**MINISTRY OF EDUCATION AND TRAINING**

**UNIVERSITY OF TECHNOLOGY AND EDUCATION**

# **FACULTY OF INFORMATION TECHNOLOGY**

# **--------------------**

**THE FIRST PROJECT REPORT**

**Build a tool to generate the genealogy tree**

**STUDENT LIST:**

1. **NGUYỄN MẠNH TIẾN 17145370**
2. **LÊ TUẤN ĐẠT 17110019**

**LECTURER: Nguyễn Đức Khoan**

# **Ho Chi Minh City** **– 2019**

**MINISTRY OF EDUCATION AND TRAINING**

**UNIVERSITY OF TECHNOLOGY AND EDUCATION**

# **FACULTY OF INFORMATION TECHNOLOGY**

# **--------------------**

**THE FIRST PROJECT REPORT**

**Build a tool to generate the genealogy tree**

**STUDENT LIST:**

1. **NGUYỄN MẠNH TIẾN 17145370**
2. **LÊ TUẤN ĐẠT 17110019**

**LECTURER: Nguyễn Đức Khoan**

# **Ho Chi Minh City** **– 2019**

**Lecturer's evaluation**

*Ho Chi Minh City, Jun, 2019*

**Lecturer**

# **Thanks**

The success of a student more or less always accompanies a lecturer. We would like to express our sincere thanks to Mr. Nguyen Duc Khoan, who directly supported the group, give us suggestions, comments and suggestions as well as provide tips to help us make the best project. Thanks to his instructions, help our team understand the knowledge to do, the presentation as well as the implementation of the project so we have completed the schedule with a lot of experience that we learn. Again, our team would like to thank the lecturer.

Projects are made within 8 weeks, just enough to complete it, However, due to many new knowledge as well as the time we do through each week is not optimal, the project will have many errors, that is inevitable. We are looking forward to receiving all the comments of our teachers to help our limited knowledge better. Sincerely thanks.

# **Preface**

The purpose and objective of this training and particularly the content is really time-being and with this training we have gained some confidence regarding to introduce the application. We also belief that way we gained some sorts of IT knowledge and if we practice much and having some expertise in the field then we will be able to survive smartly in today’s competitive environment.

The effort to write the report is a partial fulfillment to complete the course. In the report I try my best to represent all the content which we learnt in a great deal in the program in a systematic and presentable order. I divided each of the topics as an individual chapter to reflect the entire topic more prominently and clearly. In the reference I have used citation method in the entire report. Finally, I am very hopeful that the structure and topic of the report will be a useful print material to all the reader especially to the user.

**Content**

Thanks………………………………………………………………...……………...………………..........4

Preface………………………………………………………………………………………………..……..5

Chapter.1……………………………………………………………………………………………..……..7

* 1. What is the software used for …………………………………………………………………7
  2. Data, Information Input………………………………………………………………………..7
  3. Situation (purpose,feature)…………………………………………………………...………..7
  4. Expected interface………………………………………………………………...…………...7

Chapter 2 Assignment of work……………………………………………………..………………………8

Chapter 3 Design………………………………………………………………………..………………….9

3.1 Connect to Database…………………………………...………………………………………9

3.2 Draw the tree………………………………………………………………………………….10

Chapter 4 Design Program

4.1 Each class use in project……………………………………………………………………...11

4.1.1Database Design…………………………………………………………......……..12

4.1.1.1 Table In database……………………………………………....………..12

4.1.1.2 Diagram……………………………………………...…….……………12

4.1.2 Description Field…………………………………………………………...………13

4.1.3 Design Theme……………………………………………………………………...14

4.2 The sample table describes the methods in a class…………………………………………...16

Chapter 5 Installation and testing…………………………………………………………………………25

Chapter 6 Conclude……………………………………………………………………………………….26

6.1 Difficulties and solution……………………………………………………………………...26

6.2 For the further implementation………………………………………………………………26

6.3 Advantage…………………………………………………………………………………….26

6.4 Limit………………………………………………………………………………………….26

6.5 Reference……………………………………………………………………………………..27

**CHAPTER 1. PROJECT DESCRIPTION**

**1.1. What is the software used for ?**

This software used to draw a family tree that links family members. Remembering family members is difficult for large families also genealogy or genealogy is a record of the surname, age, death anniversary, role and merit of parents, grandparents, ancestors and graves of a large family or a clan. so we decided to design family tree software.

**1.2. Data, Information Input**

Family data from SQL

dbo.Member( memID, memName, rustic, gender, dob, dod, address)

dbo.Parentage( pID, pName)

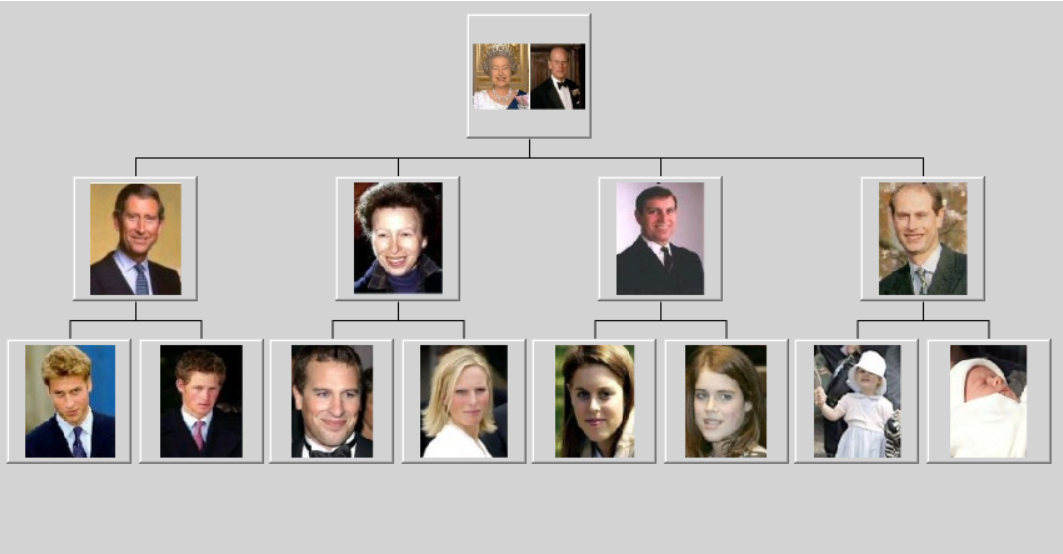
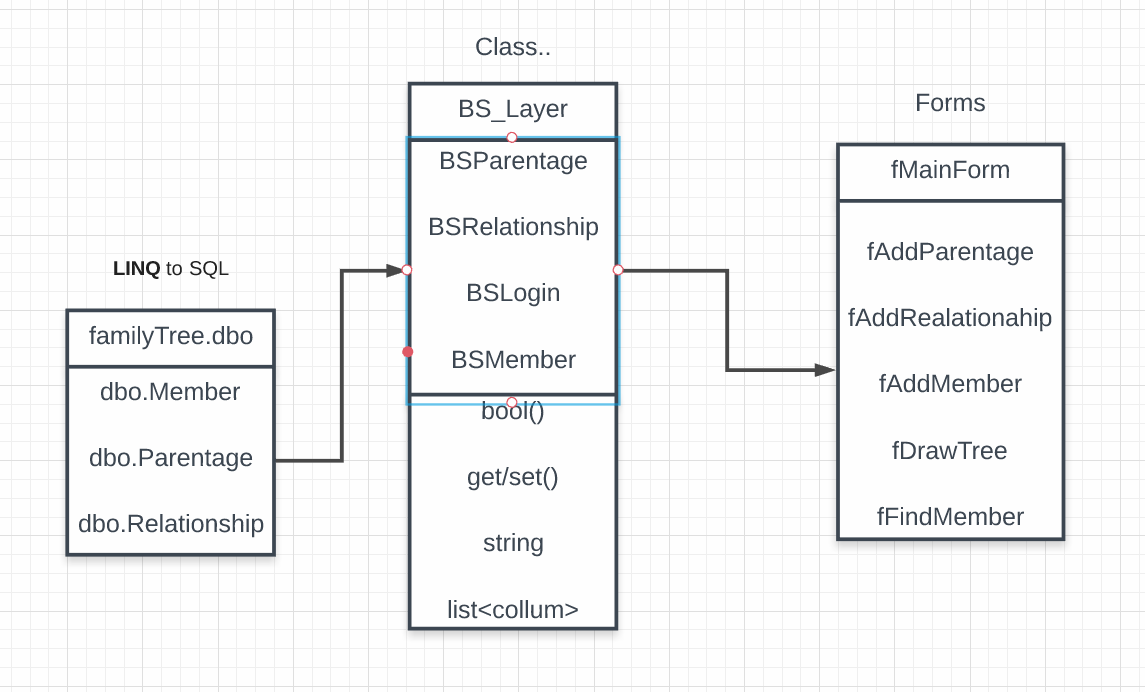
dbo.Relationship(person1ID, person2ID, relation,familyID)

**1.3. Situation (purposes, features)**

Software used for remembering family members for people who difficult in remembering family members.

The software has the ability to add or remove, delete family members, and the software has high security features

**1.4. Expected interface and UML database**



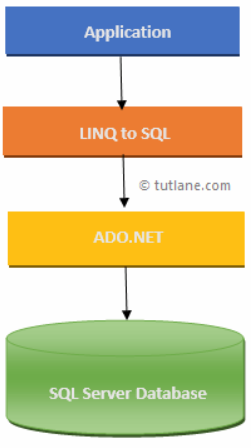
|  |  |
| --- | --- |
| **Taskwork** | **Evaluate contribution** |
| Phase 1 | Learn about the family tree and Design database and C# classes | Lê Tuấn Đạt 50%  Nguyễn Mạnh Tiến 50% |
| Phase 2 | Design UI, basic operation, searching and complete the database data connection | Lê Tuấn Đạt 45%  Nguyễn Mạnh Tiến 55% |
| Phase 3 | Design and draw a tree | Nguyễn Mạnh Tiến 80%  Lê Tuấn Đạt 20% |
| Phase 4 | Test the algorithm, check for inner error and maintain and  Write the report | Nguyễn Mạnh Tiến 35%  Lê Tuấn Đạt 65% |

**CHAPTER 2. Assignment of work**

**CHAPTER 3. Design**

**3.1 Connect to database**

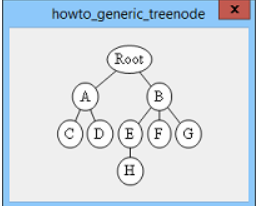
Recently, I have discussed about [getting started linq to sql](http://codesamplez.com/database/linq-to-sql-c-sharp-tutorial) in c#.net. That articles also covered for creating select query for retrieving both single and multiple results. Today, in this tutorial, I will try to give some more examples on other types of database operations, that writes/updates the database. These will cover writing and executing ‘Update’/’Insert’/’Delete’ query in the proper way.



Advantages of 3-layer Architecture

* Reusability: Moving code-behind libraries. It is possible to make changes in the presentation level without effecting the other two (business or data access layer).
* Decoupling of UI, Business logic and Data Access Layer are done.
* Maintainability: When we change one layer due to the client's needs it doesn't affect the other layers and we need to do less changes in another layer.

**3.2. Draw The Tree**

****The program draws the tree in two steps. First it arranges the nodes in the tree and then it draws them. You can do this in a single step, but arranging the nodes requires a separate pass through the tree anyway to figure out where everything goes, so you may as well break the process into two steps to make the code a little easier to understand. That also allows the program to arrange the tree without drawing it, so it can determine how big the tree will be.

* Step 1: Arranging Nodes
* Step 2 : Drawing the Tree

**CHAPTER 4. DESIGN PROGRAM**

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Class Name | Purpose | |
| 1 | DAO.Member | Retrieve data from Member table and query data from that table, add,search, delete, edit,update data from Member table | |
| 2 | DAO.Parentage | Retrieve data from Parentage table and query data from that table, add,search, delete, edit,update data from Parentage table | |
| 3 | DAO.Relation | Retrieve data from Parentage table and query data from that table, add,search, delete, edit,update data from Parentage table | |
| 4 | Idrawable | This interface requires that the class provide GetSize and Draw methods and The TreeNode class can then use those methods to draw the subtree. | |
| 5 | Treenode | The key class is TreeNode. This class has a Children list that contains references to the node’s child nodes in the tree.  This class is used to design the family tree interface.  The TreeNode class provides two constructors: one that takes a data object as a parameter and one that also includes a font | |
| 6 | Account | | This class to check the user name and password if data entered similar in the database user can assign to the program |

**4.1 Each class use in project**

|  |
| --- |
|  |

**4.1.1 Database Design**

**4.1.1.1. Table in Database**

|  |  |  |
| --- | --- | --- |
| TT | Table Name | Purpose |
| 1 | Dbo.Member | Family members data |
| 2 | Dbo.Relation | The relationship between family members, one person to another |
| 3 | Dbo.Parentage | Names of genealogy and members of those families |
| 4 | Dbo.Account | User account information and password |

**4.1.1.2 Diagram**



**4.1.2 Descripton Field**

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Field name | Type | Purpose |
| 1 | memID | Int(32) | Put the id for each person and it relation with person1ID and person2Id |
| 2 | memName | Nvarchar(50) | Name of each people in the family |
| 3 | Rustic | Nvarchar(50) | The hometown of each person |
| 4 | gender | Nchar(10) | Gender of each person |
| 5 | Dob | date | Day of birth |
| 6 | Dod | date | Day of dead |
| 7 | address | Nvarchar(50) | Address of each person |
| 8 | Person1ID | Int(32) | Id of person 1 which relation with memId in dbo.Member |
| 9 | Person2Id | Int(32) | Id of person 2 which relation with memId in dbo.Member |
| 10 | relation | Nchar(10) | Make the relation of persrson 1 to person2 |
| 11 | FamilyID | Int(32) | Id of each family id relation with pID in dbo.Parentage |
| 12 | pID | Int(32) | Id of each parentage |
| 13 | pName | Nchar(10) | Name of each parentage |
| 14 | username | Nchar(10) | The name of user account |
| 15 | password | Nchar(10) | Password of useraccount |

**4.1.3 Design Theme**

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Form/Dialog | Purpose | Design decisions |
| 1 | Authentication | Check whether the user in the data is accessed or not | Minimalist futuristic design for the user |
| 2 | Mainform | help users deceive the tool you want on the menu bar | Optimized simple design, saving space |
| 3 | AddingParentage(AddingForm) | Help user can adding, find, modify, delete the data in parentage and you also can see data in the data below | Simple design, easy for the user |
| 4 | Addmember(AddingForm) | User can add, del,edit,find member also user can draw the tree and it will display on the right, user can see data in the table below | Simple design, easy for the user |
| 5 | AddRelation (AddingForm) | User can add, modify, delete the relation of the member 1 to member 2 and put the parentage of each relation | Simple design easy for user |

**4.2** **The sample table describes the methods in a class**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Method** | **Purpose** | **The line contains the declaration** |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46 | public bool Login(string userName, string passWord)  public Member GetMemberByID(int memID)  public List<Member>GetListMember()  public List<Member> GetAllMemOfAParentage(int pID)  public bool InsertMember(int memID, string name, string rustic, string gender, DateTime dob, DateTime dod, string curAdd)  public bool ModifyMember(int memID, string name, string rustic, string gender, DateTime dob, DateTime dod, string curAdd)  public bool DeleteMember(int memID)  public List<Member> searchMember(string memName)  public Parentage GetParentageByID(int pID)  public List<Parentage> GetListParentage()  public bool InsertParentage(int pID, string pName)  public string convertToUnSign(string s)  public bool ModifyParentage(int pID, string pName)  public bool DeleteParentage(int pID)  public List<Parentage> searchParentage(string pName)  public System.Data.Linq.Table<Relationshi  p> GetListRelationship()  public DataTable GetAllRelationOfAFamily(int pID)  public List<Member> GetAllMemOfAParentage(int pID)  public List<Member> GetChildList(int mem1Id)  public int GetIDAbove(int memID)  public bool InsertRelation(int mem1ID, int mem2ID, string relation, int pID)  public bool ModifyRelation(int p1ID, int p2ID, string relation, int pID)  public bool DeleteRelation(int p1ID, int p2ID)  public bool DeleteMemberRelation(int memID)  public Member(int memID, string description, Button picture)  public SizeF GetSize(Graphics gr, Font font)  private RectangleF Location(PointF center)  public bool IsAtPoint(Graphics gr, Font font, PointF center\_pt, PointF target\_pt)  public void Draw(float x, float y, Graphics gr, Pen pen, Brush bg\_brush, Brush text\_brush, Font font)  private RectangleF PositionImage(Button picture, RectangleF rect)  SizeF GetSize(Graphics gr, Font font);  bool IsAtPoint(Graphics gr, Font font, PointF center\_pt, PointF target\_pt);  void Draw(float x, float y, Graphics gr, Pen pen,  Brush bg\_brush, Brush text\_brush, Font font);  public List<TreeNode<T>> Children = new List<TreeNode<T>>();  private const float Hoffset = 5;  private const float Voffset = 30;  private PointF Center;  public Font MyFont = null;  public Pen MyPen = Pens.Black;  public Brush FontBrush = Brushes.Black;  public Brush BgBrush = Brushes.White;  public TreeNode(T new\_data)  public TreeNode(T new\_data, Font fg\_font)  public void AddChild(TreeNode<T> child)  public void Arrange(Graphics gr, ref float xmin, ref float ymin)  public void DrawTree(Graphics gr, ref float x, float y)  public void DrawTree(Graphics gr)  private void DrawSubtreeLinks(Graphics gr)  private void DrawSubtreeNodes(Graphics gr)  public TreeNode<T> NodeAtPoint(Graphics gr, PointF target\_pt)  public bool DeleteNode(TreeNode<T> target) | retrieve data in the table and check if the username and password are correct if true returns true otherwise returns false  Get the family members of each ID  Get member list from database sorted by id  The first is to get a listID1 containing the id of the first member whose parentageID is the same as the pID passed  The listID2 is similar but the id of the second member.  Then combine those two lists into a new list id.  Then create a member list whose members contain ids in that list.  Finally, return the member list  Insert member data into member table and return true  If query have idMember then you can assign to edit data member and return true  Delete all data member when you delete memID all data will delete relize on that memID  Search member by using membername  Get the data parentage of each pID  Get the list of parentage sorted by pID  Insert the data of parentage in pareantage table and return true  Check all data if d or D is replaced by d and d  If modquery have pid user can assign to modify the parentage table then return true  Delete the parentage if delquery have pID this can assign to delete all data relation to pID in paratage table  Search pparentage by using search pName in the list  Get the database and return the dbo.Relationship  Get all the relation of each pID  Get all list member in parentage table corresponding with memid1 and memid2  pass on a memid to get the id of the child  user selects a member to draw, the system will take that person's id and find his connection otherwise there will find parent relationship without father then the id will be the original  Insert data in Relation table and return true  If modquery have memid 1 and memid 2 user can assign to edit the data and return true  Delete all data in the relation if memid1 and memid 2 satisfying conditions in delquery  Delete data in the relation if memid have memid1 and memid2 and it can assign to del the data and return true  Call button for drawings, display information and images  Return the size needed by this node.  Return a RectangleF giving the node's location  Return True if the target is under this node.  This class we draw the person by 2 step:  Draw a boader and then draw the picture  Find a rectangle to draw the image centered in the   * rectangle as large as possible without stretching.   Return the object’s needed size  Return true if the node is above this point.  Draw the object centered at (x, y).  Child nodes in the tree.  Space to skip horizontally between siblings and vertically between generations  The node's center after arranging..  Drawing Properties  Constructor  Add a treenode to out ChildrenList  Arrange the node and its children in the allowed area.   * Set xmin to indicate the right edge of our subtree. * Set ymin to indicate the bottom edge of our subtree.   Draw the subtree rooted at this node with the given upper left corner.  Draw the subtree rooted at this node  Draw the links for the subtree rooted at this node  if we have 1 child just connect the centers  Draw the nodes for the subtree rooted at this node  Return the TreeNode at this point (or null if there isn't one there).  Delete a target node from this node's subtree.  Return true if we delete the node. | DAOAccount.cs(line 29)  DAOMember.cs(line  31)  DAOMember.cs(line  41)  DAOMember(line  54)  DAOMember(line  81)  DAOMember(line  98)  DAOMember (line  118)  DAOMember (line  130)  DAOParentage(line  30)  DAOParentage(line  38)  DAO.Preantage(line  50)  DAO.Parentage(line  84)  DAO.parentage(line  60)  DAO.Parentage(line  74)  DAOPareantage(line  100)  DAORelation(line 29)  DAORelation (line 34)  DAORelation(line  47)  DAORelation(line 74)  DAORelation(line 88)  DAORelation(line  113)  DAORelation(line  125)  DAORelation(line  139)  DAORelation(line  149)  partial class Member:Idrawable(line  13)  partial class Member:Idrawable(line 28)  partial class Member:Idrawable(line 34)  partial class Member:Idrawable(line 43)  partial class Member:Idrawable(line 50)  partial class Member:Idrawable(line 80)  Idrawable(line13)  Idrawable(line 16)    Idrawable(line19)  Treenode.cs(line16)  Treenode.cs(line20)  Treenode.cs(line 24)  Treenode.cs(line 24)  Treenode.cs(line 33)  Treenode.cs(line 45)  Treenode.cs(line53)  Treenode.cs(line 117)  Treenode.cs(line127)  Treenode.cs(line 137)  Treenode.cs(line 177)  Treenode.cs(line195)  Treenode.cs(line 212) |

Soure Treenode.cs :

<http://csharphelper.com/blog/2015/02/make-a-generic-treenode-class-in-c-part-1/>

<http://csharphelper.com/blog/2015/02/make-a-generic-treenode-class-in-c-part-2/>

<http://csharphelper.com/blog/2015/02/handle-generic-treenode-mouse-events-in-c/>

<http://csharphelper.com/blog/2015/02/draw-tree-nodes-containing-pictures-c/>

**CHAPTER 5.** **Installation and testing**

|  |  |  |
| --- | --- | --- |
| **TT** | **Situation** | **Purpose** |
| 1 | Input : data from database  Output : show the data on the datagridview | Check the program can assign the database yes or not |
| 2 | Input : the data edit, delete the data, add the data  Output : change of data when adding, editing or deleting data | Check the code make sure that user can edit, add and del the data, and the data must change when user do that |
| 3 | Input : Useser choose 1 of member want to draw the family of that member  Output : Draw family of that member | Make sure that we can draw member and draw the relation of member user choose |
| 4 | Input : user choose 1 member in the draw then the system will redraw with the node is that the member we choose  Output : Draw the family user choose in the familytree | Make sure that we can draw the new tree when user click in the picture member button |

**CHAPTER 6. Conclude**

**6.1 Difficulties and solution**

|  |  |
| --- | --- |
| **Difficulties** | **Solution** |
| How to make the relation of 2 member in the database | Make the table Relationship and make member1id and member2 id and this relationship between person 1 toward person 2 ( explained by the teacher) |
| How to draw the familytree | <http://csharphelper.com/blog/2015/02/make-a-generic-treenode-class-in-c-part-1/>   build a generic TreeNode class that can draw a tree |

**6.2. For further implementation**

1. Make the button also have the avatar and name in that draw place
2. Make the theme more beautifuller and easy for user to use

**6.3. Advantages**

1. Simple easy-to-use software for drawing genealogy
2. Know more about how to solve the problem
3. Show data from database very clear
4. View detail and user can impact on that data a lots

**6.4. Limit**

1. The interface is not yet beautiful
2. If this have a lots data the program run very slow

**6.5.Reference**

<http://csharphelper.com/blog/2015/02/make-a-generic-treenode-class-in-c-part-1/>

<http://csharphelper.com/blog/2015/02/make-a-generic-treenode-class-in-c-part-2/>

<http://csharphelper.com/blog/2015/02/handle-generic-treenode-mouse-events-in-c/>

<http://csharphelper.com/blog/2015/02/draw-tree-nodes-containing-pictures-c/>

<http://csharphelper.com/blog/2015/02/draw-family-tree-c/>

<https://www.youtube.com/watch?v=3YFM0rPsc7o&t=1046s>

<https://www.youtube.com/watch?v=AwXSRbL1DbE&t=2415s>