**HCMC UNIVERSITY OF TECHNOLOGY AND EDUCATION**

**FACULTY FOR HIGH QUALITY TRAINING**

****

**Project 1**

**Topic : Manage Project Student**

***Lecturer : Nguyễn Thiên Bảo***

**Group :**

1. **Nguyễn Đình Thiên Phước 17110062**
2. **Lê Chí Thông 17110078**
3. **Mai Nguyễn Anh Thư 17110080**

**Grades**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criteria** | **Content** | **Presenting** | **Total** |
| **Grades** |  |  |  |

**Review of instructors**

# Table of content

[Table of content 3](#_Toc45314716)

[I. Work Assignment 7](#_Toc45314717)

[1. Plan 7](#_Toc45314718)

[Table 1 Weekly Plan 7](#_Toc45314719)

[2. Assign works: 7](#_Toc45314720)

[II. Design Software 9](#_Toc45314721)

[1. Design UML 9](#_Toc45314722)

[Picture 1 Use-case Diagram 9](#_Toc45314723)

[2. Scenario 10](#_Toc45314724)

[Table 2 Scenario Manage Student Information 10](#_Toc45314725)

[Table 3 Scenario Manage subject 13](#_Toc45314726)

[Table 4 Scenario Manage Project 16](#_Toc45314727)

[Table 5 Scenario Manage Lecturer 20](#_Toc45314728)

[Table 6 Scenario Login 24](#_Toc45314729)

[Table 7 Scenario Manage Class 25](#_Toc45314730)

[Table 8 Scenario Manage Faculty 28](#_Toc45314731)

[Table 9 Scenario Decentralization 31](#_Toc45314732)

[Table 10 Scenario Multi-language 34](#_Toc45314733)

[3. Design Database 35](#_Toc45314734)

[Picture 2 ER Diagram 35](#_Toc45314735)

[Picture 3 Class diagram 36](#_Toc45314736)

[Table 11 of Datatypes 36](#_Toc45314737)

[4. Design Interface 41](#_Toc45314738)

[Picture 3 Form Home 41](#_Toc45314739)

[Picture 4 Form Login 42](#_Toc45314740)

[Picture 5 form Manage student information 42](#_Toc45314741)

[Picture 6 from Manage lecturer information 42](#_Toc45314742)

[Picture 7 form Manage project student 42](#_Toc45314743)

[Picture 8 form Manage subject 43](#_Toc45314744)

[Picture 9 Manage Faculty 43](#_Toc45314745)

[Picture 10 form Manage course 43](#_Toc45314746)

[Picture 11 form Decentralization 43](#_Toc45314747)

[Picture 12 form Manage Multi-language 43](#_Toc45314748)

[5. Interface Spectification 43](#_Toc45314749)

[Table 11 Interface Spectification 43](#_Toc45314750)

[6. Methods spectification in class 54](#_Toc45314751)

[Table 12 Category of class for layers 54](#_Toc45314752)

[Table 13 Methods spectification in class 55](#_Toc45314753)

[III. Setting and testing 56](#_Toc45314754)

[Picture 13 Testing form ManageStudentInformation 56](#_Toc45314755)

[Picture 14 Testing form ManageLecturerInformation 56](#_Toc45314756)

[Picture 15 Testing form ManageProjectStudent 57](#_Toc45314757)

[Picture 16 Testing form ManageClass 57](#_Toc45314758)

[IV. Conclusion and development direction 58](#_Toc45314759)

[1. Conclusion: 58](#_Toc45314760)

[2. Development direction: 58](#_Toc45314761)

**Category of pictures**

[Picture 1 Use-case Diagram 9](#_Toc45314723)

[Picture 2 ER Diagram 35](#_Toc45314735)

[Picture 3 Class diagram 36](#_Toc45314736)

[Picture 3 Form Home 41](#_Toc45314739)

[Picture 4 Form Login 42](#_Toc45314740)

[Picture 5 form Manage student information 42](#_Toc45314741)

[Picture 6 from Manage lecturer information 42](#_Toc45314742)

[Picture 7 form Manage project student 42](#_Toc45314743)

[Picture 8 form Manage subject 43](#_Toc45314744)

[Picture 9 Manage Faculty 43](#_Toc45314745)

[Picture 10 form Manage course 43](#_Toc45314746)

[Picture 11 form Decentralization 43](#_Toc45314747)

[Picture 12 form Manage Multi-language 43](#_Toc45314748)

[Picture 13 Testing form ManageStudentInformation 56](#_Toc45314755)

[Picture 14 Testing form ManageLecturerInformation 56](#_Toc45314756)

[Picture 15 Testing form ManageProjectStudent 57](#_Toc45314757)

[Picture 16 Testing form ManageClass 57](#_Toc45314758)

**Category of tables**

[Table 1 Weekly Plan 7](#_Toc45314719)

[Table 2 Scenario Manage Student Information 10](#_Toc45314725)

[Table 3 Scenario Manage subject 13](#_Toc45314726)

[Table 4 Scenario Manage Project 16](#_Toc45314727)

[Table 5 Scenario Manage Lecturer 20](#_Toc45314728)

[Table 6 Scenario Login 24](#_Toc45314729)

[Table 7 Scenario Manage Class 25](#_Toc45314730)

[Table 8 Scenario Manage Faculty 28](#_Toc45314731)

[Table 9 Scenario Decentralization 31](#_Toc45314732)

[Table 10 Scenario Multi-language 34](#_Toc45314733)

[Table 11 of Datatypes 36](#_Toc45314737)

[Table 11 Interface Spectification 43](#_Toc45314750)

[Table 12 Category of class for layers 54](#_Toc45314752)

[Table 13 Methods spectification in class 55](#_Toc45314753)

# Work Assignment

## Plan

#### Table 1 Weekly Plan

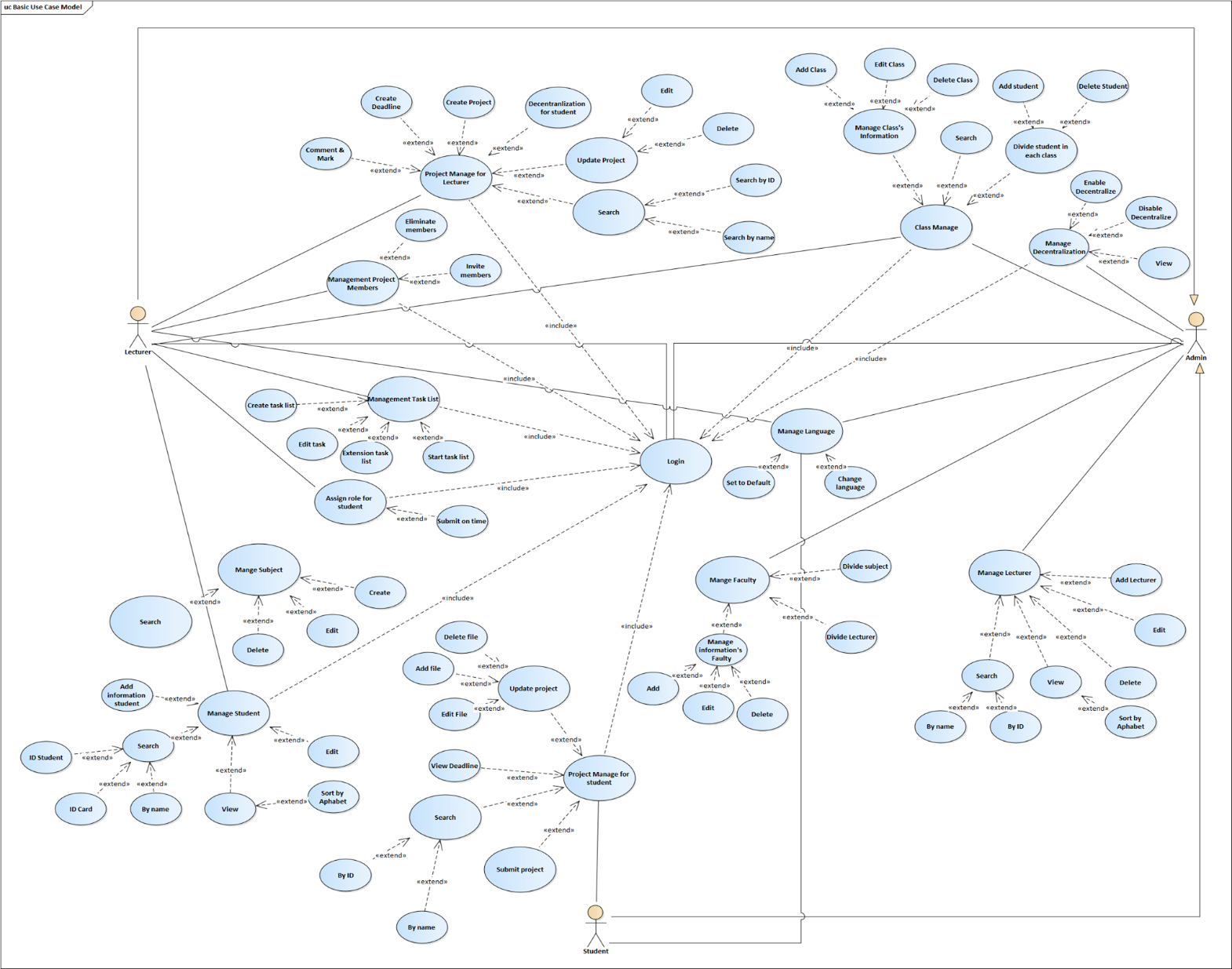
|  |  |
| --- | --- |
| Tuần | Work |
| 9 | Reseacrh about MVVM paradigm and 3 layers paradigm  Research Entity-framework CODE-FIRST |
| 10 | Design UML includes : Use-case diagram, scenario  Design interface as expected  Design database : ERD diagram, Class diagram, Create table of datatypes |
| 11 | Design official interface for the project : handle key, button, gridviewcontrol. Design 3D button, create main form and subforms, form login, form home. (branch front-end)  Design MVVM paradigms with 3 layers : View, View\_Model, Model |
| 12 | Design databases (code-first) : data processing, write query, initialization, create primary and foreign keys, create tables (branch back-end) |
| 13 | Merge two branches : front-end and back-end . Handle dump data into the forms |
| 14 | Demo and run code, test case and debug |
| 15 | Write report and presentation project for lecturer |

## Assign works:

|  |  |  |  |
| --- | --- | --- | --- |
| TT | Name of student | Work Description | Contributed |
| 1 | Nguyễn Đình Thiên Phước | Design database, code back-end, code a lot of front-end |  |
| 2 | Lê Chí Thông | Design front-end,design database, handle key, write report |  |
| 3 | Mai Nguyễn Anh Thư | Design front-end, design database, handle button, code front-end |  |

# Design Software

## Design UML



#### Picture 1 Use-case Diagram

## Scenario

#### Table 2 Scenario Manage Student Information

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC1 |
| Use-case Name | Manage Student Information |
| Actor | User (Admin) |
| Description | Let users manipulate the system to manage the student information |
| Trigger | When the user presses the login button from the home form. |
| Pre-Condition(s) | Must have an account in the system.  Account has been decentralized.  The device is connected to the internet. |
| Post-Condition(s) | User successfully logged in.  Users can successfully and add, view, edit, delete subject .  The system stores user information and reuses it for the whole system. |
| Basic Flow | 1. The system displays the login form. 2. User enter the account and password. 3. System displays checking login information. 4. Success: load the information the user has just logged in via the form home and displayed on the form management subject 5. Admin choose one of three: add , edit, delete and view 6. Add    * 1. Admin choose button Add to add a new student information      2. System require to enter student information      3. Admin enter student information      4. Admin choose save      5. System check student information in database      6. System add new information on form and in database      7. Add is complete      8. The use-case ends. 7. Edit    * 1. Admin choose button Edit to edit student information      2. Admin choose student to edit information      3. Admin enter information of student to edit      4. Admin choose save      5. The system sends updated information to the database      6. System edit student information on form and in database      7. Edit is complete      8. The use-case ends. 8. Delete    * 1. Admin choose student to delete      2. Admin press button Delete      3. Admin confirmed want to delete      4. System through database check the constraints of the selected student information      5. System delete student information on form and in database 9. View    * 1. Admin choose search box to view student information      2. Admin enter name of student need to search      3. System check name of student in database      4. System show list student information and grades of student      5. View is complete      6. The use-case end |
| Alternative Flow | 1. Login   If admin enter username or password is incorrect   * System notify login failed and login required again  1. Add 2. If admin don’t want to add a new student information  * Admin choose button Ignore * A new information isn’t added  1. If student information isn’t invalid  * System notify student information isn’t invalid , require admin re-enter * Return step a.3  1. Edit 2. If admin don’t want to edit student information  * Admin choose button Ignore * Student information isn’t edited  1. If subject information isn’t invalid  * System notify student information isn’t invalid , require admin re-enter * Return step b.3  1. Delete   If admin don’t want to delete student information   * Admin choose button Ignore * Student information isn’t deleted  1. Search 2. If admin don’t want to view information  * Admin choose button Ignore * Student information isn’t search  1. If information hasn’t in database  * System show messenger box “No results are found” * Student information isn’t view  1. The systems show message box which content is “No results are found” |

#### Table 3 Scenario Manage subject

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC2 |
| Use-case Name | Management subject |
| Actor | User (Admin) |
| Description | Let users manipulate the system to manage the subject |
| Trigger | When the user presses the login button from the home form. |
| Pre-Condition(s) | 1. Must have an account in the system. 2. Account has been decentralized. 3. The device is connected to the internet. |
| Post-Condition(s) | 1. User successfully logged in. 2. Users can successfully and create, search, edit, delete subject 3. The system stores user information and reuses it for the whole system. |
| Basic Flow | 1. The system displays the login form. 2. User enter the account and password. 3. System displays checking login information. 4. Success: load the information the user has just logged in via the form home and displayed on the form management subject 5. User can select functions: create, edit, search, delete    1. Create       1. User choose button Create to create a new subject       2. System require to enter subject information       3. User enter subject information       4. User choose save       5. System check subject information in database       6. System add new subject on form and in database       7. Add complete       8. The use-case end    2. Edit       1. User choose button Edit to edit subject information       2. User choose subject to edit information       3. User enter information of subject to edit       4. User choose save       5. The system sends updated information to the database       6. System edit subject information on form and in database       7. Edit is complete       8. The use-case end    3. Delete       1. Admin choose subject to delete       2. Admin press button Delete       3. Admin confirmed want to delete       4. System through database check the constraints of the selected subject       5. System delete subject on form and in database       6. The use-case end    4. Search       1. Admin choose search box to search subject       2. Admin enter keyword subject need to search       3. System check keyword subject in database       4. System show list subject information       5. Search is complete       6. The use-case end |
| Alternative Flow | 1. Incorrect account and password: when the user enters the wrong account and password.    1. The system displays the login form again and attached the account does not exist or is wrong on the account.    2. The system displays the login form again and encloses the wrong password. 2. Not create subject: when the user enter ID subject is the same or enter incomplete information.    1. The systems required re-enter information    2. If user want not create subject , they can choose button Cancel 3. Not edit subject: When the user Re-enter incomplete information    1. The systems required re-enter information    2. If user want not create subject , they can choose button Cancel 4. Not search subject: when keywords haven’t in the system    1. The systems show message box which content is “No results are found” |

#### Table 4 Scenario Manage Project

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC3 |
| Use-case Name | Project Management |
| Actor | User (Admin) |
| Description | Let users manipulate the system to manage the subject |
| Trigger | When the user presses the login button from the home form. |
| Pre-Condition(s) | Must have an account in the system.  Account has been decentralized.  The device is connected to the internet. |
| Post-Condition(s) | User successfully logged in.  Users can successfully and create, search, edit, delete, view, mark project  The system stores user information and reuses it for the whole system. |
| Basic Flow | 1. The system displays the login form.  2. User enter the account and password.  3. System displays checking login information.  4. Success: load the information the user has just logged in via the form home and displayed on the form management subject  5. User can select functions: create, edit, search, delete, view, mark project   1. Create    * 1. User choose button Create to create a new project      2. System require to enter project information      3. User enter project information      4. User choose save      5. System check project information in database      6. System add new project on form and in database      7. Add is complete      8. The use-case end 2. Edit    * 1. User choose button Edit to edit project information      2. User choose project to edit information      3. User enter information of project to edit      4. User choose save      5. The system sends updated information to the database      6. System edit project information on form and in database      7. Edit is complete      8. The use-case end 3. Delete    * 1. Admin choose project to delete      2. Admin press button Delete      3. Admin confirmed want to delete      4. System through database check the constraints of the selected project      5. System delete project on form and in database 4. Search    * 1. Admin choose search box to search project      2. Admin enter keyword project need to search ID or name of project      3. System check keyword project in database      4. System show list project information      5. Search is complete      6. The use-case end 5. View    * 1. Admin choose view to see list of project      2. System show list of project information      3. View is complete      4. Use case ends. 6. Mark    * 1. Admin choose search box to search project      2. Admin enter keyword project need to search ID or name of project      3. System check keyword project in database      4. System show list project information      5. Admin choose project and see information in order to mark      6. Admin will mark for project of student.      7. System will save grades of project on form and in database.      8. Use case ends. |
| Alternative Flow | 1. Incorrect account and password: when the user enters the wrong account and password. 2. The system displays the login form again and attached the account does not exist or is wrong on the account. 3. The system displays the login form again and encloses the wrong password. 4. Not create subject: when the user enter ID subject is the same or enter incomplete information 5. The systems required re-enter information 6. If user want not create subject , they can choose button Cancle 7. Not edit subject: when the user Re-enter incomplete information 8. The systems required re-enter information 9. If user want not create subject , they can choose button Cancel 10. Not search subject: when keywords haven’t in the system   The systems show message box which content is “No results are found”   1. Not mark project   a. The system required re-enter mark of project.  b. If user do not want to mark project, they can choose button Cancel. |

#### Table 5 Scenario Manage Lecturer

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC4 |
| Use-case Name | Manage Lecturer |
| Actor | Admin |
| Description | Allow administrators to manage lecturer . Include: manage lecturer’s information , statistic amount of project of lecturer ,search lecturer’s information, sort by alpha |
| Trigger | Admin login  Choose manage lecturer to form home  Choose one of four function to form manage lecturer |
| Pre-Condition(s) | Must have an account in the system.  Account has been granted access  The device is connected to the internet. |
| Post-Condition(s) | User selection function was successfully implemented |
| Basic Flow | 1. Admin choose Function Manage Lecturer’s Information to form manage lecturer 2. Add    * 1. Admin choose button Add to add a new lecturer      2. System require to enter information of a new lecturer’s information      3. Admin enter information of a new lecturer’s information      4. Admin choose save      5. System checks information of lecturer      6. System add a new lecturer on form      7. Add is complete      8. The use-case end 3. Edit    * 1. Admin choose button Edit to edit lecturer’s information      2. Admin choose lecturer’s information to edit information      3. Admin enters information of lecturer to edit      4. Admin choose save      5. System checks information of lecturer      6. System updates the information of lecturers      7. Edit is complete      8. The use-case end 4. Delete    * 1. Admin choose lecturer to delete      2. Admin press button Delete      3. Admin confirmed want to delete      4. System checks the constraints of the selected lecturer’s information      5. System delete a lecturer      6. Delete is complete      7. Use-case end 5. Admin choose Function Statistic Amount of Project of Lecturer to form manage lecturer 6. Semester    * 1. Admin choose Semester to static by semester      2. Admin choose lecturer to static      3. System performs the statistics      4. Statistics are displayed to the user      5. Static is complete      6. The use-case end 7. A school year    * 1. Admin choose A school year to static by semester      2. Admin choose lecturer to static      3. System performs the statistics      4. Statistics are displayed to the user      5. Static is complete      6. The use-case end 8. Admin choose function Search lecturer’s information to form manage lecturer 9. Search by id    * 1. Admin choose search by id to form Search lecturer’s information      2. Admin enter id of lecturer      3. System performs the search      4. System returns search results      5. Search is complete      6. The use-case end 10. Search by id-card number     * 1. Admin choose search by id-card to form Search lecturer’s information       2. Admin enter id-card of lecturer       3. System performs the search       4. System returns search results       5. Search is complete       6. The use-case end 11. Search by name     * 1. Admin choose search by name to form Search lecturer’s information       2. Admin enter name of lecturer       3. System performs the search       4. System returns search results       5. Search is complete       6. The use-case end 12. Admin choose function Sort by alpha to form manage lecturer 13. Admin click function sort 14. The system will sort users based on alphabetical 15. Sort is complete 16. The use-case end |
| Alternative Flow | * Manage lecturer’s information  1. Add 2. If admin don’t want to add a new lecturer’s information  * Admin choose button Ignore * A new lecturer’s information isn’t added  1. If user information isn’t invalid  * System notify lecturer’s information isn’t invalid, require admin re-enter * Return step a.3  1. Edit 2. If admin don’t want to edit user information  * Admin choose button Ignore * Lecturer’s information isn’t edited  1. If user information isn’t invalid  * System notify lecturer’s information isn’t invalid, require admin re-enter * Return step b.3  1. Delete   If admin don’t want to delete lecturer   * Admin choose button Ignore * Lecturer isn’t deleted * Statistic amount of project of lecturer  1. Semester   If admin don’t want to static by semester   * Admin choose button Ignore * Static isn’t performed  1. A school year   If admin don’t want to static by semester   * Admin choose button Ignore * Static isn’t performed * Search lecturer’s information  1. Search by id 2. If admin don’t want to search by id  * Admin choose button Ignore * Search isn’t performed  1. If id isn’t invalid  * System notify lecturer’s id isn’t invalid, require admin re-enter * Return step a.2  1. Search by id-card number 2. If admin don’t want to search by id-card number  * Admin choose button Ignore * Search isn’t performed  1. If id-card number isn’t invalid  * System notify id-card number of lecturer isn’t invalid, require admin re-enter * Return step b.2  1. Search by name 2. If admin don’t want to search by name  * Admin choose button Ignore * Search isn’t performed  1. If name number isn’t invalid  * System notify lecturer’s name isn’t invalid, require admin re-enter * Return step c.2 * Sort by alpha   If admin don’t want to continue sorting by name   * Admin re-click sort * Sort is cancelled |
| Non-Functional Requiement |  |

#### Table 6 Scenario Login

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC5 |
| Use-case Name | Login |
| (Actor) | User |
| Activation conditions | When user click login button from form homepage |
| Conditions before implementation | User must have account in the system |
| Request after implementation | Display name and service of user after login and load information in form homepage. |
| Main event stream | 1. System display login form. 2. User input username and pasword 3. System display to check information login. 4. Success: System load information of user after login to form homepage. 5. Use case ends. |
| Extra event stream | 1. User name and password incorrect: when user imput wrong user name and password. 2. System re-display form login and attach account is not exist or wrong user name 3. System re-display form login and attach account is not exist or wrong password. 4. User do not have account: when user click login button but do no have account   Notify admin in order to provide account. |

#### Table 7 Scenario Manage Class

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC6 |
| Use-case Name | Management Class |
| Actor | Admin |
| Description | Let users manipulate the system to manage the class |
| Trigger | When the user presses the login button from the home form. |
| Pre-Condition(s) | Must have an account in the system.  Account has been decentralized.  The device is connected to the internet. |
| Post-Condition(s) | 1. User successfully logged in.  2. Users can successfully perform operations with manage class’s information and manage student of class  3. The system stores user information and reuses it for the whole system. |
| Basic Flow | 1. The system displays the login form.  2. User enter the account and password.  3. System displays checking login information.  4. Success: load the information the user has just logged in via the form home and displayed on the form management class  5. User can select functions: Manage class’s information and manage student of class  - Manage class’s information  a. Add  a.1. User choose button Add to create a new class  a.2.System require to enter class information  a.3.User enter class information  a.4.User choose button Save  a.5.System check class information in database  a.6.System add new class on form and in database  a.7.Add is complete  a.8.The use-case end  b. Edit  b.1. User choose button Edit to edit class information  b.2.User choose class to edit information  b.3.User enter information of class to edit  b.4. User press button save  b.5. The system sends updated information to the database  b.6.System edit class information on form and in database  b.7.Edit is complete  b.8.The use-case end  c. Delete  c.1. Admin choose class to delete  c.2.Admin press button Delete  c.3. Admin confirmed want to delete  c.4.System through database check the constraints of the selected class  c.5.System delete class on form and in database  c.6. Delete is complete  c.7. The use-case end  - Manage student of class  a. Add  a.1. User choose a class and press button Add student  a.2.System show the list student  a.3.User chose the students need add in class  a.4.User press button Save  a.5.System check student information in database  a.6.System add students in database  a.7.Add is complete  a.8.The use-case end  b. Delete  b.1. Admin choose those students need delete  b.2.Admin press button Delete  b.3.System through database check the constraints of the selected students  b.4.System delete student on form and in database  b.5.Delete is complete  b.6.The use-case end |
| Alternative Flow | 1. Incorrect account and password: when the user enters the wrong account and password. 2. The system displays the login form again and attached the account does not exist or is wrong on the account. 3. The system displays the login form again and encloses the wrong password. 4. Not add class: when the user enter ID class is the same or enter incomplete information 5. The systems required re-enter information 6. If user want not add class, they can press button Cancel 7. Not edit class: when the user Re-enter incomplete information 8. The systems required re-enter information 9. If user want not edit class, they can press button Cancel |

#### Table 8 Scenario Manage Faculty

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC7 |
| Use-case Name | Manage Department |
| Actor | Admin |
| Description | Allow administrators to manage department. Include:manage department’s information, divide lecturer in each department , divide subject in each department |
| Trigger | 1. Admin login 2. Choose manage department to form home 3. Choose one of four function to form manage department |
| Pre-Condition(s) | 1. Must have an account in the system. 2. Account has been granted access 3. The device is connected to the internet. |
| Post-Condition(s) | Department selection function was successfully implemented |
| Basic Flow | * Admin choose Function Manage Department’s Information to form manage department  1. Add   a.1.Admin choose button Add to add a new department  a.2.System require to enter information of a new department’s information  a.3.Admin enter information of a new department’s information  a.4.Admin choose save  a.5.System checks information of department  a.6.System add a new department on form  a.7.Add is complete  a.8.The use-case end   1. Edit   b.1.Admin choose button Edit to edit department’s information  b.2.Admin choose department’s information to edit information  b.3.Admin enters information of department to edit  b.4.Admin choose save  b.5.System checks information of department  b.6.System updates the information of departments  b.7.Edit is complete  b.8.The use-case end   1. Delete   c.1. Admin choose department to delete  c.2.Admin press button Delete  c.3.Admin confirmed want to delete  c.4.System checks the constraints of the selected department’s information  c.5.System delete a department  c.6.Delete is complete  c.7.Use-case end   * Admin choose Function Divide Lecturer in each department  1. Admin choose department 2. Admin click button Add Lecturer to add lecturer for department 3. Admin choose lecturers from list lecturer 4. Admin click button Save 5. System performs division 6. Divide lecturer is complete 7. Use-case end  * Admin choose Function Divide Subject in each department  1. Admin choose department 2. Admin click button Add Subject to add subject for department 3. Admin choose subjects from list subject 4. Admin click button Save 5. System performs division 6. Divide subject is complete 7. Use-case end |
| Alternative Flow | * Manage department’s information  1. Add  * If admin don’t want to add a new department’s information * Admin choose button Ignore * A new department’s information isn’t added * If department information isn’t invalid * System notify department’s information isn’t invalid, require admin re-enter * Return step a.3  1. Edit  * If admin don’t want to edit department information * Admin choose button Ignore * Department’s information isn’t edited * If department information isn’t invalid * System notify department’s information isn’t invalid, require admin re-enter * Return step b.3  1. Delete  * If admin don’t want to delete department * Admin choose button Ignore * Department isn’t deleted * Divide Lecturer in each department * If admin don’t want to divide Lecturer * Admin choose button Ignore * Divide lecturer isn’t performed * Divide Subject in each department * If admin don’t want to divide Subject * Admin choose button Ignore * Divide subject isn’t performed |
| Non-Functional Requiement |  |

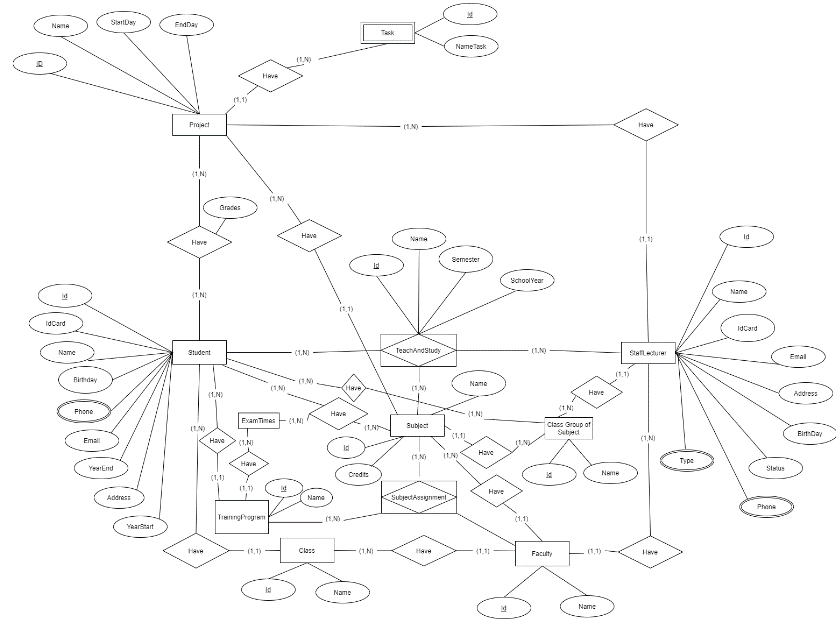
#### Table 9 Scenario Decentralization

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC8 |
| Use-case Name | Decentralized Management |
| Actor | User (Admin) |
| Description | Let users manipulate the system to manage the decentralization |
| Trigger | When the user presses the login button from the home form. |
| Pre-Condition(s) | Must have an account in the system.  Account is an admin.  The device is connected to the internet. |
| Post-Condition(s) | User successfully logged in.  Users can successfully and view, delete, edit, decentralize user  The system stores user information and reuses it for the whole system. |
| Basic Flow | 1. The system displays the login form. 2. User enter the account and password. 3. System displays checking login information. 4. Success: load the information the user has just logged in via the form home and displayed on the form decentralized management 5. User can select functions: view, delete, edit, decentralize user    1. View       1. User choose button View to view list of decentralized user       2. User can view all of information of decentralized user       3. The use-case end 6. Edit    * 1. User choose button Edit to edit decentralization      2. User choose user to edit      3. User enter information of user to edit or add decentralization      4. User choose save      5. The system sends updated information to the database      6. b.6.System edit decentralization of user on form and in database      7. b.7.Edit is complete      8. b.8.The use-case end 7. Disable    * 1. Admin choose user to disable decentralization      2. Admin press disable tick to disable decentralization of user      3. Admin confirmed want to disable      4. System through database check the constraints of the selected user      5. System disable on form and in database 8. Decentralize    * 1. Admin choose user to decentralize      2. Admin press tick to enable decentralization of user      3. Admin confirmed want to delete decentralization      4. System through database check the constraints of the selected user      5. Search decentralize on form and in database      6. The use-case end |
| Alternative Flow | 1. View   If admin don’t want to view a list of decentralized user   * Admin choose button Ignore  1. Edit 2. If admin don’t want to edit decentralization  * Admin choose button Ignore * Department’s information isn’t edited  1. If user information isn’t invalid  * System notify user information isn’t invalid, require admin re-enter * Return step b.3  1. Delete   If admin don’t want to delete decentralization   * Admin choose button Ignore * Decentralization isn’t deleted  1. Decentralization   If admin don’t want to delete decentralization  • Admin choose button Ignore  • Decentralization isn’t decentralized |

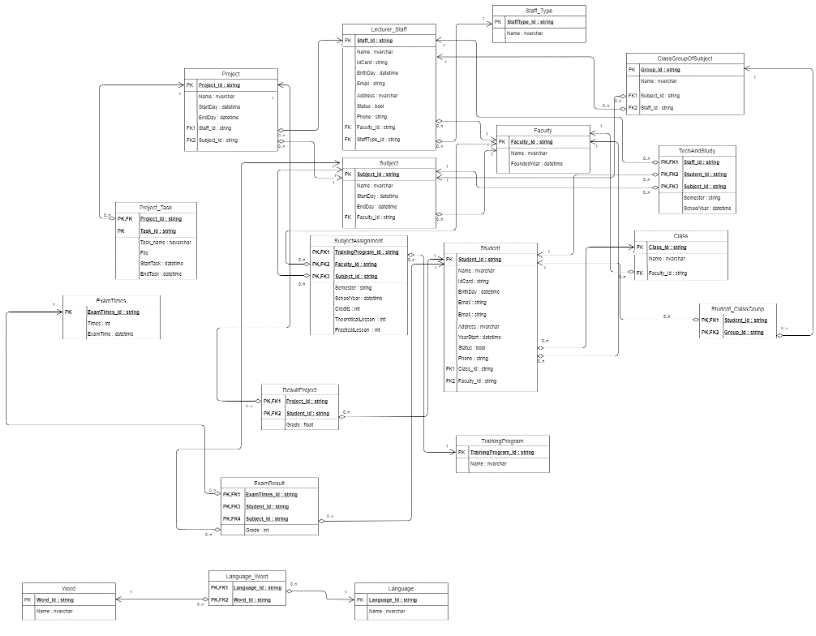
#### Table 10 Scenario Multi-language

|  |  |
| --- | --- |
| Use-case | Description |
| Use-case ID | UC10 |
| Use-case Name | Management Language |
| Actor | User |
| Description | Let users manipulate the system to manage the language |
| Trigger | When the user presses the language button from the home form. |
| Pre-Condition(s) | 1.Users open the home form from browser  2.The device is connected to the internet. |
| Post-Condition(s) | 1. User open the home form successfully  2. Users can successfully and choose language or set language default  3. The system confirms language and uses it for the whole system. |
| Basic Flow | 1. The system displays the home form  2. The system display set language button  3. User click set language button to change language or set default  4. System confirm that the user has selected  5. System uses language for the whole system |
| Alternative Flow | * If user do not want to change language, they can choose button cancel. * If user do not click language button, system will set language automatically. |

## Design Database



#### Picture 2 ER Diagram



#### Picture 3 Class diagram

#### Table 11 of Datatypes

1. Project ( Project\_ID , Name ,StartDay, EndDay, Staff\_ID ,Subject\_ID)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Project\_ID | Mã số của project | String (9) |
| ProjectName | Tên của project | String( 30) |
| S\_Day | Ngày bắt đầu của project | Datetime (mm/dd/yy) |
| E\_Day | Ngày kết thúc của project | Datetime (mm/dd/yy) |
| Staff\_ID | Mã số của nhân viên | String (9) |
| Subject\_ID | Mã số môn học | String (9) |

1. Project\_Task ( Project\_ID, Task, StartTask, EndTask )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Project\_ID | Mã số của project | String (9) |
| Task |  | String (9) |
| S\_Task | Ngày bắt đầu của task | Datetime (mm/dd/yy) |
| E\_Task | Ngày kết thúc của task | Datetime (mm/dd/yy) |
|  |  |  |

1. ResultProject ( Project\_ID, Student\_ID, Grade )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Project\_ID | Mã số của project | String (9) |
| Student\_ID | Mã số sinh viên | String (9) |
| Grade | Điểm của sinh viên | Int |

1. Student ( Student\_ID, ID\_Card, Name,BirthDay, Email, Address, YearStart, Status, Class\_ID, Faculty\_ID )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Student\_ID | Mã số sinh viên | String (9) |
| Student\_Name | Tên sinh viên | String(30) |
| Card\_ID | Sô chứng minh nhân dân của sinh viên | String (9) |
| BirthDay | Ngày sinh của sinh viên | Datetime (mm/dd/yy) |
| Email | Địa chỉ email của sinh viên | String (30) |
| Address | Địa chỉ của sinh viên | String (30) |
| S\_Year | Năm bắt đầu học của sinh viên | Datetime (YY) |

1. Staff - Lecturer (Staff\_ID , Name, ID\_Card, Birthday, Email, Address, Status, Faculty\_ID, StaffType\_ID )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Staff\_ID | Mã số nhân viên | String (9) |
| Staff\_Name | Tên của nhân viên | String(30) |
| Card\_ID | Sô chứng minh nhân dân của nhân viên | String (9) |
| BirthDay | Ngày sinh của nhân viên | Datetime (mm/dd/yy) |
| Email | Địa chỉ email của nhân viên | String (30) |
| Address | Địa chỉ của nhân viên | String (30) |
| Status | Trạng thái của nhân viên | Bool |
| Faculty\_ID | Mã số khoa | String(9) |
| StaffType\_ID | Mã số loại nhân viên | String(9) |

1. Staff type ( StaffType\_ID, Name Staff type )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| StaffType\_ID | Mã số loại nhân viên | String(9) |
| Staff\_Name | Tên loại nhân viên | String(30) |

1. Subject ( Subject\_ID, Name, Credits, Faculty\_ID )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Subject\_ID | Mã số môn học | String(9) |
| Name | Tên môn học | String(30) |
| Credits | Số tín chỉ môn học | int |

1. Subject assignment (TrainingProgram\_ID , Faculty\_ID , Subject\_ID , Staff\_ID, Year, Semester , Credits, Theoretical lesson, Practical lesson )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| TrainingProgram\_ID | Mã sô chương trình đào tạo | String(9) |
| Course\_ID | Mã số khoá học | String(9) |
| Subject\_ID | Mã môn học | String(9) |
| Year | Năm học | Datetime |
| Semester | Học kì | String(9) |
| Credits | Số tín chỉ môn học | Int |
| Theoretical lesson | Số tín chỉ lý thuyết | int |
| Practical lesson | Số tín chỉ thực hành | int |

1. TeachAndStudy (Staff\_ID, Student\_ID, Subject\_ID, Semester, Year)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Subject\_ID | Mã số môn học | String(9) |
| Staff\_ID | Mã nhân viên | String(9) |
| Student\_ID | Mã số sinh viên | String (9) |
| Semester | Học kì | String(9) |
| Year | Năm học | Datetime |

1. ClassGroupOfSubject ( Group\_ID ,Name, Subject\_ID, Staff\_ID )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Group\_ID | Mã nhóm | String(9) |
| Name | Tên Nhóm | String(30) |
| Subject\_ID | Mã số môn học | String(9) |
| Staff\_ID | Mã nhân viên | String(9) |

1. Student\_ClassGroup (Student\_ID, Group\_ID)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Student\_ID | Mã số sinh viên | String (9) |
| Group\_ID | Mã nhóm | String(9) |

1. Class (Class\_ID, Name, Faculty\_ID , Course\_ID , TrainingProgram\_ID )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Class\_ID | Mã số lớp | String(9) |
| Name | Tên lớp | String(30) |
| Faculty\_ID | Mã số khoa | String(9) |
| TrainingProgram\_ID | Mã số chương trình đào tạo | String(9) |

1. Faculty ( Faculty\_ID, Name , FoundedYear)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Faculty\_ID | Mã số khoa | String(9) |
| Name | Tên khoa | String(30) |
| FoundedYear | Năm thành lập | Datetime |

1. The course ( Course\_ID, YearStart, YearEnd)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Course\_ID | Mã số khoá học | String(9) |
| S\_year | Năm bắt đầu học của sinh viên | Datetime |
| E\_Year | Năm kết thúc của sinh viên | Datetime |

1. Training program ( TrainingProgram\_ID , Name)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| TrainingProgram\_ID | Mã chương trình đào tạo | String(9) |
| Name | Tên của chương trình đào tạo | String(30) |

1. Exam times( ExamTimes\_ID, TrainingProgram\_ID, Subject\_ID, Times , exam time )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| ExamTimes\_ID | Mã của lần thi | String(9) |
| TrainingProgram\_ID | Mã chương trình đào tạo | String(9) |
| Subject\_ID | Mã số môn học | String(9) |
| Times | Lần thi | String(9) |
| Exam time | Thời gian thi | Datetime |

1. Exam Result ( Times, TrainingProgram\_ID, Subject\_ID, Student\_ID, Grade )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Times | Lần thi | String(9) |
| TrainingProgram\_ID | Mã chương trình đào tạo | String(9) |
| Subject\_ID | Mã số môn học | String(9) |
| Grade | Điểm kết quả thi của sinh viên | int |

1. Language ( Language\_ID , Name)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Language\_ID | Mã ngôn ngữ | String(9) |
| Name | Tên ngôn ngữ | String(30) |

1. Word ( Word\_ID, Name)

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Word\_ID | Mã của từ | String(9) |
| Name | Tên của từ | String(30) |

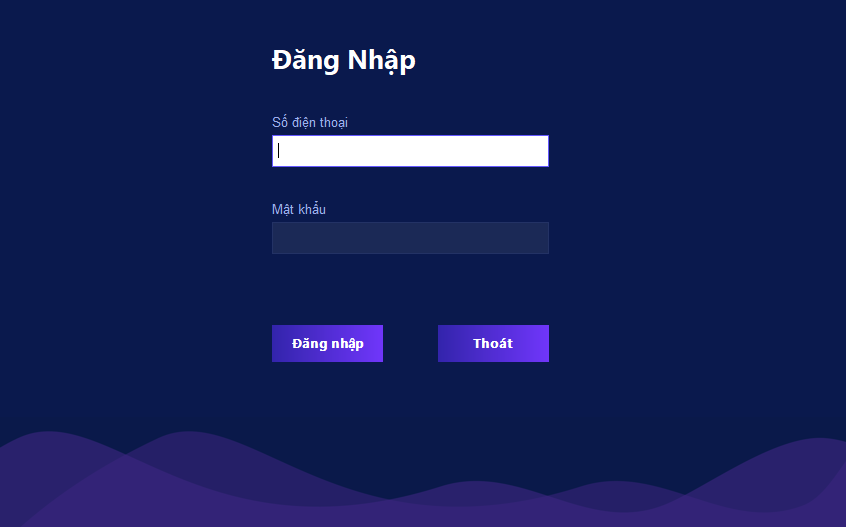
1. Language\_Word ( Language\_ID, Word\_ID )

|  |  |  |
| --- | --- | --- |
| Tên thuộc tính | Diễn giải | Kiểu dữ liệu |
| Language\_ID | Mã của ngôn ngữ | String(9) |
| Word\_ID | Mã của từ | String(9) |

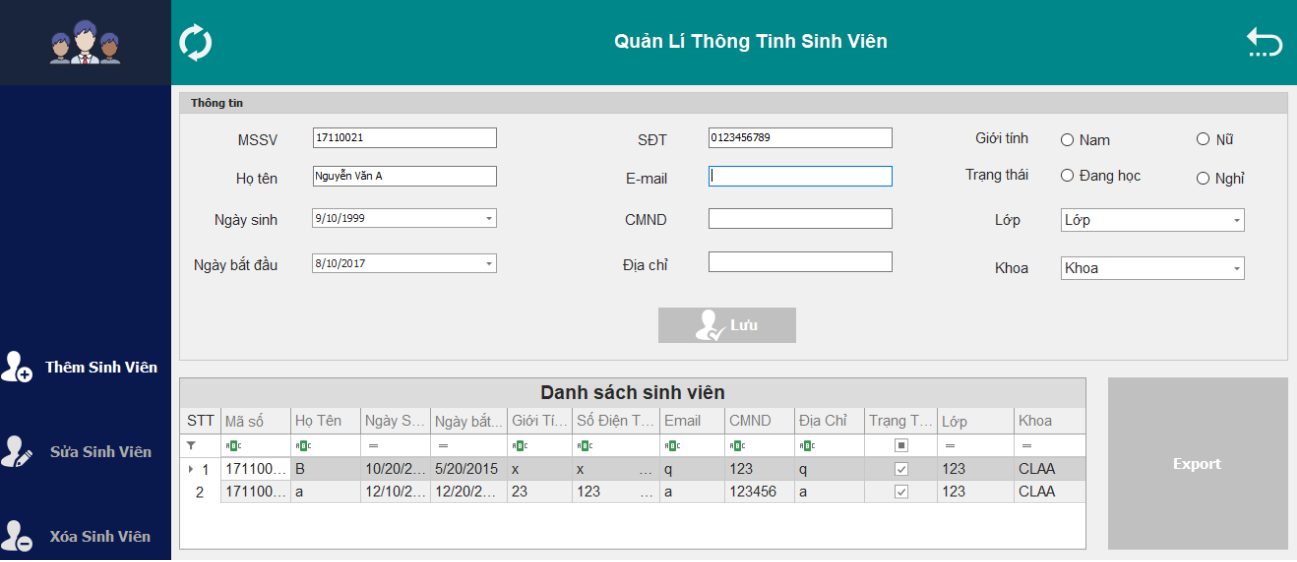
## Design Interface



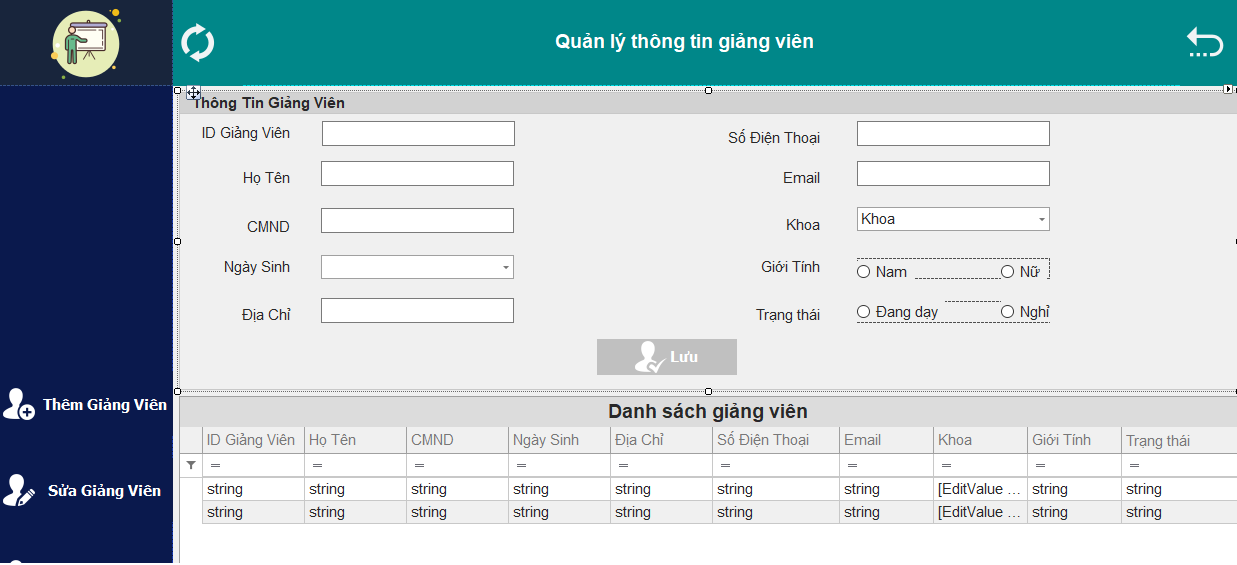
#### Picture 3 Form Home



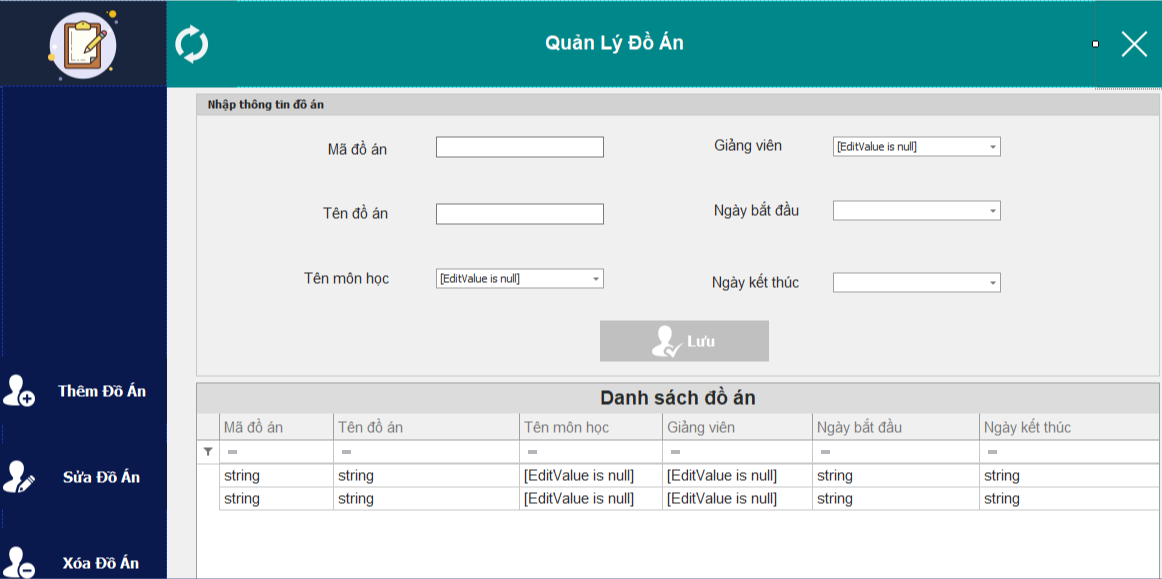
#### Picture 4 Form Login



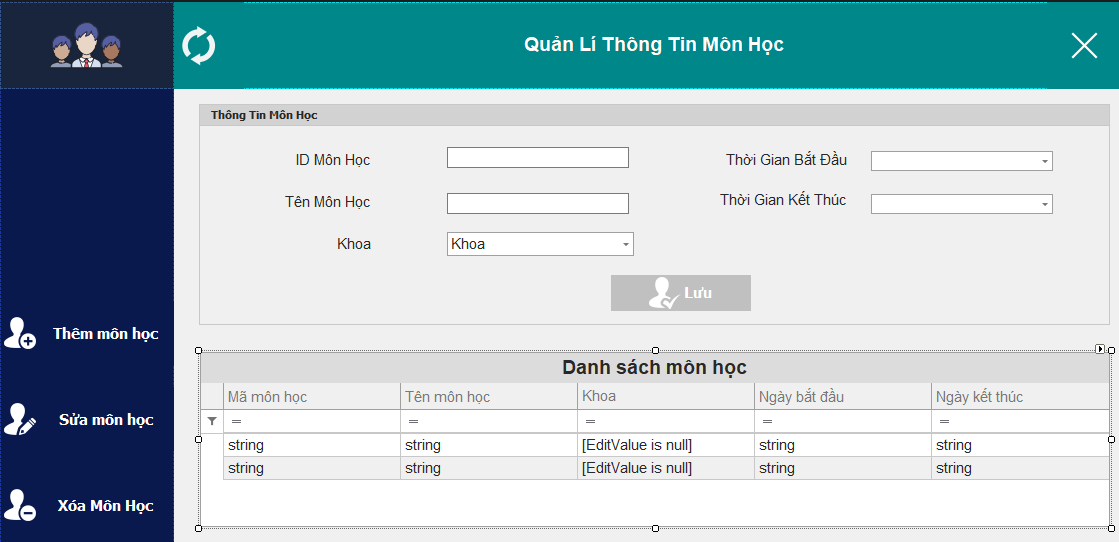
#### Picture 5 form Manage student information



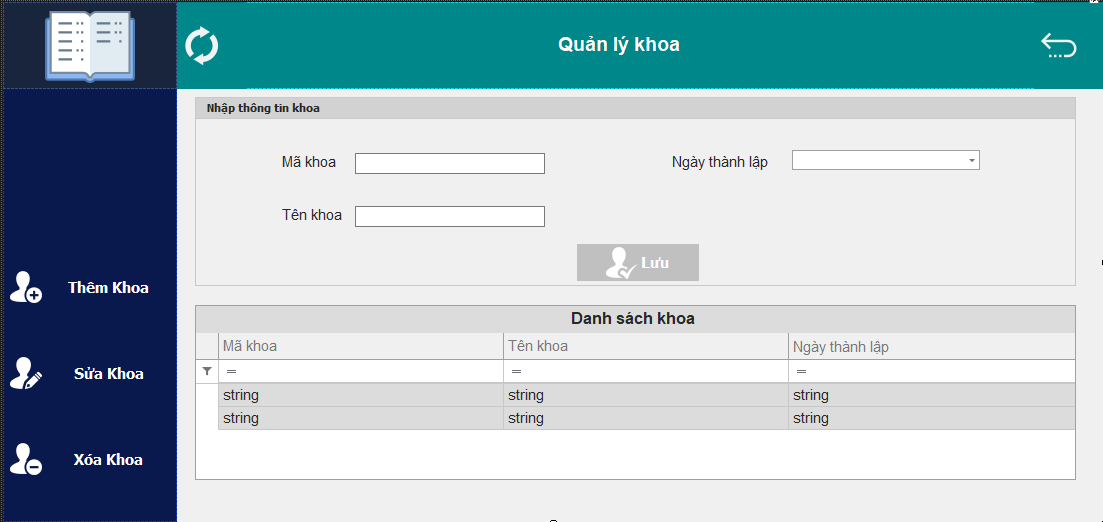
#### Picture 6 from Manage lecturer information



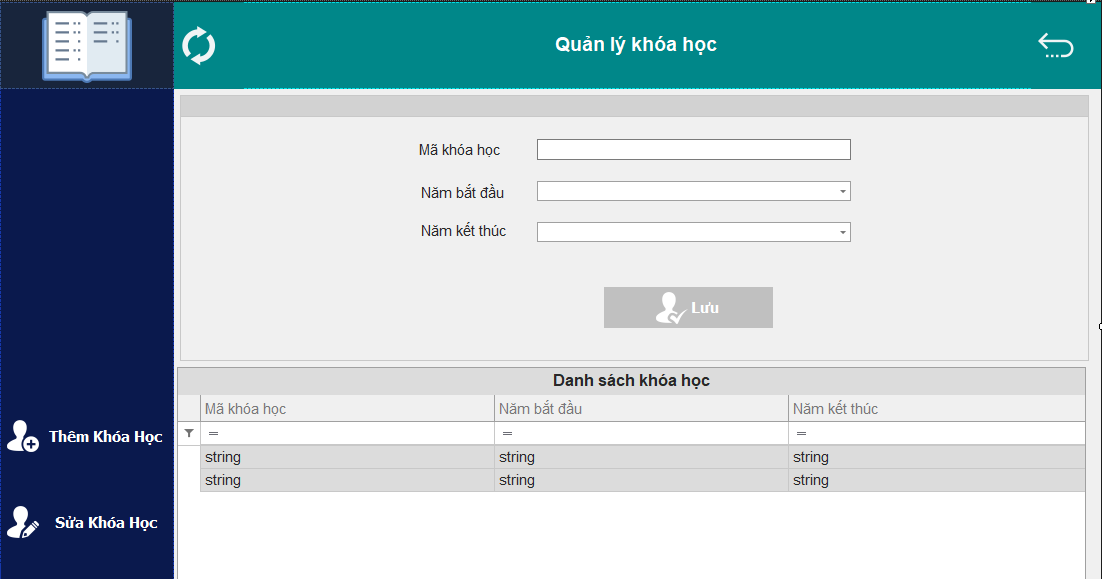
#### Picture 7 form Manage project student



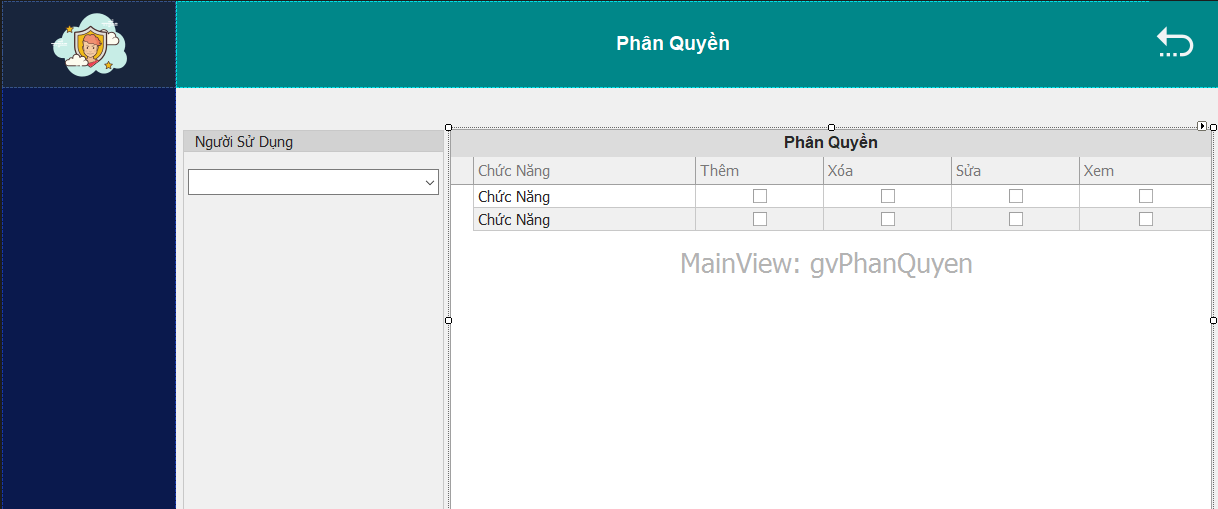
#### Picture 8 form Manage subject



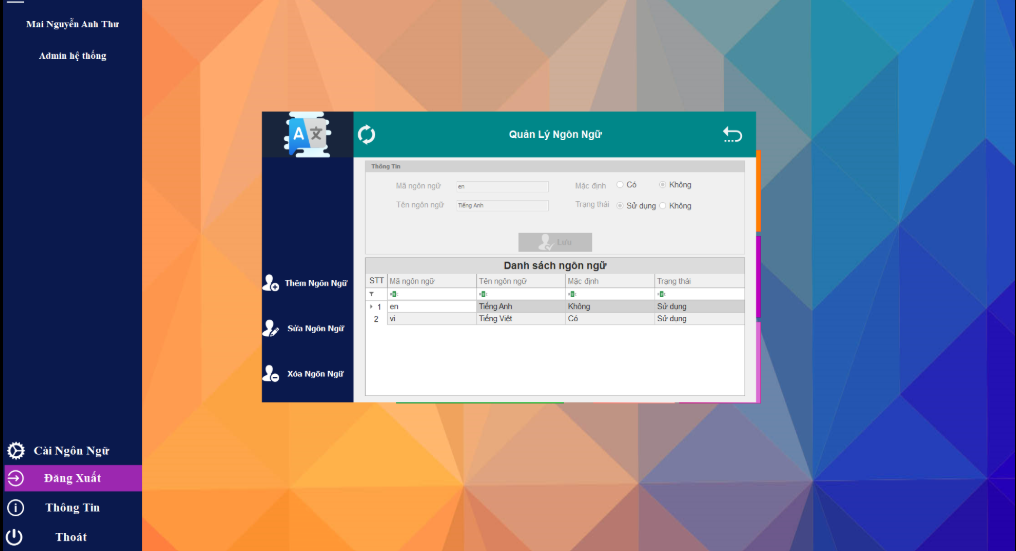
#### Picture 9 Manage Faculty

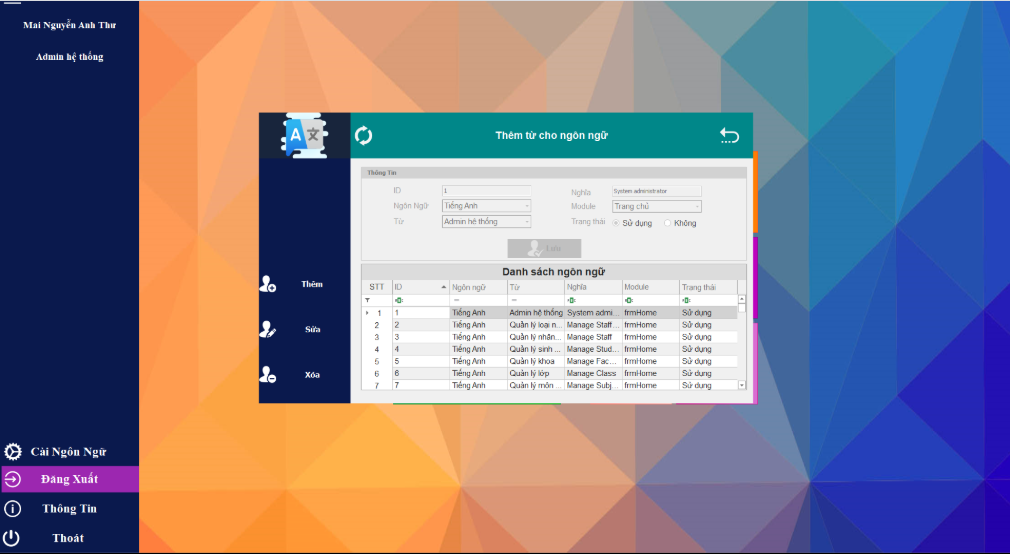


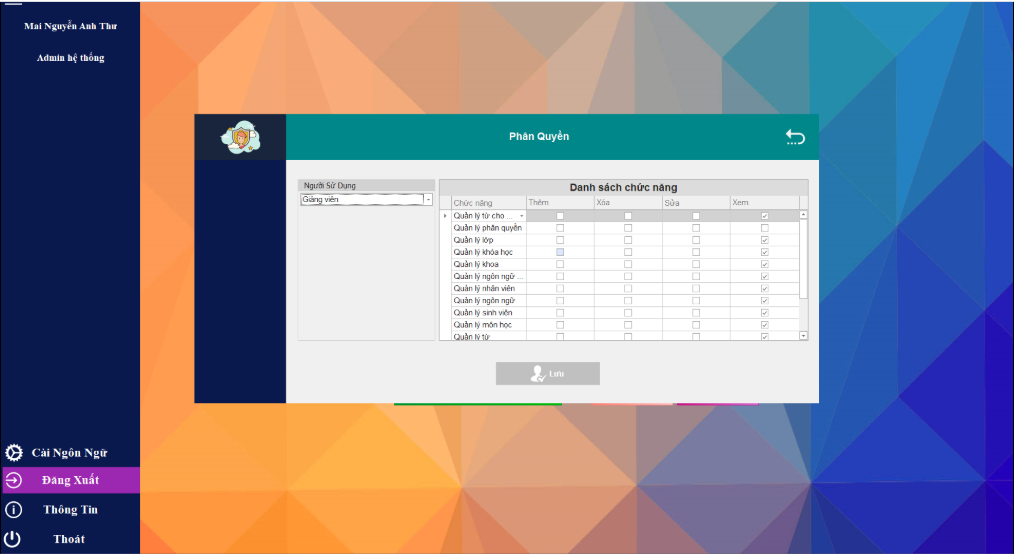
#### Picture 10 form Manage course



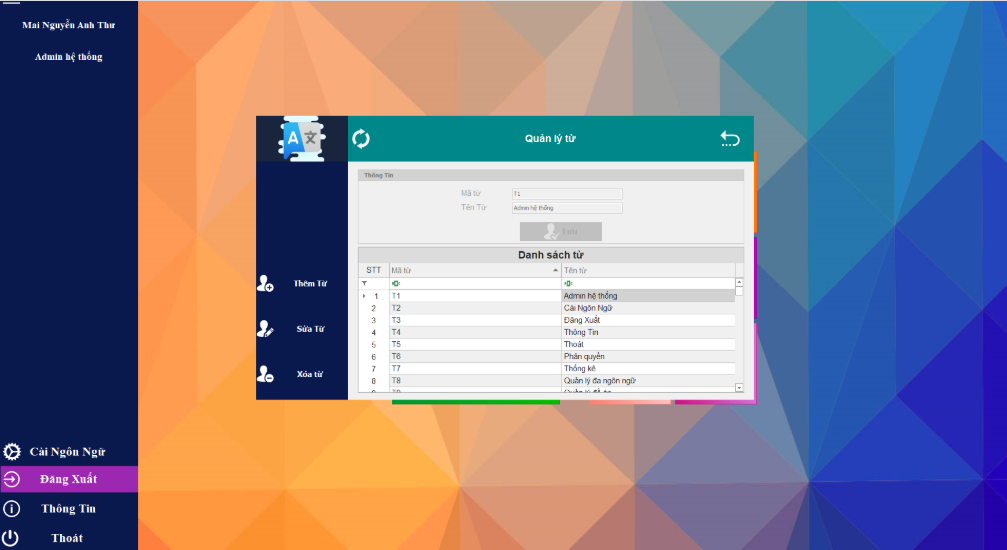
#### Picture 11 form Decentralization

****





#### Picture 12 form Manage Multi-language



#### Picture 13 form Manage words

## Interface Spectification

#### Table 12 Interface Spectification

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Name | Classify | Make-up | Function |
| 1 | btnAdd, btnDelete, btnUpdate | Button | //Add  this.btnAdd.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)  | System.Windows.Forms.AnchorStyles.Left)  | System.Windows.Forms.AnchorStyles.Right)));  this.btnAdd.Appearance.BackColor = System.Drawing.Color.Transparent;  this.btnAdd.Appearance.Font = new System.Drawing.Font("Tahoma", 11.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.btnAdd.Appearance.ForeColor = System.Drawing.Color.White;  this.btnAdd.Appearance.Options.UseBackColor = true;  this.btnAdd.Appearance.Options.UseFont = true;  this.btnAdd.Appearance.Options.UseForeColor = true;  this.btnAdd.AppearanceHovered.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(60)))), ((int)(((byte)(71)))), ((int)(((byte)(100)))));  this.btnAdd.AppearanceHovered.Font = new System.Drawing.Font("Tahoma", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.btnAdd.AppearanceHovered.ForeColor = System.Drawing.Color.White;  this.btnAdd.AppearanceHovered.Options.UseBackColor = true;  this.btnAdd.AppearanceHovered.Options.UseFont = true;  this.btnAdd.AppearanceHovered.Options.UseForeColor = true;  this.btnAdd.ButtonStyle = DevExpress.XtraEditors.Controls.BorderStyles.NoBorder;  this.btnAdd.ImageOptions.Image = ((System.Drawing.Image)(resources.GetObject("btnAdd.ImageOptions.Image")));  this.btnAdd.Location = new System.Drawing.Point(0, 369);  this.btnAdd.Name = "btnAdd";  this.btnAdd.Size = new System.Drawing.Size(171, 64);  this.btnAdd.TabIndex = 16;  this.btnAdd.TabStop = false;  this.btnAdd.Text = "Thêm Sinh Viên";  this.btnAdd.Click += new System.EventHandler(this.btnAdd\_Click);  // btnUpdate  //  this.btnUpdate.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)  | System.Windows.Forms.AnchorStyles.Left)  | System.Windows.Forms.AnchorStyles.Right)));  this.btnUpdate.Appearance.BackColor = System.Drawing.Color.Transparent;  this.btnUpdate.Appearance.Font = new System.Drawing.Font("Tahoma", 11.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.btnUpdate.Appearance.ForeColor = System.Drawing.Color.White;  this.btnUpdate.Appearance.Options.UseBackColor = true;  this.btnUpdate.Appearance.Options.UseFont = true;  this.btnUpdate.Appearance.Options.UseForeColor = true; this.btnUpdate.AppearanceHovered.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(60)))), ((int)(((byte)(71)))), ((int)(((byte)(100)))));  this.btnUpdate.AppearanceHovered.Font = new System.Drawing.Font("Tahoma", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.btnUpdate.AppearanceHovered.Options.UseBackColor = true; this.btnUpdate.AppearanceHovered.Options.UseFont = true;  this.btnUpdate.ButtonStyle = DevExpress.XtraEditors.Controls.BorderStyles.NoBorder;  this.btnUpdate.ImageOptions.Image = ((System.Drawing.Image)(resources.GetObject("btnUpdate.ImageOptions.Image")));  this.btnUpdate.Location = new System.Drawing.Point(0, 455);  this.btnUpdate.Name = "btnUpdate";  this.btnUpdate.Size = new System.Drawing.Size(171, 64);  this.btnUpdate.TabIndex = 18;  this.btnUpdate.TabStop = false;  this.btnUpdate.Text = "Sửa Sinh Viên";  this.btnUpdate.Click += new System.EventHandler(this.btnUpdate\_Click);  // btnDelete  //  this.btnDelete.Anchor = ((System.Windows.Forms.AnchorStyles)((((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Bottom)  | System.Windows.Forms.AnchorStyles.Left)  | System.Windows.Forms.AnchorStyles.Right)));  this.btnDelete.Appearance.BackColor = System.Drawing.Color.Transparent;  this.btnDelete.Appearance.Font = new System.Drawing.Font("Tahoma", 11.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.btnDelete.Appearance.ForeColor = System.Drawing.Color.White;  this.btnDelete.Appearance.Options.UseBackColor = true;  this.btnDelete.Appearance.Options.UseFont = true;  this.btnDelete.Appearance.Options.UseForeColor = true;  this.btnDelete.AppearanceHovered.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(60)))), ((int)(((byte)(71)))), ((int)(((byte)(100)))));  this.btnDelete.AppearanceHovered.Font = new System.Drawing.Font("Tahoma", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.btnDelete.AppearanceHovered.Options.UseBackColor = true;  this.btnDelete.AppearanceHovered.Options.UseFont = true;  this.btnDelete.ButtonStyle = DevExpress.XtraEditors.Controls.BorderStyles.NoBorder;  this.btnDelete.ImageOptions.Image = ((System.Drawing.Image)(resources.GetObject("btnDelete.ImageOptions.Image")));  this.btnDelete.Location = new System.Drawing.Point(0, 541);  this.btnDelete.Name = "btnDelete";  this.btnDelete.Size = new System.Drawing.Size(170, 64);  this.btnDelete.TabIndex = 17;  this.btnDelete.TabStop = false;  this.btnDelete.Text = "Xóa Sinh Viên";  this.btnDelete.Click += new System.EventHandler(this.btnDelete\_Click); | Button use in order to set status for form  When you click btnAdd so 2 button remaining will be locked  When you click btnUpdate  so all of data will be showed into groupboxcontrol  When you click btnDelete 1 line will delete in gridviewcontrol and Database |
| 2 | txtFullName\_KeyPress | TextBox | this.txtFullName.Location = new System.Drawing.Point(136, 75);  this.txtFullName.Margin = new System.Windows.Forms.Padding(4);  this.txtFullName.Name = "txtFullName";  this.txtFullName.Size = new System.Drawing.Size(187, 21);  this.txtFullName.TabIndex = 21;  this.txtFullName.KeyPress += new System.Windows.Forms.KeyPressEventHandler(this.txtFullName\_KeyPress); | User input for the program in groupboxcontrol and result will show into the gridviewcoontrol |
| 3 | gvStudentList | Gridviewcontrol | this.gvStudentList.Appearance.EvenRow.BackColor = System.Drawing.Color.FromArgb(((int)(((byte)(240)))), ((int)(((byte)(240)))), ((int)(((byte)(240)))));  this.gvStudentList.Appearance.EvenRow.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.gvStudentList.Appearance.EvenRow.Options.UseBackColor = true;  this.gvStudentList.Appearance.EvenRow.Options.UseFont = true;  this.gvStudentList.Appearance.HeaderPanel.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.gvStudentList.Appearance.HeaderPanel.Options.UseFont = true;  this.gvStudentList.Appearance.OddRow.BackColor = System.Drawing.Color.White;  this.gvStudentList.Appearance.OddRow.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.gvStudentList.Appearance.OddRow.Options.UseBackColor = true;  this.gvStudentList.Appearance.OddRow.Options.UseFont = true;  this.gvStudentList.Appearance.Row.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.gvStudentList.Appearance.Row.Options.UseFont = true;  this.gvStudentList.Appearance.ViewCaption.Font = new System.Drawing.Font("Arial", 14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.gvStudentList.Appearance.ViewCaption.Options.UseFont = true;  this.gvStudentList.AppearancePrint.Row.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.gvStudentList.AppearancePrint.Row.Options.UseFont = true;  this.gvStudentList.Columns.AddRange(new DevExpress.XtraGrid.Columns.GridColumn[] {  this.colStudentID,  this.colFullName,  this.colBirthday,  this.colStartYear,  this.ColSex,  this.colPhoneNumber,  this.colEmail,  this.colIDCard,  this.colAddress,  this.colStatus,  this.colClass,  this.colFaculty});  this.gvStudentList.DetailHeight = 377;  this.gvStudentList.GridControl = this.gcListStudent;  this.gvStudentList.Name = "gvStudentList";  this.gvStudentList.OptionsBehavior.Editable = false;  this.gvStudentList.OptionsSelection.MultiSelect = true;  this.gvStudentList.OptionsView.EnableAppearanceEvenRow = true;  this.gvStudentList.OptionsView.EnableAppearanceOddRow = true;  this.gvStudentList.OptionsView.ShowAutoFilterRow = true;  this.gvStudentList.OptionsView.ShowGroupPanel = false;  this.gvStudentList.OptionsView.ShowViewCaption = true;  this.gvStudentList.ViewCaption = "Danh sách sinh viên";  this.gvStudentList.CustomDrawRowIndicator += new DevExpress.XtraGrid.Views.Grid.RowIndicatorCustomDrawEventHandler(this.gvStudentList\_CustomDrawRowIndicator);  this.gvStudentList.SelectionChanged += new DevExpress.Data.SelectionChangedEventHandler(this.gvStudentList\_SelectionChanged);  this.gvStudentList.RowCountChanged += new System.EventHandler(this.gvStudentList\_RowCountChanged); | When user input information in groupboxcontrol  .The information will be showed in Gridviewcontrol |
| 4 | radNam\_CheckedChanged | RadioButton | this.radNam.AutoSize = true;  this.radNam.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.radNam.Location = new System.Drawing.Point(15, 0);  this.radNam.Name = "radNam";  this.radNam.Size = new System.Drawing.Size(57, 21);  this.radNam.TabIndex = 39;  this.radNam.TabStop = true;  this.radNam.Text = "Nam";  this.radNam.UseVisualStyleBackColor = true;  this.radNam.CheckedChanged += new System.EventHandler(this.radNam\_CheckedChanged); | Choose Sex Male/Female |
| 6 | grpInformationStudent | GroupBoxControl | this.grpInformationStudent.Controls.Add(this.panel2);  this.grpInformationStudent.Controls.Add(this.panel1);  this.grpInformationStudent.Controls.Add(this.dteStartYear);  this.grpInformationStudent.Controls.Add(this.label10);  this.grpInformationStudent.Controls.Add(this.btnSave);  this.grpInformationStudent.Controls.Add(this.lkeFaculty);  this.grpInformationStudent.Controls.Add(this.lkeClass);  this.grpInformationStudent.Controls.Add(this.label12);  this.grpInformationStudent.Controls.Add(this.label11);  this.grpInformationStudent.Controls.Add(this.label9);  this.grpInformationStudent.Controls.Add(this.txtAddress);  this.grpInformationStudent.Controls.Add(this.label8);  this.grpInformationStudent.Controls.Add(this.txtIDCard);  this.grpInformationStudent.Controls.Add(this.txtEmail);  this.grpInformationStudent.Controls.Add(this.txtPhoneNumber);  this.grpInformationStudent.Controls.Add(this.dteBirthday);  this.grpInformationStudent.Controls.Add(this.txtFullName);  this.grpInformationStudent.Controls.Add(this.label7);  this.grpInformationStudent.Controls.Add(this.label6);  this.grpInformationStudent.Controls.Add(this.label5);  this.grpInformationStudent.Controls.Add(this.label4);  this.grpInformationStudent.Controls.Add(this.label3);  this.grpInformationStudent.Controls.Add(this.label2);  this.grpInformationStudent.Controls.Add(this.txtID);  this.grpInformationStudent.Controls.Add(this.label1);  this.grpInformationStudent.Location = new System.Drawing.Point(182, 94);  this.grpInformationStudent.Margin = new System.Windows.Forms.Padding(4);  this.grpInformationStudent.Name = "grpInformationStudent";  this.grpInformationStudent.Size = new System.Drawing.Size(1127, 273);  this.grpInformationStudent.TabIndex = 43;  this.grpInformationStudent.Text = "Thông tin"; | User input data in GroupBoxControl |
| 7 | colFullName | GridViewControl | this.colFullName.AppearanceCell.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(163)));  this.colFullName.AppearanceCell.Options.UseFont = true;  this.colFullName.AppearanceHeader.Font = new System.Drawing.Font("Arial", 11.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));  this.colFullName.AppearanceHeader.Options.UseFont = true;  this.colFullName.Caption = "Họ Tên";  this.colFullName.FieldName = "StrStudentName";  this.colFullName.Name = "colFullName";  this.colFullName.Visible = true;  this.colFullName.VisibleIndex = 1;  this.colFullName.Width = 68; | This column display information in GridViewCntrol |

## Methods spectification in class

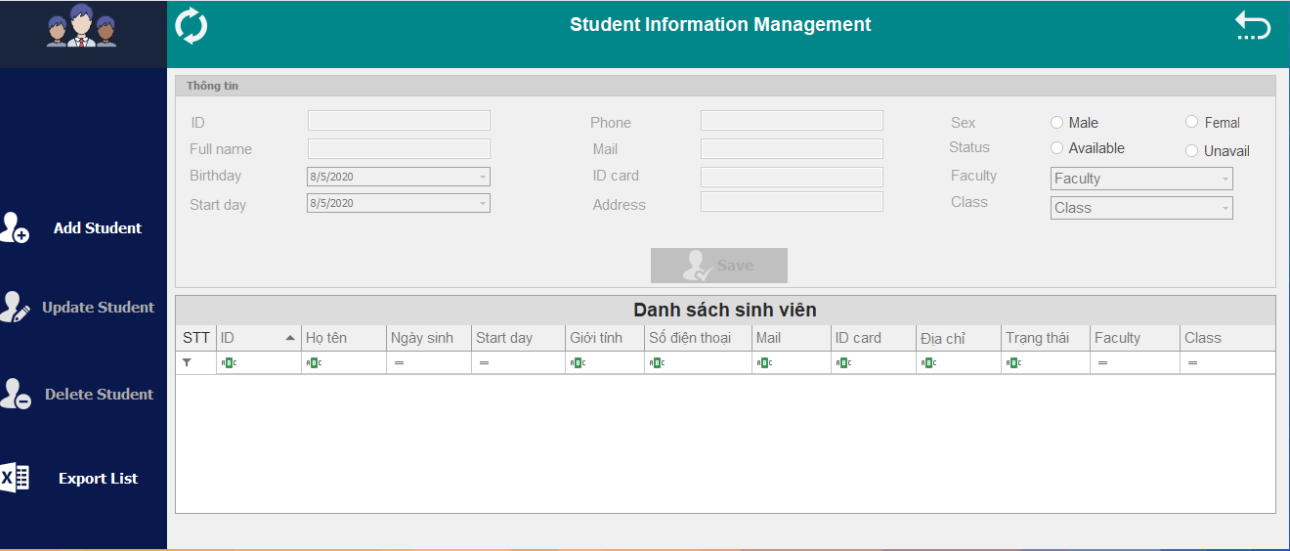
#### Table 13 Category of class for layers

|  |  |  |  |
| --- | --- | --- | --- |
| No | Name of class | Purpose | Student in charge |
| 1 | Interface | Interface is a solotion to support multiple inheritance  Declare functions to and contain no content.The default is always public  Support inheritance from the ViewModel to other classes. |  |
| 2 | Model | Holds the data and has nothing to do with any the business logic |  |
| 3 | ViewModel | The intermediate layer between View and Model can be considered as a replacement component Controller in MVC model .It contains the code needed to perform data binding, command |  |
| 4 | View | It simply holds the formatted data and essentially delegates to the Model. |  |

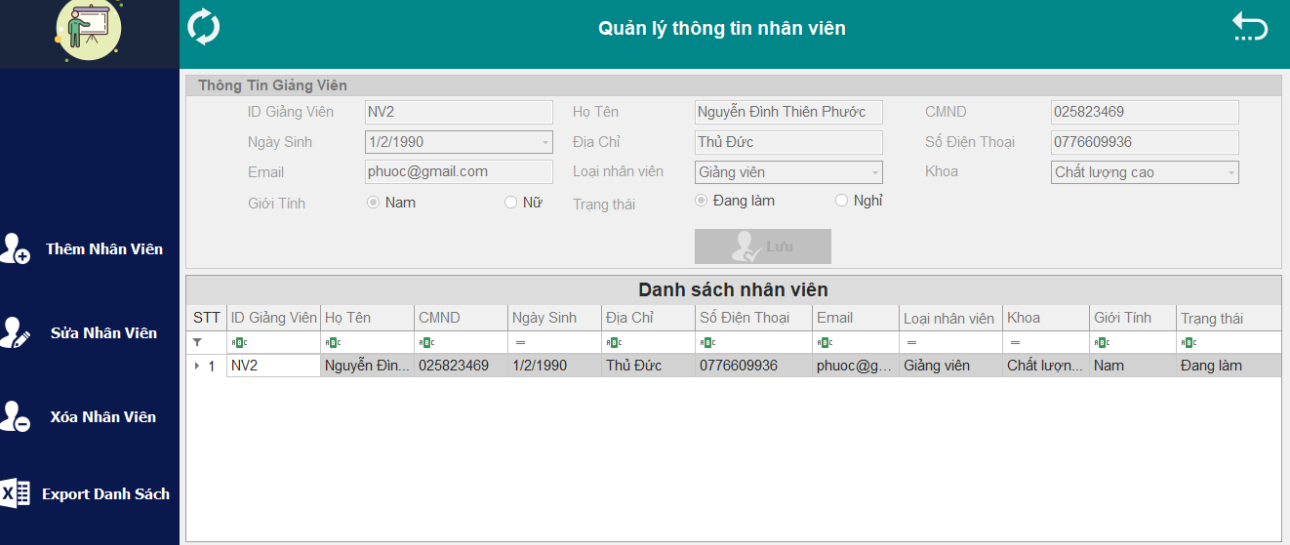
#### Table 14 Methods spectification in class

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Method | Purpose | FileName, line declare |
| 1 | bool addNewStudent(StudentModel student);  bool updateCurrentStudent(StudentModel student);  bool deleteCurrentStudent(StudentModel student);  input :  output : | Add, update, delete information from form to the database  Interaction between form and database | ManageProjectStudent\_Interface/IStudent.cs  (15,16,17) |
| 2 | private void \_loadData()  input :  output: | Loading data | frmManageStudentInformation.cs  (87) |
| 3 | private void \_getData()  input:  output: | Take data user input in order to adding into database | frmManageStudentInformation.cs  (142) |
| 4 | private void \_setStatusForm()  input:  output: | handle event button for set status for form | frmManageStudentInformation.cs  (40) |
| 5 | private void gvStudentList\_SelectionChanged(object sender, DevExpress.Data.SelectionChangedEventArgs e) | Select 1 line in gridview display in groupbox | frmManageStudentInformation.cs  (222) |

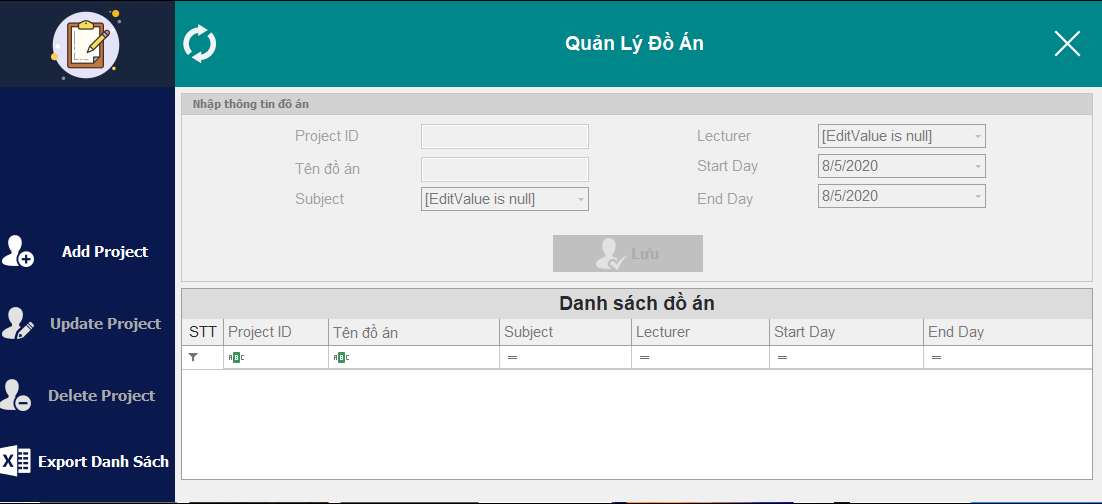
# Setting and testing



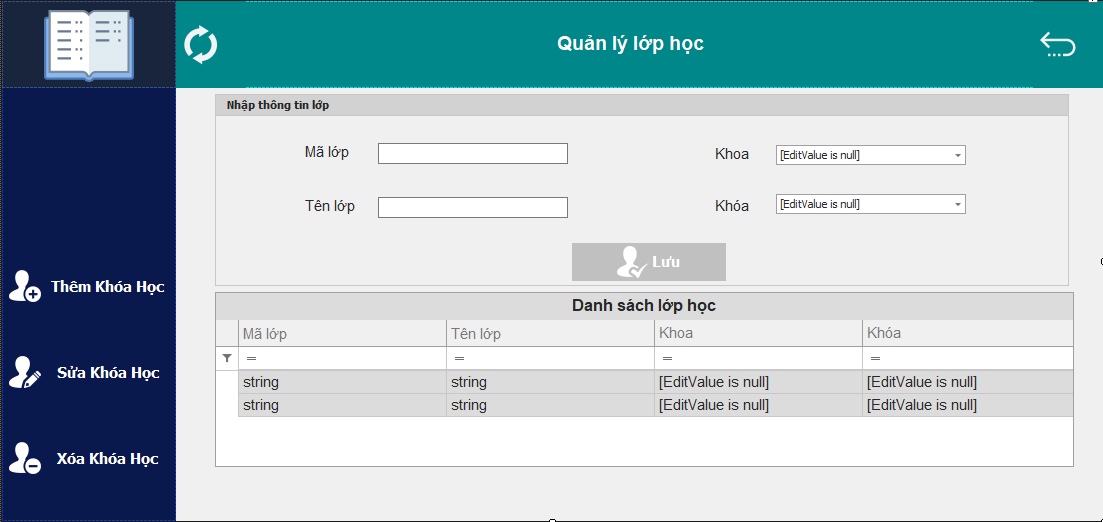
#### Picture 14 Testing form ManageStudentInformation



#### Picture 15 Testing form ManageLecturerInformation



#### Picture 16 Testing form ManageProjectStudent



#### Picture 17 Testing form ManageClass

# Conclusion and development direction

## Conclusion:

Basically, my group completed the basic task of requirement’s final project.There are some advantages and disanvantages of the software.

* Advantages:
* Interface is easy to approach, good-looking .User support tools :Dev Express, Guna.Interface in order to have a beautiful interface
* Software is easy to use, form ManageStudent and ManageProject can export file Excel.
* Disadvantages:

Capacity of project is quite heavy

## Development direction:

* Develop MultiLanguage have many languages.
* Students can view progress of deadline of them (form ViewDeadline)