

**Ho Chi Minh City University of Technology and Education**

**FACULTY FOR HIGH QUALITY TRAINING**

**INFORMATION TECHNOLOGY**

**PROJECT REPORT**

**TOPIC: LAB MEMBER MANAGEMENT**

**SUBJECT: WINDOW PROGRAMMING**





**Instructor : Huynh Xuan Phung**

**Group 1 :**

**Nguyen Truong Hoang Hieu - 17110030**



**Ho Chi Minh City, July 3rd, 2020**

SCORE

|  |  |  |  |
| --- | --- | --- | --- |
| Criteria | Content | Present | Total |
| Score |  |  |  |

Instructor’s comments

**………………………………………………………………………………………………..………………………………………………………………….………………………….……………………………………………………………………………………………………………..……………………………………………………………………………..……………………..………………………………………………………………………..……………………………..………………………………………………………………..……………………………………..………………………………………………………..……………………………………………..………………………………………………..……………………………………………………..………………………………………..……………………………………………………………..………………………………..……………………………………………………………………..…………………………..**

Instructor

(***Sign and Fullname***)

### Thanks

The success of a student more or less always accompanies a lecturer. We would like to

express our sincere thanks to Mr. Huynh Xuan Phung , who directly supported the

group, give us suggestions, comments and suggestions as well as provide tips to help us

make the project. Thanks for your instructions which help our team understand the

problems to do, the presentation as well as the implementation of the project so we have

completed the project 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 us improve our knowledge. Sincerely thanks.

*Ho Chi Minh city, July 3rd, 2020*

**CONTENTS**

[Thanks 4](#_Toc10798)

[List of images 7](#_Toc10302)

[List of tables 8](#_Toc4053)

[I. Introduction 9](#_Toc27168)

[1. Purpose 9](#_Toc4254)

[2. Input Information 9](#_Toc13296)

[3. Feature 10](#_Toc8387)

[4. Expected Interface 10](#_Toc17504)

[II. Assignment for work 13](#_Toc4089)

[III. Design 13](#_Toc27600)

[1. Layer design 13](#_Toc13977)

[A. List of entities used in the program 13](#_Toc11459)

[B. Methods in one entities: 14](#_Toc12896)

[2. Database Design: 21](#_Toc26154)

[3. Field: 21](#_Toc17946)

[3.1. Person entity: 21](#_Toc6353)

[3.2. Risk entity: 22](#_Toc23500)

[3.3. Task entity: 22](#_Toc26984)

[3.4. Requirement entity: 23](#_Toc25887)

[3.5. Project entity: 23](#_Toc19401)

[4. Interface Design: 24](#_Toc29711)

[4.1. Beginning Form 24](#_Toc26976)

[4.2. Main Form 25](#_Toc30256)

[4.3. Project Hour Form 26](#_Toc9189)

[4.4. Team members Form 27](#_Toc3850)

[4.5. Requirement Form 28](#_Toc24257)

[4.6. Risk Form 29](#_Toc12885)

[4.7. Task Form 30](#_Toc19678)

[IV. Testing 32](#_Toc25889)

[V. Conclusion 33](#_Toc11697)

[1. Student Evalution 33](#_Toc1322)

[2. Difficulties 33](#_Toc31667)

[3. Solution 33](#_Toc12268)

[VI. Reference 34](#_Toc24500)

List of images

*[Image](#_Toc18594)* [1](#_Toc18594) *[: Default Form](#_Toc18594)* [11](#_Toc18594)

*[Image](#_Toc11285)* [2](#_Toc11285)  *[: Project Hours Tab](#_Toc11285)* [11](#_Toc11285)

*[Image](#_Toc12513)* [3](#_Toc12513)  *[: Team Members Tab](#_Toc12513)* [12](#_Toc12513)

*[Image](#_Toc29273)* [4](#_Toc29273)  *[: Requirements Tab](#_Toc29273)* [12](#_Toc29273)

*[Image](#_Toc32469)* [5](#_Toc32469)  *[:Risks Tab](#_Toc32469)* [14](#_Toc32469)

*[Image](#_Toc21223)* [6](#_Toc21223)  *[:Tasks Tab](#_Toc21223)* [16](#_Toc21223)

*[Image](#_Toc18254)* [7](#_Toc18254) *[: Lab member manage form](#_Toc18254)* [30](#_Toc18254)

*[Image](#_Toc20371)* [8](#_Toc20371) *[: Main Form](#_Toc20371)* [31](#_Toc20371)

*[Image](#_Toc5745)* [9](#_Toc5745) *[:Change Project Properties Form](#_Toc5745)* [31](#_Toc5745)

*[Image](#_Toc11883)* [10](#_Toc11883) *[: Project Hour Form](#_Toc11883)* [32](#_Toc11883)

*[Image](#_Toc72)* [11](#_Toc72) *[: Team members Form](#_Toc72)* [33](#_Toc72)

*[Image](#_Toc11858)* [12](#_Toc11858) *[: Add team members Form](#_Toc11858)* [33](#_Toc11858)

*[Image](#_Toc6782)* [13](#_Toc6782) *[: Requirement Form](#_Toc6782)* [34](#_Toc6782)

*[Image](#_Toc28395)* [14](#_Toc28395) *[: Create Requirement Form](#_Toc28395)* [34](#_Toc28395)

*[Image](#_Toc17251)* [15](#_Toc17251) *[: Risk Form](#_Toc17251)* [35](#_Toc17251)

*[Image](#_Toc4025)* [16](#_Toc4025) *[: Create Risk Form](#_Toc4025)* [35](#_Toc4025)

*[Image](#_Toc23094)* [17](#_Toc23094) *[: Task Form](#_Toc23094)* [36](#_Toc23094)

*[Image](#_Toc19076)* [18](#_Toc19076) *[: Add task hour Form](#_Toc19076)* [36](#_Toc19076)

*[Image](#_Toc19620)* [19](#_Toc19620) *[: Create task Form](#_Toc19620)* [37](#_Toc19620)

List of tables

*[Table](#_Toc27167)* [1](#_Toc27167) *[:Input information](#_Toc27167)* [9](#_Toc27167)

*[Table](#_Toc26491)* [2](#_Toc26491) *[: Assignment for work](#_Toc26491)* [13](#_Toc26491)

*[Table](#_Toc14288)* [3](#_Toc14288) *[: List of entities](#_Toc14288)* [14](#_Toc14288)

*[Table](#_Toc10847)* [4](#_Toc10847) *[:List of Methods](#_Toc10847)*  [14](#_Toc10847)

*[Table](#_Toc30692)* [5](#_Toc30692) *[: Database design](#_Toc30692)* [21](#_Toc30692)

[Table 6](#_Toc10930) *[: Person](#_Toc10930)* [21](#_Toc10930)

[Table 7](#_Toc10342) *[: Risk](#_Toc10342)* [22](#_Toc10342)

*[Table](#_Toc29297)* [8](#_Toc29297) *[: Task](#_Toc29297)* [22](#_Toc29297)

*[Table](#_Toc7125)* [9](#_Toc7125) *[: Requirement](#_Toc7125)* [23](#_Toc7125)

[Table 10](#_Toc9656) *[: Project](#_Toc9656)* [23](#_Toc9656)

*[Table](#_Toc16841)* [11](#_Toc16841) *[: Testing situations](#_Toc16841)* [32](#_Toc16841)

I. Introduction

1. Purpose

Lab member management software is a software that was designed to manage general information about lab such as project name, project owner, project description . Beside that the project allow users store project hours, team members, requirement, risks and tasks.

The software can also store many project in the system and project can be retrieve and create a new one.

2. Input Information

|  |  |
| --- | --- |
| **Object** | **Input Information** |
| Project Overview | Project Name  Project Owner  Project Description |
| Project Hours | Generals Properties  Project Hours |
| Team Members | First Name  Last Name  Salary |
| Requirement | Requirement Description  Type  Priority |
| Risk | Risk Description  Risk Priority  Risk Likelyhood |
| Task | Task name  Task hours  Task Descriptions  Task Owner  Task Category |

*Table 1**:Input information*

3. Feature

The application provides user many features:

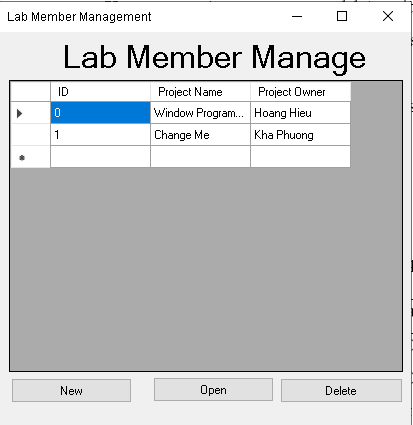
User can create new ,open, or delete exited project in the system.

User can edit information about project such as name , owner , description.

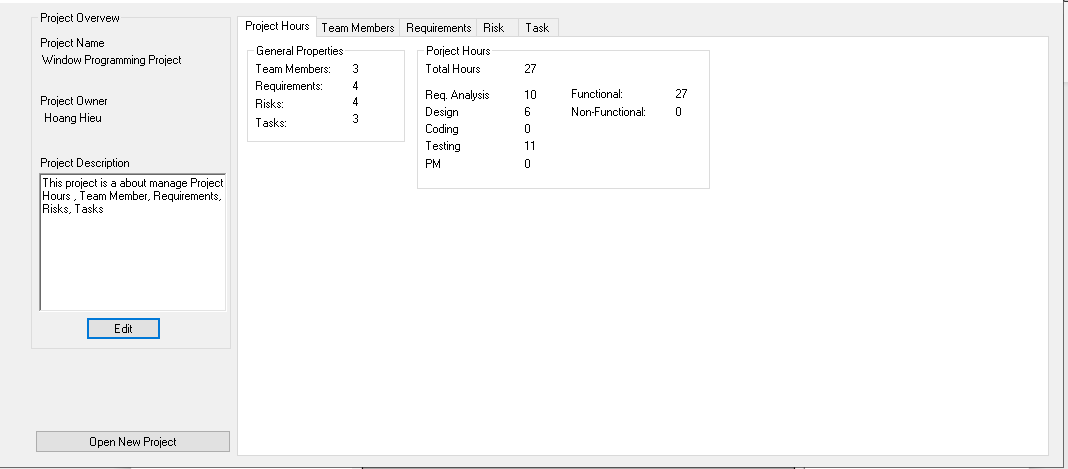
Project hours can be stored by tasks.

Software store data about team members, requirements, risks, tasks.

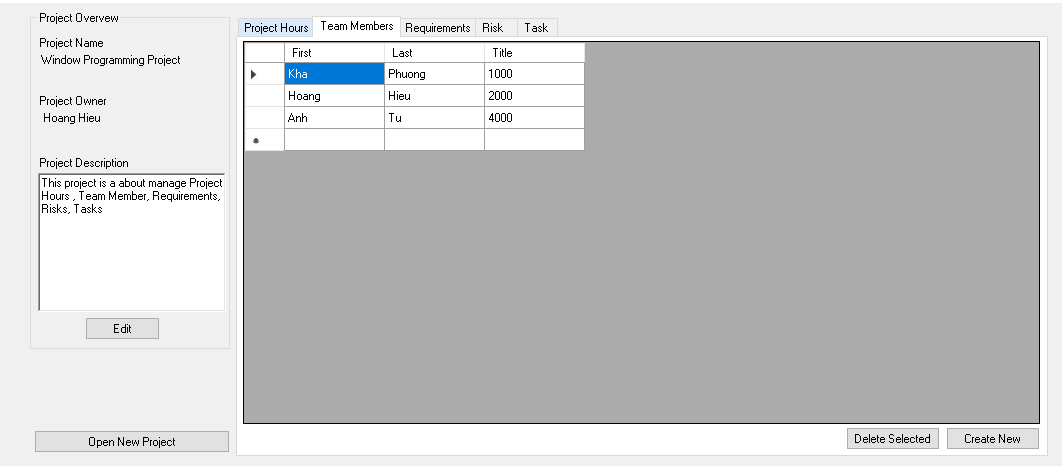
4. Expected Interface



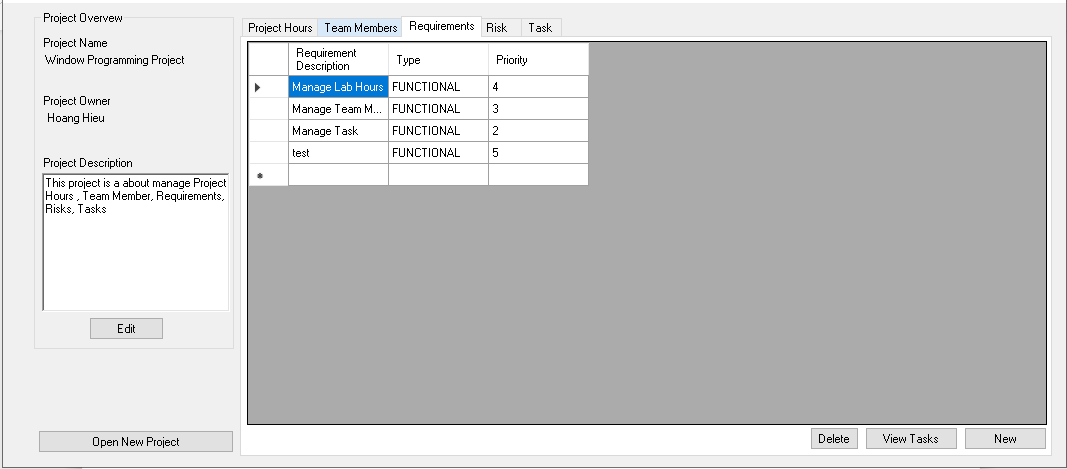
*Image 1**: Default Form*



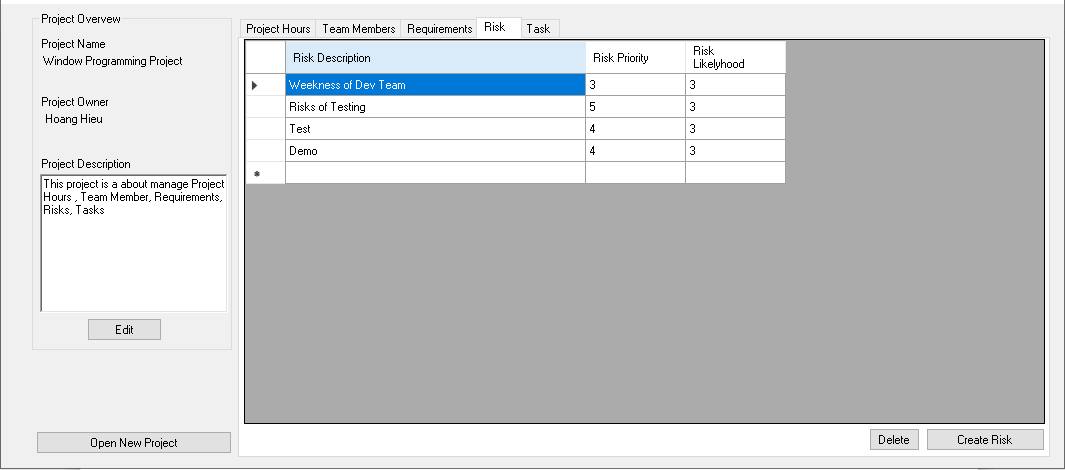
*Image 2* *: Project Hours Tab*



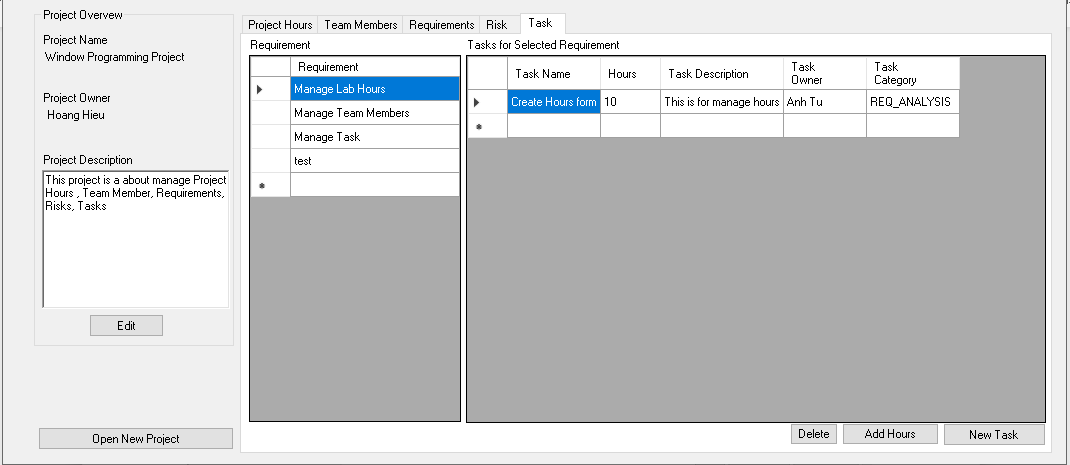
*Image 3* *: Team Members Tab*



*Image 4* *: Requirements Tab*



*Image 5* *:Risks Tab*



*Image 6* *:Tasks Tab*

II. Assignment for work

*Table 2**: Assignment for work*

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Full Name** | **Task description** | **Percent contribution** |
| 17110030 | Nguyen Truong Hoang Hieu | -Design database -Design application interface  -Code  -Make report | 100% |

III. Design

1. Layer design

### A. List of entities used in the program

*Table 3**: List of entities*

|  |  |  |
| --- | --- | --- |
| **Oder** | **Entity’s name** | **Purpose** |
| 1 | Person | Store data about person (ID, name,salary) |
| 2 | Project Manager - Person | Store data about project manager |
| 3 | Project Owner - Person | Store data about project owner |
| 4 | Task Owner - Person | Store data about task owner |
| 5 | Task | Store data about task such as name, description, task owner and hours |
| 6 | Risk | Store data about risk such as decriptions, likelihood, priority |
| 7 | Requirement - Task | Store data about requirement such as decriptions, priority, tasks |
| 8 | Projectn -  Person  Requirement  Risks  Task | Store data about project(ID, Name, Owner, Manager, Description, Risk Requirement) |

### B. Methods in one entities:

*Table 4**:List of Methods*

|  |  |  |  |
| --- | --- | --- | --- |
| **Order** | **Method** | **Purpose** | **File’s name** |
| 1 | buttonNew\_Click()  Input: none  Output: none | Create new project in the system | Form1.cs  (35) |
| 2 | buttonOpen\_Click()  Input: none  Output: none | Open a exited project in the system | Form1.cs  (43) |
| 3 | setUpDataGrid (List<Project> projects)  Input: List of projects  Output: none | Show all project in the system to datagrid | Form1.cs  (50) |
| 4 | UI\_deleteBtn\_Click  Input: none  Output: none | Delete a project in the system | Form1.cs  (71) |
| 5 | Init()  Input: none  Output: none | Anything that you need to run the first time the Form loads, things like initialize the column names, refresh some data, make a new method for it, and add it here so it is updated. | Form2.cs  (48) |
| 6 | refresh()  Input: none  Output: none | Anything that needs to update AFTER the initial start. This might include things like change of "project properties", refreshing team list, refreshing requirements list, etc.This will only be called if the WHOLE WINDOW needs a refresh. Otherwise, use the individual methods like refreshTeamMembers() to just update the team members list. No need to update the whole window if everything else stayed the same. | Form2.cs  (57) |
| 7 | updateHomePage()  Input: none  Output: none | Update the home page with all the details of the currentProject | Form2.cs  (69) |
| 8 | updateRisksPage()  Input: none  Output: none | Update the risk tab control with all the details of the currentProject | Form2.cs  (131) |
| 9 | initializeProjectProperties()  Input: none  Output: none | This shold really be called "refreshProjectProperties" because that's what it does. It just updates the project properties after a potential update. | Form2.cs  (148) |
| 10 | initializeTeamTab()  Input: none  Output: none | Adds columns to the team tab. These are them populated by "refreshTeamMembers" | Form2.cs  (156) |
| 11 | refreshTeamMembers()  Input: none  Output: none | This takes the currentProject, extracts the List<Person> team from it, and populates the teamGridView, which is on the Team Members tab. | Form2.cs  (167) |
| 12 | buttonCreateUser\_Click()  Input: none  Output: none | Create new team member | Form2.cs  (178) |
| 13 | UI\_editButton\_Click()  Input: none  Output: none | Shows the dialog to edit the basic properties shown on the right side.  TO-DO: change refresh() to initializeProjectProperties() since we do not need to update the whole Window. | Form2.cs  (191) |
| 14 | UI\_deleteTeamMemberButton\_Click()  Input: none  Output: none | This is an example of a successful edit in the Project.It deletes the team member that is currently highlighted. Then updates the DATABASE using data.updateProject. Then refreshes the local team members DataGRidView using refreshTeamMembers(); | Form2.cs  (205) |
| 15 | buttonCreateTask\_Click()  Input: none  Output: none | Create new task in the system | Form2.cs  (211) |
| 16 | buttonNewRequirement\_Click  Input: none  Output: none | Implementation code for Requirements Tab | Form2.cs  (225) |
| 17 | updateRequirementsPage()  Input: none  Output: none | Update the current project: getLatestProjectFromDatabase();  Then use currentProject.requirements which holds all the requirements to get  each Requirement. | Form2.cs  (236) |
| 18 | deleteBtn\_Click  Input: none  Output: none | Delete requirement from requirement tab | Form2.cs  (251) |
| 19 | updateTasksPage()  Input: none  Output: none | Update the task page with all the details of the currentProject | Form2.cs  (260) |
| 20 | button2\_Click()  Input: none  Output: none | Create new risk in the current project | Form2.cs  (272) |
| 21 | updateTasksTabTasks()  Input: none  Output: none | Update the current project: getLatestProjectFromDatabase();  Then use currentProject.requirements which holds all the requirements to get  each Task for selected requirement. | Form2.cs  (283) |
| 22 | buttonDeleteTask\_Click()  Input: none  Output: none | Delete Task for selected requirement | Form2.cs  (302) |
| 23 | buttonAddHours\_Click()  Input: none  Output: none | Add hours for Task selected in selected requirement | Form2.cs  (316) |
| 24 | UI\_NewProjectBtn\_Click()  Input: none  Output: none | Move to the Form1 to create new project | Form2.cs  (326) |
| 25 | UI\_addButton\_Click()  Input: none  Output: none | Add new person to the current project | AddPerson.cs  (17) |
| 26 | UI\_AddHrsBtn\_Click()  Input: none  Output: none | Add hours for selected task to the current project | AddTaskHours.cs  (27) |
| 28 | UI\_newPersonBtn\_Click  Input: none  Output: none | Move to AddPerson.cs to add new person | AddTeamMember.cs  (27) |
| 29 | fillPersonDataGrid()  Input: none  Output: none | Fill in datagrid view person in the current project | AddTeamMember.cs  (33) |
| 30 | addMember\_Click  Input: none  Output: none | Add selected member to the current project | AddTeamMember.cs  (50) |
| 31 | UI\_delete\_TeamMem\_Click  Input: none  Output: none | Delete selected member to the current project | AddTeamMember.cs  (56) |
| 32 | buttonReqSave\_Click()  Input: none  Output: none | Save all details about requirement to current project | CreateRequirement.cs  (43) |
| 33 | button1\_Click\_1()  Input: none  Output: none | Save all details about risk to current project | CreateRisk.cs  (23) |
| 34 | buttonSaveTask\_Click  Input: none  Output: none | Save all details about task to current project | CreateTask.cs  (28) |
| 35 | setBasicProjectProperties()  Input: none  Output: none | Set all basic project properties | ProjectBasics.cs  (28) |
| 36 | fillPersonDataGrid()  Input: none  Output: none | Fill in data frid view all person in current project | ProjectBasics.cs  (33) |
| 37 | UI\_newPersonBtn\_Click()  Input: none  Output: none | Add new person to current project | ProjectBasics.cs  (51) |
| 38 | UI\_save\_Click()  Input: none  Output: none | Save all deatails about project to the system | ProjectBasics.cs  (60) |

2. Database Design:

*Table 5**: Database design*

|  |  |  |
| --- | --- | --- |
| **Order** | **Table’s name** | **Purpose** |
| 1 | Person | Save and show details about person in the system |
| 2 | Project | Save and show all project in the system |
| 3 | Requirement | Save and show details about requirement in the system |
| 4 | Risk | Save and show details about risks in the system |
| 5 | Task | Save and show details about tasks in the system |

3. Field:

### 3.1. Person entity:

Table 6*: Person*

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity name** | **Data type** | **Primary key** | **Decription** |
| PersonID | nvarchar(50) | x | Person id |
| fName | nvarchar(50) |  | First name |
| lName | nchar(10) |  | Last name |
| Salary | nvarchar(50) |  | Salary |

### 3.2. Risk entity:

Table 7*: Risk*

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity name** | **Data type** | **Primary key** | **Decription** |
| description | nvarchar(50) | X | Decription about the risk |
| likelihood | nvarchar(50) |  | Likelihood of risk |
| priority | nvachar(50) |  | Priority of the risk |

### 3.3. Task entity:

*Table 8**: Task*

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity name** | **Data type** | **Primary key** | **Decription** |
| name | char(10) | X | Name of task |
| description | char(10) |  | Decription of task |
| taskOwner | nvachar(50) |  | Task Owner |
| category | nvarchar(50) |  | Category of task such as REQ\_ANALYSIS,DESIGN,  CODING,TESTING,PM |
| hoursExpended | nvarchar(50) |  | Number of hours need to complete this task |

### 3.4. Requirement entity:

*Table 9**: Requirement*

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity name** | **Data type** | **Primary key** | **Decription** |
| description | nvarchar(50) | X | Decription of requirement |
| priority | nvarchar(50) |  | Priority of requirement |
| Tasks Name | nvachar(50) |  | Name of task in requirement |
| Requirement Category | Nvarchar(50) |  | Requirement Category such as FUNCTIONAL,NONFUNCTIONAL |

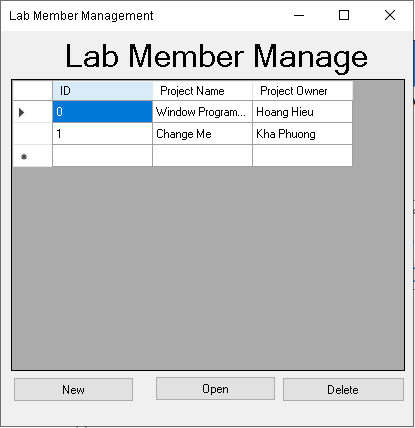
### 3.5. Project entity:

Table 10*: Project*

|  |  |  |  |
| --- | --- | --- | --- |
| **Entity name** | **Data type** | **Primary key** | **Decription** |
| ProjectID | char(10) | X | Project ID |
| projectName | nchar(10) |  | Project name |
| projectManager | nchar(10) |  | Manager of the project |
| projectOwner | nchar(10) |  | Owner of the project |
| projectDescription | nchar(10) |  | Description of the project |
| requirements | nchar(10) |  | Requirements of project |
| risks | nchar(10) |  | Risk of project |
| team | nchar(10) |  | Team member of project |

4. Interface Design:

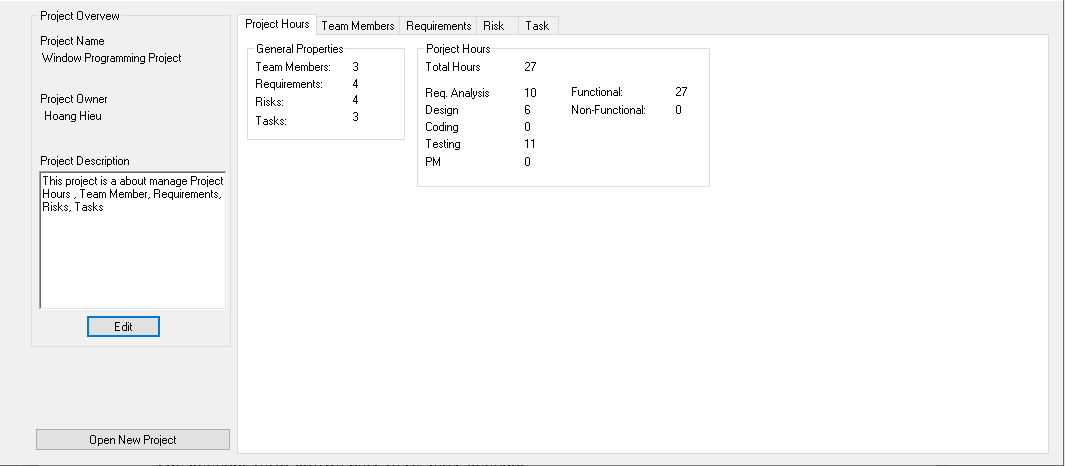
### 4.1. Beginning Form



*Image 7**: Lab member manage form*

This form show list of project in the system you can create new, open or delete project in the system.

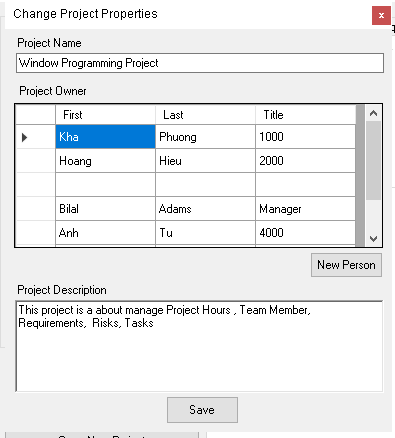
### 4.2. Main Form



*Image 8**: Main Form*

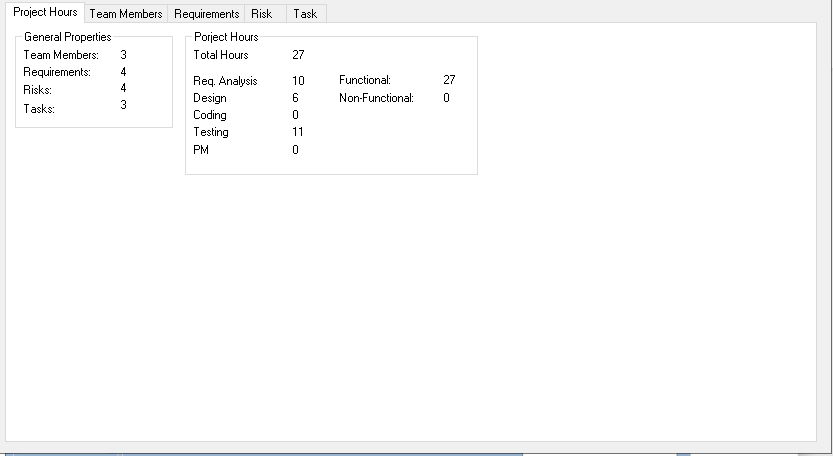
- In the left hand side, the form show all details about project such as project name , project owner, and project description .

- User can edit project detail by click edit button and it show change project properties



*Image 9**:Change Project Properties Form*

### 4.3. Project Hour Form



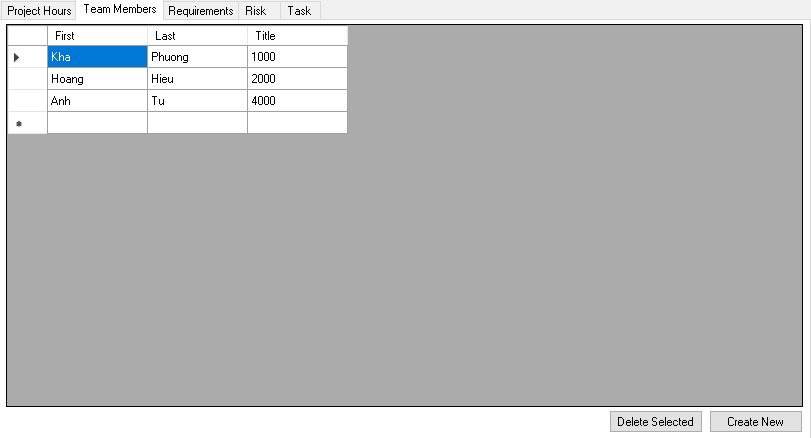
*Image 10**: Project Hour Form*

- This form show general properties of project and project hour.

-General properties show number of team member, requirements , risks and tasks in the project.

-Project hours show the totals hours to do this project and specific tasks

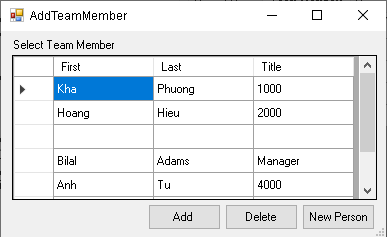
### 4.4. Team members Form



*Image 11**: Team members Form*

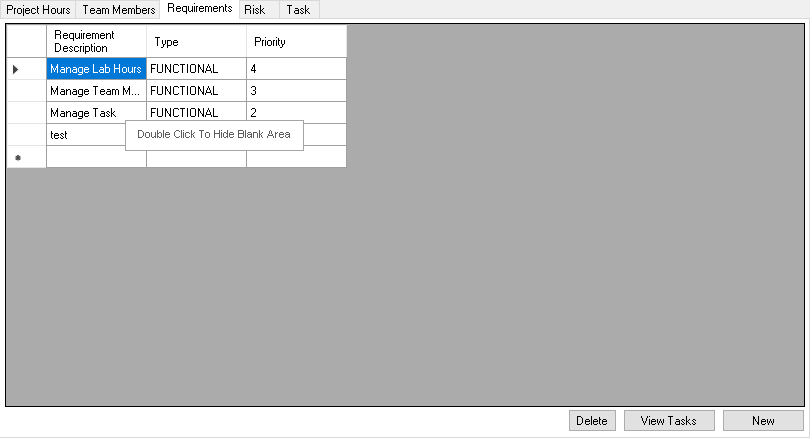
- The team members form shows list of team member in this project .

-You can also create new team members or delete a team member



*Image 12**: Add team members Form*

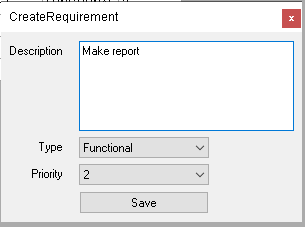
### 4.5. Requirement Form



*Image 13**: Requirement Form*

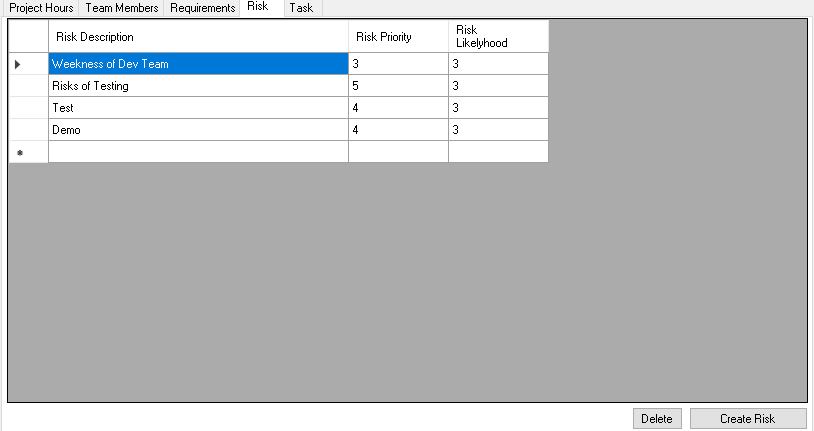
- This requirement form show list of requirement in the project and all details of requirement.

- You can create new delete requirement in this project



*Image 14**: Create Requirement Form*

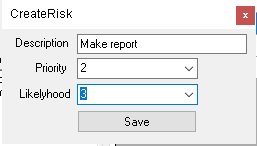
### 4.6. Risk Form



*Image 15**: Risk Form*

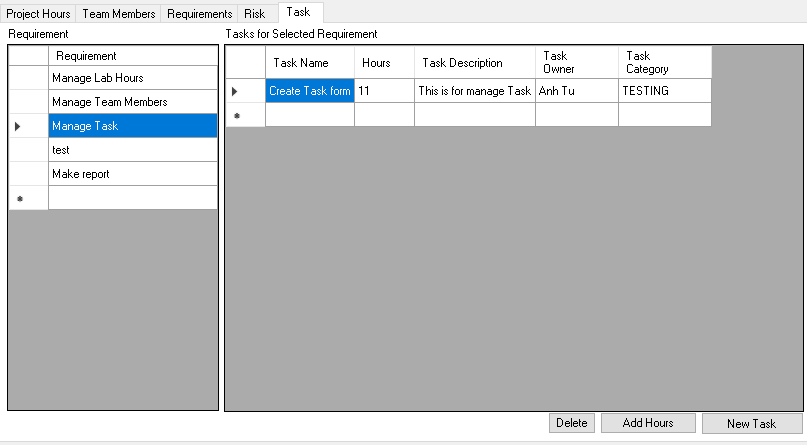
- This form show list of risk in the project and all detail about risk in this project.

- User can aslo create new or delete risk.



*Image 16**: Create Risk Form*

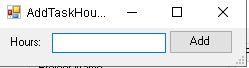
### 4.7. Task Form



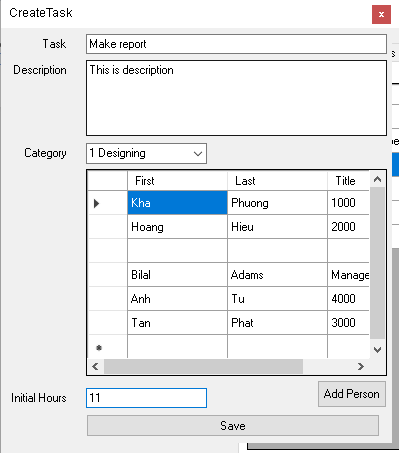
*Image 17**: Task Form*

- In the left data grid view show all requirement in the project and with each requirement selected , the right data grid view show all tasks in selected requirement

- User can also delete task, add extend hour for task and create new task



*Image 18**: Add task hour Form*



*Image 19**: Create task Form*

IV. Testing

*Table 11**: Testing situations*

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Situations** | **Purpose** | **Explaination** |
| 1 | Data input: details information of project, team member, tasks, requirements and risks  Expected result:  All information show on data grid view and user can manage it | To make sure data can be easy for user to store, retrieve and make change | The point of the tool is to manage projects, team member, tasks, requirements, and risks. |
| 2 | Data input:  Requirements , Task  Expected result:  For each selected requirement show all tasks for it. | To manage tasks for each requirement | Each requirement (functional and nonfunctional) has a list of tasks that need to be performed for the requirement to be completed. |
| 3 | Data input:  Hours on tasks  Expected result:  Hours for completed project was show | Hours on these tasks are recorded and summarized on the tab hours project. | Hours on these tasks are recorded and summarized on the home page. |

V. Conclusion

1. Student Evalution

-My team self-assesses that 90% of the project has been completed.

-Design a project with Window Programming knowledge.

-Simple GUI design which is easy to use

-Apply all knowledge about C# and winform to do this project

-The code is quite clean and reuseable

1. Difficulties

-The problem when design the system, hard to find the way to structure project.

-Some time data grid view doesn’t refresh when I add new value.

-Many bug occur when coding

3. Solution

- Spend more time to design structure of system .

- Searching for solution on internet about the problem.

-Ask friends and lectures for help.

-Debug many times and try as many solutions as possible.

VI. Reference

- Watson, K., Nagel, C., Pedersen, J. H., Reid, J. D., & Skinner, M. (2010). Beginning Visual C# 2010. John Wiley & Sons.

- Faison, T. (2002). Component-based development with visual C# (pp. 575-577). Hungry Minds.

- Petzold, C. (2002). Programming Microsoft Windows with C# (pp. 89-98). Redmond, Washington: Microsoft Press.

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

**-<https://stackoverflow.com/>**