Int(4), ContactTypeID Varchar(40), Data Varchar(40));

CREATE TABLE IF NOT EXISTS ContactType (ContactTypeID Varchar(40) PRIMARY KEY);

CREATE TABLE IF NOT EXISTS GroupType (GroupTypeID Varchar(40) PRIMARY KEY);

CREATE TABLE IF NOT EXISTS Groups (GroupID Int(4) PRIMARY KEY AUTO\_INCREMENT, GroupName Varchar(40), GroupTypeID Varchar(40), MeetingDay Varchar(40), StartTime Varchar(5), EndTime Varchar(5), Location Varchar(40));

CREATE TABLE IF NOT EXISTS GroupMember (GroupID Int(4), MemberID Int(4))

## SQL Statements

INSERT INTO Family (FamilyID, HouseNam, Street, Town, County, PostCode) VALUES (‘Surname’, ‘House Name’, ‘Street’, ‘Town’, ‘County’, ‘Post Code’);"

UPDATE Member SET FirNam = 'First Name', DOB = 'Date of Birth' WHERE MemberID = 'Member ID' AND FamilyID = 'Surname’

DELETE FROM Member WHERE MemberID = 'Member ID' AND FamilyID = 'Surname'

# Pseudo Code

## Quick Sort

For my system I am going to have to use a few sorts for the database, including member details, groups and events. The best method for this will be a Quicksort as it is more efficient than other methods such as Bubble Sort.

Procedure Quicksort (List, First, Last)

PivotValue = List(First)

LeftPointer = First + 1

RightPointer = Last

While LeftPointer <= RightPointer

While List(LeftPointer) < PivotValue AND LeftPointer <= RightPointer

LeftPointer = LeftPointer + 1

EndWhile

While List(RightPointer) > PivotValue AND LeftPointer <= RightPointer

RightPointer = RightPointer – 1

EndWhile

If Left < Right

Then Swap (List(LeftPointer), List(RightPointer))

EndIf

EndWhile

Pivot = RightPointer

Swap (List(First), List(Pivot))

Quicksort (List, First, Pivot – 1)

Quicksort (List, Pivot +1, Last

EndProcedure

# UI Designs

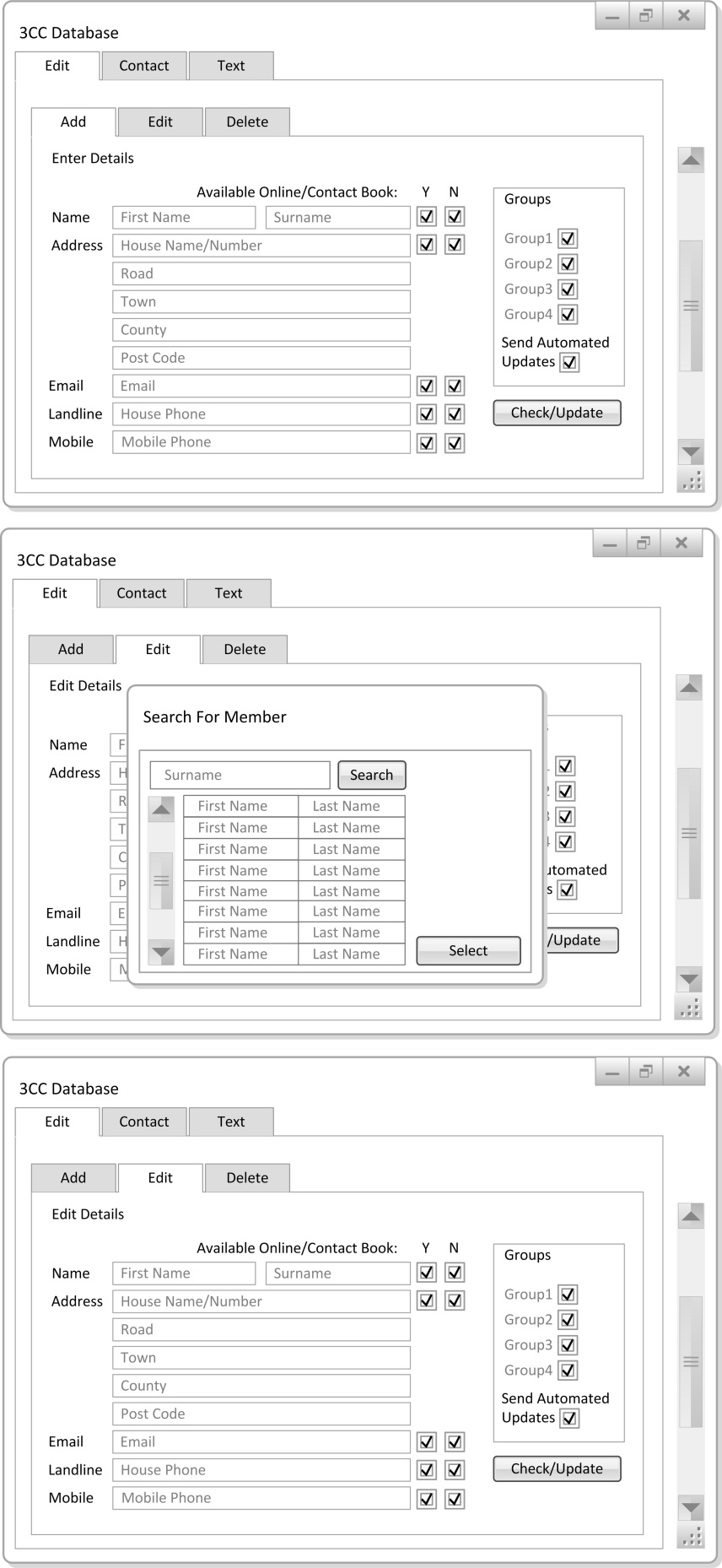
## Login and Loading Screens

This is the design for the login and loading screen of the system, it includes a simple and easy to use username and password entry box. The login button is large and obvious allowing the system to follow its objective of being simple to use. Hopefully any new user of the system will be comfortable with this layout.

The loading screen shows to the user that the database is being connected too and loaded into the system. Without this screen, for example having it load in the background without notifying the user could lead to confusion over whether the system was working or not. So it is important to have this, it is simple and very much easy to understand for the user.

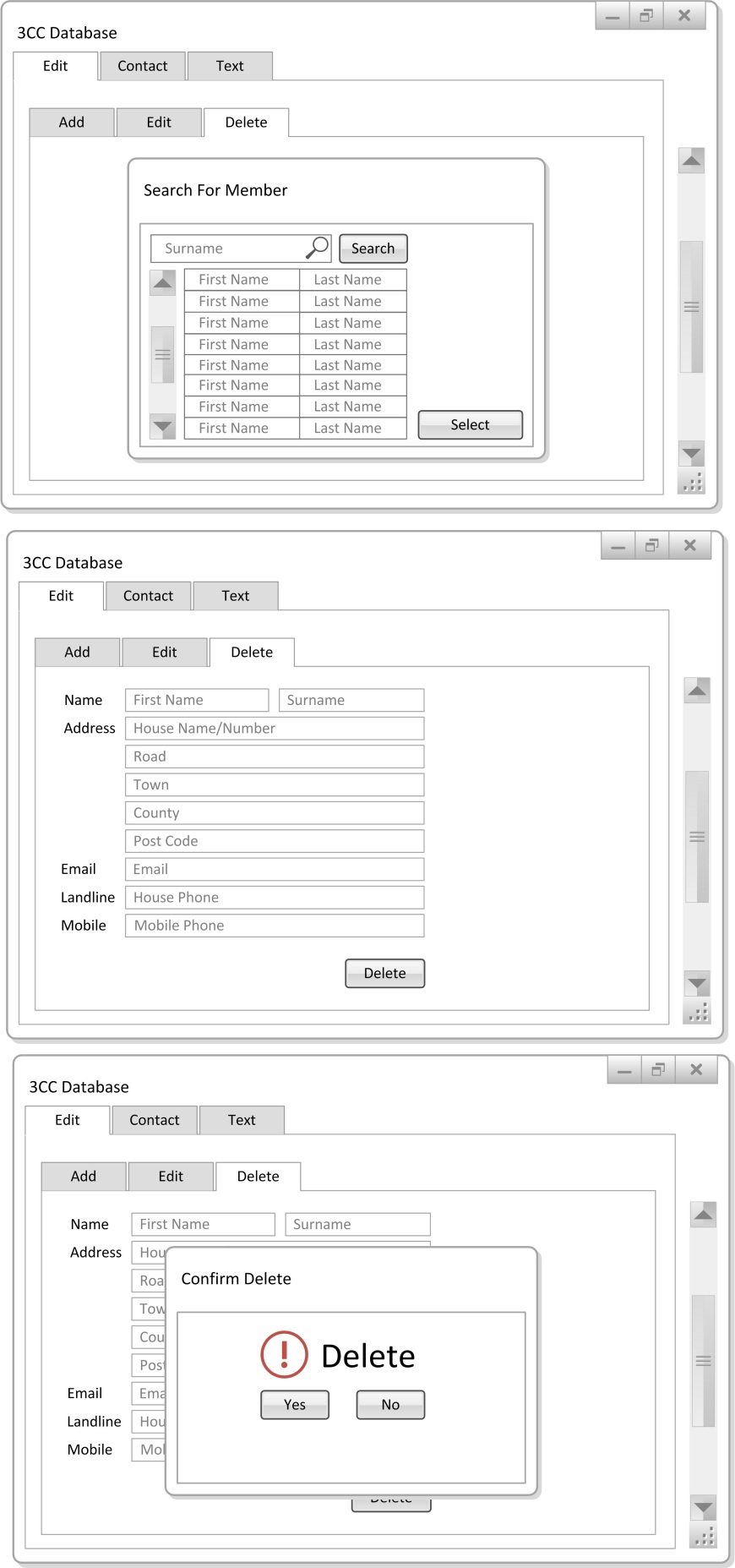
## Adding, Editing and Deleting Members

The process of adding new members will present the user with this screen, this allows them to enter all data and validate it. The usage of tabs at the top of the page means that navigation should be simple and easy to do. Using separate boxes for the entry of each data type allows them to all be validated so that they work successfully within the system. The user can select the groups that the member attends using a simple tick box to the right with the option to send automated updates via email there too. It also allows the admin to select whether said user wants their data available online or in the contact book, to abide with the Data Protection act and the end users requests. This is again by simple tick boxes which will automatically initially be on yes to save the user time.

With a simple click onto the edit tab the user will be greeted with a search box that will allow them to search via surname for the chosen member. The user can then highlight and select the chosen user for editing. The use of a small pop-up box allows for simple and smooth operation of the system as having a full screen search would be a waste of screen space.

Having selected the chosen member a screen very much similar to the add member screen will be presented to them. Except the data will already be in ready for the admin to change and then again validate and save.

If data is in the incorrect form it will be highlighted and when correctly changed the text will return to its normal form and the user will be able to save the data.

Again the use of tabs and tick boxes allows easy use and conforms to objective number one.

The delete tab show the same search pop-up as for edit allowing the user to quickly select which member they want to delete for whatever reason.

Obviously to ensure that the wrong member is not deleted the select button does not delete the data but brings up all the data kept on said member to allow the administration to ensure they have selected the correct member.

Then as a second layer of assurance the pressing of the delete button will bring up a pop-up box similar to that of the search box validating the delete a final time ensuring that meticulously entered data is not wrongly deleted.

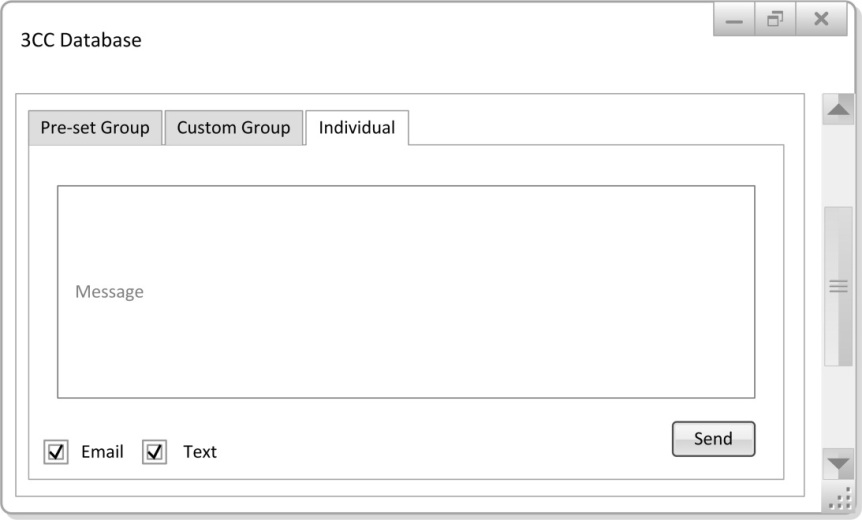
## 

## Contact Screen Designs

These screens show how members of the church will be contacted through the system. There will be three methods of contact, Pre-Set and Custom groups and Individuals. This allows the user to use the pre-set groups created and assigned when adding members to send bulk emails, but also allows the freedom to send out emails to one or more people in a custom group.

For the pre-set group a simple search pop-up again similar to that of the add function is used. This allows the admin to search through all the groups created and allocated when adding members and use their details stored in the database to contact them.

The custom group uses a similar design but with tick-boxes beside names so that multiple searches can be made and multiple members chosen. The selected members are then listed below the search to ensure the right members have been selected.

For a single member the search pop-up used is almost identical to that of the add function allowing simple and easy searching for members. The use of such a similar search means that the end users should be able to pick up the navigation and functionality of the system quickly.

All the search functions bring the user to the same contact form, which consists of a very simple text box for the message, a tick-box system for selecting between email and text and a large and self-explanatory send button.

The simple and easy to use nature of the contact system will allow the user to pick it up quickly and conforms too many of the numbered objectives.

# Security Measures

## Password Protection and Encryption

Only those with the knowledge of the system password will be able to connect to the database and access any data, to ensure the security of this password it will request changing every thirty days. Also if left inactive for fifteen minutes or more the system will log out to ensure that someone does not tamper with or steal data from the system. No parts of the system will be accessible without logging in and the login details will only be known by the three administrators.

Hopefully this will allow the system to abide with the data protection act and objective three in the scope. Security is very important for this system as there is a lot of private data held within it. The database can be encrypted with an AES encryption, meaning that only program capable of reading and changing the data held within will be the churches database system, containing the key.

# Testing Strategy

## Data set for member entry

Typical

Family: Carter, 20, London Rd, Chiddingfold, Berkshire, GU27 1NN

Family Member: Rachel, [r.carter@gmail.com](mailto:r.carter@gmail.com), [rachelcarter@deloitte.com](mailto:rachelcarter@deloitte.com), 01428675984, 01420976538, 07653849573, 26/08/1957

Group Type: Life Group

Group: Saturday Mens, Tue, 11:00, 13:00, Hammer Downstairs

Erroneous

Family: C4rter, %\*&\*&, L0ndon Rd, Ch1dd1ngfold, B3rkshire, GU176 12M

Family Member: Rache;, [r~carter@gmail.coom](mailto:r~carter@gmail.coom), [rach3lc?carter@deloitte.com](mailto:rach3lc?carter@deloitte.com), 014286234965, 1420976538, 077653849573, 30/02/1957

Group Type: ;ife Gr0up

Group: Saturd4y Mens, Non-Select, 111:00, 133:00, H4mmer D00wnstairs

Extreme

Family: CarterKnattFischerPriceBrittonFoxRodgers, 1000000000000000000000000000000000000000, PortsmouthBrightonHaslemereLiphook Road, Haslemereeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee, Berkshireeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee, GU1 1NW

Family Member: Raaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaachel, [rachelcarterthisismyemailbrah@gmail.com](mailto:rachelcarterthisismyemailbrah@gmail.com), [rachelcarterthisisntmyemailyo@gmail.com](mailto:rachelcarterthisisntmyemailyo@gmail.com), 00000000000, 01111111111, 07777777777, 07888888888, 29/02/2008

Group Type: This Group Is A Prayer Machine Of A Group

Group: Prayer Prayer Prayer Prayer Prayer Group, Fri, 12:59, 13:00, Hammer Church Downstairs In The LivingRm

# System Testing

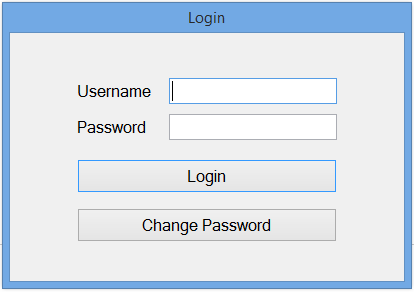
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test | | Expected Outcome | Actual Outcome | Screen Shot (See Appendix) |
| Typical  Username: Admin  Password: Password | | Log in Successful | Successful login after entering the correct user details from the Database | 1A, 1B |
| Erroneous  Username: User  Password: Pass | | Log in Unsuccessful | Unsuccessful login, with “Login Failed” message appearing. This is because the entered login details did not match those in the database | 2A, 2B |
| Data Set | | Expected Outcome | Actual Outcome | Screen Shot (See Appendix) |
| Typical | Family | | | |
| Carter  20  London Rd  Chiddingfold  Berkshire  GU27 1NN | | Successful Entry  Message “Added to database” | Successful data entry, with notification “Added to database”  SQL statement correctly entered the text into the family table | 3A, 3B, 3C |
| Typical | Family Member | | | |
| Rachel  [r.carter@gmail.com](mailto:r.carter@gmail.com)  [rachelcarter@deloitte.com](mailto:rachelcarter@deloitte.com)  01428675984  01420976538  07653849573  26/08/1957 | | Successful Entry  Message “Added to database” | Successful data entry, with notification “Family added to database”  SQL statement correctly entered the text into the member table. Shown with PHP MyAdmin screen shot. | 4A, 4B, 4C, 4D |
| Typical | Group Type | | | |
| Life Group | | Successful Entry  Message “Added to database” | Successful data entry, with notification “Added to database”. SQL statement correctly entered the type into the group type table. | 5A, 5B, 5C |
| Typical | Group | | | |
| Saturday Mens  Tue  11:00  13:00  Hammer Downstairs | | Successful Entry  Message “Added to database” | Successful data entry, with notification “Added to database”. SQL statement correctly entered the type into the group table. | 6A, 6B, 6C |
| Erroneous | Family | | | |
| C4rter  %\*&\*&  L0ndon Rd  Ch1dd1ngfold  B3rkshire  GU176 12M | | Un successful entry  “Data Invalid Please Adjust” | Unsuccessful data entry with notification “Invalid data please adjust”. The SQL statement was not even run as the RegEx ensured that the invalid data was not allowed through. | 7A, 7B |
| Erroneous | Family Member | | | |
| Rache;  [r~carter@gmail.coom](mailto:r~carter@gmail.coom)  [rach3lc?carter@deloitte.com](mailto:rach3lc?carter@deloitte.com)  014286234965  1420976538  077653849573  30/02/1957 | | Un successful entry  “Data Invalid Please Adjust” | Unsuccessful data entry with notification “Invalid data please adjust”. None of the SQL was run as the RegEx ensured that the invalid data was not allowed through. Even the date which was in the right format just not an actual date. | 8A, 8B |
| Erroneous | Group Type | | | |
| ;ife Gr0up | | Un successful entry  “Data Invalid Please Adjust” | Unsuccessful data entry with notification “Invalid data please adjust”. The SQL statement was not even run as the RegEx ensured that the invalid data was not allowed through. | 9A, 9B |
| Erroneous | Group | | | |
| Saturd4y Mens  Non-Select  111:00  133:00  H4mmer D00wnstairs | | Un successful entry  “Data Invalid Please Adjust” | Unsuccessful data entry with notification “Invalid data please adjust”. The SQL statement was not even run as the RegEx ensured that the invalid data was not allowed through. | 10A, 10B |
| Extreme | Family | | | |
| CarterKnattFischerPriceBrittonFoxRodgers  1000000000000000000000000000000000000000  PortsmouthBrightonHaslemereLiphook Road  Haslemereeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee  Berkshireeeeeeeeeeeeeeeeeeeeeeeeeeeeeeee  GU1 1NW | | Successful Entry  Message “Added to database” | Successful data entry, with notification “Family Added to database”  SQL statement correctly entered the text into the family table, with no length or format errors and fully validated with RegEx | 11A, 11B |
| Extreme | Family Member | | | |
| Raaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaachel  [rachelcarterthisismyemailbrah@gmail.com](mailto:rachelcarterthisismyemailbrah@gmail.com)  [rachelcarterthisisntmyemailyo@gmail.com](mailto:rachelcarterthisisntmyemailyo@gmail.com)  00000000000  01111111111  07777777777  29/02/2008 | | Successful Entry  Message “Added to database” | Successful data entry, with notification “Member Added to database”  SQL statement correctly entered the text into the family table, with no length or format errors and fully validated with RegEx | 12A, 12B, 12C, 12D |
| Extreme | Group Type | | | |
| This Group Is A Prayer Machine Of A Group | | Successful Entry  Message “Added to database” | Partially successful data entry, passed RegEx and was inserted into database via SQL however the text was over 40 characters so was in fact erroneous data. | 13A, 13B |
| Extreme | Group | | | |
| Prayer Prayer Prayer Prayer Prayer Group  Fri  12:59  13:00  Hammer Church Downstairs In The LivingRm | | Successful Entry  Message “Added to database” | Successful data entry, with notification “Added to database”  SQL statement correctly entered the text into the group table, with no length or format errors and fully validated with RegEx | 14A, 14B |
| Contact Form – Test Text | | Successful Text | Texted myself through the program successfully using the Clickatell SMTP to SMS service, the text does not display what is sent as it is a trial text before credit is purchased. | 15A, 15B |
| Contact Form – Test Email | | Successful Email | Emailed myself through the program successfully using the Gmail SMTP server | 16A, 16B |
| Edit | | Successful Edit | Successfully edited and updated the member ‘Rachel’ in the database. | 17A, 17B, 17C, 17D |
| Delete | | Successful Delete | Successfully Deleted the member ‘Raaaaaaaaaaaaaaachel’ from the extreme test from the database. | 18A, 18B, 18C, 18D |
| Adding Members to Groups | | Successful Add | Successfully added member ‘Rachel’ to the group Saturday Mens | 19A, 19B |
| Changing Password | | Successful Password Change | Changed the password from Password to NewPassword | 20A, 20B, 20C |

# 

# System’s Maintenance

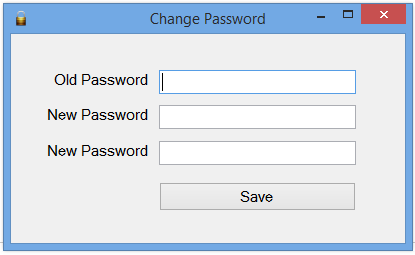
# System Overview

## Forms

Login Form – frmLogin.vb

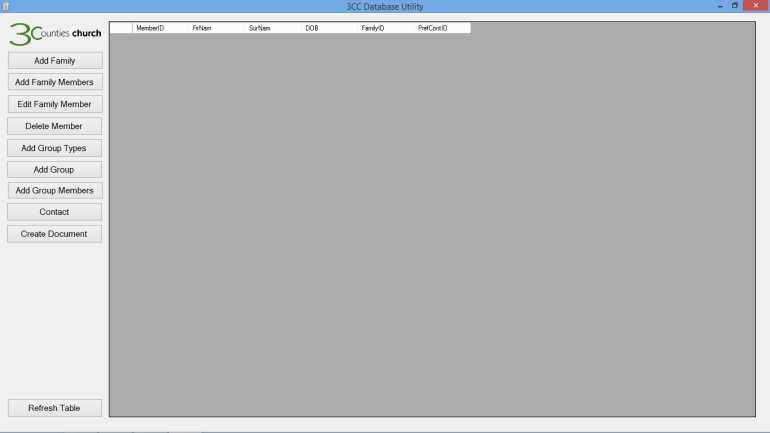
This form takes the username and password data from the database and checks it against the user entered. It allows them to login using the enter key as well as the login button and displays an asterisk as a password character. It also includes a button which takes them to the change password form.

Change Password – frmChangePassword.vb

This form allows the user to change the password taking and updating the data in the database, it requires entry of the password twice to ensure correct re-entry and again uses the asterisk password character.

Main Menu – frmMainMenu.vb

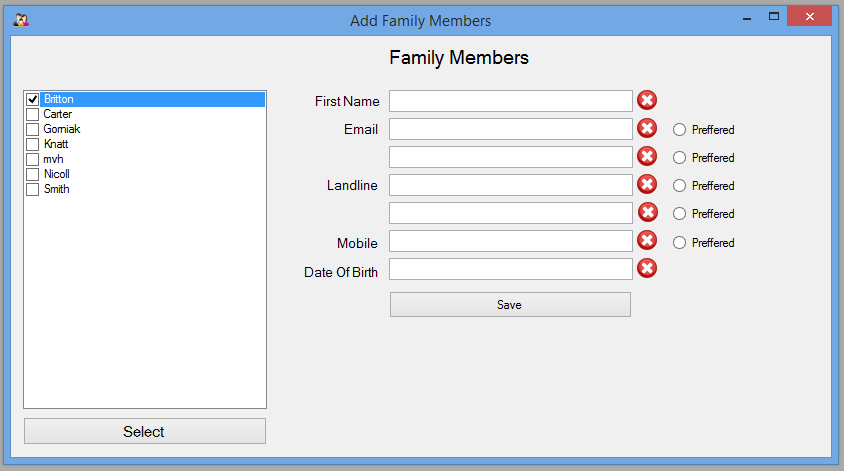
This form is the main navigational page for the program, it allows the user to go to all the other forms in the program to do the different tasks. It also displays all the data in the database in a datagrid view for the user to see.

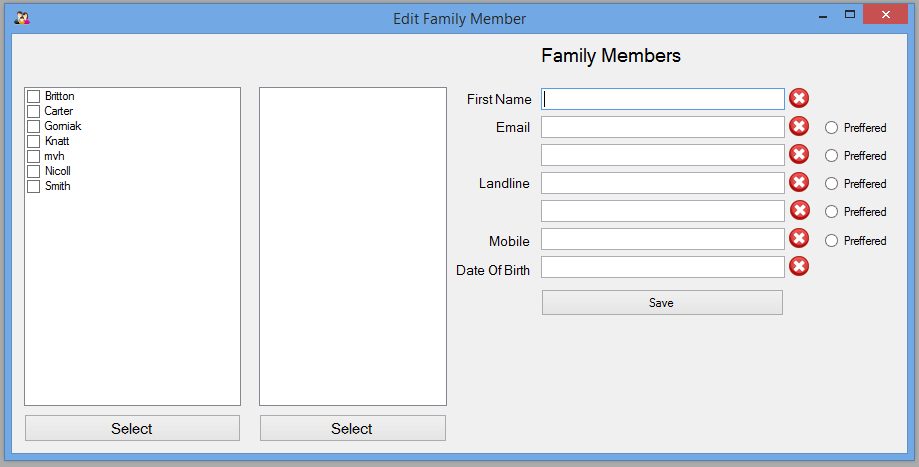
The only processing this form handles is the creation of the contact book which does not need its own form to be done. This uses the Microsoft Office format and imports the dll into the project allowing the program to take all the data from the database and sort it into a simple and easy to read booklet.

Add Family – frmAddFamily.vb

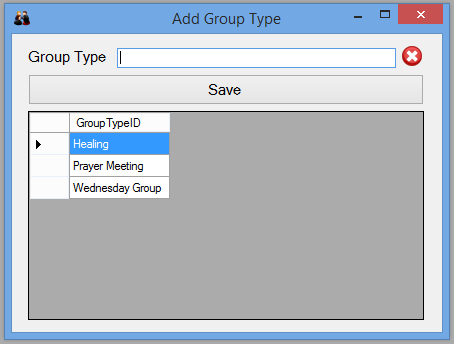
This form allows the user to add family’s to the system, it assumes that a family will share one household and therefore address. So in this form the family surname is entered and used as the primary key and their address is entered. All data entered is validated using regex and cannot be saved unless it is valid, the actual SQL entry also uses a try catch validation to ensure the database is open. The form also contains a data grid view which shows the details of all family’s entered into the database already to avoid double enteries.

Add Family Members – frmAddFamilyMembers.vb

This form allows the user/admin to assign family members to the families that they have already created and saved. It has fully RegEx validated inputs for first name, date of birth and multiple contact information entry. It uses the surname/primary key from the family table to link multiple family members to their family. The chosen family is selected from a list box that is created from the family table. Again all the SQL commands use a try catch to ensure that the server is open and the data is valid.

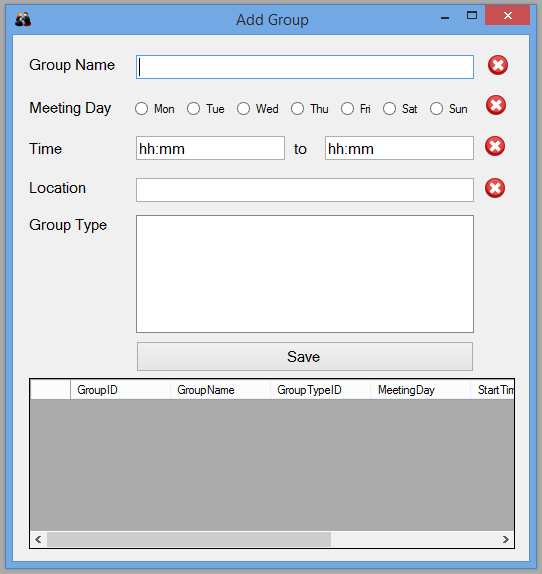
Edit Family Members – frmEditFamilyMembers.vb

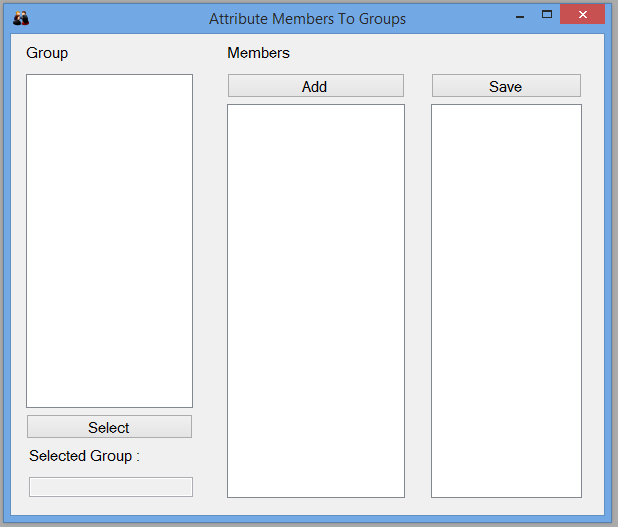
This form does a similar action to the add family member form however it allows the user to edit already entered members instead using SQL update statements to save the data to the database. This form again uses ticked list boxes to select the family and then the member and is fully validated.

Add Group Type – frmAddGroupType.vb

This form allows the user to enter types of groups that go on within the church for example prayer or life groups. It uses a single RegEx validated text box group type entry and saves it as the primary key in its own SQL table. The form also has a data grid to display the group types that have already been entered to avoid double entries.

Add Group – frmAddGroup.vb

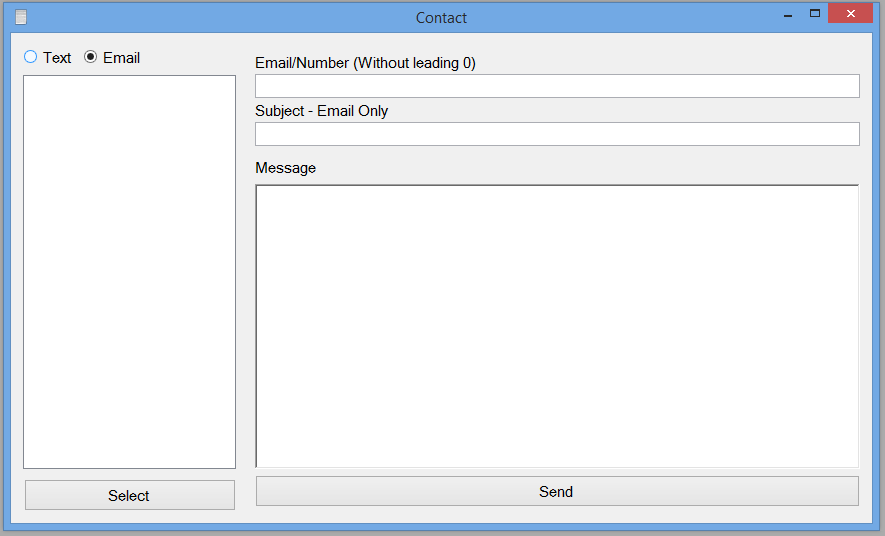
This form is for entering groups that go on within the church, it has a simple interface with fully RegEx validated text box entry for the majority of the data. The day the group is on is selected by simple radio buttons labelled with all the weekdays. The selection of group type to associate the group with is done again by a simple ticked list box. All the data is entered via an SQL statement that is contained within a try catch to ensure data is saved properly. At the bottom of the form is a data grid so that all entered groups can be seen and so that no duplicate entries are added.

Add Members to Group – frmMemberGroup.vb

This form is for attributing members to the groups saved within the database using the group ID and member ID. This form is a very simple set up with two checked list boxes to select the group and then the members with an add button then showing them in a standard list box to show which members have been entered and a save button to confirm the members that have been entered.

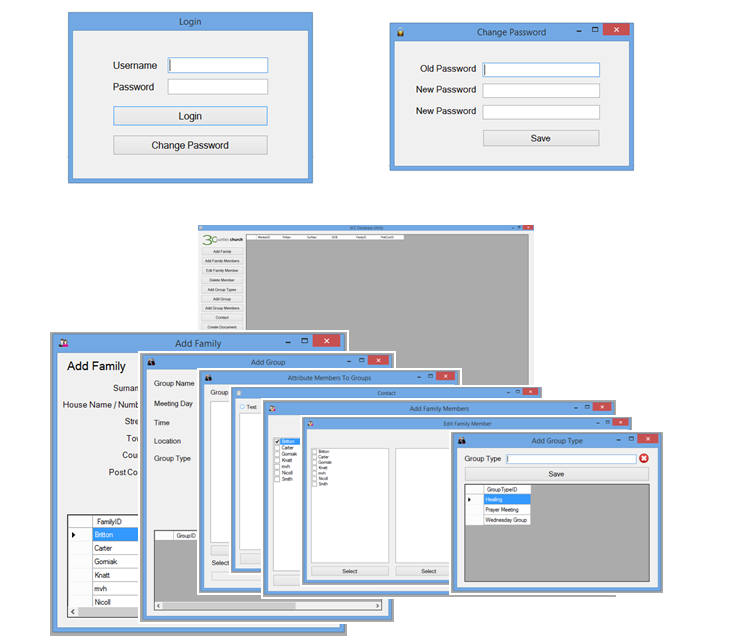
Contact – frmContact.vb

This form is where all member contact takes place in the form of either text or email messages. Contact details can be entered manually or a member or group selected from the checked list box on the left. The email and texts are sent through any email server e.g. Gmail the emails obviously being sent to the chosen recipient whereas the texts are sent to Clickatells smtp to sms service with the members number sent within the email. This is a paid and completely secure service with which credits can be bought online via their website.



## Form layout

After login all forms are accessed from the main menu as this is the main screen and main point of navigation.



## Sub Routines

frmLogin.vb

Region – DB connection

Contains the connections strings e.g. database name, password etc for the database and the query for selecting username and password, for use within the form.

Sub – frmLogin\_Load

This form hides the main menu form so that it will start running and opening the database but won’t open till the user is logged in. It also sets the password character for password entry to an asterisk using the command “txtPassword.PasswordChar = "\*"”. The final function of this sub is to allocate the pressing of the enter key to the pressing of the login button to allow swift logging in.

Sub – getInfo

This sub gets the username and password from the database using a data reader and a simple loop to be used in the login click sub to ensure that the data entered is the same as that in the database.

Sub – btnLogin\_Click

This sub calls the getInfo sub and checks the collected data against that entered by the user using a simple if statement. Revealing a error message box if it is incorrect and deleting any text in the password text box.

Sub – btnChangePassword

This sub opens the change password form.

frmChangePassword.vb

Region – DB Connection

This region contains the connection strings for the database and two of the three SQL queries used within the form. It also declares a few other variables including the array for the current username and password.

Sub – btnSave\_Click

This sub procedure uses a slightly more complex If, else statement to ensure that the old password entered is equal to that saved in the database and that the two new passwords entered are identical. If this is the case it then deletes the old password and updates it with the new one using a simple SQL statement and gives a message to tell you that it is finished. Otherwise it comes up with an error message telling the user that either the old password is incorrect or the two new passwords are not identical.

Sub – getInfo

This sub gets the username and password from the database using a data reader and a simple loop to be used in the save click sub to ensure that the data entered is the same as that in the database and to change the data within the database.

Sub – frmChangePassword\_Load

This sub hides the login form so that it is the only form open. It also sets the password character as an asterisk for all three password text boxes using the PasswordChar property and runs the getInfo sub.

frmMainMenu.vb

Region – DB Connection

This region contains the database connection strings to be used within the form.

Sub – CreateDatabase

This sub contains the DDL for the database so that on the programs first run the tables will be created. It also ensures that no tables have been deleted and recreates them if so. It uses a try catch to ensure that the database is open, if not an error message is displayed and the program closes.

Sub – frmMain\_Load

This sub opens the login screen and runs the CreateDatabase sub and the DataGridView sub to get the ball rolling.

Sub – DataGridView

This sub uses a MySQl data adapter to fill a dataset to fill the data grid view on the main page with the information on all members currently saved in the system. It only runs if the database is open.

Subs – Form Button Clicks

These sub procedures open the other forms in the program

* btnAdd
* btnEdit
* btnAddGroup
* btnContact
* btnAddFamilyMembers
* btnGroupType
* btnAddGroupMembers

Sub – btnCreateDocument\_Click

This sub takes the data from the database using SQL Select statements and formats it into a Microsoft Word Document .docx format. It uses the Microsoft Office .dll to import this functionality it then uses various loops, if and SQL statements to take the data from the database and format it into a contact book.

Sub – btnRefresh\_Click

This sub runs the DataGridView sub whenever the refresh button is clicked so that the newest data will show in the data grid view.

frmAddFamily.vb

Region – DB Connection

This region contains the database connection strings to be used within the form.

Region – RegEx Add Member

This region contains all the RegEx statements and all the sub procedures for the validation of all the text on this form. There are three RegEx statements for just text, text and numbers and English postcodes. There is a textchanged sub for each text box which runs the subsequent validation sub using an if statement to check the text against the RegEx and to change the ticks and crosses based on this.

Sub – btnSave\_Click

This sub is the most complex of the form as it handles the saving of all data with multiple fail safes and checks to ensure correct entry. The top of the nested If statement asks the user whether they are sure they want to save and on selecting yes it goes into the second If statement which first ensures that all the data is valid. Then inside a try catch to ensure the database is running and saving is successful the connection is opened and an SQL statement with all the data in is created and executed. All the text boxes are then cleared a message box displayed to show the user that is was successful and the data grid view refreshed by running its sub.

However if the database is closed or the saving fails the text is kept within the text boxes and a connection error is displayed. Also if some of the data is not valid a message box appears asking the user to adjust the data accordingly.

Sub – DataGridView

This sub displays all the data of families that have already been entered into the system using a MySQL data adapter to fill a dataset to be shown in a data grid view.

frmAddFamilyMember.vb

Region – DB Connection

This region contains the database connection strings to be used within the form.

Region – RegEx

This region contains all the RegEx statements and all the sub procedures for the validation of all the text on this form. There is a textchanged sub for each text box which runs the subsequent validation sub using an if statement to check the text against the RegEx and to change the ticks and crosses based on this.

Sub - frmAddFamilyMembers\_Load

This sub runs the Checked list box filling procedure when the form loads

Sub – FillFamilyCheckBox

This sub uses a MySQL data adapter to fill a data table with all the families in the system which are then entered into the checked list box using a loop.

Sub – btnSave\_Click

This sub saves all the data to the database the top of the nested If statement asks the user whether they are sure they want to save and on selecting yes it goes into the second If statement which first ensures that all the data is valid. Then inside a try catch to ensure the database is running and saving is successful the connection is opened and an SQL statement with all the data in is created and executed. All the text boxes are then cleared a message box displayed to show the user that is was successful and the data grid view refreshed by running its sub.

However if the database is closed or the saving fails the text is kept within the text boxes and a connection error is displayed. Also if some of the data is not valid a message box appears asking the user to adjust the data accordingly.

Sub – BtnSelect\_Click

This sub selects the family chosen from the database so that when saved the family member is associated with the family.

frmEditFamilyMembers.vb

Region – DB Connection

This region contains the database connection strings to be used within the form.

Region – RegEx

This region contains all the RegEx statements and all the sub procedures for the validation of all the text on this form. There is a textchanged sub for each text box which runs the subsequent validation sub using an if statement to check the text against the RegEx and to change the ticks and crosses based on this.

Sub - frmEditFamilyMembers\_Load

This sub runs the first Checked list box filling procedure when the form loads

Sub – FillFamilyListBox

This sub uses a MySQL data adapter to fill a data table with all the families in the system which are then entered into the checked list box using a loop.

Sub – FillFamilyMembersListBox

This sub uses a MySQL data adapter to fill a data table with all the family members in the chosen family which are then entered into the checked list box using a loop.

Sub – btnSelectFamily\_Click

This sub selects the chosen family then runs the FillFamilyMemberListBox sub so that the family members can be shown.

Sub – btnSave\_Click

This sub updates the data in the database the top of the nested If statement asks the user whether they are sure they want to save and on selecting yes it goes into the second If statement which first ensures that all the data is valid. Then inside a try catch to ensure the database is running and updating is successful the connection is opened and an SQL statement with all the data in is created and executed. All the text boxes are then cleared a message box displayed to show the user that is was successful.

However if the database is closed or the updating fails the text is kept within the text boxes and a connection error is displayed. Also if some of the data is not valid a message box appears asking the user to adjust the data accordingly.

frmAddGroupType.vb

Region – DB Connection

This region contains the database connection strings to be used within the form.

Region – RegEx

This region contains the RegEx statement and validation procedures for the text box on this form.

Sub –DataGridView

This sub displays a list of all the group types that have already been entered into the system using a MySQL data adapter to fill a dataset to be shown in a data grid view.

Sub – btnSave\_Click

This sub saves the group type to the database, it ensure the data is valid and that the database is open, if not appropriate error messages appear. It then saves the data to the database, notifies the user, clears the text and refreshes the data grid view.

frmAddGroup.vb

Region – DB Connection

This region contains the database connection strings to be used within the form.

Region – Day Validation

This region contains the RegEx validation for the radio button day selection in this form.

Region – Other Validation

This region contains all the rest of the RegEx statements and the sub procedures for the validation of the text on this form. There is a textchanged sub for each text box which runs the subsequent validation sub using an if statement to check the text against the RegEx and to change the ticks and crosses based on this.

Sub – btnSave\_Click

This sub saves all the data to the database it ensures the data is valid and the database is open before saving. It contains try catches and if statements to do this. It also translates the radio button selection into the full day text using if statements and then inserts it into the database, notifies the user and clears all text on the form.

Sub – btnSelect\_Click

This sub selects the chosen group type so that it can be inserted into the database when the group is saved.

Sub – frmAddGroup\_Load

This runs the data grid view sub when the form loads.

Sub – DataGridView

This sub displays a list of all the groups that have already been entered into the system using a MySQL data adapter to fill a dataset to be shown in a data grid view.

frmMemberGroup.vb

Region – DB Connection

This region contains the database connection strings to be used within the form.

Sub – FillroupCLB

This sub fills the checked list box for group display and selection using a MySQL data adapter filling a dataset then entered in using a loop function.

Sub – FillMemberCLB

This sub fills the checked list box for member display and selection using a MySQL data adapter filling a dataset then entered in using a loop function.

Sub – frmMemberGroup\_Load

This sub runs the two checked list box subs when the form loads.

Sub – btnSelect\_Click

This sub selects and finalises the group chosen to add members too, so that its ID is entered into the SQL table that links groups and members.

Sub – btnAdd\_Click

This sub selects and adds members to the third list box to display chosen group members. So that when saved the member ID is placed in the SQL table that attributes members to groups.

Sub – btnSave\_Click

This button saves the selected member into the selected group using the primary key ID of the member and the same of the group so that they can be linked together within the program.

frmContact.vb

Function – SendEmail

This function uses the SMTP client function of VB to send emails using the information typed in on the form plus the server information in the send sub. It contains a try catch to ensure that the server and the port is open or correct.

Sub – btnSend\_Click

This sub first discerns whether the user wants to send a text or an email using a simple if else statement. For an email it takes the recipient, subject and text body from the form and inserts them into the email function along with the server port and login details of the chosen email service. It then displays a message box confirming the email was sent and to and from whom.

For texts it does much the same also inserts Clickatell’s SMTP to SMS syntax including account information and mobile number so that their servers can send it as text message to the members mobile phone.

Sub – btnSelect\_Click

This sub takes the selected members details and inserts it into the contact form from the database using a simple SQL select statement from the chosen members row of the table.

# Variable lists

## Login

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| query | String | Contains SQL statement |
| Command | MySqlCommand | Command to execute SQL contains statement |
| userInfo(0, 1) | String | Takes user name and password |
| dr | DataReader | Reads data from database then adds to string |

## Change Password

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| query1-3 | String | Contains SQL statement |
| Command1-3 | MySqlCommand | Command to execute SQL contains statement |
| userInfo(0, 1) | String | Takes user name and password |
| dr | DataReader | Reads data from database then adds to string |

## Main Menu

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| cmd | MySqlCommand | Command to execute SQL contains statement |
| sbCmd | String | Contains SQL statement for all tables |
| query | String | Contains SQL statement |
| adp | MySqlDataAdapter | Data adapter to fill dataset or datatable |
| ds | Dataset | Dataset to fill grid view |
| oWord | Word.Application | Word document application opener |
| oDoc | Word.Document | Word document |
| oPara1 | Word.Paragraph | Word Paragraph of text |
| oPara2 | Word.Paragraph | Word Paragraph of text |
| Year | String | Takes current year to enter in document |
| str | String | Takes data from database |

## Add Family

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| regName | Regex | RegEx for validating name |
| regHouse | Regex | RegEx for validating house name/number |
| regPost | Regex | RegEx for validating post code |
| IsMatch | Boolean | Checks the Regex with the text inputted |
| SqlCommand | String | Contains SQL statement |
| Command | MySqlCommand | Command to execute SQL contains statement |
| query | String | Contains SQL statement |
| adp | MySqlDataAdapter | Data adapter to fill dataset or table |
| ds | Dataset | Dataset to fill grid view |

## Add Family Members

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| regName | Regex | RegEx for validating name |
| regEmail | Regex | RegEx for validating email address |
| regDOB | Regex | RegEx for validating date of birth |
| query | String | Checks the Regex with the text inputted |
| adp | MySqlDataAdapter | Data adapter to fill dataset or table |
| dt | DataTable | Data table to fill grid view |
| SqlCommand | String | Contains SQL statement |
| Command | MySqlCommand | Command to execute SQL contains statement |
| MemID | Integer | Takes the member ID to check against table |
| dr | DataReader | Reads data from database then adds to string |
| SqlCommand1-7 | String | Contains SQL statement |
| Command1-7 | MySqlCommand | Command to execute SQL contains statement |

## Edit Family Members

|  |  |  |
| --- | --- | --- |
| MemberInfo(0, 3) | String |  |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| regName | Regex | RegEx for validating name |
| regEmail | Regex | RegEx for validating email address |
| regDOB | Regex | RegEx for validating date of birth |
| isMatch | Boolean | Checks the Regex with the text inputted |
| query | String | Contains SQL statement |
| adp | MySqlDataAdapter | Data adapter to fill dataset or table |
| dt1 | DataTable | Data table to fill grid view |
| Checked | String | Gives data of checked member |
| query1-6 | String | Contains SQL statement |
| SqlCommand1-6 | String | Contains SQL statement |
| Command1-6 | MySqlCommand | Command to execute SQL contains statement |
| dr1-6 | DataReader | Reads data from database then adds to string |
| MemberEmail1-2 | String | Takes email to check in SQL for inner joins |
| MemberLandline1-2 | String | Takes Landline to check in SQL for inner joins |
| MemberMobile | String | Takes Mobile to check in SQL for inner joins |

## Delete Family Member

|  |  |  |
| --- | --- | --- |
| MemberInfo(0, 3) | String |  |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| regName | Regex | RegEx for validating name |
| regEmail | Regex | RegEx for validating email address |
| regDOB | Regex | RegEx for validating date of birth |
| isMatch | Boolean | Checks the Regex with the text inputted |
| query | String | Contains SQL statement |
| adp | MySqlDataAdapter | Data adapter to fill dataset or table |
| dt1 | DataTable | Data table to fill grid view |
| query1-6 | String | Contains SQL statement |
| SqlCommand1-2 | String | Contains SQL statement |
| Command1-6 | MySqlCommand | Command to execute SQL contains statement |
| dr1-6 | DataReader | Reads data from database then adds to string |
| MemberEmail1-2 | String | Takes email to check in SQL for inner joins |
| MemberLandline1-2 | String | Takes Landline to check in SQL for inner joins |
| MemberMobile | String | Takes Mobile to check in SQL for inner joins |

## Add group Type

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| regName | Regex | RegEx for validating name |
| isMatch | Boolean | Checks the Regex with the text inputted |
| query | String | Contains SQL statement |
| adp | MySqlDataAdapter | Data adapter to fill dataset or table |
| ds | Dataset | Dataset to fill grid view |
| SqlCommand | String | Contains SQL statement |
| Command | MySqlCommand | Command to execute SQL contains statement |

## Add Group

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| regName | Regex | RegEx for validating name |
| isMatch | Boolean | Checks the Regex with the text inputted |
| SqlCommand | String | Contains SQL statement |
| Command | MySqlCommand | Command to execute SQL contains statement |
| query | String | Contains SQL statement |
| adp | MySqlDataAdapter | Data adapter to fill dataset or table |
| ds | Dataset | Dataset to fill grid view |
| dt1 | DataTable | Data table to fill grid view |

## Contact

|  |  |  |
| --- | --- | --- |
| connectionString | String | Contains connection data for database |
| Connection | MySqlConnection | MySql connector containing the connection string |
| FromAddress | String | Address of the sender |
| Subject | String | Subject of the email |
| Body | String | Main body of the email |
| UserName | String | Username of Gmail account used |
| Password | String | password of Gmail account used |
| Server | String | Address of the SMTP Server |
| Port | Integer | Port number of the SMTP server |
| Email | MailMessage | All put together into one email message |
| query1-2 | String | Contains SQL statement |
| adp | MySqlDataAdapter | Data adapter to fill dataset or table |
| dt1 | DataTable | Data table to fill grid view |
| Checked | String | Gives data of checked member |
| dr2 | DataReader | Reads data from database then adds to string |
| command2 | MySqlCommand | Command to execute SQL contains statement |
| MemberID | String | Takes the member ID to check against table |

# Database Maintenance

Database Tables

|  |  |  |  |
| --- | --- | --- | --- |
| **Member** | | |  |
| **ID** | PK | Int | Unique primary key, for each member |
| **FirNam** |  | String | First name of member |
| **SurNam** |  | String | Surname of stranger |
| **DOB** |  | String | DOB as string |
| **FamilyID** | FK | Unique But Not Necessarily Int | Foriegn key, attributing them to a family |
| **PrefContID** |  | Int | Preffered method of contact |
|  |  |  |  |
| **Family** | | |  |
| **FamilyID** | PK | Unique but not necessarily int | Unique primary key, surname or number |
| **House Num/Nam** |  | String | House name or number |
| **Town** |  | String | Town or city |
| **County** |  | String | County of residence |
| **PostCode** |  | String | Postal code |
|  |  |  |  |
| **Contact** | | |  |
| **ContactID** | PK | Int | Unique primary key, for each contact detail |
| **MemberID** | FK | Int | Foriegn, attributing each contact data to member |
| **ContactTypeID** | FK | String | Foriegn, attributing each detail to type eg email |
| **Data** |  | String | The actual contact detail |
|  |  |  |  |
| **ContactType** | | |  |
| **ContactTypeID** | PK | String | Primary key, showing contact data type |
|  |  |  |  |
| **GroupType** | | |  |
| **GroupTypeID** | PK | String | Unique primary key for group type, eg cell |
|  |  |  |  |
| **Group** | | |  |
| **GroupID** | PK | Int | Unique primary key for each group |
| **GrName** |  | String | Group name |
| **GroupTypeID** | FK | Int | Foriegn key showing group type |
| **MeetingDay** |  | String | Meeting day of group |
| **StartTime** |  | String | Start time of group |
| **EndTime** |  | String | End time of group |
| **Location** |  | String | location of group |
|  |  |  |  |
| **GroupMember** | | |  |
| **GroupID** | FK | Int | Foriegn Key showing group |
| **MemberID** | FK | Int | Foriegn key showing which member to the group |

Relationships in family: If they are under 18, an over 18 in the family has parental responsibility and can be contacted for them. Any other under 18's are considered siblings, two over 18's are considers as partners next of kin, or parents.

## DDL

CREATE TABLE IF NOT EXISTS Member (MemberID Int(4) PRIMARY KEY AUTO\_INCREMENT, FirNam Varchar(40), SurNam Varchar(40), DOB Varchar(10), FamilyID Varchar(40));

CREATE TABLE IF NOT EXISTS Family (FamilyID Varchar(40) PRIMARY KEY, HouseNam Varchar(40), Street Varchar(40), Town Varchar(40), County Varchar(40), PostCode Varchar(12));

CREATE TABLE IF NOT EXISTS Contact (ContactID Int(5) PRIMARY KEY AUTO\_INCREMENT, MemberID Int(4), ContactTypeID Varchar(40), Data Varchar(40));

CREATE TABLE IF NOT EXISTS ContactType (ContactTypeID Varchar(40) PRIMARY KEY);

CREATE TABLE IF NOT EXISTS GroupType (GroupTypeID Varchar(40) PRIMARY KEY);

CREATE TABLE IF NOT EXISTS Groups (GroupID Int(4) PRIMARY KEY AUTO\_INCREMENT, GroupName Varchar(40), GroupTypeID Varchar(40), MeetingDay Varchar(40), StartTime Varchar(5), EndTime Varchar(5), Location Varchar(40));

CREATE TABLE IF NOT EXISTS GroupMember (GroupID Int(4), MemberID Int(4))

CREATE TABLE IF NOT EXISTS LogInDetails (username Varchar(40), password Int(40))

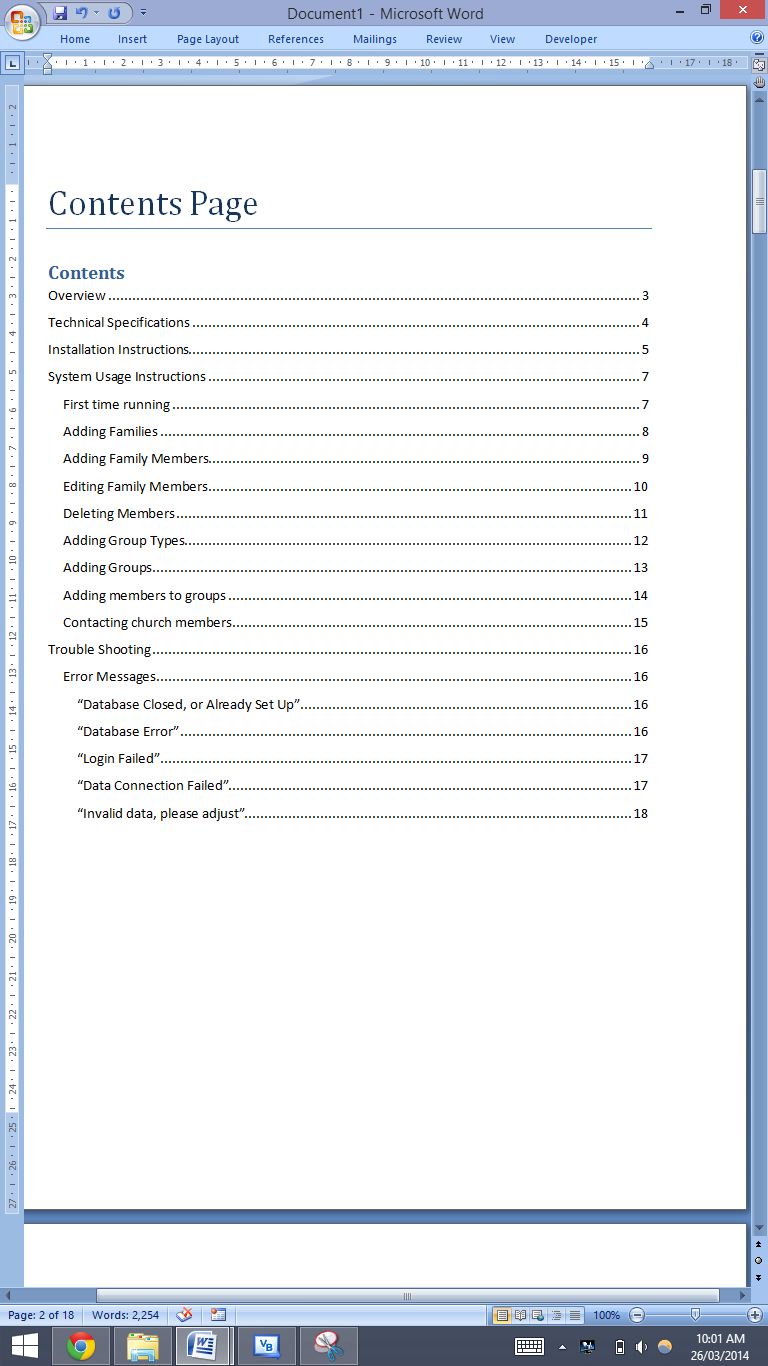
# User Manual

Three Counties Church

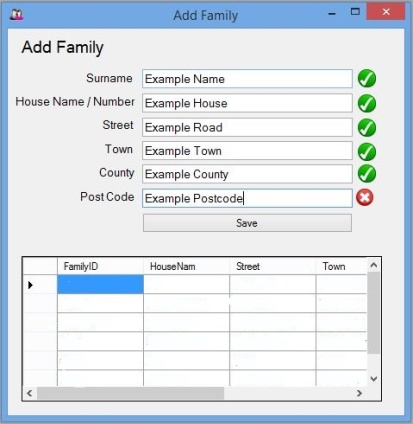
Database System

User Guide

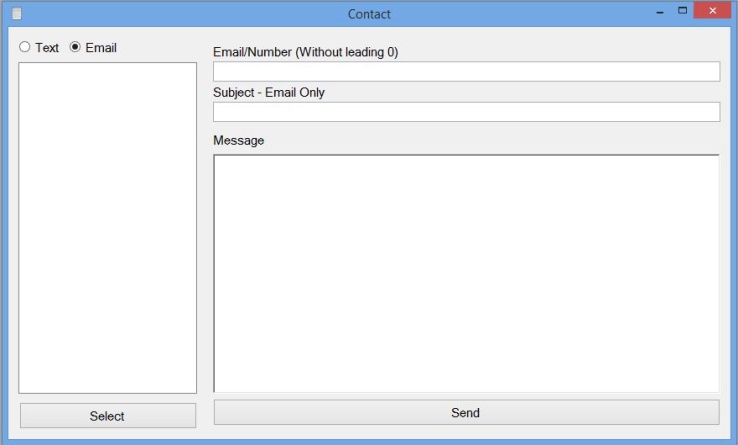
# Contents Page



# Overview

This program has been designed to give you a smooth, easy to use and highly comprehensive database system. It allows entry of large amounts of church member data including multiple contact details and a range of other information. The data is stored securely in a password protected local server that is easily backed up for extra peace of mind.

The system has many other features including the ability to create a contact book from the database in the widely supported and easily formatted .docx file type. It also has a group feature that allows the entering of different types of groups and their time and location information, with the ability to link members to it allowing emails or texts to be sent to groups of people e.g. prayer or life groups.

The program requires admin login on entry ensuring that it complies with the Data Protection Act and allow the user to change the password periodically for extra safety. The system has the ability to send emails to custom emails or those saved in the database. It also allows the sending of texts (as long as text credits have been purchased from Clickatell ([www.clickatell.co.uk](http://www.clickatell.co.uk))) to one or more recipients at once.

# Technical Specifications

Minimum System Requirements

* Windows XP Service pack 3 or higher (32bit or 64bit)
* 1.1 GHz Processor or greater
* 1GB Ram
* 1GB Free Space
* On board graphics
* Monitor with a resolution of 640 by 480 or higher
* Available USB port for program transfer
* Microsoft Office 2007 or higher for document view

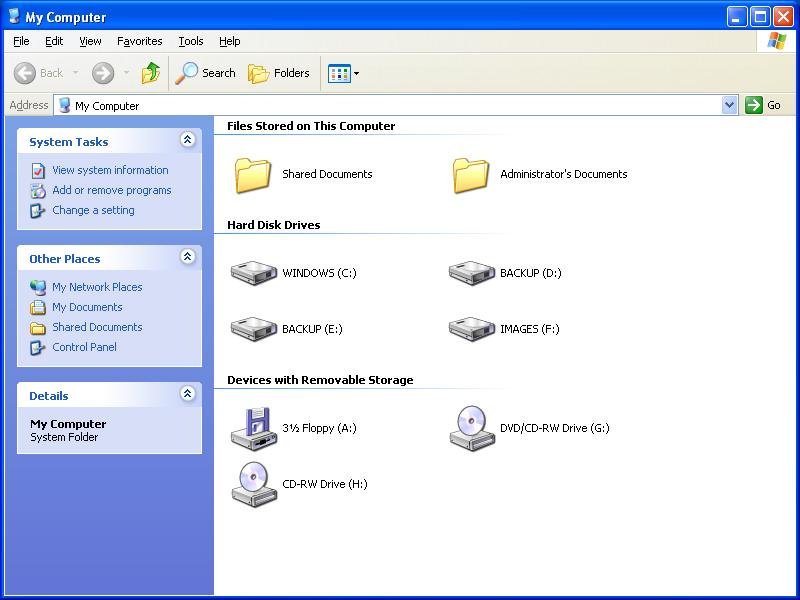
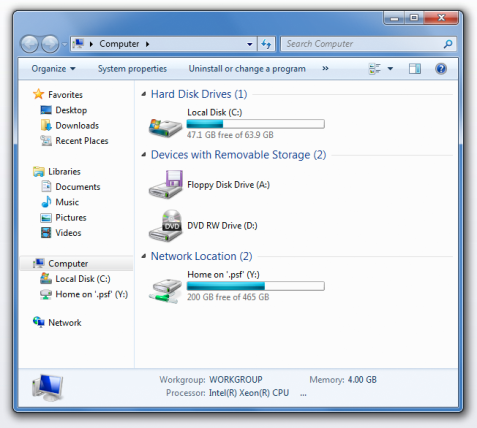
Recommended System Requirements

* Windows 7 (64bit)
* 2.1 GHz Processor or greater
* 4GB Ram
* 2GB Free Space
* On board graphics (Intel 3000, AMD HD 6000)
* Monitor with resolution of 1366 by 768 or greater
* Available USB port for program transfer
* Microsoft Office 2007 or higher for document view

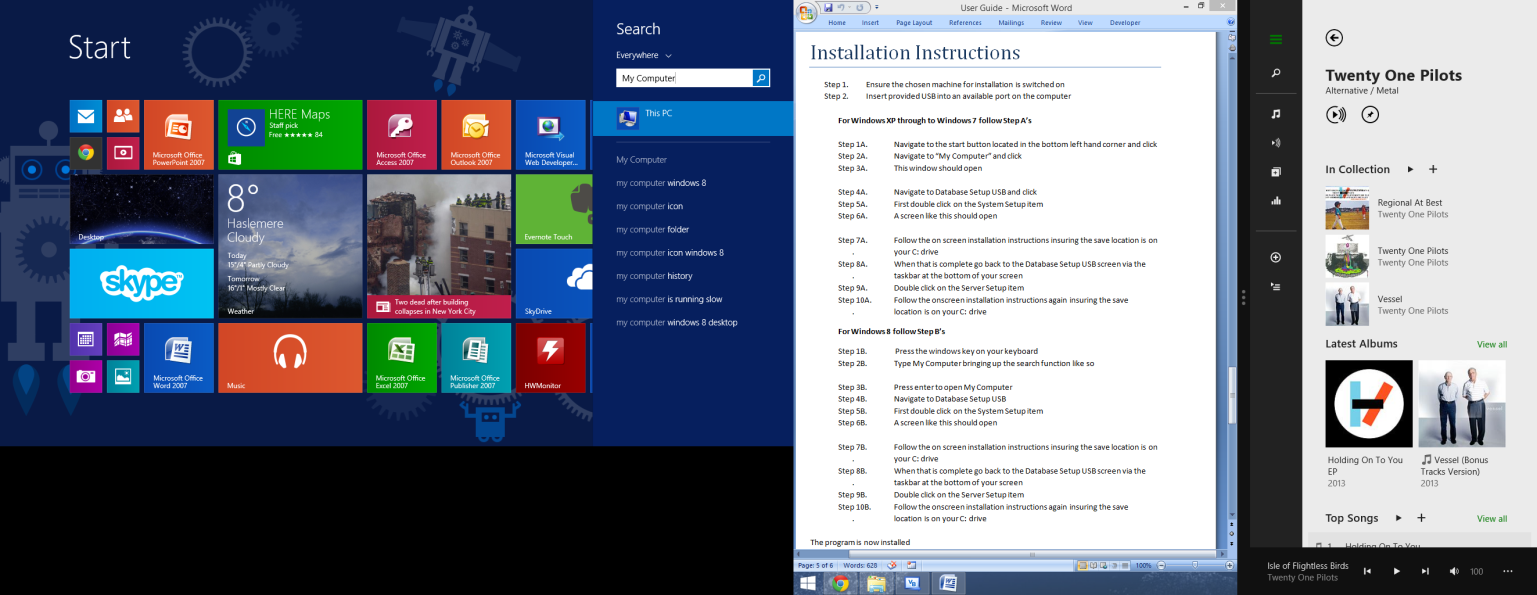
# Installation Instructions

1. Ensure the chosen machine for installation is switched on
2. Insert provided USB into an available port on the computer

**For Windows XP through to Windows 7 follow Step A’s**

1. Navigate to the start button located in the bottom left hand corner and click
2. Navigate to “My Computer” and click
3. This window should open
4. Navigate to Database Setup USB and click
5. First double click on the System Setup item
6. Select yes and ok on any messages
7. When that is complete go back to the Database Setup USB screen via the . taskbar at the bottom of your screen
8. Double click on the WampServer item
9. Follow on screen instructions for installation

**For Windows 8 follow Step B’s**

1. Press the windows key on your keyboard
2. Type My Computer bringing up the search function like so
3. Press enter to open My Computer
4. Navigate to Database Setup USB
5. First double click on the System Setup item
6. Select yes and ok on any messages
7. When that is complete go back to the Database Setup USB screen via the . taskbar at the bottom of your screen
8. Double click on the WampServer item
9. Follow on screen instructions for installation

The program is now installed

# System Usage Instructions

## First time running

When first opening the system you first need to open the server software and then the main program.

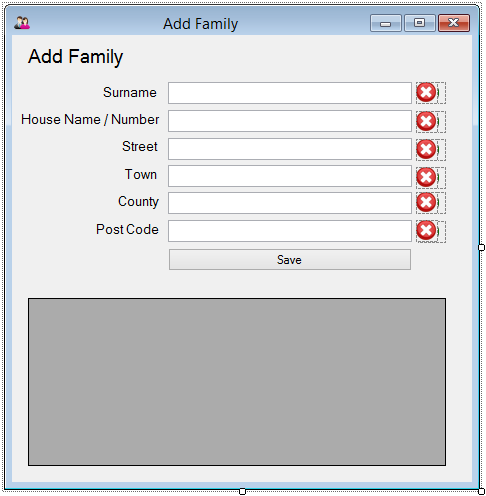
1. First navigate to your installed your installed programs via the start menu or the metro
2. Find or search for WampServer and select “start WampServer”
3. A security message will appear, select yes and the server will open in the background.

Then you can open the main program

1. Again navigate to the start menu or the metro and find the program
2. You will be first greeted by a Login screen
3. The username is Admin and the password is set at Password, but before logging in you must first change the password
4. Select change password
5. Type in the set password “Password”
6. Then type in your new custom password, ensuring it is secure
7. You can then login for the first time

## Adding Families

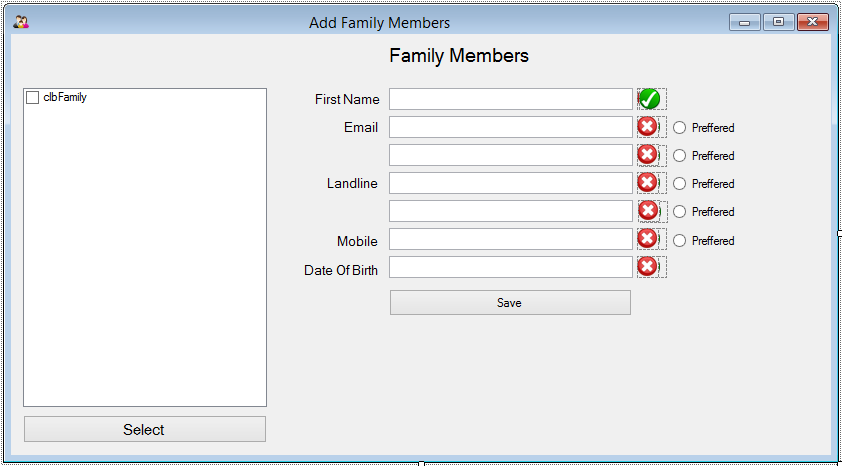
1. Navigate to the Add Family button and a new form should open like so:



1. Now you can enter the details of the family into the form, the system assumes that families share a surname and an address so those details are entered here
2. As you enter text the crosses will become ticks when the data is valid, certain characters are invalid so try to keep it text or numbers only. The post code has to be in capital letters and the two parts separated by a space, the system only allows valid UK post codes.
3. When you have finished entering all the data and it is all valid click the save button
4. A message box should appear commenting on the success of the save, if an error message appears please check the troubleshooting section of this guide.
5. The text in the form should now have disappeared and the family appeared in the grid below.
6. You can now enter another family or close the form and continue with any other operation.

## Adding Family Members

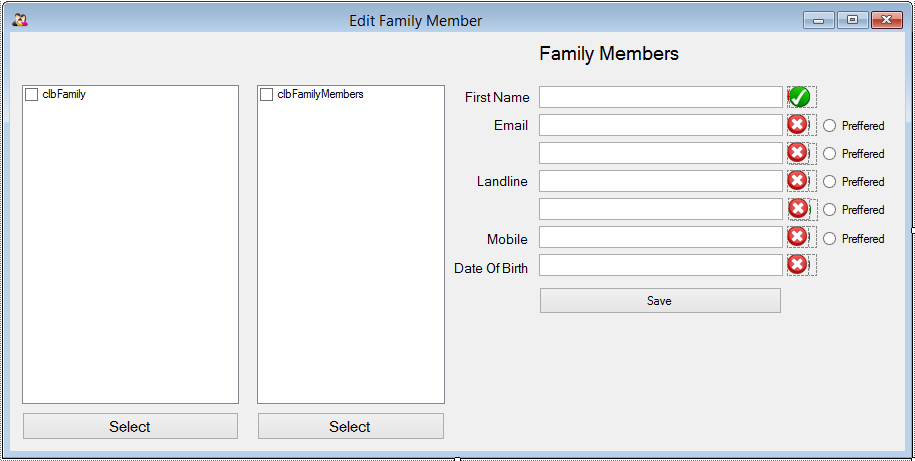
1. Navigate to the Add Family Members button on the main menu, this for should open



1. Scroll through the available families saved by surname and tick the chosen family
2. Finalise this by clicking select
3. Type in the details of the family member, all of the text boxes are validated so only valid text will be allowed. This means that only valid emails and valid dates of birth will be accepted. Select which email and phone number is the preferred contact.
4. Click save to add the data to the database, a message shall appear stating whether the save was successful or not, if you get an error message check the troubleshooting section of this guide.

## Editing Family Members

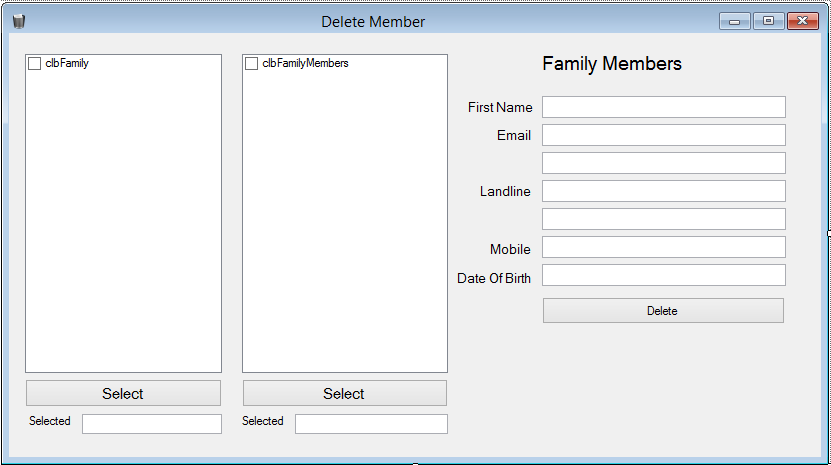
1. Navigate to the Edit Family Members button on the main menu, this window should appear



1. Scroll through the available families saved by surname and tick the chosen family
2. Press select to finalise
3. Scroll through the available members tick the chosen member for editing
4. Press select to finalise
5. The member’s data will appear in the text boxes allowing you to edit and save their data much like in the Add Members window.

## Deleting Members

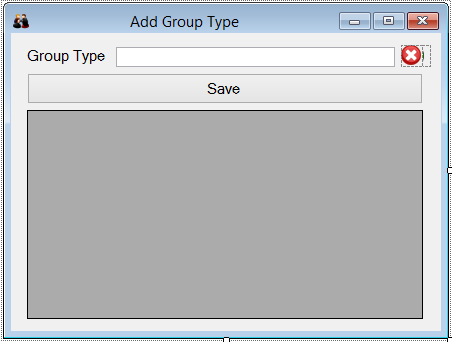
1. Navigate to the Delete Family Members button on the main menu, this window should appear



1. Scroll through the available families saved by surname and tick the chosen family
2. Press select to finalise
3. Scroll through the available members tick the chosen member for editing
4. Press select to finalise
5. The member’s data will appear in the text boxes allowing you to ensure you have the right member and then delete

## Adding Group Types

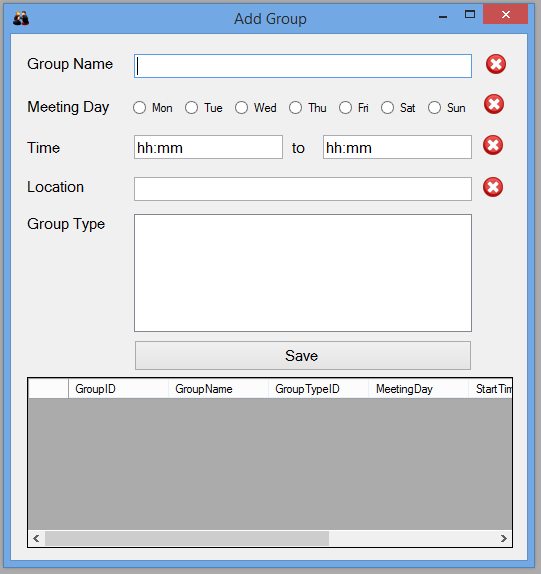
1. Navigate to the Add Group Type button on the main menu and this window should open



1. Type in the name of the group type into the text box, only plain text and a select few characters are allowed
2. Click save and then yes, the save should be successful, if you see an error message go to the trouble shooting section of this guide
3. The group types should appear in the table below and you can continue to add more.

## Adding Groups

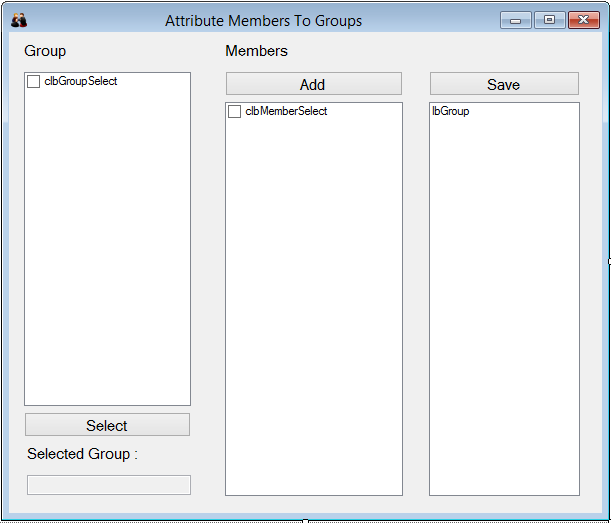
1. Navigate to the Add Group button on the main menu, a form like this should open



1. Fill in all the data on the form following all the stated text forms
2. Select the chosen group type and click save, if successful a message will appear stating this, if not go to the troubleshooting section of this guide.
3. The group should appear in the table below and you can continue to add new groups

## Adding members to groups

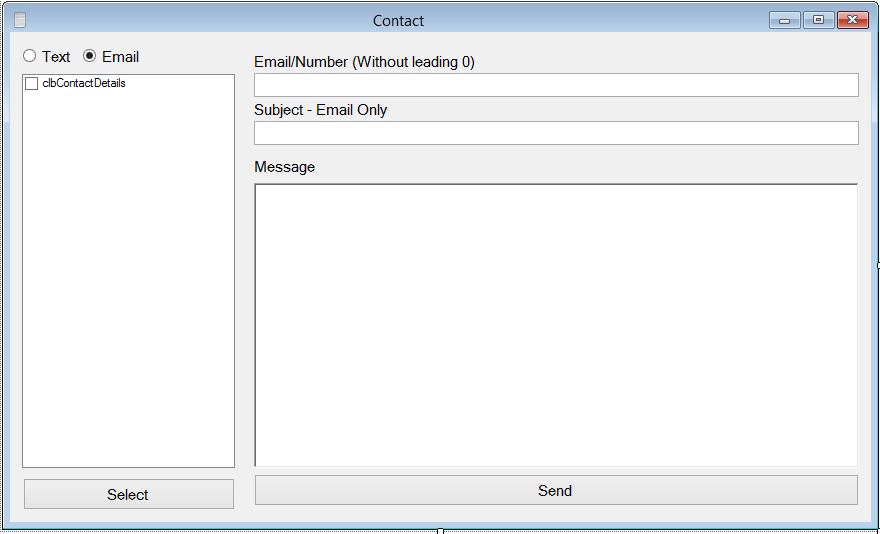
1. Navigate to the Add Group Members button on the main menu, this window will open



1. Select the group to add members too from the list on the left
2. Press select to finalise the decision and the selected group should appear below
3. Then select any members that are to be added to the group using the next list, this can be done one by one or multiple at a time.
4. Press the add button to add them to the list of selected members
5. When you are happy with the chosen members press the save button, if this brings an error message check the troubleshooting section of this guide
6. You can then add members to more groups or close this form and continue with any other functionality of the system.

## Contacting church members

1. Navigate to the Contact button on the main menu, this form should appear



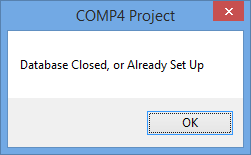
1. To send an email select the email tick box and for text messaging select the text button.
2. To send to a custom number/email address type the addressee’s details (either an email or a mobile number without the leading zero.

# Trouble Shooting

## Error Messages

### “Database Closed, or Already Set Up”

This message is not necessarily a bad thing. When the system first runs it creates the tables for the database and fills some with the appropriate data, then every time afterwards it checks that they have already been created so that is does not overwrite things like password. Therefore this message will appear on every opening of the program. Continue using the program as usual and unless another error message appears before the login screen is useable everything is fine.



### “Database Error”

This message will appear on opening if the database server is not open or switched on. If this happens ensure you have opened wamp server by making sure the W icon is visible and coloured green on the taskbar (see second picture). If it is red click on it and select start all services, if it green select restart all services. If however it is not there the database server is not switched on, therefore you need to open it, this is done by searching wampserver in the start menu and opening it. After saying yes to the message that pops up the database should be open and the system can be restarted and ran as usual.

## Capture.JPG

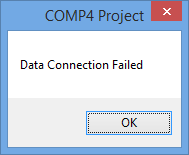
### “Login Failed”

This message will appear when the password or username you have entered is incorrect, the user is Admin. The password initially is Password but you should reset it at first use. Ensure that caps lock is not on and that if using numbers NumLock is on, if none of these work make sure that you are using the correct password. This could also happen if the database has closed whilst trying to login, if none of the above works first try resetting the database as described in the “Database Error” trouble shooting guide above. If this also fails contact the programme creator to get it reset.

## 

### “Data Connection Failed”

This message means that when entering data into the database something went wrong, to rectify this ensure all your data has ticks next to it and that all the data is less than 40 characters long. Then try again. If this is unsuccessful try resetting the database as described in the “Database Error” trouble shooting guide above.

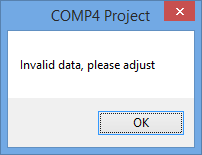
****

### “Invalid data, please adjust”

This error message will pop up only if the data you have entered into the form is invalid, for example using numbers or unaccepted characters in a name. The data is valid when the cross aside the text becomes a tick, so ensure that all the ticks are visible before saving (Unless with contact details were not all need filling in). If this does not fix it close and reopen the form. Most text inputs are limited to 40 characters.

The allowed characters are as follows:

Name (Surname, Firstname, Street, Town, County, Group Name\*, Group Location\*)

* A-z (Capital and lower case)
* -
* ‘
* **No Numbers**

House Name/Number

* A-z (Capital and lower case)
* -
* Any number

Post Code

* Any valid postcode

Date of birth

* Any valid date

Phone number

* 0 followed by 10 numbers

Email - A-z0-9.\_%+-

* A-z (Capital and lower case), 0-9, ., \_, %, +, -
* Followed by @
* Followed by A-z (Capital and lower case), 0-9, ., -
* Followed by .
* Followed by .co.uk, .com, .net (or any other email ending)

Time

* HH:MM

\*Group name and location can contain numbers

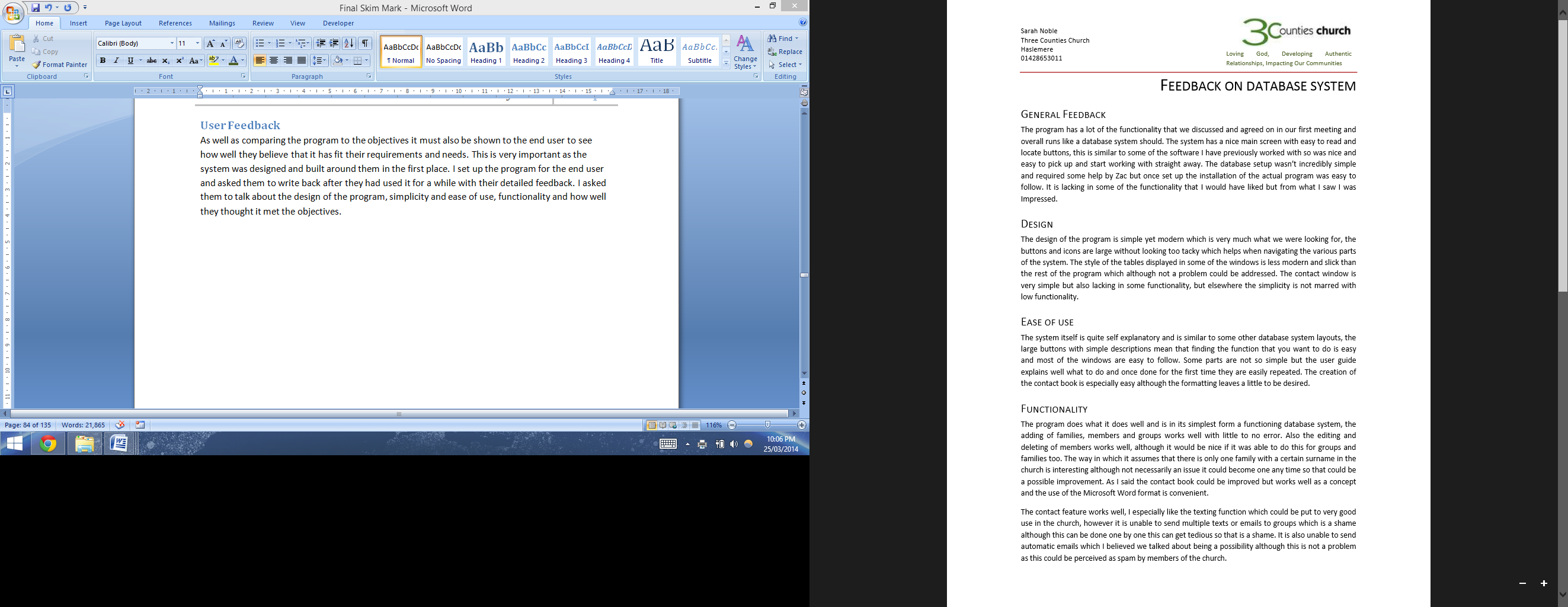
# Appraisal

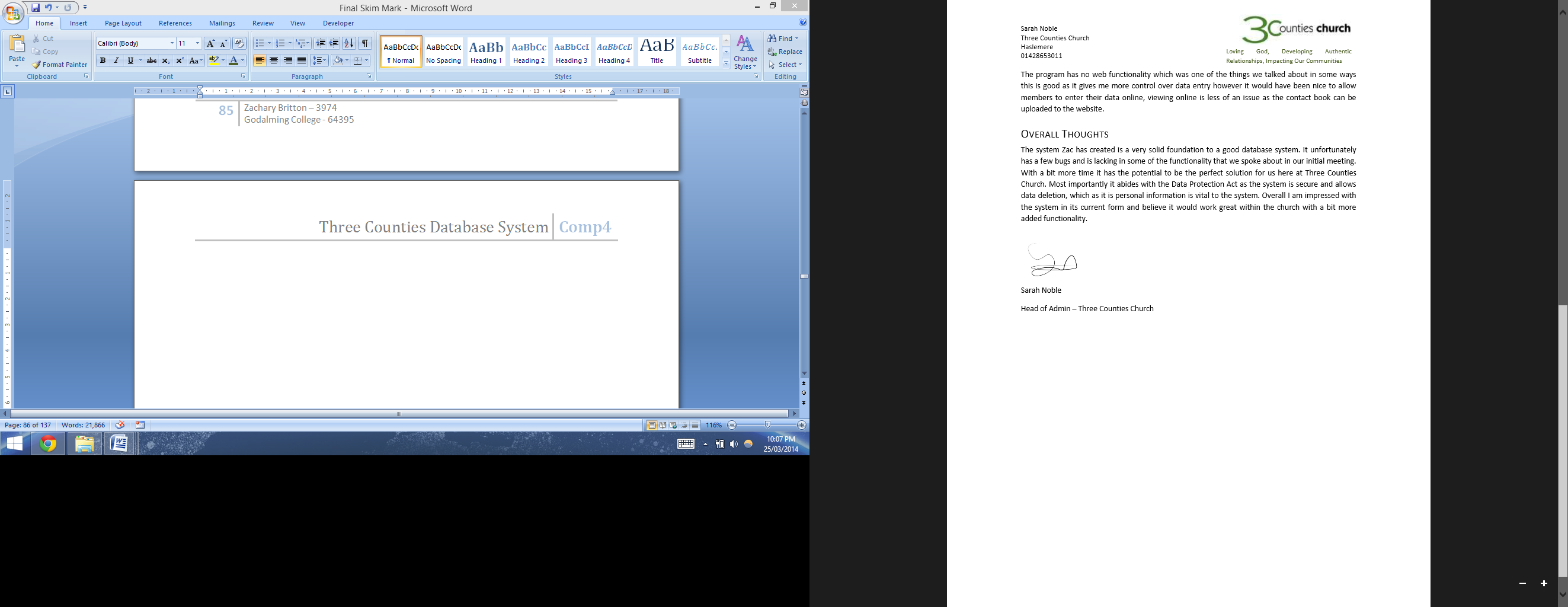
## Comparison Against Numbered Objectives

|  |  |
| --- | --- |
| 1. The system will be able to display the details stored in the database quickly and simply    1. The data should be displayed in under 5 seconds    2. To display the data a maximum of 6 button presses will be used    3. The display of the data will be in a simple table | This objective has been achieved; the opening of the data grid on the main menu that displays the data runs in the background while the user is logging in. Therefore it passes 1a and 1b. However cannot be opened until login is successful therefore is secure.  The data is displayed in a simple data grid view table which is very simple and easy to use. 1c |
| 1. The system will be based around a solid and secure server based database    1. The software will be password protected    2. The database will be stored remotely and be accessible from multiple computers | This objective has also been achieved the database itself is password protected and the software requires password entry 1a.  However 2b has not been achieved, it would be entirely possible and rather easy to place the database on a server in a remote location however this was not implemented into the program as local hosting was easier when coding. |
| 1. The system will have adequate security and privacy features and abide with the data protection act    1. The software will request a password reset every month    2. The software will allow editing and removal of data as to abide with the data protection act    3. The software will automatically log out after 10 minutes of inactivity | Part 3a has not been achieved although the user guide recommends that the password is changed regularly it is not done through the program although with more time this could be achieved.  3b has been achieved as the system allows editing and deleting of members from the system be it name or contact detail change, this ensures that it abides with the data protection act.  3c was not achieved as the system will stay logged in until closed however could be achieved with a simple timer. |
| 1. The system will allow the admin to send out automated and custom emails and texts    1. It will allow emails and texts to be sent on set dates and times automatically    2. The system will be capable of efficiently sending emails and texts in under a minute    3. The software will allow single or multiple recipients from custom or set groups | The system is unable to send automated emails although the normal email feature is fully functional. Therefore 4a has not been achieved.  4b however has been achieved as the system can send emails using Gmail’s SMTP server and text using Clickcatell’s SMTP to SMS service again via Gmail’s SMTP servers. They are sent quickly and efficiently and as long as the recipient is connected to the internet/has phone signal they are received within a minute.  The system can send emails or texts to a single recipient one at a time but not to multiple recipients although this could be added later on, 4c. |
| 1. The system will allow quick and functional sorting by means of surname age and other attributes    1. The system will allow the searching of and within groups    2. The system will allow simple and complex searching by tick box attributes or by name, age or other physical detail    3. The search will give results in under 10 seconds | The system does not currently allow searching of data although all data is able to be viewed; this was due to time constraints. 5a  However the system does allow simple searching via surname when adding and editing family members. 5b  The only method of searching used, shown above, gives almost instant results. 5c |
| 1. The system will allow flexible grouping with custom groups with automated contact and notification systems    1. The system will allow over 30 different groups    2. Groups selectable and editable in data input window    3. Automated emailing for group activities and events can be set    4. The system will allow for the input of meeting and event time and dates for groups | The system allows an almost unlimited number of groups with many different types and any location or time. This means 6a has been achieved.  6b and 6c have not been achieved as although the data can be entered it is not possible to remove it however this could be a possible extension of the program. Also even though times and locations are entered and members are attributed to groups it is not possible to email groups, automated or not.  6d has been successfully achieved as all the information regarding groups can be entered, including type (Healing etc) day, time and location. All the data that is needed to organise and notify people of events. |
| 1. The system will allow church members to access and edit their own details online and see others too    1. The web app should be accessible via the church website login section    2. It will show only data that the members want to be visible online as to abide with privacy laws    3. The editing process should be simple and only take a couple of clicks to accomplish    4. The web app should allow members to add data of their family members | This part of the program was never completed, partly due to the database being local and partly due to technical and time based constraints. It also saved issues with data protection as data is not visible online.  Therefore 7a, 7b, 7c and 7d have not been achieved. |
| 1. The system will allow members to select whether they want their data published or not    1. They should be able to select whether they want it available in the contact book and/or online    2. This should be accomplished by a simple tick box operation either online or by the administrator | This part was also not successfully completed as the contact book creation part of the program was one of the last to be completed. I made the adding members screen before realising that I needed a way of notifying what data was to be entered and what wasn’t.  So unfortunately 8a and 8b have not been completed. |
| 1. The system will allow the production of the contact book in various forms and types    1. The book will be creatable in .pdf or .docx formats    2. It should be created by the system in under a minute | The system does create a contact book in the form of a .docx word document which can be saved as a .pdf for emailing if need be. 9a  9b is also achieved as the system quickly creates the document with a relatively large data set however beyond a certain amount of data it would probably take longer than a minute. |
| 1. The system will have functional automatic validation and manual validation for member submitted data    1. The system will use RegEX validation to ensure all data entered is in the right format and can be processed correctly by the system    2. The system will validate instantly, and upon saving data | All parts of objective 10 have been achieved the whole system is fully and meticulously validated ensuring that all data entries are successful and all data e.g. emails and numbers can be used accurately and successfully by the system.  It uses Regular Expressions to do all of this therefore achieves 8a the data validates as it is saved so achieves 8b. |

## User Feedback

As well as comparing the program to the objectives it must also be shown to the end user to see how well they believe that it has fit their requirements and needs. This is very important as the system was designed and built around them in the first place. I set up the program for the end user and asked them to write back after they had used it for a while with their detailed feedback. I asked them to talk about the design of the program, simplicity and ease of use, functionality and how well they thought it met the objectives.





## User Feedback Analysis

Overall Sarah Noble was pleased with the system especially the look and simplicity of the way the forms worked and linked together. The general functionality of the system was approved of however she did think that it was lacking certain features that I had put in the scope.

Mrs Noble very much approved of the design of the system saying it was clean and simple and she liked the large buttons and icons of the system. This was something I put a lot of effort in as I wanted it to be clean and simple but also easy to read and follow.

Again she liked the ease of use of the system saying it was relatively self-explanatory and that any parts that were mildly confusing where well explained in the user guide. This was one of the main objectives of the system so I am glad that she also thought that the system was easy to use.

Mrs Noble was again pleased with the current functionality of the system although it was lacking in a few things that she would have liked. As a simple database storage system she thought that it worked well and functioned like any database should, I agree with this as although it is lacking it I in its simplest form a database system. She did notice the odd error or bug which is to be expected. The way the system assumes that there is only one family, living in the same house she thought to be interesting although not a problem as is.

She very much liked the contact system especially in the way it sent texts, which she thought would be a very well used feature within the church. The few parts of the system such as group and automated emails that have not been created yet could definitely be added and she agrees that these would definitely benefit the system and its functionality. Although she believed that some parts not being added had their benefits too, for example not adding web functionality gave her more control over the data and ensured the security of the data.

Overall Mrs Noble was impressed with the system, even though it was missing certain features she thought that it had great potential. She was happy with the security of the system and thought that it would abide with the data protection act. I am also happy with the system and with this feedback especially that about the design of the system, obviously there is much more that the system could do which would improve it greatly but this could obviously be added with time.

## Possible Extensions

There are a few things that could be added to the system to improve it, such as the un-finished objectives and the odd other improvement be it aesthetically or functionally.

### Improved Group System

The grouping system is lacking an editing or deleting function which is pretty important and could be added, again using a new form and some SQL delete statements and a bit of RegEx validation. This small addition would improve the system a lot as it is pretty essential. Modifying the home table would be nice to allow the viewing of group data within the system would be another good and somewhat essential function. Also the ability to use the group information to send automatic and manual group emails would be very useful and would make the contact form much more functional. This would allow the admin to notify people of group events and regular meetings.

### Online Functionality

Creating a VbWeb application for the church system is a potential extension to the program as although not essential would make data entry and updating much more convenient for the administrators and for the members of the church. Although this could potentially throw up some security and validation errors if well thought out it could be a great improvement. It would reduce some of the workload for the admin and allow easier access to the data for the church.

### Improved Security

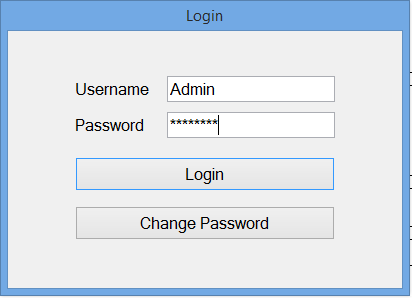
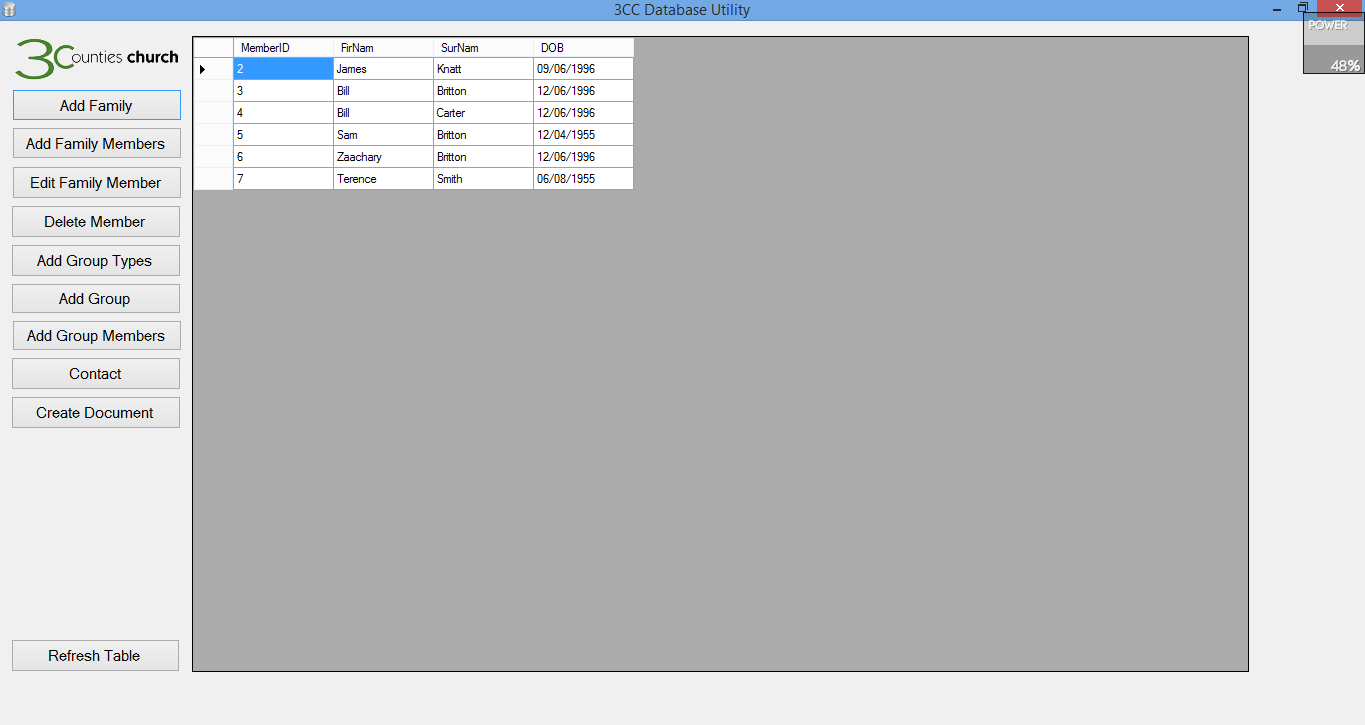
The system although relatively secure could be improved greatly to ensure greater data security. For example the addition of allowing the user to choose what data is published is essential to ensure that it abides to the data protection act. Also including the ability to have multiple logins some with restricted access would add security. The system is also lacking in a back up procedure which is almost vital for this kind of database. The database could also potentially be more secure with the use of data encoding so that even if hacked into it would be unreadable.

All of these suggestions could improve the program greatly and with time could be implemented into the system with ease.

# Appendix

## Screen Shots – Testing

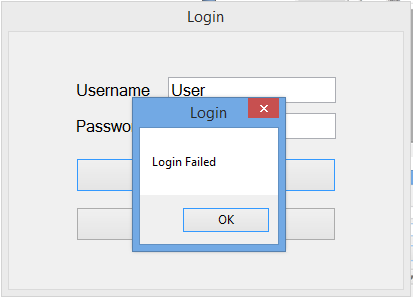
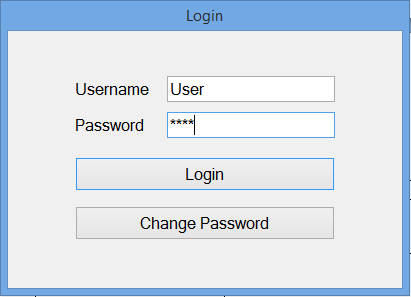
### 1

1B

1A

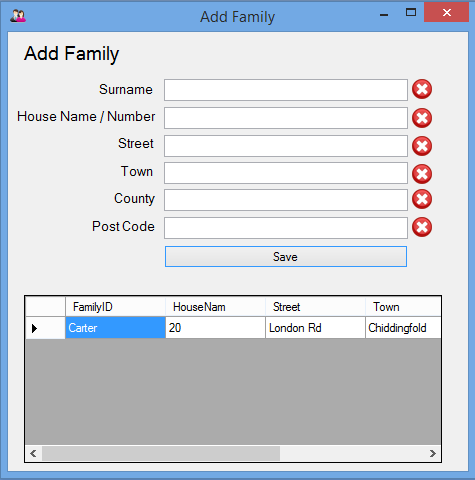
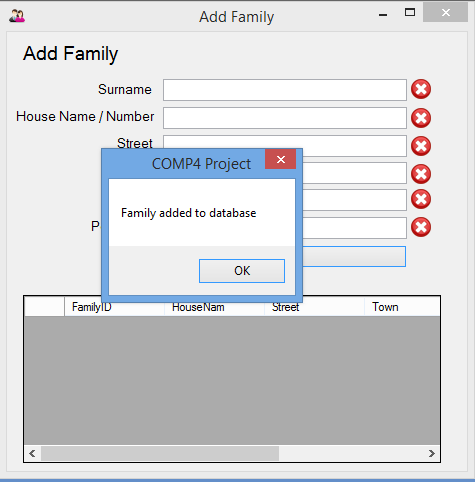
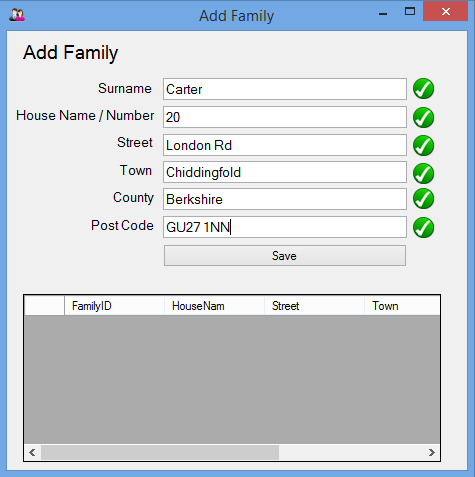
### 2



2A

2B

### 3

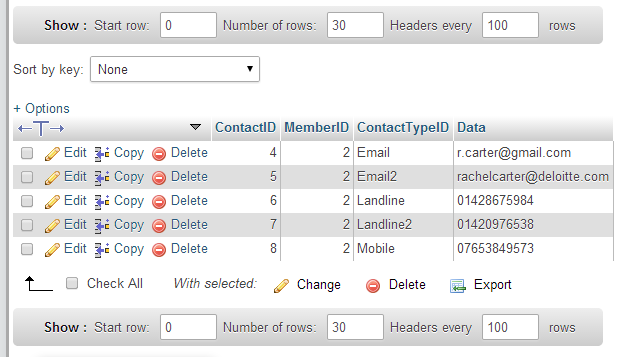
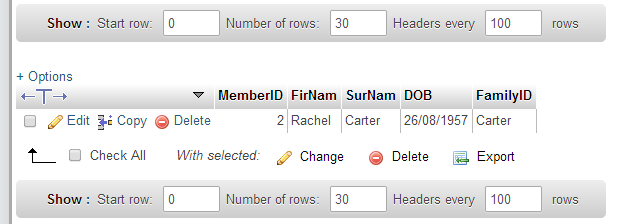
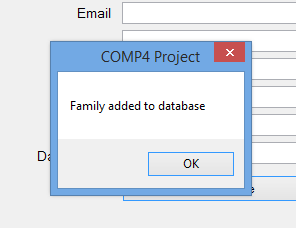
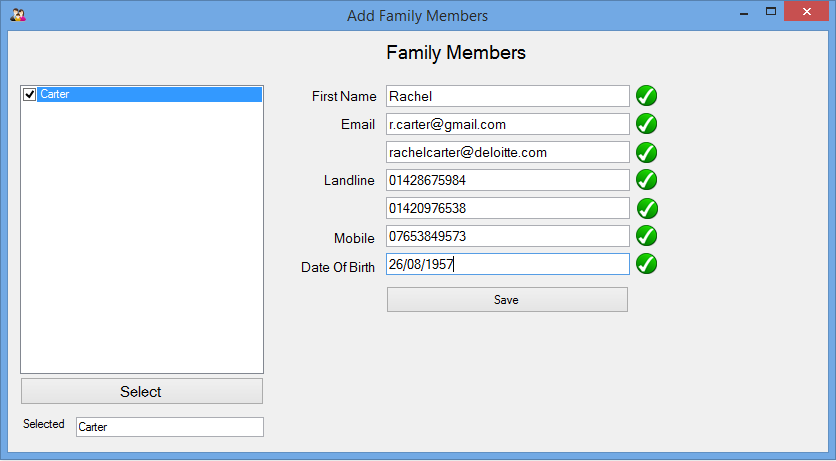


3C

3A

3B

### 4



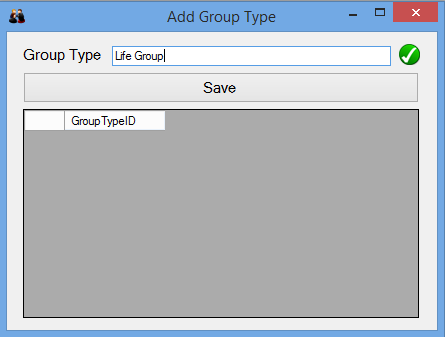
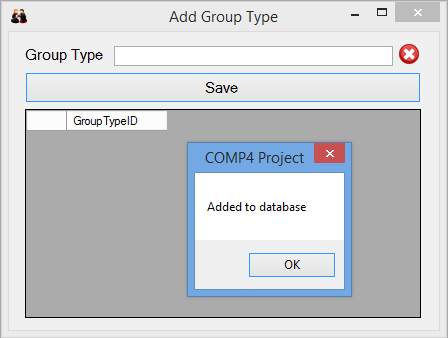
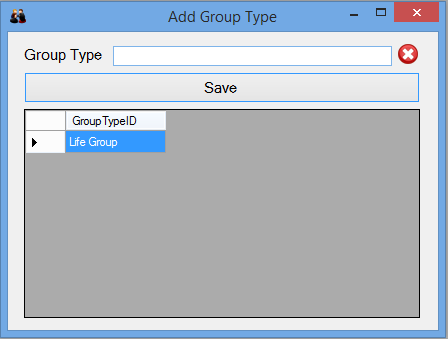
4D

4C

4B

4A

### 5

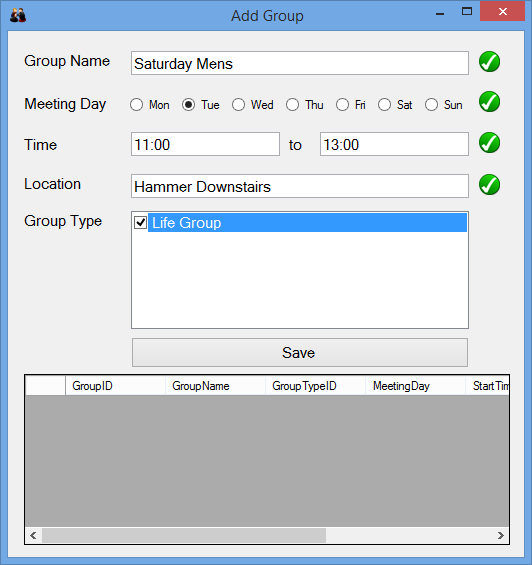
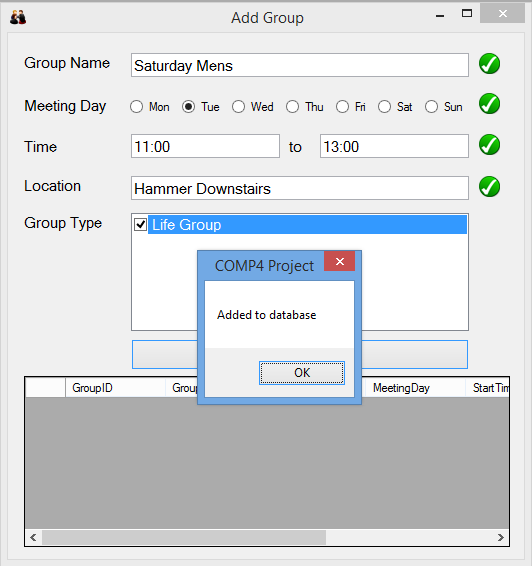
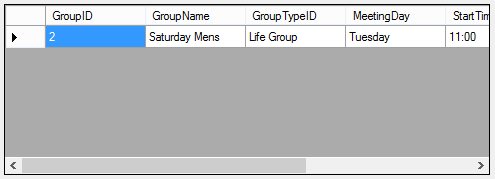
  

5A

5B

5C

### 6

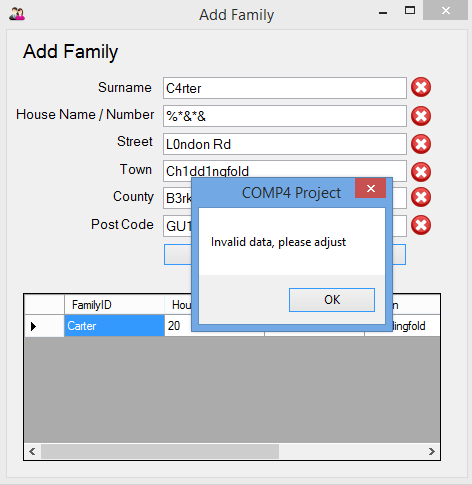
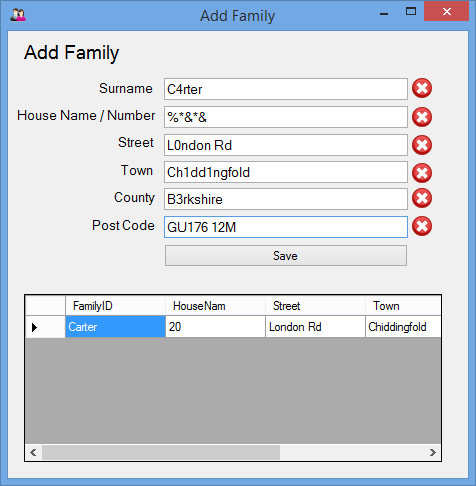
  

6C

6B

6A

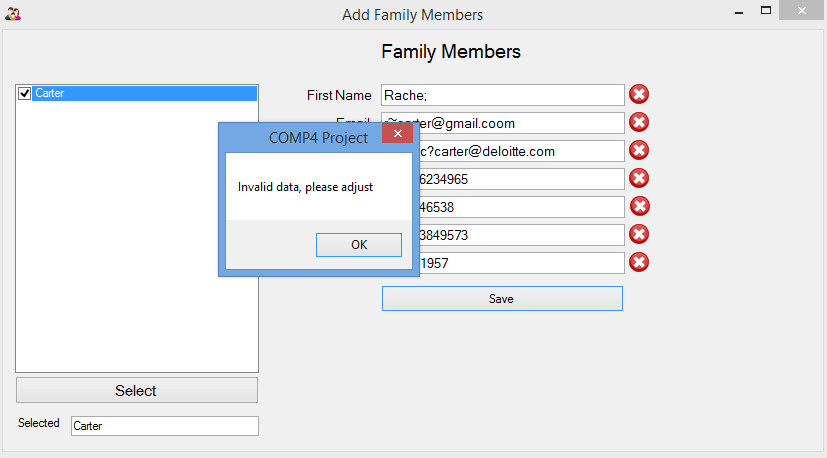
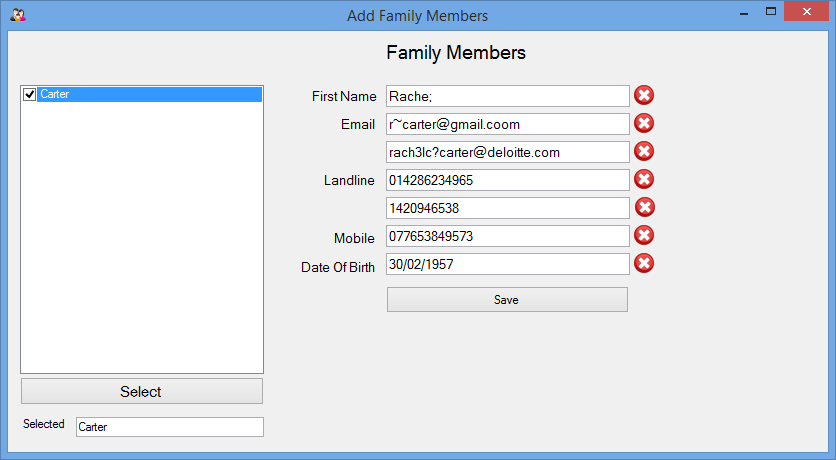
### 7



7A

7B

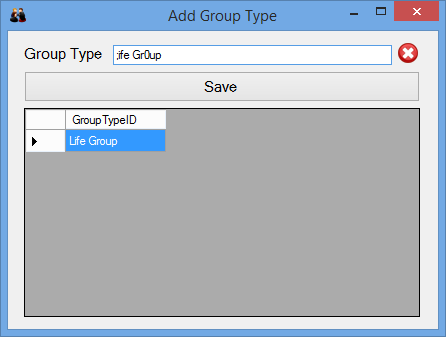
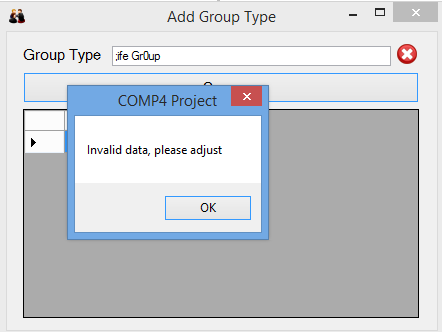
### 8



8A

8B

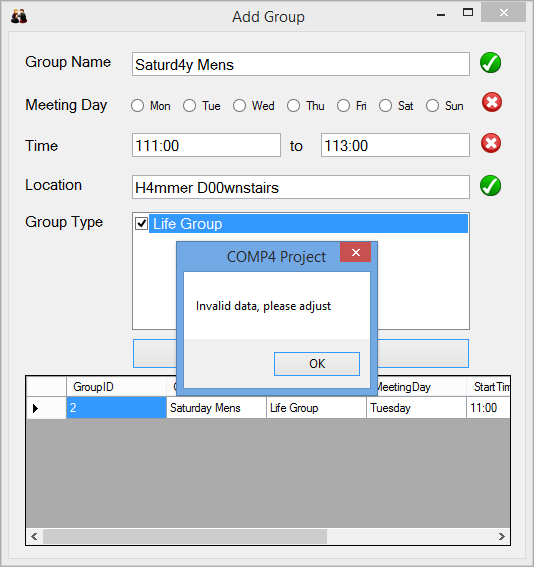
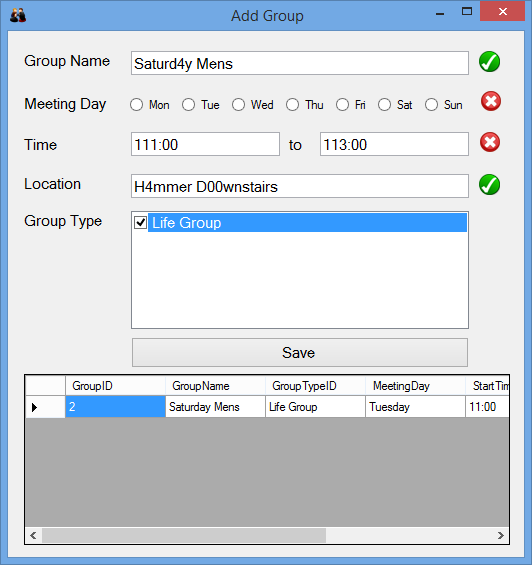
### 9

9A

9B

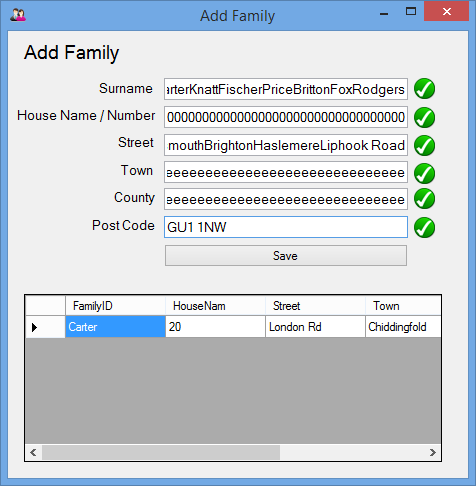
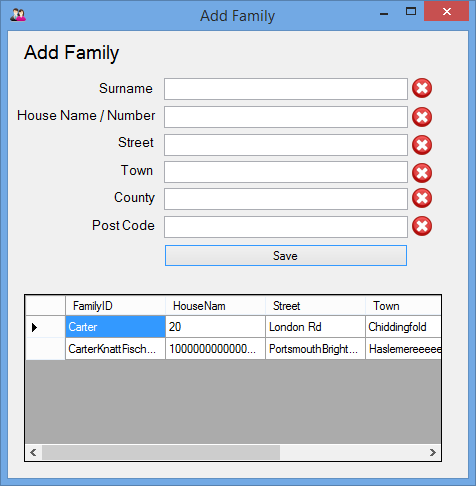
### 10



10B

10A

### 11

11B

11A

### 12

### 

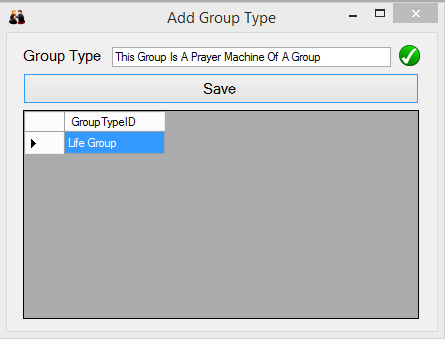
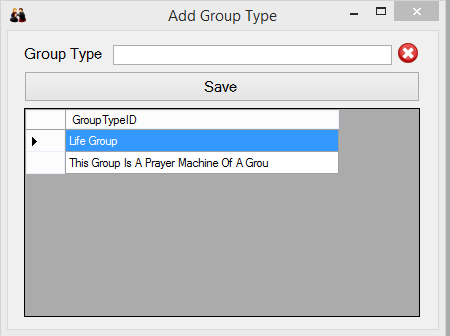
12D

12C

12B

12A

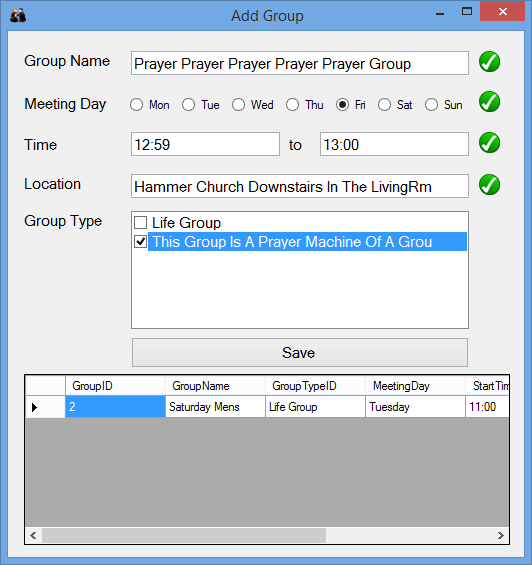
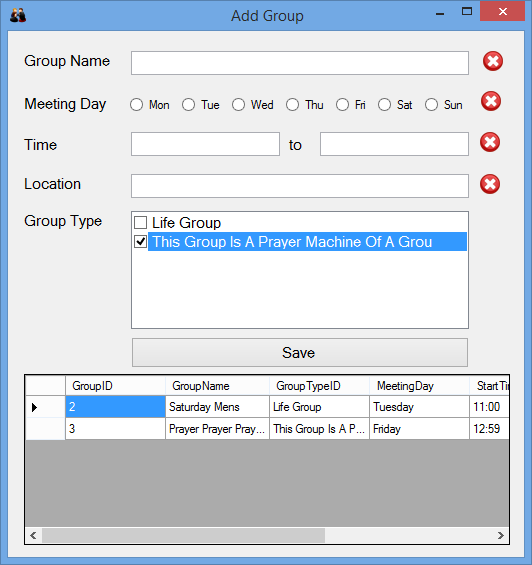
### 13

13A

13B

### 14

14A

14B

### 15

### IMG_9646.jpg

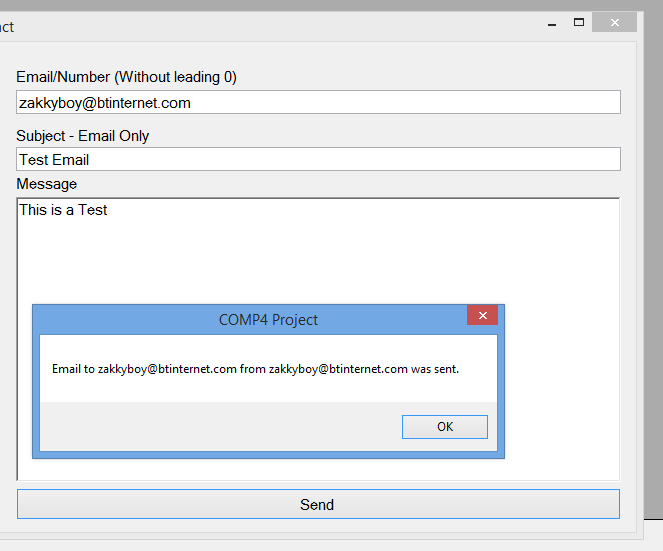
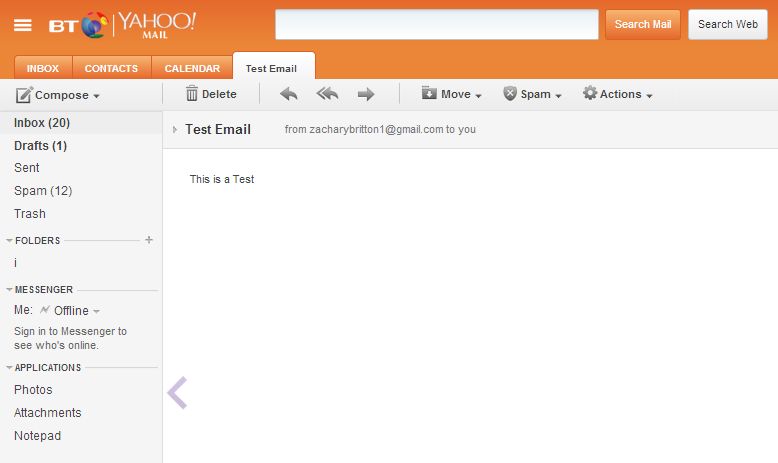
15B

15A

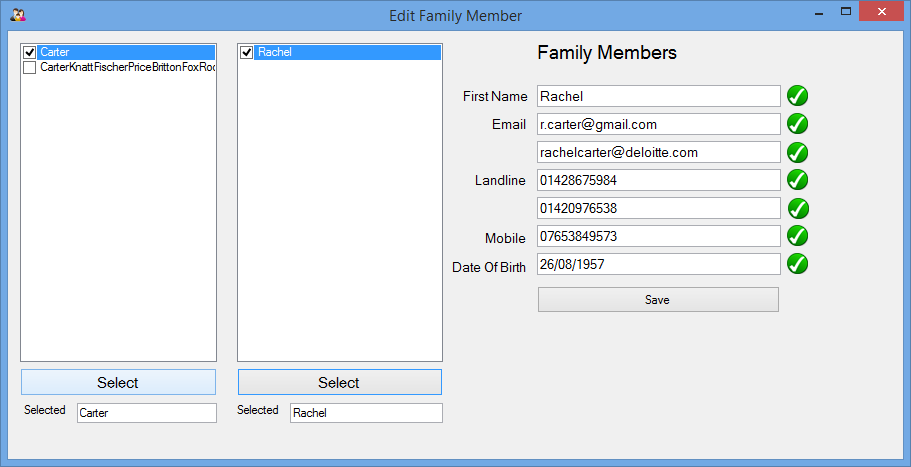
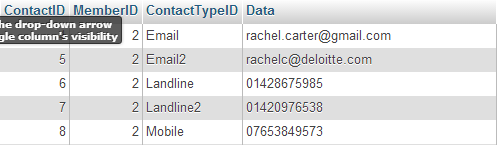
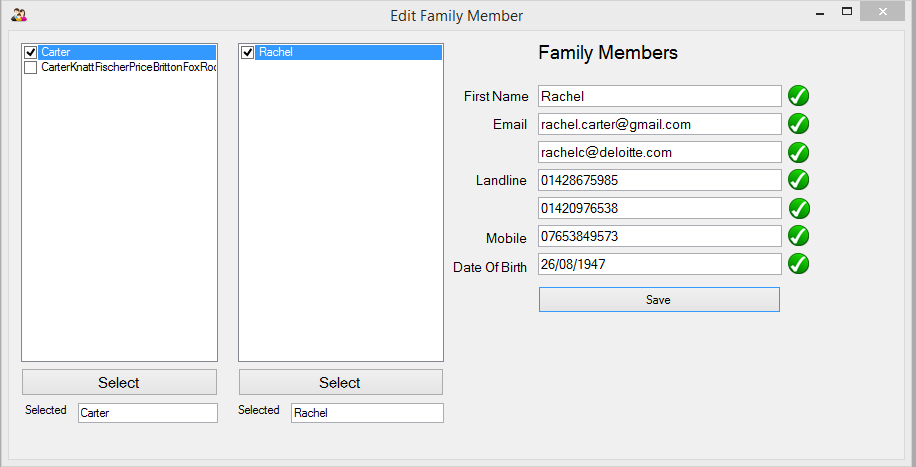
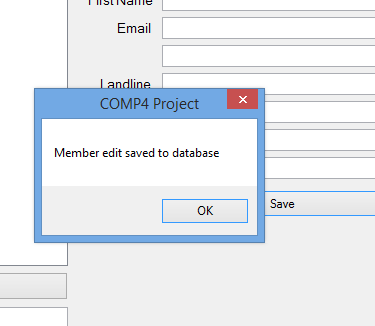
### 16

16A

16B

### 17

17D

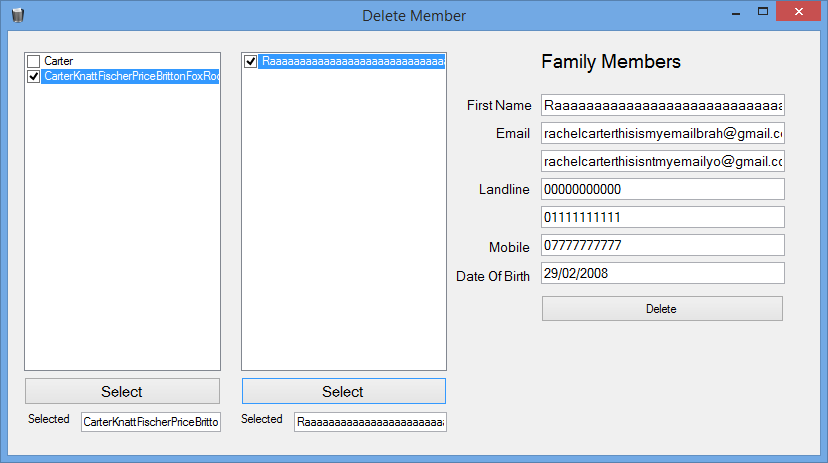
17D

17C

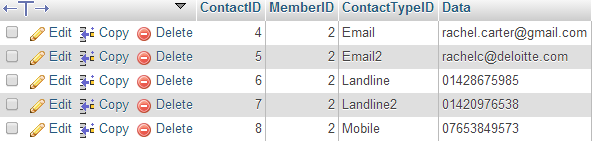
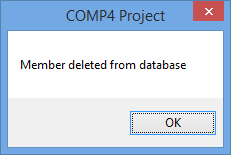
17A

17B

### 18



18A

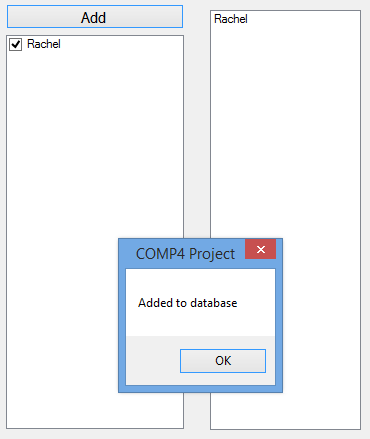
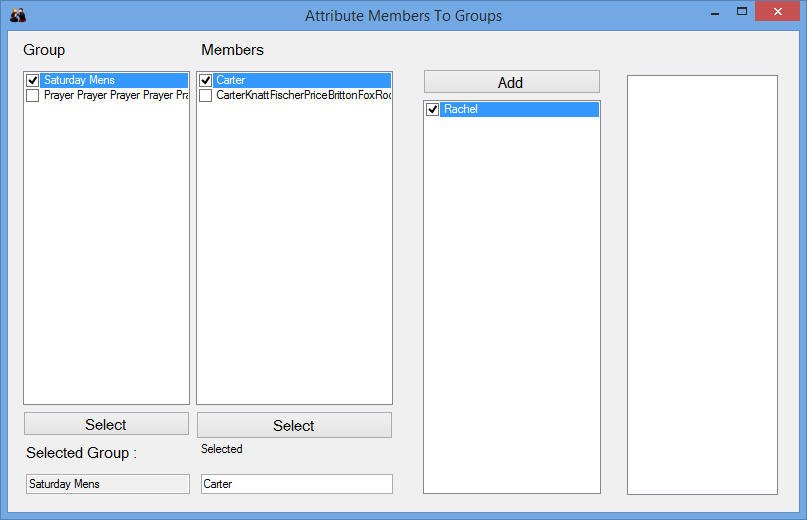


18C

18B

18D

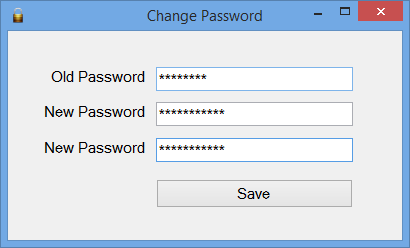
### 19



19B

19A

### 20



20C

20B

20A

## Code

### frmLogin.vb

Imports MySql.Data.MySqlClient

Public Class frmLogin

#Region "DB connection"

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim connection As New MySqlConnection(connectionString)

Dim query As String = "select username,password from LogInDetails"

Dim command As New MySqlCommand(query, connection)

Dim userInfo(0, 1) As String

#End Region

Private Sub frmLogin\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

frmMain.Hide()

txtPassword.Text = ""

txtPassword.PasswordChar = "\*"

Me.AcceptButton = btnLogin

End Sub

Sub getInfo()

Dim dr As MySql.Data.MySqlClient.MySqlDataReader

connection.Open()

dr = command.ExecuteReader()

While dr.Read

userInfo(0, 0) = dr("Username")

userInfo(0, 1) = dr("Password")

End While

connection.Close()

End Sub

Private Sub btnLogin\_Click(sender As System.Object, e As System.EventArgs) Handles btnLogin.Click

getInfo()

If txtUsername.Text = userInfo(0, 0) And txtPassword.Text = userInfo(0, 1) Then

Me.Hide()

frmMain.Show()

Else

MsgBox("Login Failed", , "Login")

txtPassword.Text = ""

End If

End Sub

Private Sub btnChangePassword\_Click(sender As System.Object, e As System.EventArgs) Handles btnChangePassword.Click

frmChangePassword.Show()

End Sub

End Class

### frmChangePassword.vb

Imports MySql.Data.MySqlClient

Public Class frmChangePassword

#Region "DB Connection"

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

#End Region

Dim connection As New MySqlConnection(connectionString)

Dim query1 As String = "select username,password from LogInDetails"

Dim query2 As String = "DELETE from logindetails;"

Dim command1 As New MySqlCommand(query1, connection)

Dim command2 As New MySqlCommand(query2, connection)

Dim userInfo(0, 1) As String

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs) Handles btnSave.Click

'Change password

If txtOldPwrd.Text = userInfo(0, 1) And txtNewPwrd.Text = txtNewPwrd2.Text Then

Dim query3 As String

query3 = "INSERT INTO LogInDetails (username, password) VALUES ('" & userInfo(0, 0) & "' , '" & txtNewPwrd.Text & "');"

Dim command3 As New MySqlCommand(query3, connection)

connection.Open()

command2.ExecuteNonQuery()

command3.ExecuteNonQuery()

connection.Close()

txtOldPwrd.Text = ""

txtNewPwrd.Text = ""

txtNewPwrd2.Text = ""

MsgBox("Password Changed")

Me.Hide()

frmLogin.Show()

Else

MsgBox("Current password incorrect or new passwords do not match")

txtOldPwrd.Text = ""

txtNewPwrd.Text = ""

txtNewPwrd2.Text = ""

End If

End Sub

Sub getInfo()

Dim dr As MySql.Data.MySqlClient.MySqlDataReader

connection.Open()

dr = command1.ExecuteReader()

While dr.Read

userInfo(0, 0) = dr("Username")

userInfo(0, 1) = dr("Password")

End While

connection.Close()

End Sub

Private Sub frmChangePassword\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

frmLogin.Hide()

txtOldPwrd.Text = ""

txtNewPwrd.Text = ""

txtNewPwrd2.Text = ""

txtOldPwrd.PasswordChar = "\*"

txtNewPwrd.PasswordChar = "\*"

txtNewPwrd2.PasswordChar = "\*"

getInfo()

End Sub

End Class

### frmMainMenu.vb

Imports MySql.Data.MySqlClient

Imports System.Text.RegularExpressions

Imports Microsoft.Office.Interop

Public Class frmMain

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

Sub CreateDatabase()

Try

Connection.Open()

Dim cmd As MySqlCommand

Dim sbCmd As String

sbCmd = "CREATE TABLE IF NOT EXISTS Member (MemberID Int(4) PRIMARY KEY AUTO\_INCREMENT, FirNam Varchar(40), SurNam Varchar(40), DOB Varchar(10), FamilyID Varchar(40));" & \_

"CREATE TABLE IF NOT EXISTS Family (FamilyID Varchar(40) PRIMARY KEY, HouseNam Varchar(40), Street Varchar(40), Town Varchar(40), County Varchar(40), PostCode Varchar(12));" & \_

"CREATE TABLE IF NOT EXISTS Contact (ContactID Int(5) PRIMARY KEY AUTO\_INCREMENT, MemberID Int(4), ContactTypeID Varchar(40), Data Varchar(40));" & \_

"CREATE TABLE IF NOT EXISTS ContactType (ContactTypeID Varchar(40) PRIMARY KEY);" & \_

"CREATE TABLE IF NOT EXISTS GroupType (GroupTypeID Varchar(40) PRIMARY KEY);" & \_

"CREATE TABLE IF NOT EXISTS Groups (GroupID Int(4) PRIMARY KEY AUTO\_INCREMENT, GroupName Varchar(40), GroupTypeID Varchar(40), MeetingDay Varchar(40), StartTime Varchar(5), EndTime Varchar(5), Location Varchar(40));" & \_

"CREATE TABLE IF NOT EXISTS GroupMember (GroupID Int(4), MemberID Int(4));" & \_

"CREATE TABLE IF NOT EXISTS LogInDetails (username Varchar(40), password Varchar(40));" & \_

"INSERT INTO ContactType (ContactTypeID) VALUES ('Email');" & \_

"INSERT INTO ContactType (ContactTypeID) VALUES ('Email2');" & \_

"INSERT INTO ContactType (ContactTypeID) VALUES ('Landline');" & \_

"INSERT INTO ContactType (ContactTypeID) VALUES ('Landline2');" & \_

"INSERT INTO ContactType (ContactTypeID) VALUES ('Mobile');" & \_

"INSERT INTO LogInDetails (username, password) VALUES ('Admin', 'Password');"

cmd = New MySqlCommand(sbCmd, Connection)

cmd.ExecuteNonQuery()

Connection.Close()

MsgBox("Database Running and Tables Created")

Catch

MsgBox("Database Closed, or Already Set Up")

End Try

End Sub

Private Sub frmMain\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

'frmLogin.Show()

Me.CreateDatabase()

Me.DataGridView()

End Sub

Sub DataGridView()

Try

Dim query As String = "SELECT MemberID, FirNam, SurNam, DOB from Member;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim ds As New DataSet()

Connection.Close()

Connection.Open()

adp.Fill(ds, "Main\_Table")

Connection.Close()

dgvMain.DataSource = ds

dgvMain.DataMember = "Main\_Table"

Catch

MsgBox("Database Error")

Me.Close()

End Try

End Sub

Private Sub btnAdd\_Click(sender As System.Object, e As System.EventArgs) Handles btnAddFamily.Click

frmAddFamily.Show()

End Sub

Private Sub btnEdit\_Click(sender As System.Object, e As System.EventArgs) Handles btnEditFamilyMembers.Click

frmEditFamilyMember.Show()

End Sub

Private Sub btnAddGroup\_Click(sender As System.Object, e As System.EventArgs) Handles btnAddGroup.Click

frmAddGroup.Show()

End Sub

Private Sub btnCreateDocument\_Click(sender As System.Object, e As System.EventArgs) Handles btnCreateDocument.Click

Dim oWord As Word.Application

Dim oDoc As Word.Document

Dim oPara1 As Word.Paragraph, oPara2 As Word.Paragraph

'Start Word and open the document template.

oWord = CreateObject("Word.Application")

oWord.Visible = True

oDoc = oWord.Documents.Add

'Insert title at the beginning of the document, taking year from computer

Dim Year As String

Year = Date.Today.Year

oPara1 = oDoc.Content.Paragraphs.Add

oPara1.Range.Text = "3CC Contact Book " & Year

oPara1.Range.ParagraphFormat.Alignment = Word.WdParagraphAlignment.wdAlignParagraphCenter

oPara1.Range.Font.Size = 14

oPara1.Range.Font.Bold = True

oPara1.Format.SpaceAfter = 24 '24 pt spacing after paragraph.

oPara1.Range.InsertParagraphAfter()

Dim query As String = "SELECT MemberID, FirNam, SurNam, DOB from Member;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim ds As New DataSet()

Connection.Open()

adp.Fill(ds)

Dim str As String = ds.Tables(0).Rows(0).Item(0).ToString()

Connection.Close()

'Insert another paragraph.

oPara2 = oDoc.Content.Paragraphs.Add(oDoc.Bookmarks.Item("\endofdoc").Range)

oPara2.Range.Text = str

oPara2.Range.Font.Bold = False

oPara2.Format.SpaceAfter = 24

oPara2.Range.InsertParagraphAfter()

End Sub

Private Sub btnRefresh\_Click(sender As System.Object, e As System.EventArgs) Handles btnRefresh.Click

Me.DataGridView()

End Sub

Private Sub btnContact\_Click(sender As System.Object, e As System.EventArgs) Handles btnContact.Click

frmContact.Show()

End Sub

Private Sub btnAddFamilyMembers\_Click(sender As System.Object, e As System.EventArgs) Handles btnAddFamilyMembers.Click

frmAddFamilyMembers.Show()

End Sub

Private Sub btnGroupType\_Click(sender As System.Object, e As System.EventArgs) Handles btnGroupType.Click

frmAddGroupType.Show()

End Sub

Private Sub btnAddGroupMembers\_Click(sender As System.Object, e As System.EventArgs) Handles btnAddGroupMembers.Click

frmMemberGroup.Show()

End Sub

Private Sub btnDeleteMember\_Click(sender As System.Object, e As System.EventArgs) Handles btnDeleteMember.Click

frmDeleteMember.Show()

End Sub

End Class

### frmAddFamily.vb

Imports System.Text.RegularExpressions

Imports MySql.Data.MySqlClient

Public Class frmAddFamily

Private Sub frmAddMember\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.DataGridView()

End Sub

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

#Region "RegEx Add Member"

'Validation for all text input

Private regName = New Regex("^[-‘ A-z]+$"), regHouse = New Regex("^[- A-z\d]+$"), \_

regPost = New Regex("^(GIR ?0AA|[A-PR-UWYZ]([0-9]{1,2}|([A-HK-Y][0-9]([0-9ABEHMNPRV-Y])?)|[0-9][A-HJKPS-UW]) ?[0-9][ABD-HJLNP-UW-Z]{2})$")

'Surname (text change sub run)

Private Sub txtSurname\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtSurname.TextChanged

Me.SurnameValidation()

End Sub

'House Name (text change sub run)

Private Sub txtHouseName\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtHouseName.TextChanged

Me.HouseNameValidation()

End Sub

'Street (text change sub run)

Private Sub txtStreet\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtStreet.TextChanged

Me.StreetValidation()

End Sub

'Town (text change sub run)

Private Sub txtTown\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtTown.TextChanged

Me.TownValidation()

End Sub

'County (text change sub run)

Private Sub txtCounty\_TextChanged\_1(sender As System.Object, e As System.EventArgs) Handles txtCounty.TextChanged

Me.CountyValidation()

End Sub

'Post Code (text change sub run)

Private Sub txtPostCode\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtPostCode.TextChanged

Me.PostCodeValidation()

End Sub

Sub SurnameValidation()

Dim IsMatch As Boolean = regName.IsMatch(txtSurname.Text)

If IsMatch Then

SurNamCross.Visible = False

SurNamTick.Visible = True

Else

SurNamTick.Visible = False

SurNamCross.Visible = True

End If

End Sub

Sub HouseNameValidation()

Dim IsMatch As Boolean = regHouse.IsMatch(txtHouseName.Text)

If IsMatch Then

HouseCross.Visible = False

HouseTick.Visible = True

Else

HouseTick.Visible = False

HouseCross.Visible = True

End If

End Sub

Sub StreetValidation()

Dim IsMatch As Boolean = regName.IsMatch(txtStreet.Text)

If IsMatch Then

StreetCross.Visible = False

StreetTick.Visible = True

Else

StreetTick.Visible = False

StreetCross.Visible = True

End If

End Sub

Sub TownValidation()

Dim IsMatch As Boolean = regName.IsMatch(txtTown.Text)

If IsMatch Then

TownCross.Visible = False

TownTick.Visible = True

Else

TownTick.Visible = False

TownCross.Visible = True

End If

End Sub

Sub CountyValidation()

Dim IsMatch As Boolean = regName.IsMatch(txtCounty.Text)

If IsMatch Then

CountyCross.Visible = False

CountyTick.Visible = True

Else

CountyTick.Visible = False

CountyCross.Visible = True

End If

End Sub

Sub PostCodeValidation()

Dim IsMatch As Boolean = regPost.IsMatch(txtPostCode.Text)

If IsMatch Then

PostCross.Visible = False

PostTick.Visible = True

Else

PostTick.Visible = False

PostCross.Visible = True

End If

End Sub

#End Region

#Region "Save and clear data"

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs) Handles btnSave.Click

'Checking to see if sure

If MsgBox("Are You Sure", MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then

'Ensure all data is valid then save

If SurNamTick.Visible = True And HouseTick.Visible = True And StreetTick.Visible = True And TownTick.Visible = True And CountyTick.Visible = True And PostTick.Visible = True Then

Try

'Add to database

Connection.Open()

Dim SqlCommand As String

SqlCommand = "INSERT INTO Family (FamilyID, HouseNam, Street, Town, County, PostCode) VALUES ('" & txtSurname.Text & "','" & txtHouseName.Text & "', '" & txtStreet.Text & "', '" & txtTown.Text & "','" & txtCounty.Text & "','" & txtPostCode.Text & "');"

Dim Command As New MySqlCommand(SqlCommand, Connection)

Command.ExecuteNonQuery()

Connection.Close()

'Clear Text

txtSurname.Text = ""

txtHouseName.Text = ""

txtStreet.Text = ""

txtTown.Text = ""

txtCounty.Text = ""

txtPostCode.Text = ""

'Notify user that data saved

MsgBox("Family added to database")

Me.DataGridView()

Catch

MsgBox("Data Connection Failed")

End Try

Else

MsgBox("Invalid data, please adjust")

End If

End If

End Sub

#End Region

'Data grid view for all data entered, runs at start and when new data is entered

'Visual display to show what is already there to prevent re-entries

Sub DataGridView()

Dim query As String = "SELECT \* from Family;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim ds As New DataSet()

Connection.Open()

adp.Fill(ds, "Family\_Table")

Connection.Close()

dgvFamily.DataSource = ds

dgvFamily.DataMember = "Family\_Table"

End Sub

End Class

### frmAddFamilyMembers.vb

Imports System.Text.RegularExpressions

Imports MySql.Data.MySqlClient

Public Class frmAddFamilyMembers

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

Private Sub frmAddFamilyMembers\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.FillFamilyCheckBox()

End Sub

#Region "Regex"

Private regName = New Regex("^[-‘ A-z]+$"), \_

regEmail = New Regex("^[A-z0-9.\_%+-]+@[A-z0-9.-]+\.[A-z]{2,4}$"), regNumber = New Regex("^0[0-9]{10}$"), \_

regDOB = New Regex("^(((0[1-9]|[12]\d|3[01])\/(0[13578]|1[02])\/((19|[2-9]\d)\d{2}))|((0[1-9]|[12]\d|30)\/(0[13456789]|1[012])\/((19|[2-9]\d)\d{2}))|((0[1-9]|1\d|2[0-8])\/02\/((19|[2-9]\d)\d{2}))|(29\/02\/((1[6-9]|[2-9]\d)(0[48]|[2468][048]|[13579][26])|((16|[2468][048]|[3579][26])00))))$")

Private Sub txtFirstName\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtFirstName.TextChanged

Me.FirstNameValidation()

End Sub

Private Sub txtEmail\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtEmail.TextChanged

Me.EmailValidation()

End Sub

Private Sub txtEmail2\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtEmail2.TextChanged

Me.EmailValidation2()

End Sub

Private Sub txtLandline\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtLandline.TextChanged

Me.LandlineValidation()

End Sub

Private Sub txtLandline2\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtLandline2.TextChanged

Me.LandlineValidation2()

End Sub

Private Sub txtMobile\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtMobile.TextChanged

Me.MobileValidation()

End Sub

Private Sub txtDOB\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtDOB.TextChanged

Me.DOBValidation()

End Sub

'Validation subd using regex and the .visible operator on tick and cross picture boxes to display result

Sub FirstNameValidation()

Dim IsMatch As Boolean = regName.IsMatch(txtFirstName.Text)

If IsMatch Then

FirNamCross.Visible = False

FirNamTick.Visible = True

Else

FirNamTick.Visible = False

FirNamCross.Visible = True

End If

End Sub

Sub EmailValidation()

Dim IsMatch As Boolean = regEmail.IsMatch(txtEmail.Text)

If IsMatch Then

EmailCross.Visible = False

EmailTick.Visible = True

Else

EmailTick.Visible = False

EmailCross.Visible = True

End If

End Sub

Sub EmailValidation2()

Dim IsMatch As Boolean = regEmail.IsMatch(txtEmail2.Text)

If IsMatch Then

Email2Cross.Visible = False

Email2Tick.Visible = True

Else

Email2Tick.Visible = False

Email2Cross.Visible = True

End If

End Sub

Sub LandlineValidation()

Dim IsMatch As Boolean = regNumber.IsMatch(txtLandline.Text)

If IsMatch Then

LandlineCross.Visible = False

LandlineTick.Visible = True

Else

LandlineTick.Visible = False

LandlineCross.Visible = True

End If

End Sub

Sub LandlineValidation2()

Dim IsMatch As Boolean = regNumber.IsMatch(txtLandline2.Text)

If IsMatch Then

Landline2Cross.Visible = False

Landline2Tick.Visible = True

Else

Landline2Tick.Visible = False

Landline2Cross.Visible = True

End If

End Sub

Sub MobileValidation()

Dim IsMatch As Boolean = regNumber.IsMatch(txtMobile.Text)

If IsMatch Then

MobileCross.Visible = False

MobileTick.Visible = True

Else

MobileTick.Visible = False

MobileCross.Visible = True

End If

End Sub

Sub DOBValidation()

Dim IsMatch As Boolean = regDOB.IsMatch(txtDOB.Text)

If IsMatch Then

DOBCross.Visible = False

DOBTick.Visible = True

Else

DOBTick.Visible = False

DOBCross.Visible = True

End If

End Sub

#End Region

Sub FillFamilyCheckBox()

'Fill checked list box

Dim query As String = "SELECT FamilyID from Family;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamily.DisplayMember = "clbFamily"

clbFamily.ValueMember = "clbFamily"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamily.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs) Handles btnSave.Click

'Checking to see if sure

If MsgBox("Are You Sure", MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then

'Ensure all data is valid then save

If FirNamTick.Visible = True And DOBTick.Visible = True Then

Try

'Add to database member

Connection.Open()

Dim SqlCommand As String

SqlCommand = "INSERT INTO Member (FirNam, SurNam, DOB, FamilyID) VALUES ('" & txtFirstName.Text & "','" & SelectedSurname.Text & "', '" & txtDOB.Text & "', '" & SelectedSurname.Text & "');"

Dim Command As New MySqlCommand(SqlCommand, Connection)

Command.ExecuteNonQuery()

Connection.Close()

'Find Member ID for contact

Dim MemID As Integer

Dim dr As MySql.Data.MySqlClient.MySqlDataReader

Dim SqlCommand7 As String = "SELECT MemberID FROM Member WHERE FirNam = '" & txtFirstName.Text & "' AND SurNam = '" & SelectedSurname.Text & "'"

Dim Command7 As New MySqlCommand(SqlCommand7, Connection)

Connection.Open()

dr = Command7.ExecuteReader()

While dr.Read

MemID = dr("MemberID")

End While

Connection.Close()

'Add to database contact

Connection.Open()

If EmailTick.Visible = True And txtEmail.Text <> "" Then

Dim SqlCommand2 As String

SqlCommand2 = "INSERT INTO Contact (MemberID, ContactTypeID, Data) VALUES ('" & MemID & "', 'Email', '" & txtEmail.Text & "');"

Dim Command2 As New MySqlCommand(SqlCommand2, Connection)

Command2.ExecuteNonQuery()

End If

If Email2Tick.Visible = True And txtEmail2.Text <> "" Then

Dim SqlCommand3 As String

SqlCommand3 = "INSERT INTO Contact (MemberID, ContactTypeID, Data) VALUES ('" & MemID & "', 'Email2', '" & txtEmail2.Text & "');"

Dim Command3 As New MySqlCommand(SqlCommand3, Connection)

Command3.ExecuteNonQuery()

End If

If LandlineTick.Visible = True And txtLandline.Text <> "" Then

Dim SqlCommand4 As String

SqlCommand4 = "INSERT INTO Contact (MemberID, ContactTypeID, Data) VALUES ('" & MemID & "', 'Landline', '" & txtLandline.Text & "');"

Dim Command4 As New MySqlCommand(SqlCommand4, Connection)

Command4.ExecuteNonQuery()

End If

If Landline2Tick.Visible = True And txtLandline2.Text <> "" Then

Dim SqlCommand5 As String

SqlCommand5 = "INSERT INTO Contact (MemberID, ContactTypeID, Data) VALUES ('" & MemID & "', 'Landline2', '" & txtLandline2.Text & "');"

Dim Command5 As New MySqlCommand(SqlCommand5, Connection)

Command5.ExecuteNonQuery()

End If

If MobileTick.Visible = True And txtMobile.Text <> "" Then

Dim SqlCommand6 As String

SqlCommand6 = "INSERT INTO Contact (MemberID, ContactTypeID, Data) VALUES ('" & MemID & "', 'Mobile', '" & txtMobile.Text & "');"

Dim Command6 As New MySqlCommand(SqlCommand6, Connection)

Command6.ExecuteNonQuery()

End If

Connection.Close()

'Clear Text

txtFirstName.Text = ""

SelectedSurname.Text = ""

txtEmail.Text = ""

txtEmail2.Text = ""

txtLandline.Text = ""

txtLandline2.Text = ""

txtMobile.Text = ""

txtDOB.Text = ""

'Notify user that data saved

MsgBox("Family added to database")

Catch

MsgBox("Data Connection Failed")

End Try

Else

MsgBox("Invalid data, please adjust")

End If

End If

End Sub

Private Sub btnSelect\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelect.Click

Dim Checked As String = clbFamily.SelectedItem.ToString()

SelectedSurname.Text = Checked

End Sub

End Class

### frmDeleteMember.vb

Imports MySql.Data.MySqlClient

Public Class frmDeleteMember

Dim MemberInfo(0, 3) As String

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

Sub FillFamilyListBox()

'Fill checked list box

Dim query As String = "SELECT FamilyID from Family;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamily.DisplayMember = "clbFamily"

clbFamily.ValueMember = "clbFamily"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamily.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Sub FillFamilyMembersListBox()

'Fill checked list box

Dim query As String = "SELECT FirNam FROM Member WHERE FamilyID = '" & SelectedFamily.Text & "';"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamilyMembers.DisplayMember = "clbFamilyMembers"

clbFamilyMembers.ValueMember = "clbFamilyMembers"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamilyMembers.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Sub FillText()

Dim dr As MySql.Data.MySqlClient.MySqlDataReader

Dim query As String = "Select MemberID, FirNam, SurNam, DOB FROM Member WHERE FirNam = '" & SelectedMember.Text & "' AND FamilyID = '" & SelectedFamily.Text & "'"

Dim command As New MySqlCommand(query, Connection)

Connection.Open()

dr = command.ExecuteReader()

While dr.Read

MemberInfo(0, 0) = dr("MemberID")

MemberInfo(0, 1) = dr("FirNam")

MemberInfo(0, 2) = dr("SurNam")

MemberInfo(0, 3) = dr("DOB")

End While

Connection.Close()

Dim dr2 As MySql.Data.MySqlClient.MySqlDataReader

Dim query2 As String = "SELECT Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email'"

Dim command2 As New MySqlCommand(query2, Connection)

Dim MemberEmail As String

Connection.Open()

dr2 = command2.ExecuteReader()

While dr2.Read

MemberEmail = dr2("Data")

End While

Connection.Close()

Dim dr3 As MySql.Data.MySqlClient.MySqlDataReader

Dim query3 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email2'"

Dim command3 As New MySqlCommand(query3, Connection)

Dim MemberEmail2 As String

Connection.Open()

dr3 = command3.ExecuteReader()

While dr3.Read

MemberEmail2 = dr3("Data")

End While

Connection.Close()

Dim dr4 As MySql.Data.MySqlClient.MySqlDataReader

Dim query4 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline'"

Dim command4 As New MySqlCommand(query4, Connection)

Dim MemberLandline As String

Connection.Open()

dr4 = command4.ExecuteReader()

While dr4.Read

MemberLandline = dr4("Data")

End While

Connection.Close()

Dim dr5 As MySql.Data.MySqlClient.MySqlDataReader

Dim query5 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline2'"

Dim command5 As New MySqlCommand(query5, Connection)

Dim MemberLandline2 As String

Connection.Open()

dr5 = command5.ExecuteReader()

While dr5.Read

MemberLandline2 = dr5("Data")

End While

Connection.Close()

Dim dr6 As MySql.Data.MySqlClient.MySqlDataReader

Dim query6 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Mobile'"

Dim command6 As New MySqlCommand(query6, Connection)

Dim MemberMobile As String

Connection.Open()

dr6 = command6.ExecuteReader()

While dr6.Read

MemberMobile = dr6("Data")

End While

Connection.Close()

txtFirstName.Text = MemberInfo(0, 1)

txtEmail.Text = MemberEmail

txtEmail2.Text = MemberEmail2

txtLandline.Text = MemberLandline

txtLandline2.Text = MemberLandline2

txtMobile.Text = MemberMobile

txtDOB.Text = MemberInfo(0, 3)

End Sub

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs) Handles btnSave.Click

'Checking to see if sure

If MsgBox("Are You Sure", MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then

'Ensure all data is valid then save

Try

'Add to database member

Connection.Open()

Dim SqlCommand As String

SqlCommand = "DELETE FROM Member WHERE MemberID = '" & MemberInfo(0, 0) & "' AND FamilyID = '" & MemberInfo(0, 2) & "'"

Dim Command As New MySqlCommand(SqlCommand, Connection)

Command.ExecuteNonQuery()

Connection.Close()

'Add to database contact

Connection.Open()

Dim SqlCommand2 As String

SqlCommand2 = "DELETE FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "'"

Dim Command2 As New MySqlCommand(SqlCommand2, Connection)

Command2.ExecuteNonQuery()

Connection.Close()

'Clear Text

txtFirstName.Text = ""

SelectedFamily.Text = ""

SelectedMember.Text = ""

txtEmail.Text = ""

txtEmail2.Text = ""

txtLandline.Text = ""

txtLandline2.Text = ""

txtMobile.Text = ""

txtDOB.Text = ""

'Notify user that data saved

MsgBox("Member deleted from database")

clbFamily.ClearSelected()

clbFamilyMembers.ClearSelected()

Me.FillFamilyMembersListBox()

Catch

MsgBox("Data Connection Failed")

End Try

End If

End Sub

Private Sub btnSelectMember\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelectMember.Click

Dim Checked As String = clbFamilyMembers.SelectedItem.ToString()

SelectedMember.Text = Checked

Me.FillText()

End Sub

Private Sub btnSelectFamily\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelectFamily.Click

Dim Checked As String = clbFamily.SelectedItem.ToString()

SelectedFamily.Text = Checked

Me.FillFamilyMembersListBox()

End Sub

Private Sub frmDeleteMember\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.FillFamilyListBox()

End Sub

End Class

### frmEditFamilyMembers.vb

Imports System.Text.RegularExpressions

Imports MySql.Data.MySqlClient

Public Class frmEditFamilyMember

Dim MemberInfo(0, 3) As String

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

Private Sub frmEditFamilyMember\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.FillFamilyListBox()

End Sub

#Region "Regex"

Private regName = New Regex("^[-‘ A-z]+$"), \_

regEmail = New Regex("^[A-z0-9.\_%+-]+@[A-z0-9.-]+\.[A-z]{2,4}$"), regNumber = New Regex("^0[0-9]{10}$"), \_

regDOB = New Regex("^(((0[1-9]|[12]\d|3[01])\/(0[13578]|1[02])\/((19|[2-9]\d)\d{2}))|((0[1-9]|[12]\d|30)\/(0[13456789]|1[012])\/((19|[2-9]\d)\d{2}))|((0[1-9]|1\d|2[0-8])\/02\/((19|[2-9]\d)\d{2}))|(29\/02\/((1[6-9]|[2-9]\d)(0[48]|[2468][048]|[13579][26])|((16|[2468][048]|[3579][26])00))))$")

Private Sub txtFirstName\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtFirstName.TextChanged

Me.FirstNameValidation()

End Sub

Private Sub txtEmail\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtEmail.TextChanged

Me.EmailValidation()

End Sub

Private Sub txtEmail2\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtEmail2.TextChanged

Me.EmailValidation2()

End Sub

Private Sub txtLandline\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtLandline.TextChanged

Me.LandlineValidation()

End Sub

Private Sub txtLandline2\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtLandline2.TextChanged

Me.LandlineValidation2()

End Sub

Private Sub txtMobile\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtMobile.TextChanged

Me.MobileValidation()

End Sub

Private Sub txtDOB\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtDOB.TextChanged

Me.DOBValidation()

End Sub

'Validation subd using regex and the .visible operator on tick and cross picture boxes to display result

Sub FirstNameValidation()

Dim IsMatch As Boolean = regName.IsMatch(txtFirstName.Text)

If IsMatch Then

FirNamCross.Visible = False

FirNamTick.Visible = True

Else

FirNamTick.Visible = False

FirNamCross.Visible = True

End If

End Sub

Sub EmailValidation()

Dim IsMatch As Boolean = regEmail.IsMatch(txtEmail.Text)

If IsMatch Then

EmailCross.Visible = False

EmailTick.Visible = True

Else

EmailTick.Visible = False

EmailCross.Visible = True

End If

End Sub

Sub EmailValidation2()

Dim IsMatch As Boolean = regEmail.IsMatch(txtEmail2.Text)

If IsMatch Then

Email2Cross.Visible = False

Email2Tick.Visible = True

Else

Email2Tick.Visible = False

Email2Cross.Visible = True

End If

End Sub

Sub LandlineValidation()

Dim IsMatch As Boolean = regNumber.IsMatch(txtLandline.Text)

If IsMatch Then

LandlineCross.Visible = False

LandlineTick.Visible = True

Else

LandlineTick.Visible = False

LandlineCross.Visible = True

End If

End Sub

Sub LandlineValidation2()

Dim IsMatch As Boolean = regNumber.IsMatch(txtLandline.Text)

If IsMatch Then

Landline2Cross.Visible = False

Landline2Tick.Visible = True

Else

Landline2Tick.Visible = False

Landline2Cross.Visible = True

End If

End Sub

Sub MobileValidation()

Dim IsMatch As Boolean = regNumber.IsMatch(txtMobile.Text)

If IsMatch Then

MobileCross.Visible = False

MobileTick.Visible = True

Else

MobileTick.Visible = False

MobileCross.Visible = True

End If

End Sub

Sub DOBValidation()

Dim IsMatch As Boolean = regDOB.IsMatch(txtDOB.Text)

If IsMatch Then

DOBCross.Visible = False

DOBTick.Visible = True

Else

DOBTick.Visible = False

DOBCross.Visible = True

End If

End Sub

#End Region

Sub FillFamilyListBox()

'Fill checked list box

Dim query As String = "SELECT FamilyID from Family;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamily.DisplayMember = "clbFamily"

clbFamily.ValueMember = "clbFamily"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamily.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Sub FillFamilyMembersListBox()

'Fill checked list box

Dim query As String = "SELECT FirNam FROM Member WHERE FamilyID = '" & SelectedFamily.Text & "';"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamilyMembers.DisplayMember = "clbFamilyMembers"

clbFamilyMembers.ValueMember = "clbFamilyMembers"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamilyMembers.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Private Sub btnSelectFamily\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelectFamily.Click

Dim Checked As String = clbFamily.SelectedItem.ToString()

SelectedFamily.Text = Checked

Me.FillFamilyMembersListBox()

End Sub

Private Sub btnSelectMember\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelectMember.Click

Dim Checked As String = clbFamilyMembers.SelectedItem.ToString()

SelectedMember.Text = Checked

Me.FillText()

End Sub

Sub FillText()

Dim dr As MySql.Data.MySqlClient.MySqlDataReader

Dim query As String = "Select MemberID, FirNam, SurNam, DOB FROM Member WHERE FirNam = '" & SelectedMember.Text & "' AND FamilyID = '" & SelectedFamily.Text & "'"

Dim command As New MySqlCommand(query, Connection)

Connection.Open()

dr = command.ExecuteReader()

While dr.Read

MemberInfo(0, 0) = dr("MemberID")

MemberInfo(0, 1) = dr("FirNam")

MemberInfo(0, 2) = dr("SurNam")

MemberInfo(0, 3) = dr("DOB")

End While

Connection.Close()

Dim dr2 As MySql.Data.MySqlClient.MySqlDataReader

Dim query2 As String = "SELECT Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email'"

Dim command2 As New MySqlCommand(query2, Connection)

Dim MemberEmail As String

Connection.Open()

dr2 = command2.ExecuteReader()

While dr2.Read

MemberEmail = dr2("Data")

End While

Connection.Close()

Dim dr3 As MySql.Data.MySqlClient.MySqlDataReader

Dim query3 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email2'"

Dim command3 As New MySqlCommand(query3, Connection)

Dim MemberEmail2 As String

Connection.Open()

dr3 = command3.ExecuteReader()

While dr3.Read

MemberEmail2 = dr3("Data")

End While

Connection.Close()

Dim dr4 As MySql.Data.MySqlClient.MySqlDataReader

Dim query4 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline'"

Dim command4 As New MySqlCommand(query4, Connection)

Dim MemberLandline As String

Connection.Open()

dr4 = command4.ExecuteReader()

While dr4.Read

MemberLandline = dr4("Data")

End While

Connection.Close()

Dim dr5 As MySql.Data.MySqlClient.MySqlDataReader

Dim query5 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline2'"

Dim command5 As New MySqlCommand(query5, Connection)

Dim MemberLandline2 As String

Connection.Open()

dr5 = command5.ExecuteReader()

While dr5.Read

MemberLandline2 = dr5("Data")

End While

Connection.Close()

Dim dr6 As MySql.Data.MySqlClient.MySqlDataReader

Dim query6 As String = "Select Data FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Mobile'"

Dim command6 As New MySqlCommand(query6, Connection)

Dim MemberMobile As String

Connection.Open()

dr6 = command6.ExecuteReader()

While dr6.Read

MemberMobile = dr6("Data")

End While

Connection.Close()

txtFirstName.Text = MemberInfo(0, 1)

txtEmail.Text = MemberEmail

txtEmail2.Text = MemberEmail2

txtLandline.Text = MemberLandline

txtLandline2.Text = MemberLandline2

txtMobile.Text = MemberMobile

txtDOB.Text = MemberInfo(0, 3)

End Sub

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs) Handles btnSave.Click

'Checking to see if sure

If MsgBox("Are You Sure", MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then

'Ensure all data is valid then save

If FirNamTick.Visible = True And DOBTick.Visible = True Then

Try

'Add to database member

Connection.Open()

Dim SqlCommand As String

SqlCommand = "UPDATE Member SET FirNam = '" & txtFirstName.Text & "', DOB = '" & txtDOB.Text & "' WHERE MemberID = '" & MemberInfo(0, 0) & "' AND FamilyID = '" & MemberInfo(0, 2) & "'"

Dim Command As New MySqlCommand(SqlCommand, Connection)

Command.ExecuteNonQuery()

Connection.Close()

'Add to database contact

Connection.Open()

If EmailTick.Visible = True And txtEmail.Text <> "" Then

Dim SqlCommand2 As String

SqlCommand2 = "UPDATE Contact SET Data = '" & txtEmail.Text & "' WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email'"

Dim Command2 As New MySqlCommand(SqlCommand2, Connection)

Command2.ExecuteNonQuery()

Else

Dim SqlCommand2 As String

SqlCommand2 = "DELETE FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email'"

Dim Command2 As New MySqlCommand(SqlCommand2, Connection)

Command2.ExecuteNonQuery()

End If

If Email2Tick.Visible = True And txtEmail2.Text <> "" Then

Dim SqlCommand3 As String

SqlCommand3 = "UPDATE Contact SET Data = '" & txtEmail2.Text & "' WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email2'"

Dim Command3 As New MySqlCommand(SqlCommand3, Connection)

Command3.ExecuteNonQuery()

Else

Dim SqlCommand3 As String

SqlCommand3 = "DELETE FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Email2'"

Dim Command3 As New MySqlCommand(SqlCommand3, Connection)

Command3.ExecuteNonQuery()

End If

If LandlineTick.Visible = True And txtLandline.Text <> "" Then

Dim SqlCommand4 As String

SqlCommand4 = "UPDATE Contact SET Data = '" & txtLandline.Text & "' WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline'"

Dim Command4 As New MySqlCommand(SqlCommand4, Connection)

Command4.ExecuteNonQuery()

Else

Dim SqlCommand4 As String

SqlCommand4 = "DELETE FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline'"

Dim Command4 As New MySqlCommand(SqlCommand4, Connection)

Command4.ExecuteNonQuery()

End If

If Landline2Tick.Visible = True And txtLandline2.Text <> "" Then

Dim SqlCommand5 As String

SqlCommand5 = "UPDATE Contact SET Data = '" & txtLandline2.Text & "' WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline2'"

Dim Command5 As New MySqlCommand(SqlCommand5, Connection)

Command5.ExecuteNonQuery()

Else

Dim SqlCommand5 As String

SqlCommand5 = "DELETE FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Landline2"

Dim Command5 As New MySqlCommand(SqlCommand5, Connection)

Command5.ExecuteNonQuery()

End If

If MobileTick.Visible = True And txtMobile.Text <> "" Then

Dim SqlCommand6 As String

SqlCommand6 = "UPDATE Contact SET Data = '" & txtMobile.Text & "' WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Mobile'"

Dim Command6 As New MySqlCommand(SqlCommand6, Connection)

Command6.ExecuteNonQuery()

Else

Dim SqlCommand6 As String

SqlCommand6 = "DELETE FROM Contact WHERE MemberID = '" & MemberInfo(0, 0) & "' AND ContactTypeID = 'Mobile'"

Dim Command6 As New MySqlCommand(SqlCommand6, Connection)

Command6.ExecuteNonQuery()

End If

Connection.Close()

'Clear Text

txtFirstName.Text = ""

SelectedFamily.Text = ""

SelectedMember.Text = ""

txtEmail.Text = ""

txtEmail2.Text = ""

txtLandline.Text = ""

txtLandline2.Text = ""

txtMobile.Text = ""

txtDOB.Text = ""

'Notify user that data saved

MsgBox("Member edit saved to database")

clbFamily.ClearSelected()

clbFamilyMembers.ClearSelected()

Me.FillFamilyMembersListBox()

Catch

MsgBox("Data Connection Failed")

End Try

Else

MsgBox("Invalid data, please adjust")

End If

End If

End Sub

End Class

### frmAddGroupType.vb

Imports System.Text.RegularExpressions

Imports MySql.Data.MySqlClient

Public Class frmAddGroupType

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

#Region "RegEx"

Private regName = New Regex("^[-‘ A-z]+$")

Private Sub frmAddGroupType\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.DataGridView()

End Sub

Private Sub TextBox1\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtGroupType.TextChanged

Me.GroupTypeValidation()

End Sub

Sub GroupTypeValidation()

Dim IsMatch As Boolean = regName.IsMatch(txtGroupType.Text)

If IsMatch Then

GrTypeCross.Visible = False

GrTypeTick.Visible = True

Else

GrTypeTick.Visible = False

GrTypeCross.Visible = True

End If

End Sub

#End Region

Sub DataGridView()

Dim query As String = "SELECT \* from GroupType;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim ds As New DataSet()

Connection.Open()

adp.Fill(ds, "Group\_Type")

Connection.Close()

dgvGroupType.DataSource = ds

dgvGroupType.DataMember = "Group\_Type"

End Sub

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs) Handles btnSave.Click

'Checking to see if sure

If MsgBox("Are You Sure", MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then

'Ensure all data is valid then save

If GrTypeTick.Visible = True Then

Try

'Add to database

Connection.Open()

Dim SqlCommand As String

SqlCommand = "INSERT INTO GroupType (GroupTypeID) VALUES ('" & txtGroupType.Text & "');"

Dim Command As New MySqlCommand(SqlCommand, Connection)

Command.ExecuteNonQuery()

Connection.Close()

'Clear Text

txtGroupType.Text = ""

'Notify user that data saved

MsgBox("Added to database")

Me.DataGridView()

Catch

MsgBox("Data Connection Failed")

End Try

Else

MsgBox("Invalid data, please adjust")

End If

End If

End Sub

End Class

### frmAddGroup.vb

Imports System.Text.RegularExpressions

Imports MySql.Data.MySqlClient

Public Class frmAddGroup

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

#Region "Day validation"

Private Sub rdbutMon\_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles rdbutMon.CheckedChanged

Me.DayValidation()

End Sub

Private Sub rdbutTue\_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles rdbutTue.CheckedChanged

Me.DayValidation()

End Sub

Private Sub rdbutWed\_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles rdbutWed.CheckedChanged

Me.DayValidation()

End Sub

Private Sub rdbutThu\_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles rdbutThu.CheckedChanged

Me.DayValidation()

End Sub

Private Sub rdbutFri\_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles rdbutFri.CheckedChanged

Me.DayValidation()

End Sub

Private Sub drbutSat\_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles rdbutSat.CheckedChanged

Me.DayValidation()

End Sub

Private Sub rdbutSun\_CheckedChanged(sender As System.Object, e As System.EventArgs) Handles rdbutSun.CheckedChanged

Me.DayValidation()

End Sub

Sub DayValidation()

DayTick.Visible = True

DayCross.Visible = False

End Sub

#End Region

#Region "Other Validation"

Private regname As New Regex("^[-‘ A-z0-9]+$"), regtime As New Regex("^[0-9]{2}:([0-9]{2})$")

Private Sub txtGroupName\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtGroupName.TextChanged

Me.GroupNamValidation()

End Sub

Private Sub txtStartTime\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtStartTime.TextChanged

Me.TimeValidation()

End Sub

Private Sub txtEndTime\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtEndTime.TextChanged

Me.TimeValidation()

End Sub

Private Sub txtLocation\_TextChanged(sender As System.Object, e As System.EventArgs) Handles txtLocation.TextChanged

Me.LocationValidation()

End Sub

Sub GroupNamValidation()

Dim IsMatch As Boolean = regname.IsMatch(txtGroupName.Text)

If IsMatch Then

GrNamTick.Visible = True

GrNamCross.Visible = False

Else

GrNamTick.Visible = False

GrNamCross.Visible = True

End If

End Sub

Sub LocationValidation()

Dim IsMatch As Boolean = regname.IsMatch(txtLocation.Text)

If IsMatch Then

LocTick.Visible = True

LocCross.Visible = False

Else

LocTick.Visible = False

LocCross.Visible = True

End If

End Sub

Sub TimeValidation()

Dim IsMatch As Boolean = regtime.IsMatch(txtStartTime.Text)

Dim IsMatch2 As Boolean = regtime.IsMatch(txtEndTime.Text)

If IsMatch And IsMatch2 Then

TimeTick.Visible = True

TimeCross.Visible = False

Else

TimeTick.Visible = False

TimeCross.Visible = True

End If

End Sub

#End Region

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs) Handles btnSave.Click

'Checking to see if sure

If MsgBox("Are You Sure", MsgBoxStyle.YesNo) = MsgBoxResult.Yes Then

If GrNamTick.Visible = True And DayTick.Visible = True And TimeTick.Visible = True And LocTick.Visible = True Then

'Calculate Day

Dim txtMeetingDay As String

If rdbutMon.Checked = True Then

txtMeetingDay = "Monday"

ElseIf rdbutTue.Checked = True Then

txtMeetingDay = "Tuesday"

ElseIf rdbutWed.Checked = True Then

txtMeetingDay = "Wednesday"

ElseIf rdbutThu.Checked = True Then

txtMeetingDay = "Thursday"

ElseIf rdbutFri.Checked = True Then

txtMeetingDay = "Friday"

ElseIf rdbutSat.Checked = True Then

txtMeetingDay = "Saturday"

Else

txtMeetingDay = "Sunday"

End If

'Find Group Type

Dim TypeID As String = clbGroupType.SelectedItem.ToString()

'Save to database

Try

Connection.Open()

Dim SqlCommand As String

SqlCommand = "INSERT INTO groups (groupname, grouptypeID, meetingday, starttime, endtime, location) VALUES ('" & txtGroupName.Text & "', '" & TypeID & "', '" & txtMeetingDay & "','" & txtStartTime.Text & "','" & txtEndTime.Text & "','" & txtLocation.Text & "');"

Dim Command As New MySqlCommand(SqlCommand, Connection)

Command.ExecuteNonQuery()

Connection.Close()

'Notify that it's saved

MsgBox("Added to database")

'Clear Text

txtGroupName.Text = ""

txtStartTime.Text = ""

txtEndTime.Text = ""

txtLocation.Text = ""

rdbutMon.Checked = False

rdbutTue.Checked = False

rdbutWed.Checked = False

rdbutThu.Checked = False

rdbutFri.Checked = False

rdbutSat.Checked = False

rdbutSun.Checked = False

DayTick.Visible = False

DayCross.Visible = True

Me.DataGridView()

Catch

MsgBox("Data Connection Failed")

End Try

Else

MsgBox("Invalid data, please adjust")

End If

End If

End Sub

Private Sub frmAddGroup\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.DataGridView()

Me.FillGroupTypeListBox()

End Sub

Sub DataGridView()

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim connection As New MySqlConnection(connectionString)

Dim query As String = "select \* from groups;"

Dim adp As New MySqlDataAdapter(query, connection)

Dim ds As New DataSet()

connection.Open()

adp.Fill(ds, "Group\_Table")

connection.Close()

DataGridView1.DataSource = ds

DataGridView1.DataMember = "Group\_Table"

End Sub

Sub FillGroupTypeListBox()

'Fill checked list box

Dim query As String = "SELECT \* from GroupType;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbGroupType.DisplayMember = "clbGroupType"

clbGroupType.ValueMember = "clbGroupType"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbGroupType.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

End Class

### frmMemberGroup.vb

Imports MySql.Data.MySqlClient

Public Class frmMemberGroup

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

Private Sub BtnAdd\_Click(sender As System.Object, e As System.EventArgs) Handles BtnAdd.Click

Try

Dim Member As String = clbMemberSelect.SelectedItem.ToString()

lbGroup.Items.Add(Member)

Dim dr2 As MySql.Data.MySqlClient.MySqlDataReader

Dim query2 As String = "SELECT GroupID FROM Groups WHERE GroupName = '" & txtSelectedGroup.Text & "'"

Dim command2 As New MySqlCommand(query2, Connection)

Dim GroupID As String

Connection.Open()

dr2 = command2.ExecuteReader()

While dr2.Read

GroupID = dr2("GroupID")

End While

Connection.Close()

Dim dr As MySql.Data.MySqlClient.MySqlDataReader

Dim query As String = "SELECT MemberID FROM Member WHERE FirNam = '" & Member & "' AND FamilyID = '" & SelectedFamily.Text & "'"

Dim command As New MySqlCommand(query, Connection)

Dim MemberID As String

Connection.Open()

dr = command.ExecuteReader()

While dr.Read

MemberID = dr("MemberID")

End While

Connection.Close()

'Add to database

Connection.Open()

Dim SqlCommand As String

SqlCommand = "INSERT INTO GroupMember (GroupID, MemberID) VALUES ('" & GroupID & "', '" & MemberID & "');"

Dim Command3 As New MySqlCommand(SqlCommand, Connection)

Command3.ExecuteNonQuery()

Connection.Close()

'Clear Text

clbGroupSelect.ClearSelected()

clbFamily.ClearSelected()

clbMemberSelect.ClearSelected()

'Notify user that data saved

MsgBox("Added to database")

Catch

MsgBox("Data Connection Failed")

End Try

End Sub

Sub FillGroupCLB()

'Fill checked list box

Dim query As String = "SELECT GroupName from Groups;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt2 As New DataTable

Connection.Open()

adp.Fill(dt2)

Connection.Close()

clbGroupSelect.DisplayMember = "clbGroup"

clbGroupSelect.ValueMember = "clbGroup"

If dt2.Rows.Count > 0 Then

For i As Integer = 0 To dt2.Rows.Count - 1

clbGroupSelect.Items.Add(CStr(dt2.Rows(i).Item(0)), False)

Next

End If

End Sub

Sub FillMemberCLB()

'Fill checked list box

Dim query As String = "SELECT FirNam FROM Member WHERE FamilyID = '" & SelectedFamily.Text & "';"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbMemberSelect.DisplayMember = "clbFamilyMembers"

clbMemberSelect.ValueMember = "clbFamilyMembers"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbMemberSelect.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Sub FillFamilyCLB()

'Fill checked list box

Dim query As String = "SELECT FamilyID from Family;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamily.DisplayMember = "clbFamily"

clbFamily.ValueMember = "clbFamily"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamily.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Private Sub frmMemberGroup\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.FillGroupCLB()

Me.FillFamilyCLB()

End Sub

Private Sub btnSelect\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelect.Click

Dim Checked As String = clbGroupSelect.SelectedItem.ToString()

txtSelectedGroup.Text = Checked

End Sub

Private Sub btnSave\_Click(sender As System.Object, e As System.EventArgs)

End Sub

Private Sub btnSelectFamily\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelectFamily.Click

Dim Checked As String = clbFamily.SelectedItem.ToString()

SelectedFamily.Text = Checked

Me.FillMemberCLB()

End Sub

End Class

### frmContact.vb

Imports MySql.Data.MySqlClient

Imports System.Net.Mail

Public Class frmContact

#Region "DB Connection"

'Connecting to MySQL databse

Dim connectionString As String = \_

"Database=Computing;" & \_

"Data Source=localhost;" & \_

"User Id=zac;" & \_

"Password=password;"

Dim Connection As New MySqlConnection(connectionString)

#End Region

Private Sub frmContact\_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load

Me.FillFamilyListBox()

End Sub

Function SendEmail(ByVal Recipients As List(Of String), \_

ByVal FromAddress As String, \_

ByVal Subject As String, \_

ByVal Body As String, \_

ByVal UserName As String, \_

ByVal Password As String, \_

Optional ByVal Server As String = "smtp.gmail.com", \_

Optional ByVal Port As Integer = 587, \_

Optional ByVal Attachments As List(Of String) = Nothing) As String

Dim Email As New MailMessage()

'Code for sending email including try, catch and error messages

Try

Dim SMTPServer As New SmtpClient

For Each Attachment As String In Attachments

Email.Attachments.Add(New Attachment(Attachment))

Next

Email.From = New MailAddress(FromAddress)

For Each Recipient As String In Recipients

Email.To.Add(Recipient)

Next

Email.Subject = Subject

Email.Body = Body

SMTPServer.Host = Server

SMTPServer.Port = Port

SMTPServer.Credentials = New System.Net.NetworkCredential(UserName, Password)

SMTPServer.EnableSsl = True

SMTPServer.Send(Email)

Email.Dispose()

Return "Email to " & Recipients(0) & " from " & FromAddress & " was sent."

Catch ex As SmtpException

Email.Dispose()

Return "Sending Email Failed. Smtp Error."

Catch ex As ArgumentOutOfRangeException

Email.Dispose()

Return "Sending Email Failed. Check Port Number."

Catch Ex As InvalidOperationException

Email.Dispose()

Return "Sending Email Failed. Check Port Number."

End Try

End Function

Private Sub btnSend\_Click(sender As System.Object, e As System.EventArgs) Handles btnSend.Click

'Changes send location and syntaz based on whether email or text is selected

If rdbtnEmail.Checked = True Then

'Sends email from Gmail SMTP server

Dim Recipients As New List(Of String)

Recipients.Add(txtEmail.Text)

Dim FromEmailAddress As String = Recipients(0)

Dim Subject As String = txtSubject.Text

Dim Body As String = txtBody.Text

Dim UserName As String = "zacharybritton1"

Dim Password As String = "TheGoogle1"

Dim Port As Integer = 587

Dim Server As String = "smtp.gmail.com"

Dim Attachments As New List(Of String)

MsgBox(SendEmail(Recipients, FromEmailAddress, Subject, Body, UserName, Password, Server, Port, Attachments))

txtSubject.Text = ""

txtBody.Text = ""

txtEmail.Text = ""

Else

'Sends email to clickatell SMTP to SMS server via paid text account, again using Gmail as server

Dim Recipients As New List(Of String)

Recipients.Add("sms@messaging.clickatell.com")

Dim FromEmailAddress As String = Recipients(0)

Dim Subject As String = ""

Dim Body As String = ("api\_id:3460916" & Environment.NewLine & "user:ZacharyBritton1" & Environment.NewLine & "password:TheGoogle1" & Environment.NewLine & "to:44" & txtEmail.Text & Environment.NewLine & "text:" & txtBody.Text)

Dim UserName As String = "zacharybritton1"

Dim Password As String = "TheGoogle1"

Dim Port As Integer = 587

Dim Server As String = "smtp.gmail.com"

Dim Attachments As New List(Of String)

MsgBox(SendEmail(Recipients, FromEmailAddress, Subject, Body, UserName, Password, Server, Port, Attachments))

txtSubject.Text = ""

txtBody.Text = ""

txtEmail.Text = ""

End If

End Sub

Sub FillFamilyListBox()

'Fill checked list box

Dim query As String = "SELECT FamilyID from Family;"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamily.DisplayMember = "clbFamily"

clbFamily.ValueMember = "clbFamily"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamily.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Sub FillFamilyMembersListBox()

'Fill checked list box

Dim query As String = "SELECT FirNam FROM Member WHERE FamilyID = '" & SelectedFamily.Text & "';"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbFamilyMembers.DisplayMember = "clbFamilyMembers"

clbFamilyMembers.ValueMember = "clbFamilyMembers"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbFamilyMembers.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Private Sub btnSelectFamily\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelectFamily.Click

Dim Checked As String = clbFamily.SelectedItem.ToString()

SelectedFamily.Text = Checked

Me.FillFamilyMembersListBox()

End Sub

Private Sub btnSelectMember\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelectMember.Click

Dim Checked As String = clbFamilyMembers.SelectedItem.ToString()

SelectedMember.Text = Checked

Me.FillContact()

End Sub

Sub FillContact()

Dim dr2 As MySql.Data.MySqlClient.MySqlDataReader

Dim query2 As String = "SELECT MemberID FROM Member WHERE FirNam = '" & SelectedMember.Text & "' AND FamilyID = '" & SelectedFamily.Text & "'"

Dim command2 As New MySqlCommand(query2, Connection)

Dim MemberID As String

Connection.Open()

dr2 = command2.ExecuteReader()

While dr2.Read

MemberID = dr2("MemberID")

End While

Connection.Close()

Dim query As String = "SELECT Data FROM Contact WHERE MemberID = '" & MemberID & "';"

Dim adp As New MySqlDataAdapter(query, Connection)

Dim dt1 As New DataTable

Connection.Open()

adp.Fill(dt1)

Connection.Close()

clbContactDetails.DisplayMember = "clbContactDetails"

clbContactDetails.ValueMember = "clbContactDetails"

If dt1.Rows.Count > 0 Then

For i As Integer = 0 To dt1.Rows.Count - 1

clbContactDetails.Items.Add(CStr(dt1.Rows(i).Item(0)), False)

Next

End If

End Sub

Private Sub btnSelect\_Click(sender As System.Object, e As System.EventArgs) Handles btnSelect.Click

Dim Contact As String = clbContactDetails.SelectedItem.ToString()

txtEmail.Text = Contact

End Sub

End Class