

**ỦY BAN NHÂN DÂN TP HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC SÀI GÒN  
KHOA CÔNG NGHỆ THÔNG TIN**

---



**XÂY DỰNG PHẦN MỀM THEO MÔ HÌNH PHÂN LỚP**

**TÊN ĐỀ TÀI : PHẦN MỀM QUẢN LÝ KHÓA HỌC**

**Nhóm sinh viên thực hiện:**

- 1. Bùi Trí Đạt - 3119410083**
- 2. Nguyễn Ngọc Bảo Chương - 3119410048**
- 3. Lâm Thiên Bảo - 3119410029**
- 4. Nguyễn Thị Xuân Hoài - 3119410136**
- 5. Phạm Thị Ngọc Huyền - 3119410167**

**Giảng viên hướng dẫn : Cao Minh Thành**

## MỤC LỤC

<b>BẢNG PHÂN CÔNG</b>	3
<b>Phần 1: Chức năng quản lý thông tin người học, người dạy</b>	4
I. Sơ đồ class chung	4
II. Quản lý thông tin người học	4
1. Xử lý 1: Hiển thị danh sách người học	4
2. Xử lý 2: Thêm thông tin người học	6
3. Xử lý 3: Sửa thông tin người học	9
4. Xử lý 4: Xóa thông tin người học	12
5. Xử lý 5: Tìm kiếm thông tin người học theo tên	15
III. Quản lý thông tin người dạy	18
1. Xử lý 1: Hiển thị danh sách thông tin người dạy	18
2. Xử lý 2: Thêm thông tin người dạy	20
3. Xử lý 3: Sửa thông tin người dạy	22
4. Xử lý 4: Xóa thông tin người dạy	26
5. Xử lý 5: Tìm kiếm thông tin người dạy theo tên	28
<b>Phần 2: Chức năng quản lý thông tin các khóa học online, onsite</b>	31
I. Sơ đồ class chung	31
II. Quản lý thông tin khóa học online	31
1. Xử lý 1: Hiển thị danh sách thông tin khóa học online	31
2. Xử lý 2: Thêm thông tin khóa học online	34
3. Xử lý 3: Sửa thông tin khóa học online	37
4. Xử lý 4: Xóa thông tin khóa học online	41
5. Xử lý 5: Tìm kiếm thông tin khóa học online theo mã khóa học	44
III. Quản lý thông tin khóa học onsite	46
1. Xử lý 1: Hiển thị danh sách thông tin khóa học onsite	46
2. Xử lý 2: Thêm thông tin khóa học onsite	49
3. Xử lý 3: Sửa thông tin khóa học onsite	53
4. Xử lý 4: Xóa thông tin khóa học onsite	55
5. Xử lý 5: Tìm kiếm thông tin khóa học onsite theo mã khóa học	58
<b>Phần 3: Chức năng quản lý thông tin phân công giảng dạy</b>	61
I. Sơ đồ class chung	61
II. Quản lý thông tin phân công giảng dạy	61

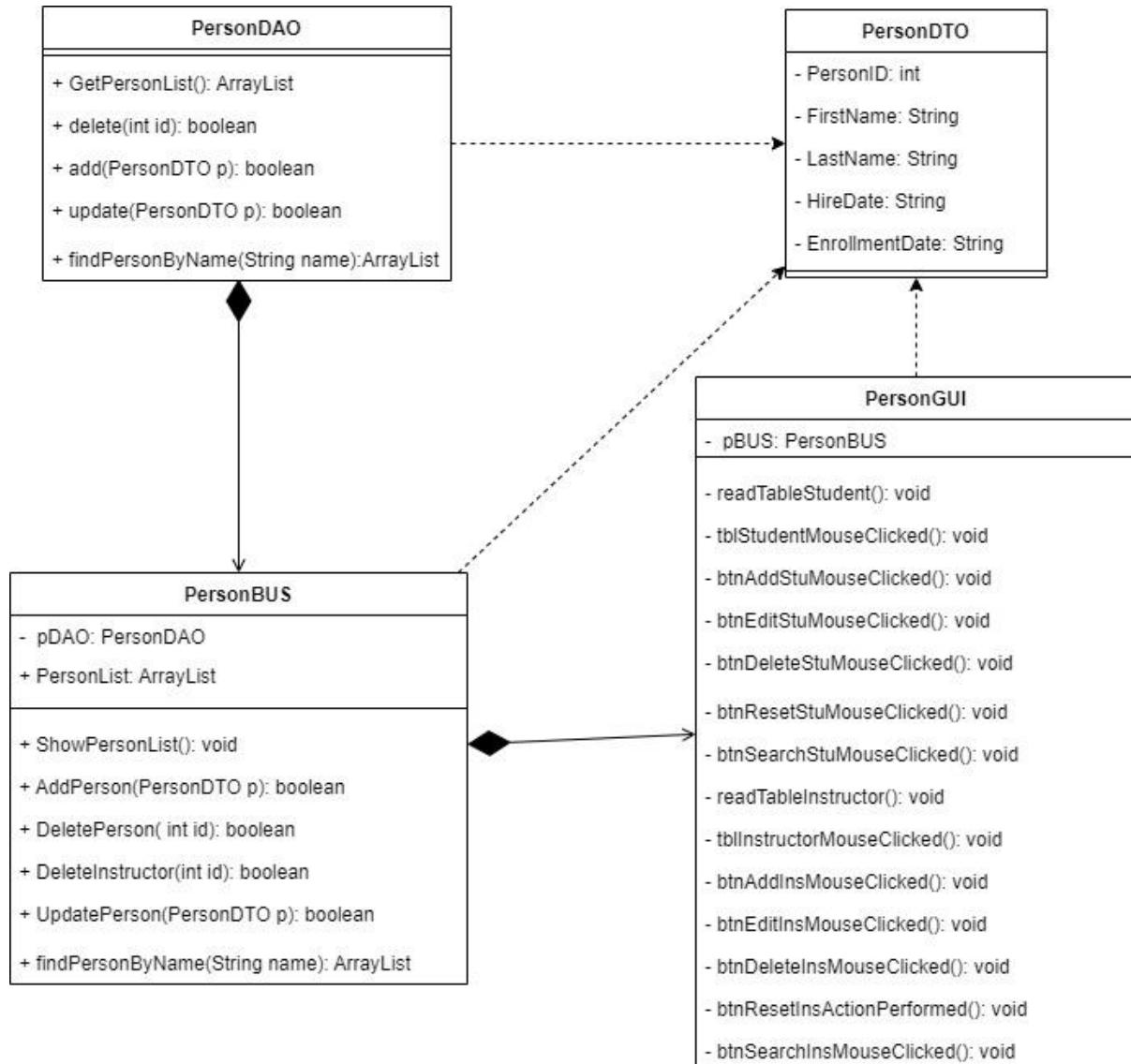
1. Xử lý 1: Hiển thị danh sách thông tin phân công giảng dạy	61
2. Xử lý 2: Thêm thông tin phân công giảng dạy	63
3. Xử lý 3: Sửa thông tin phân công giảng dạy	66
4. Xử lý 4: Xóa thông tin phân công giảng dạy	69
5. Xử lý 5: Tìm kiếm thông tin phân công giảng dạy theo tên khóa học	72
<b>Phần 4: Chức năng quản lý kết quả khóa học</b>	<b>75</b>
I. Sơ đồ class chung	75
II. Quản lý kết quả khóa học	75
1. Xử lý 1: Hiển thị danh sách kết quả khóa học	75
2. Xử lý 2: Thêm điểm cho người học	77
3. Xử lý 3: Cập nhập điểm của người học	80
4. Xử lý 4: Xóa điểm của người học	83
5. Xử lý 5: Tìm kiếm điểm bằng mã người học	86

## BẢNG PHÂN CÔNG

STT	Nội dung công việc	Người phụ trách
01	Tổng hợp và viết báo cáo	Hoài, Huyền
02	Thiết kế giao diện	Huyền, Hoài
03	Vẽ sơ đồ class và sơ đồ tuần tự chức năng 1 và 4	Hoài
04	Vẽ sơ đồ class và sơ đồ tuần tự chức năng 2 và 3	Huyền
05	Hiện thực hóa cho chức năng 1 và 3	Bảo
06	Hiện thực hóa cho chức năng 2	Chương
07	Hiện thực hóa cho chức năng 4	Đạt
08	Tổng hợp và sửa chữa lỗi	Đạt, Chương, Bảo

## Phần 1: Chức năng quản lý thông tin người học, người dạy

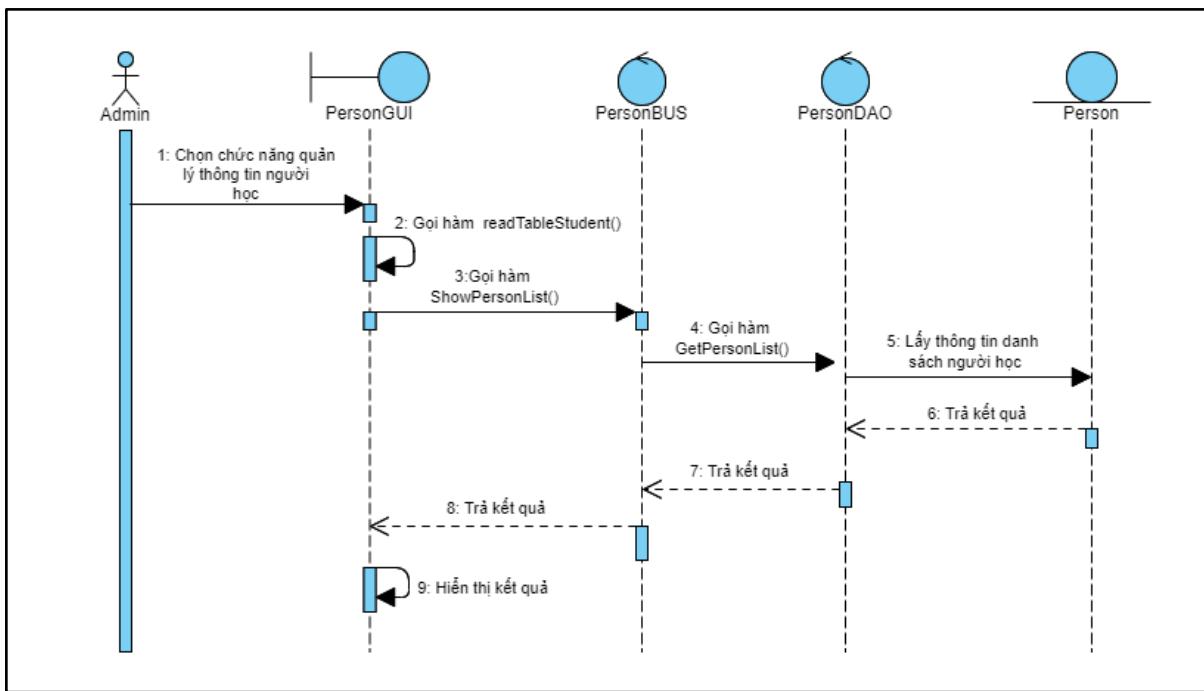
### I. Sơ đồ class chung



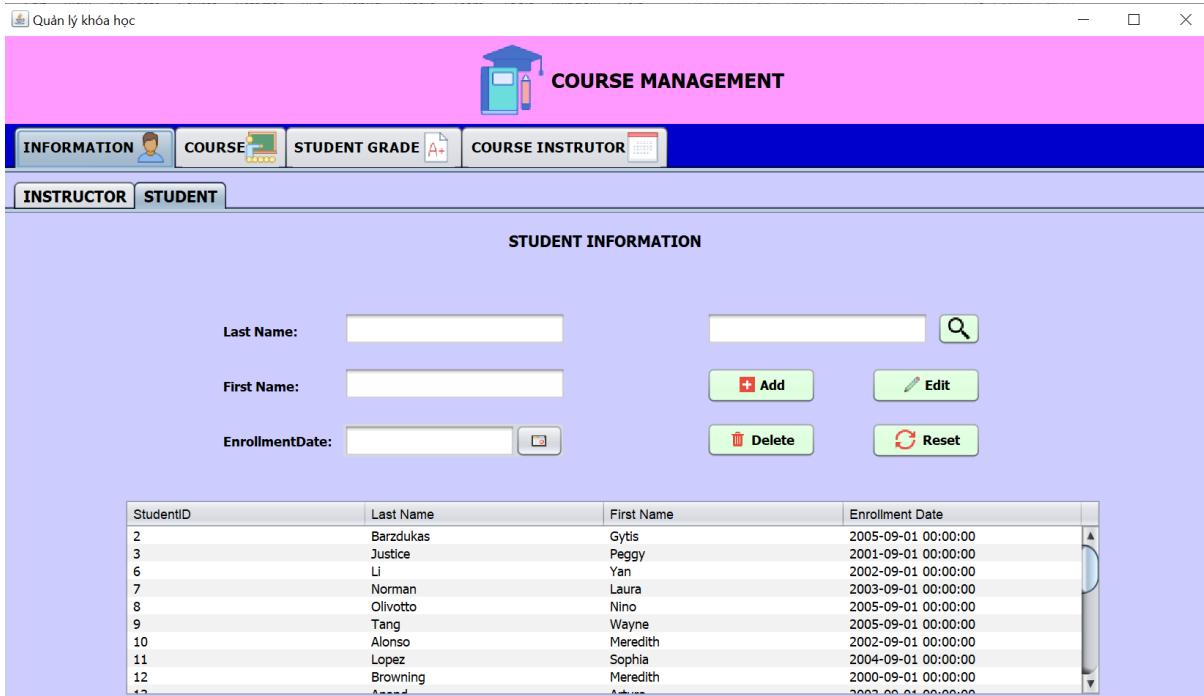
### II. Quản lý thông tin người học

#### 1. Xử lý 1: Hiển thị danh sách người học

##### 1.1. Sơ đồ tuần tự



## 1.2. Giao diện



Giao diện hiển thị danh sách thông tin người học

## 1.3. Code 3 class:

❖ DAO

```

public ArrayList<PersonDTO> GetPersonList() throws SQLException {
    ArrayList<PersonDTO> personList = new ArrayList<>();

    String sql = "select * from person";
    ResultSet rs = ConnectDatabase.GetInstance().ExcuteSELECT(sql);
    while (rs.next()) {
        PersonDTO pDTO = new PersonDTO();
        pDTO.setPersonID(rs.getInt("PersonID"));
        pDTO.setLastName(rs.getString("Lastname"));
        pDTO.setFirstName(rs.getString("Firstname"));
        pDTO.setHireDate(rs.getString("HireDate"));
        pDTO.setEnrollmentDate(rs.getString("EnrollmentDate"));
        personList.add(pDTO);
    }
    return personList;
}

```

## ❖ BUS

```

public void ShowPersonList() throws SQLException {
    PersonList = pDAO.GetPersonList();
}

```

## ❖ GUI

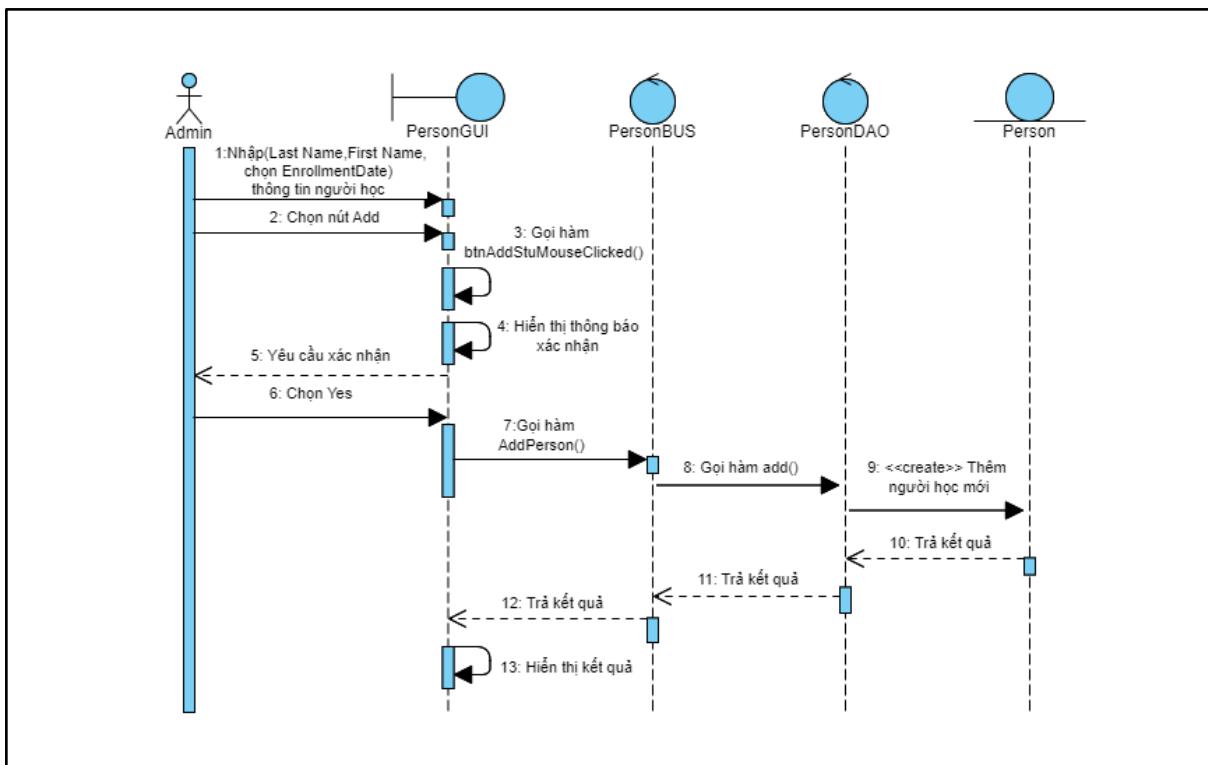
```

private void readTableStudent() {
    try {
        pBUS.ShowPersonList();
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
    modelTableStudent.setRowCount(0);
    for (PersonDTO p : PersonBUS.PersonList) {
        if (p.getHireDate() == null) {
            modelTableStudent.addRow(new Object[]{
                p.getPersonID(), p.getLastName(), p.getFirstName(), p.getEnrollmentDate()
            });
        }
    }
}

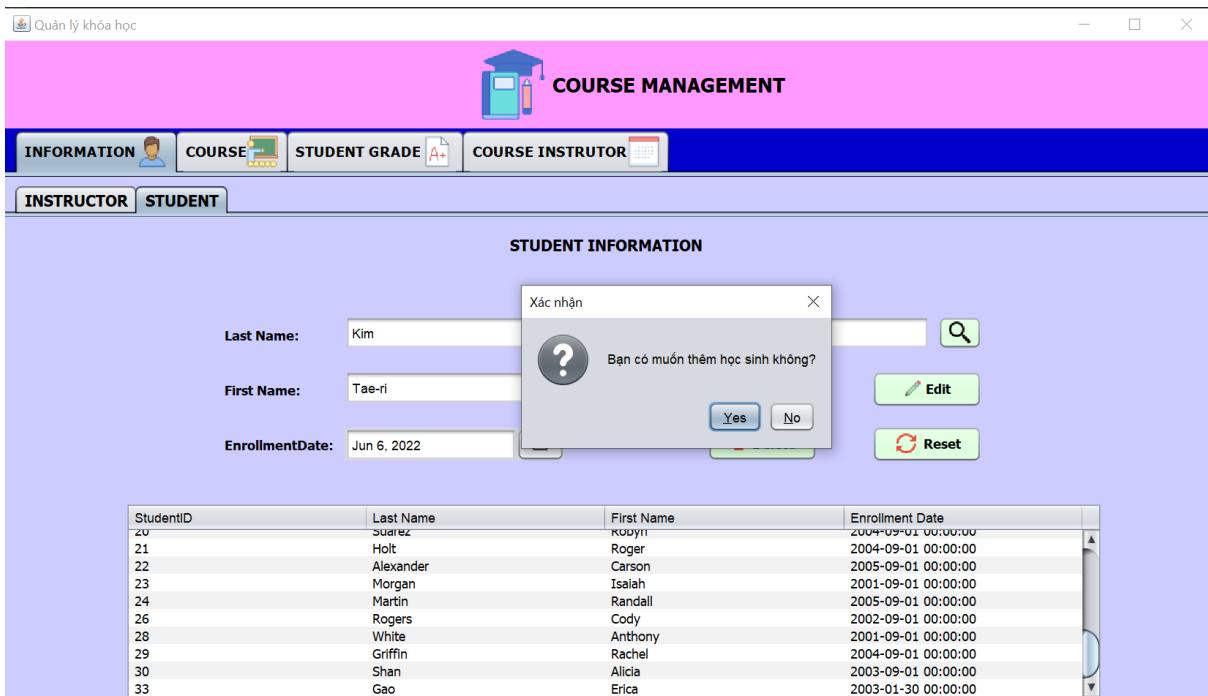
```

## 2. Xử lý 2: Thêm thông tin người học

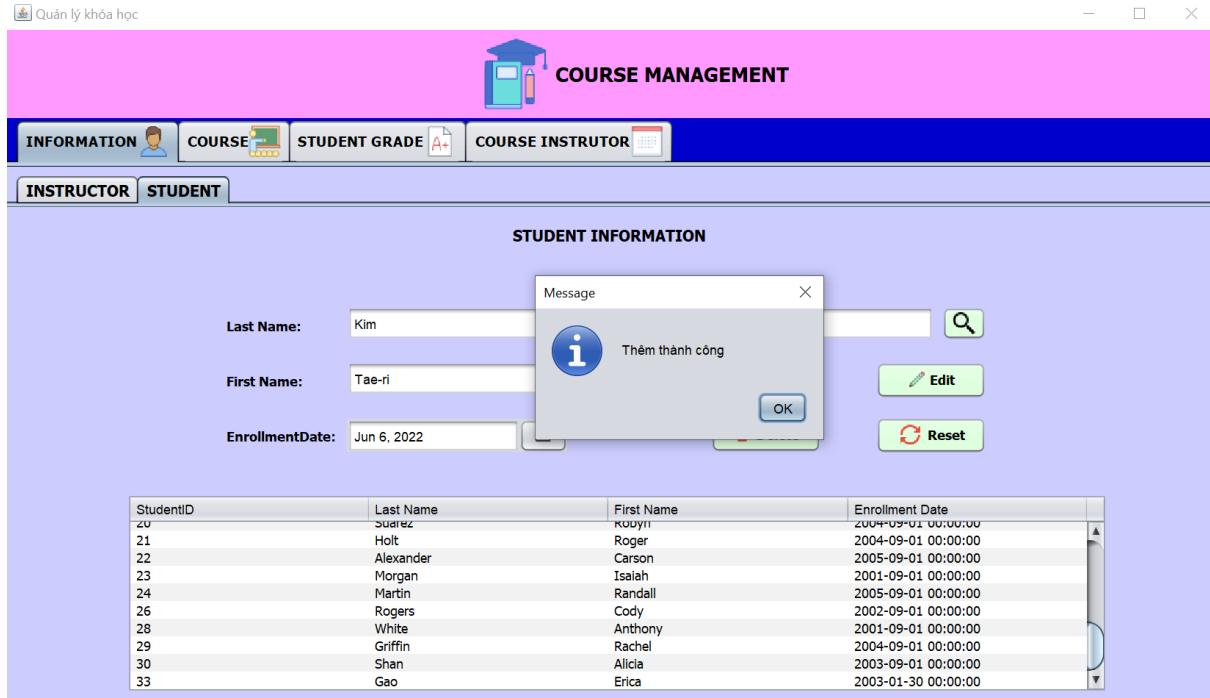
### 2.1. Sơ đồ tuần tự



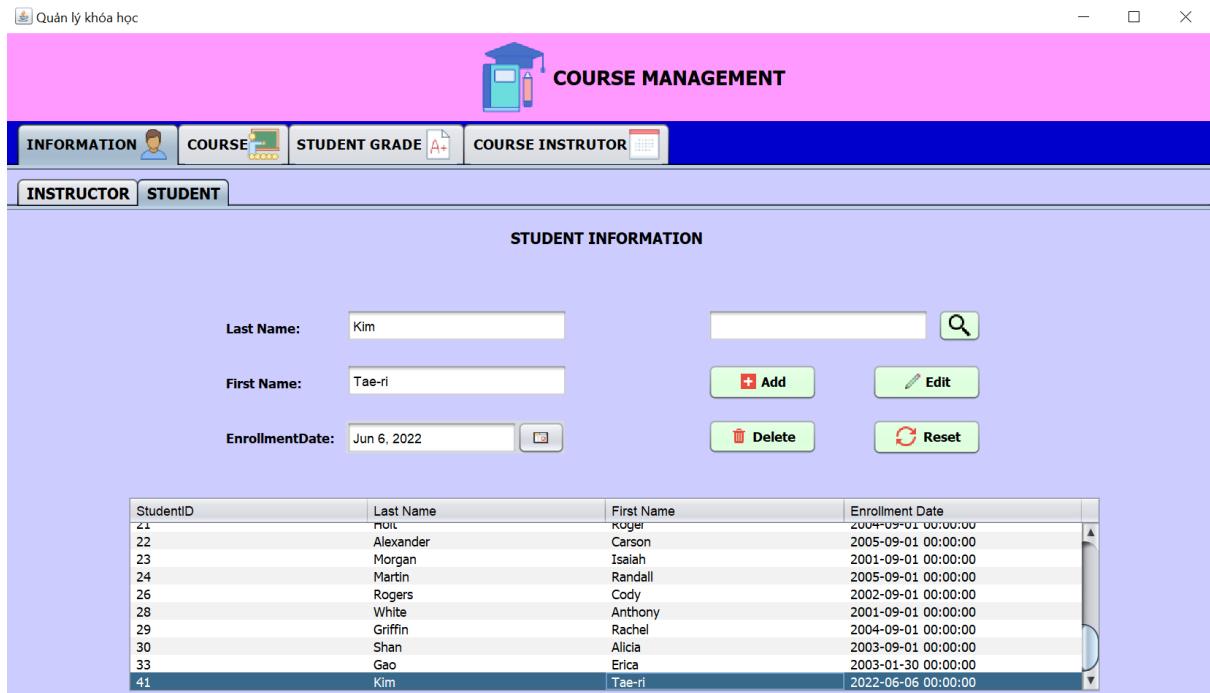
## 2.2. Giao diện



*Giao diện thêm người học mới sau khi bấm nút Add*



*Giao diện thông báo thêm người học mới thành công*



*Giao diện người học mới sau khi thêm xong*

### 2.3. Code 3 class:

❖ DAO

```

public boolean add(PersonDTO p) {
    String sql;
    if (p.getHireDate() == null) {
        LocalDate enrollmentDate = LocalDate.parse(p.getEnrollmentDate(), dtf);
        sql = String.format("INSERT INTO `person`(`Lastname`, `Firstname`, `EnrollmentDate`) VALUES ('%s', '%s', '%s')",
            p.getLastName(), p.getFirstName(), enrollmentDate);
    } else {
        LocalDate hireDate = LocalDate.parse(p.getHireDate(), dtf);
        sql = String.format("INSERT INTO `person`(`Lastname`, `Firstname`, `HireDate`) VALUES ('%s', '%s', '%s')",
            p.getLastName(), p.getFirstName(), hireDate);
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean AddPerson(PersonDTO p) {
    PersonList.add(p);
    return pDAO.add(p);
}

```

## ❖ GUI

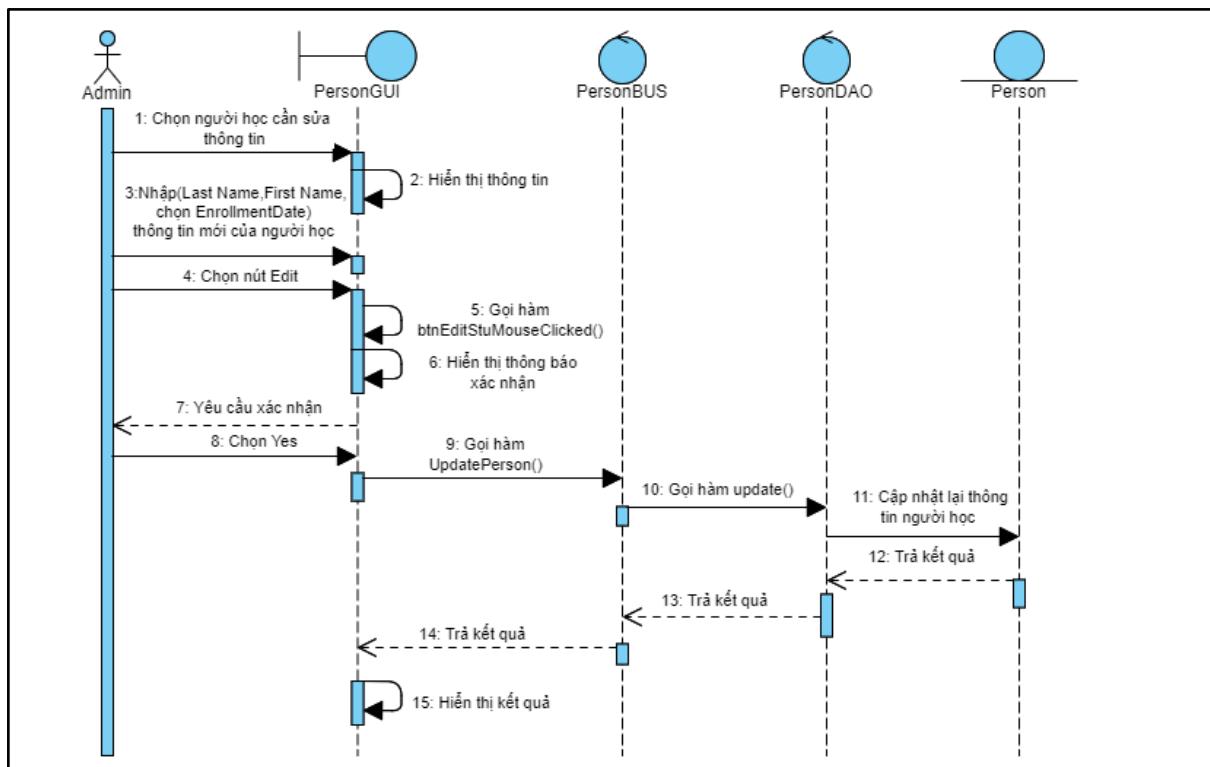
```

private void btnAddStuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn thêm học sinh không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
    if (result == JOptionPane.YES_OPTION) {
        PersonDTO pDTO = new PersonDTO();
        pDTO.setFirstName(txtFirstNameStu.getText());
        pDTO.setLastName(txtLastNameStu.getText());
        pDTO.setEnrollmentDate(formatType.format(dateEnrollment.getDate()));
        if (pBUS.AddPerson(pDTO)) {
            JOptionPane.showMessageDialog(null, "Thêm thành công");
            readTableStudent();
            cbPersonID.setModel(new DefaultComboBoxModel(ComboBoxPersonID()));
            cbPersonIdCouIns.setModel(new DefaultComboBoxModel(ComboBoxPersonInstructorID()));
        } else {
            JOptionPane.showMessageDialog(null, "Thêm thất bại");
        }
    }
}

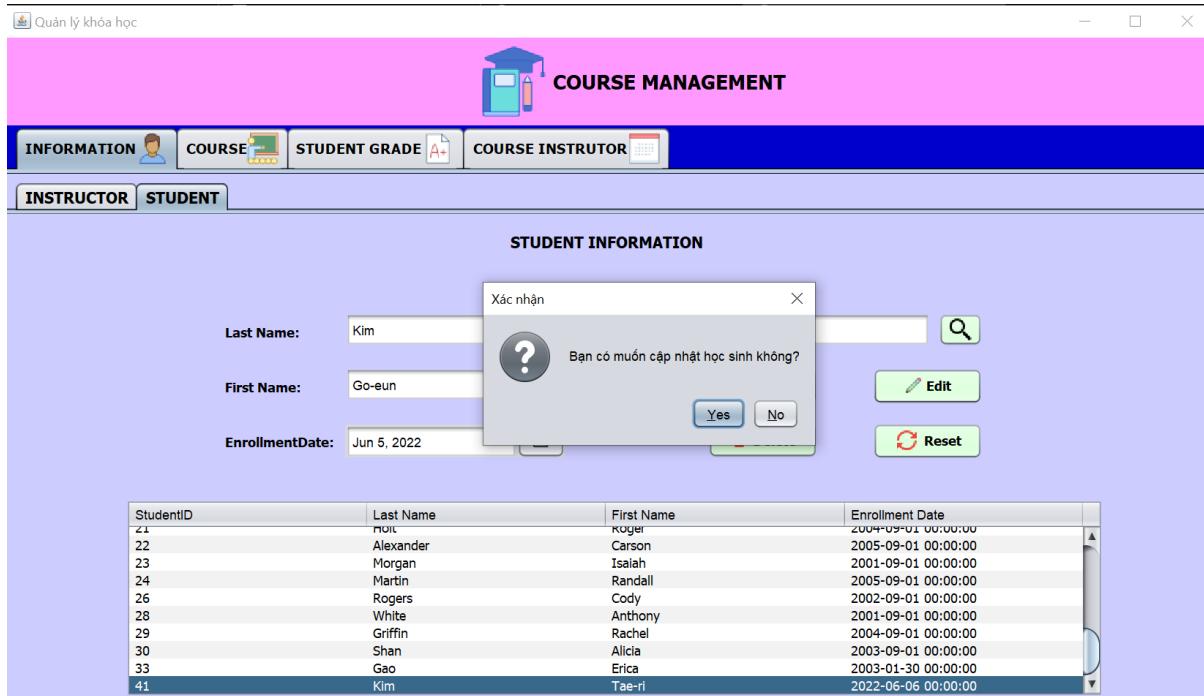
```

### 3. Xử lý 3: Sửa thông tin người học

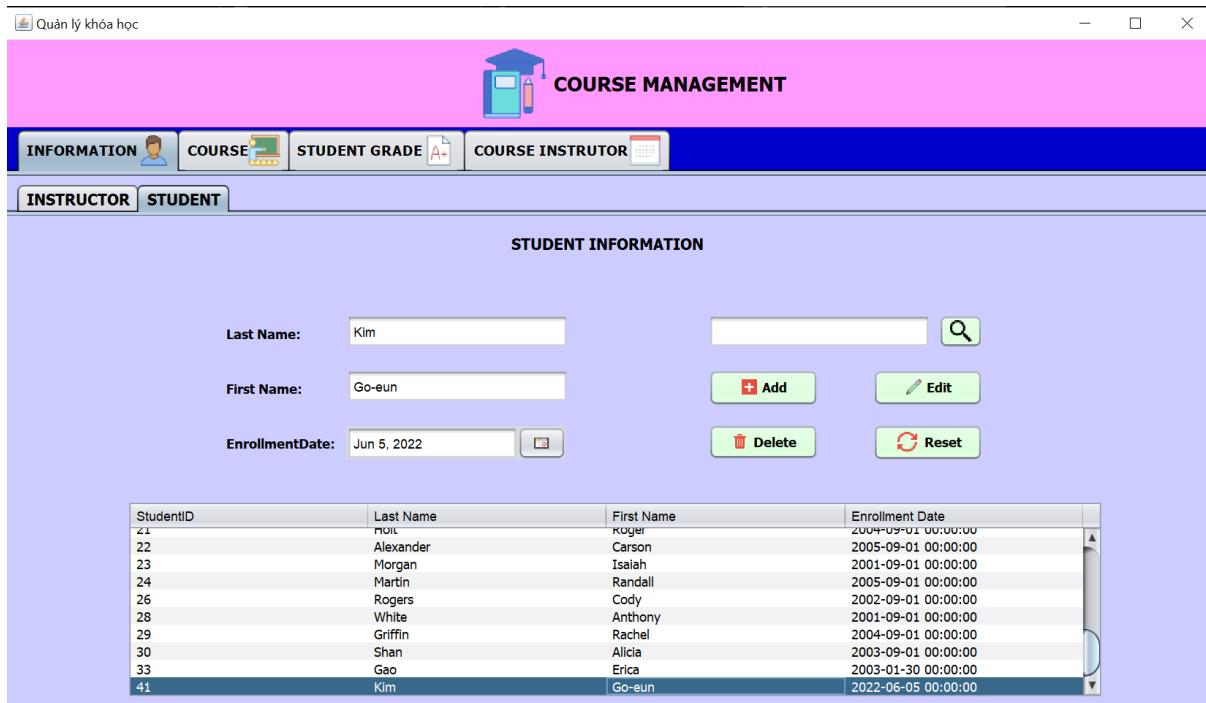
#### 3.1. Sơ đồ tuần tự



### 3.2. Giao diện



Giao diện cập nhật thông tin người học mới sau khi bấm nút Edit



*Giao diện người học mới sau khi cập nhật thông tin xong*

### 3.3. Code 3 class:

#### ❖ DAO

```
public boolean update(PersonDTO p) {
    String sql;
    if (p.getHireDate() == null) {
        LocalDate enrollmentDate = LocalDate.parse(p.getEnrollmentDate(), dtf);
        sql = String.format("UPDATE `person`"
            + "SET LastName='%s', Firstname='%s', EnrollmentDate='%s'"
            + "WHERE PersonID='%s'", p.getLastName(), p.getFirstName(), enrollmentDate, p.getPersonID());
    } else {
        LocalDate hireDate = LocalDate.parse(p.getHireDate(), dtf);
        sql = String.format("UPDATE `person`"
            + "SET LastName='%s', Firstname='%s', HireDate='%s'"
            + "WHERE PersonID='%s'", p.getLastName(), p.getFirstName(), hireDate, p.getPersonID());
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}
```

#### ❖ BUS

```

public boolean UpdatePerson(PersonDTO p) {
    for (int i = 0; i < PersonList.size(); i++) {
        if (p.getPersonID() == PersonList.get(i).getPersonID()) {
            PersonList.set(i, p);
            break;
        }
    }
    return pDAO.update(p);
}

```

## ❖ GUI

```

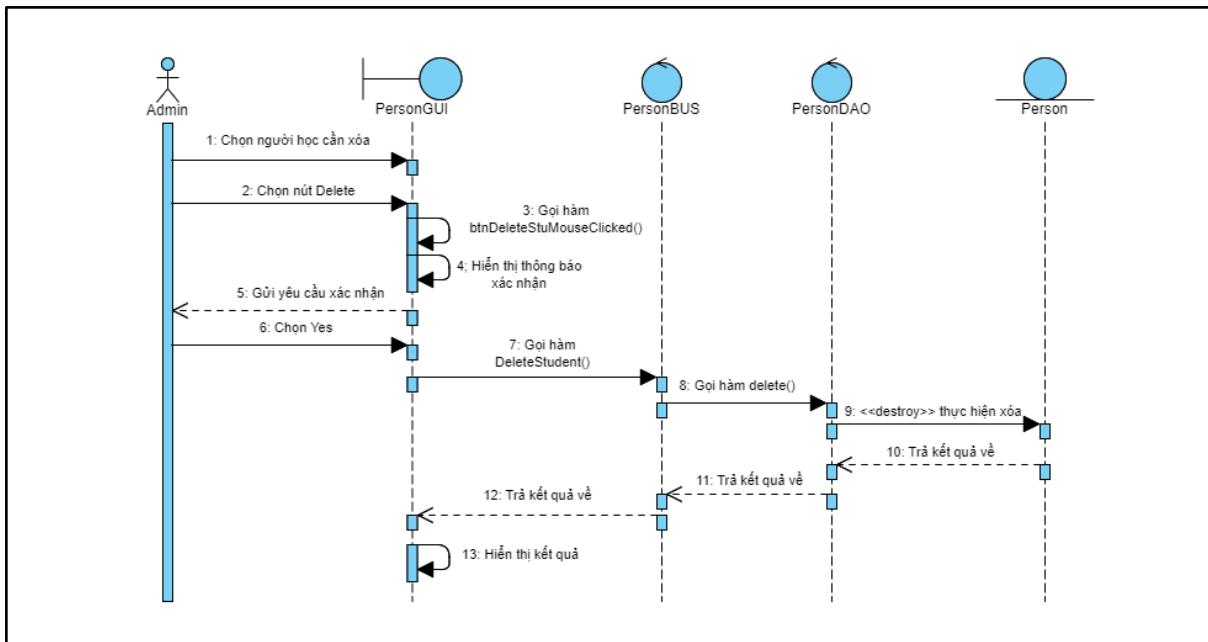
private void btnEditStuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:

    int i = tblStudent.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn học sinh cần cập nhật");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn cập nhật học sinh không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            PersonDTO pDTO = new PersonDTO();
            pDTO.setPersonID(Integer.parseInt(modelTableStudent.getValueAt(i, 0).toString()));
            pDTO.setFirstName(txtFirstNameStu.getText());
            pDTO.setLastName(txtLastNameStu.getText());
            pDTO.setEnrollmentDate(formatType.format(dateEnrollment.getDate()));
            if (pBUS.UpdatePerson(pDTO)) {
                JOptionPane.showMessageDialog(null, "Cập nhật thành công");
                txtFirstNameStu.setText("");
                txtLastNameStu.setText("");
                dateEnrollment.setDate(null);
                readTableStudent();
                cbPersonID.setModel(new DefaultComboBoxModel(ComboBoxPersonID()));
                cbPersonIdCouIns.setModel(new DefaultComboBoxModel(ComboBoxPersonInstructorID()));
            } else {
                JOptionPane.showMessageDialog(null, "Cập nhật thất bại");
            }
        }
    }
}

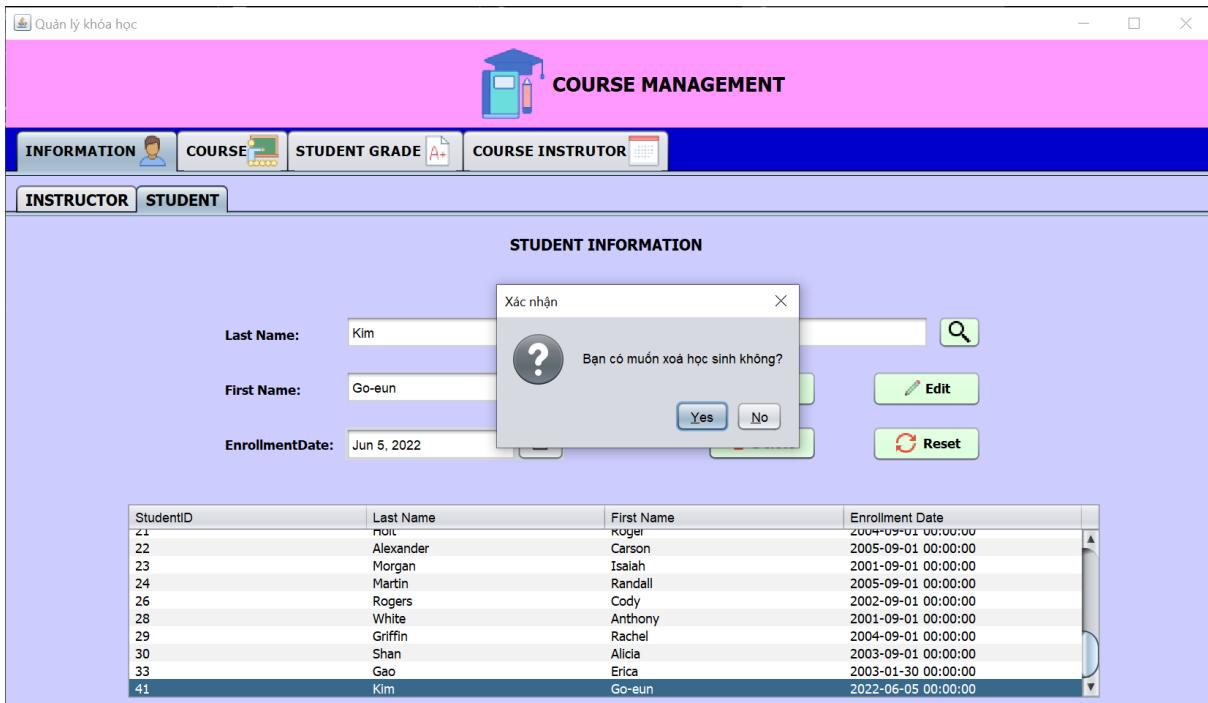
```

## 4. Xử lý 4: Xóa thông tin người học

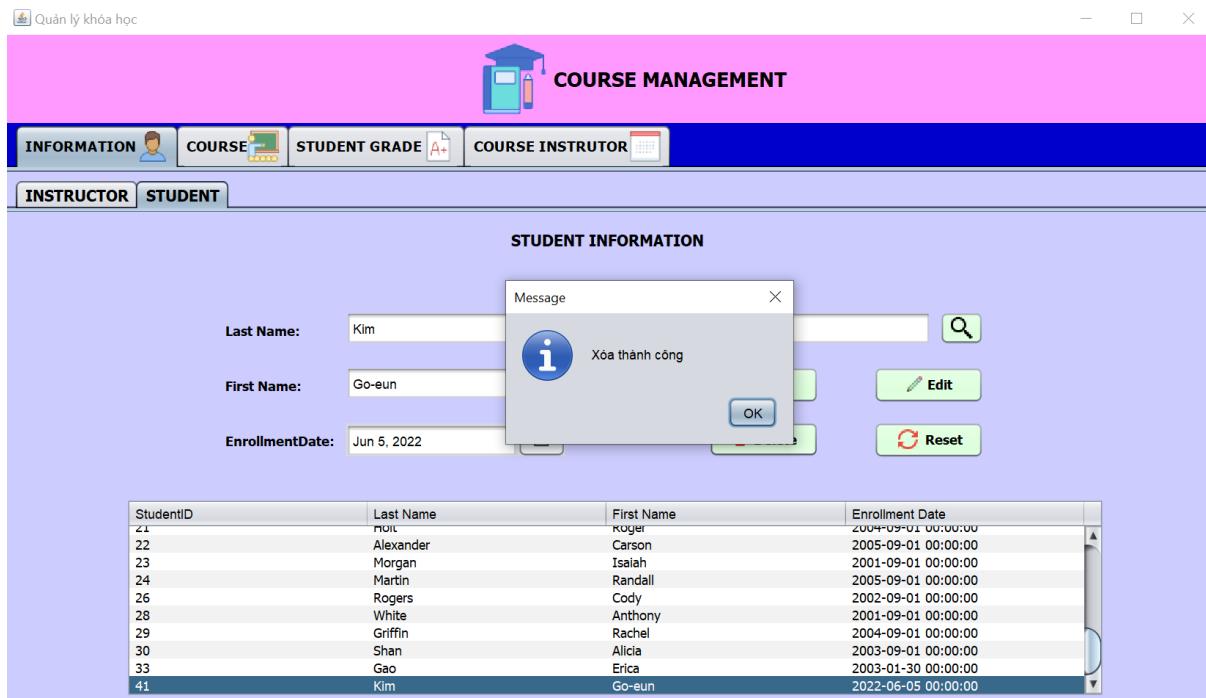
### 4.1. Sơ đồ tuần tự



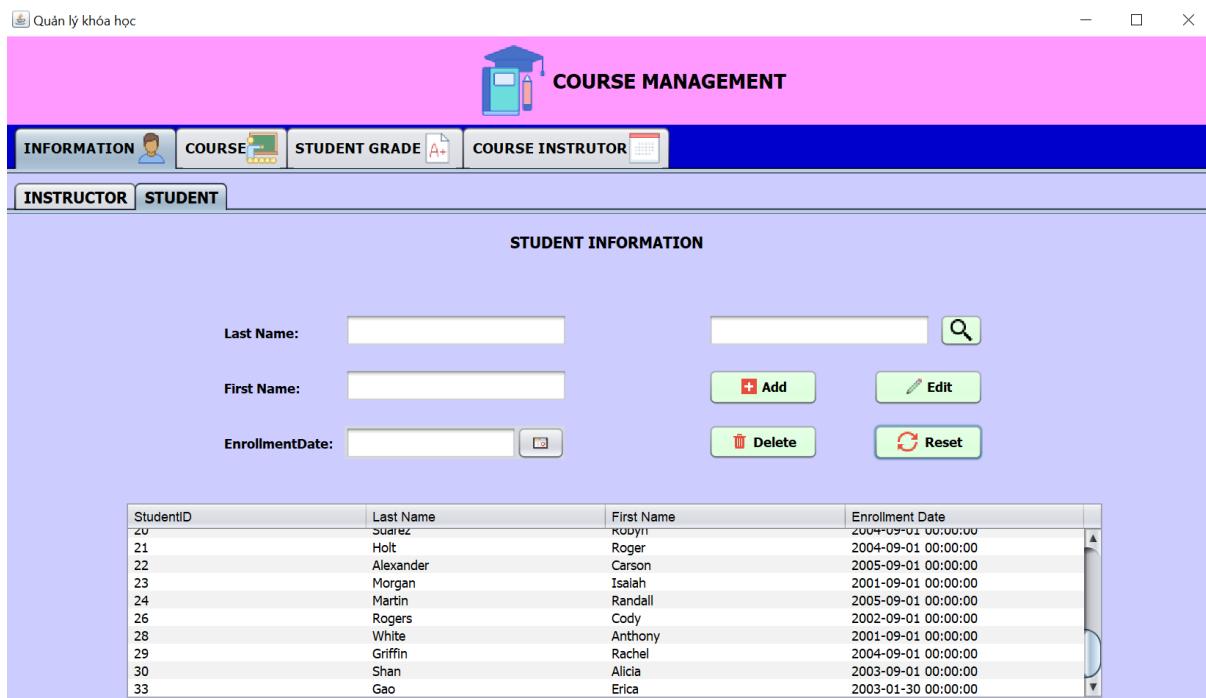
## 4.2. Giao diện



Giao diện thông báo xác nhận xóa



*Giao diện thông báo xóa thành công*



*Giao diện thông tin người học sau khi xóa*

#### 4.3. Code 3 class:

❖ DAO

```

public boolean delete(int id) {
    String sql = String.format("DELETE FROM `person` WHERE PersonID='%s'", id);
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean DeleteStudent(int id) {
    for (StudentGradeDTO stDTO : StudentGradeBUS.sgList) {
        if (stDTO.getPersonID() == id) {
            return false;
        }
    }
    for (PersonDTO person : PersonList) {
        if (person.getPersonID() == id) {
            PersonList.remove(person);
            break;
        }
    }
    return pDAO.delete(id);
}

```

## ❖ GUI

```

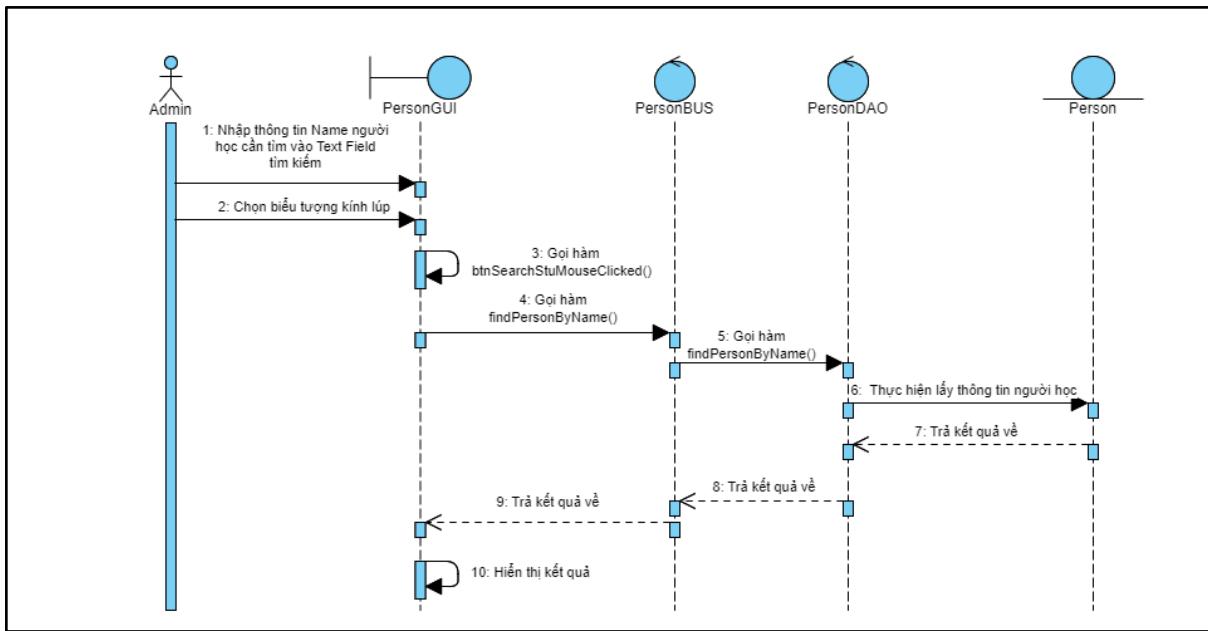
private void btnDeleteStuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here.

    int i = tblStudent.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn học sinh cần xoá");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn xoá học sinh không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            int idStudent = Integer.parseInt(modelTableStudent.getValueAt(i, 0).toString());
            if (pBUS.DeleteStudent(idStudent)) {
                JOptionPane.showMessageDialog(null, "Xóa thành công");
                txtFirstNameStu.setText("");
                txtLastNameStu.setText("");
                dateEnrollment.setDate(null);
                readTableStudent();
                cbPersonID.setModel(new DefaultComboBoxModel(ComboBoxPersonID()));
                cbPersonIdCouIns.setModel(new DefaultComboBoxModel(ComboBoxPersonInstructorID()));
            } else {
                JOptionPane.showMessageDialog(null, "Xóa thất bại");
            }
        }
    }
}

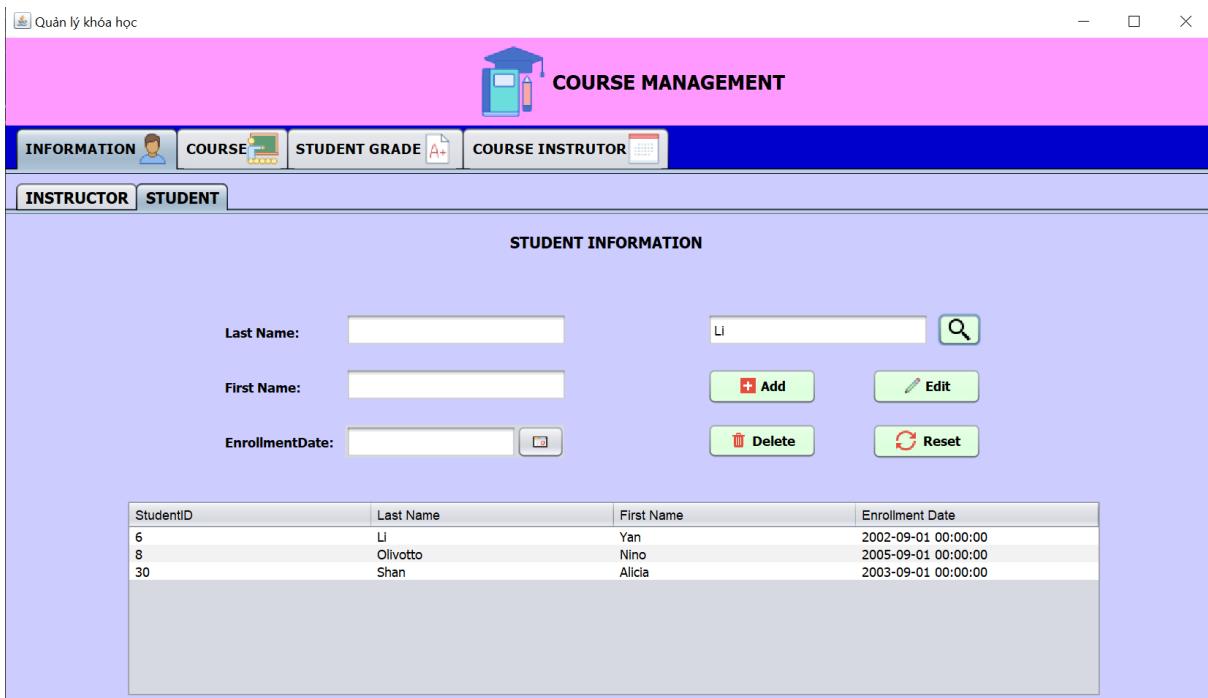
```

## 5. Xử lý 5: Tìm kiếm thông tin người học theo tên

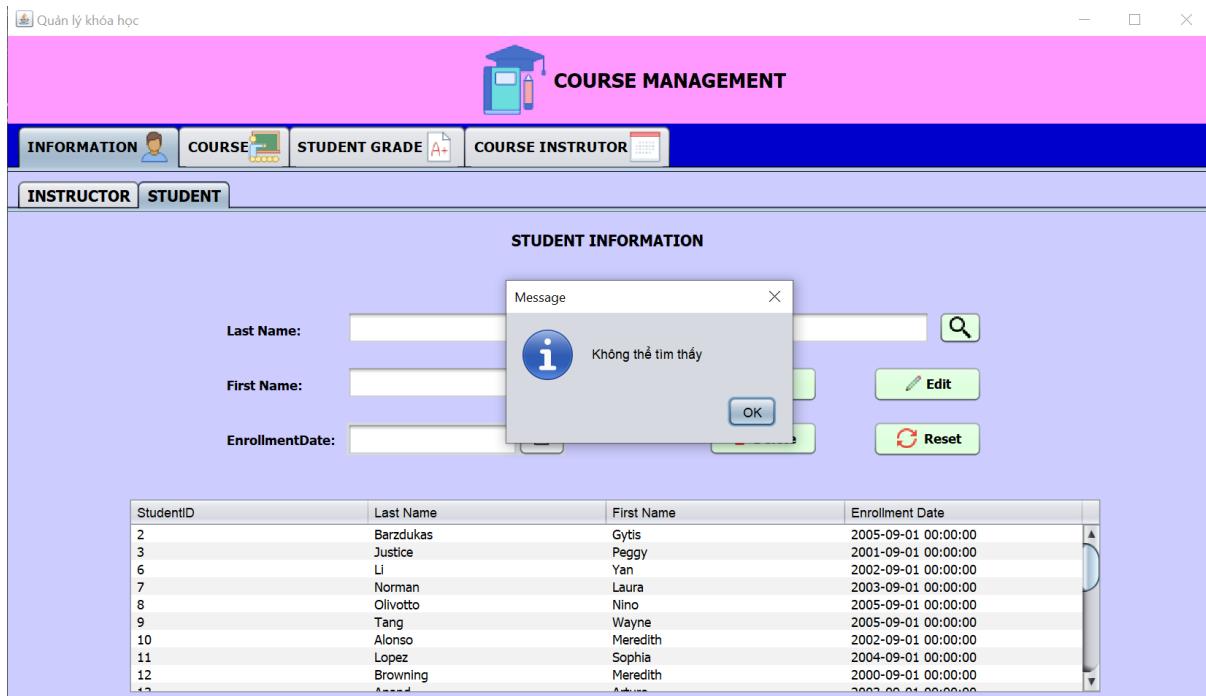
### 5.1. Sơ đồ tuần tự



## 5.2. Giao diện



*Giao diện tìm kiếm người học*



Giao diện thông báo kết quả tìm kiếm sau khi nhấn biểu tượng kính lúp trên màn hình

### 5.3. Code 3 class:

#### ❖ DAO

```
public ArrayList<PersonDTO> findPersonByName(String name) throws SQLException {
    ArrayList<PersonDTO> personList = new ArrayList<>();
    if (name.contains(" ")) {
        String[] word = name.split("\\s+");
        String sql = String.format("SELECT * FROM `person` WHERE (`Firstname` LIKE '%"+word[0]+"' AND `Lastname` LIKE '%"+word[1]+"')",
        word[0], word[1]);
        ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
        while (rs.next()) {
            PersonDTO pDTO = new PersonDTO();
            pDTO.setPersonID(rs.getInt("PersonID"));
            pDTO.setLastName(rs.getString("Lastname"));
            pDTO.setFirstName(rs.getString("Firstname"));
            pDTO.setHireDate(rs.getString("HireDate"));
            pDTO.setEnrollmentDate(rs.getString("EnrollmentDate"));
            personList.add(pDTO);
        }
    } else {
        String sql = "SELECT * FROM `person` WHERE (`Firstname` LIKE '%"+name+"%' OR `Lastname` LIKE '%"+name+"%')";
        ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
        while (rs.next()) {
            PersonDTO pDTO = new PersonDTO();
            pDTO.setPersonID(rs.getInt("PersonID"));
            pDTO.setLastName(rs.getString("Lastname"));
            pDTO.setFirstName(rs.getString("Firstname"));
            pDTO.setHireDate(rs.getString("HireDate"));
            pDTO.setEnrollmentDate(rs.getString("EnrollmentDate"));
            personList.add(pDTO);
        }
    }
    return personList;
}
```

#### ❖ BUS

```

public ArrayList<PersonDTO> findPersonByName(String name) throws SQLException {
    return pDAO.findPersonByName(name);
}

```

## ❖ GUI

```

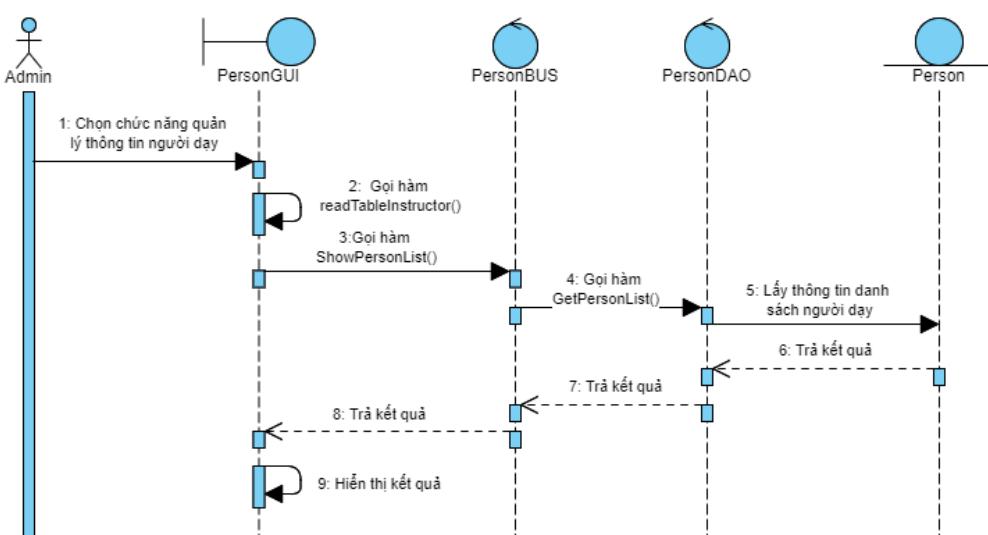
private void btnSearchStuMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    String input = txtSearchStu.getText();
    System.out.println(input);
    if (!input.isEmpty()) {
        try {
            ArrayList<PersonDTO> listPerson = pBUS.findPersonByName(input);
            if (!listPerson.isEmpty()) {
                modelTableStudent.setRowCount(0);
                for (PersonDTO p : listPerson) {
                    if (p.getHireDate() == null) {
                        modelTableStudent.addRow(new Object[]{
                            p.getPersonID(), p.getLastName(), p.getFirstName(), p.getEnrollmentDate()
                        });
                    }
                }
                JOptionPane.showMessageDialog(null, "Tim thấy rồi");
            } else {
                JOptionPane.showMessageDialog(null, "Không tìm thấy");
            }
        } catch (SQLException ex) {
            Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
        }
    } else {
        JOptionPane.showMessageDialog(null, "Không tìm thấy");
    }
}

```

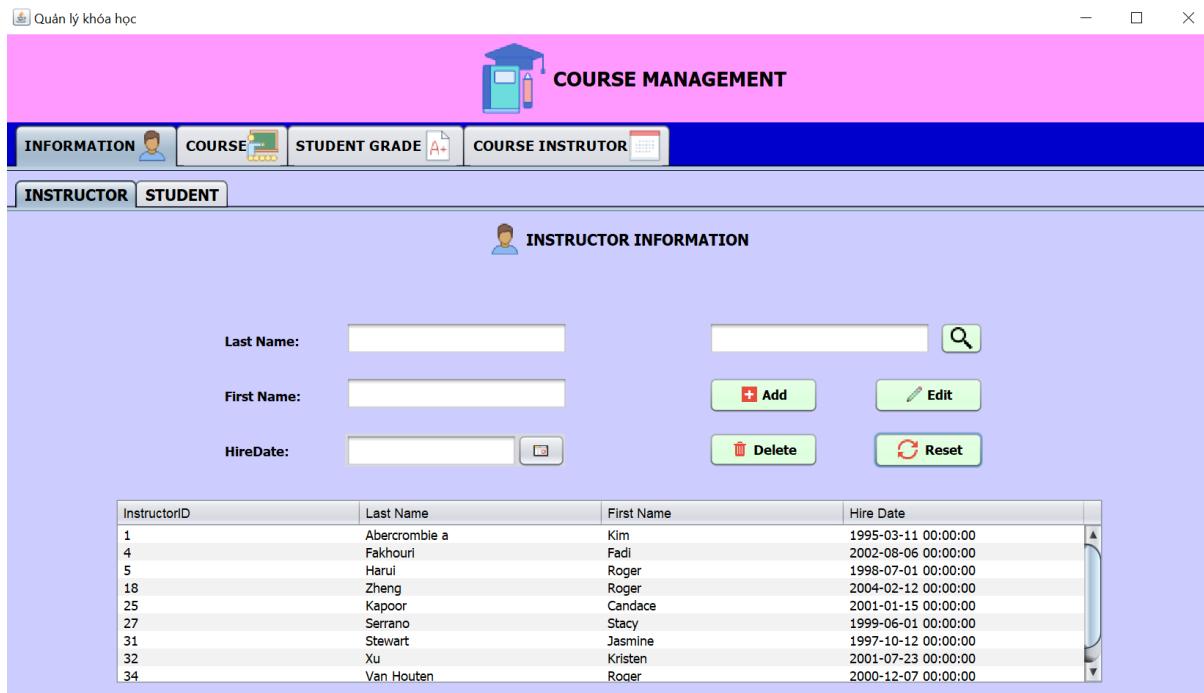
### III. Quản lý thông tin người dạy

#### 1. Xử lý 1: Hiển thị danh sách thông tin người dạy

##### 1.1. Sơ đồ tuần tự



## 1.2. Giao diện



Giao diện hiển thị danh sách thông tin người dạy

## 1.3. Code 3 class:

### ❖ DAO

```
public ArrayList<PersonDTO> GetPersonList() throws SQLException {
    ArrayList<PersonDTO> personList = new ArrayList<>();

    String sql = "select * from person";
    ResultSet rs = ConnectDatabase.GetInstance().ExcuteSELECT(sql);
    while (rs.next()) {
        PersonDTO pDTO = new PersonDTO();
        pDTO.setPersonID(rs.getInt("PersonID"));
        pDTO.setLastName(rs.getString("Lastname"));
        pDTO.setFirstName(rs.getString("Firstname"));
        pDTO.setHireDate(rs.getString("HireDate"));
        pDTO.setEnrollmentDate(rs.getString("EnrollmentDate"));
        personList.add(pDTO);
    }
    return personList;
}
```

### ❖ BUS

```

public void ShowPersonList() throws SQLException {
    PersonList = pDAO.GetPersonList();
}

```

## ❖ GUI

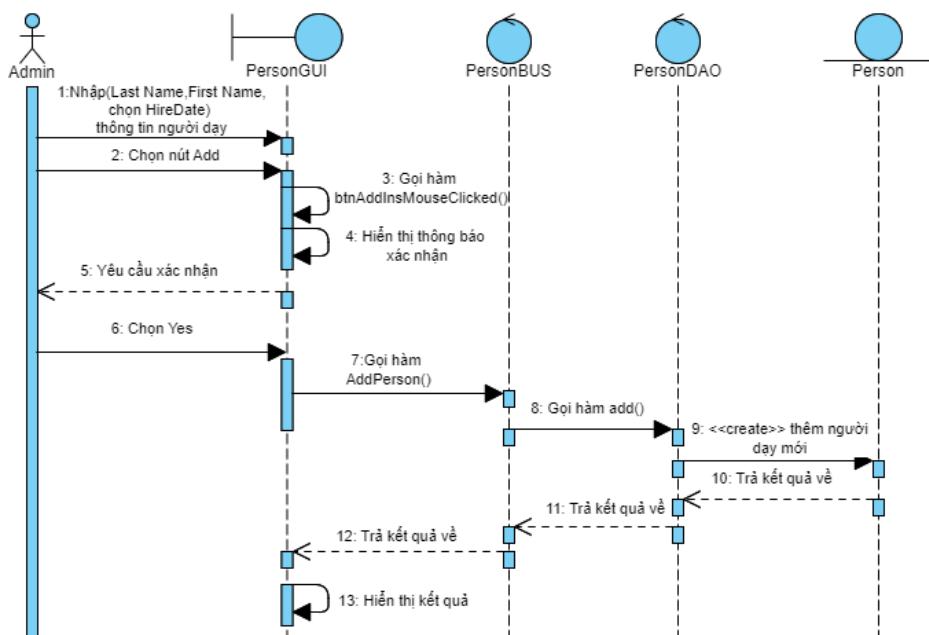
```

private void readTableInstructor() {
    try {
        pBUS.ShowPersonList();
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
    modelTableInstructor.setRowCount(0);
    for (PersonDTO p : PersonBUS.PersonList) {
        if (p.getEnrollmentDate() == null) {
            modelTableInstructor.addRow(new Object[]{
                p.getPersonID(), p.getLastName(), p.getFirstName(), p.getHireDate()
            });
        }
    }
}

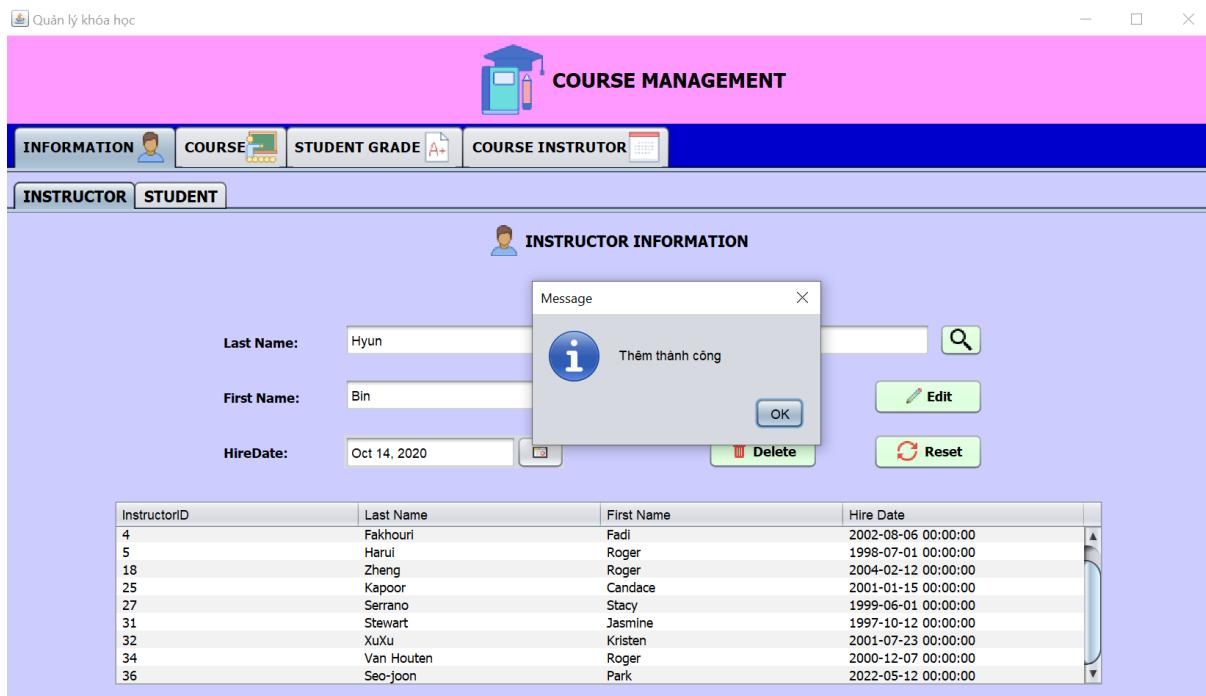
```

## 2. Xử lý 2: Thêm thông tin người dạy

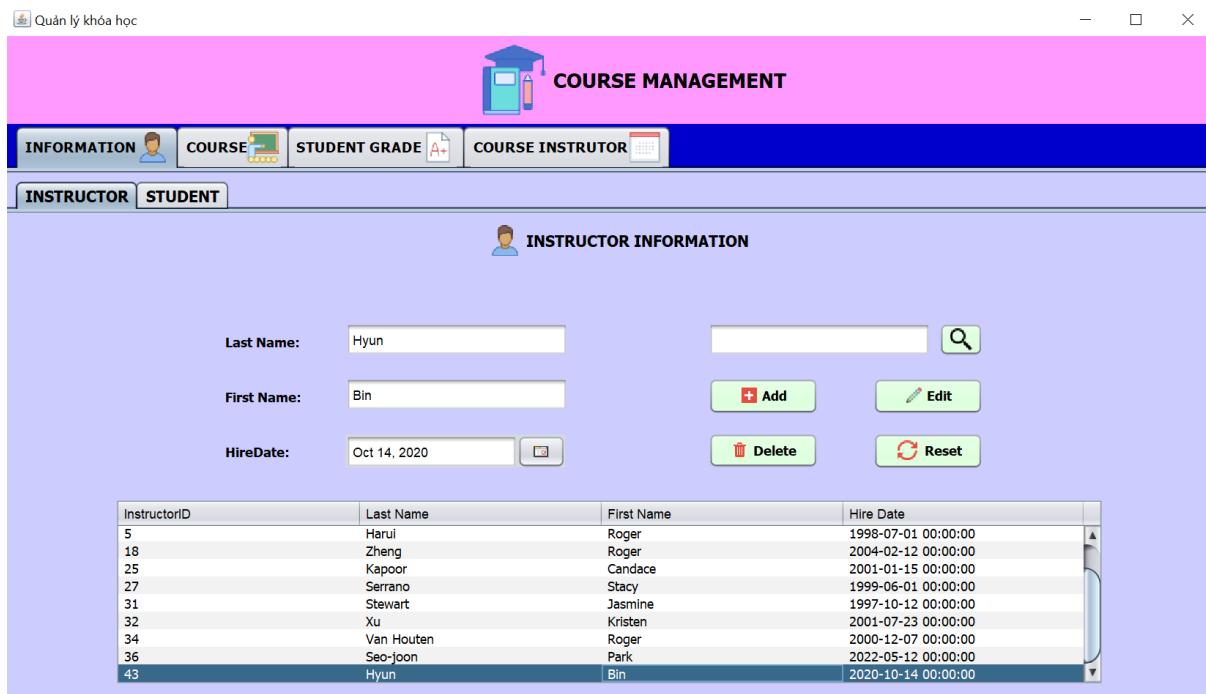
### 2.1. Sơ đồ tuần tự



### 2.2. Giao diện



Giao diện thêm người dạy thành công



Giao diện hiển thị danh sách thông tin người dạy sau khi thêm

### 2.3. Code 3 class:

❖ DAO

```

public boolean add(PersonDTO p) {

    String sql;
    if (p.getHireDate() == null) {
        LocalDate enrollmentDate = LocalDate.parse(p.getEnrollmentDate(), dtf);
        sql = String.format("INSERT INTO `person`(`Lastname`, `Firstname`, `EnrollmentDate`) VALUES ('%s', '%s', '%s')",
            p.getLastName(), p.getFirstName(), enrollmentDate);
    } else {
        LocalDate hireDate = LocalDate.parse(p.getHireDate(), dtf);
        sql = String.format("INSERT INTO `person`(`Lastname`, `Firstname`, `HireDate`) VALUES ('%s', '%s', '%s')",
            p.getLastName(), p.getFirstName(), hireDate);
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean AddPerson(PersonDTO p) {
    PersonList.add(p);
    return pDAO.add(p);
}

```

## ❖ GUI

```

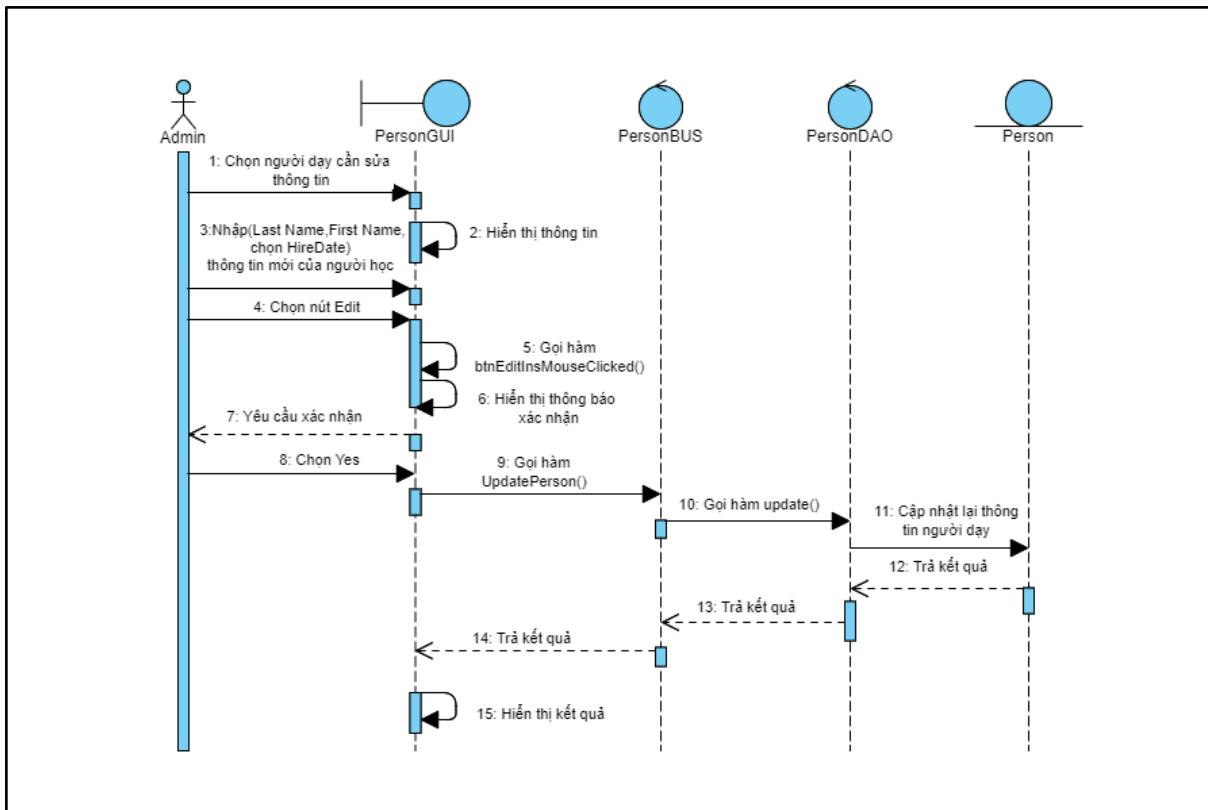
private void btnAddInsMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:

    PersonDTO pDTO = new PersonDTO();
    pDTO.setFirstName(txtFirstNameIns.getText());
    pDTO.setLastName(txtLastNameIns.getText());
    pDTO.setHireDate(formatType.format(dateHire.getDate()));
    if (pBUS.AddPerson(pDTO)) {
        JOptionPane.showMessageDialog(null, "Thêm thành công");
        readTableInstructor();
        cbPersonID.setModel(new DefaultComboBoxModel(ComboBoxPersonID()));
        cbPersonIdCouIns.setModel(new DefaultComboBoxModel(ComboBoxPersonInstructorID()));
    } else {
        JOptionPane.showMessageDialog(null, "Thêm thất bại");
    }
}

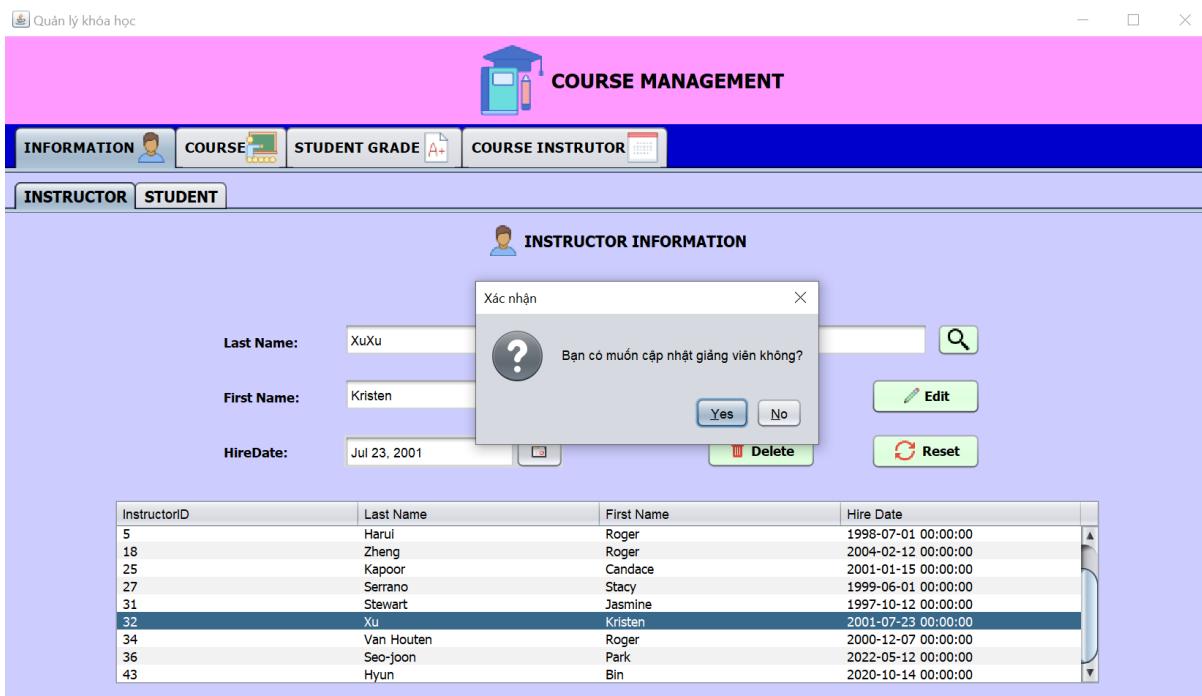
```

### 3. Xử lý 3: Sửa thông tin người dạy

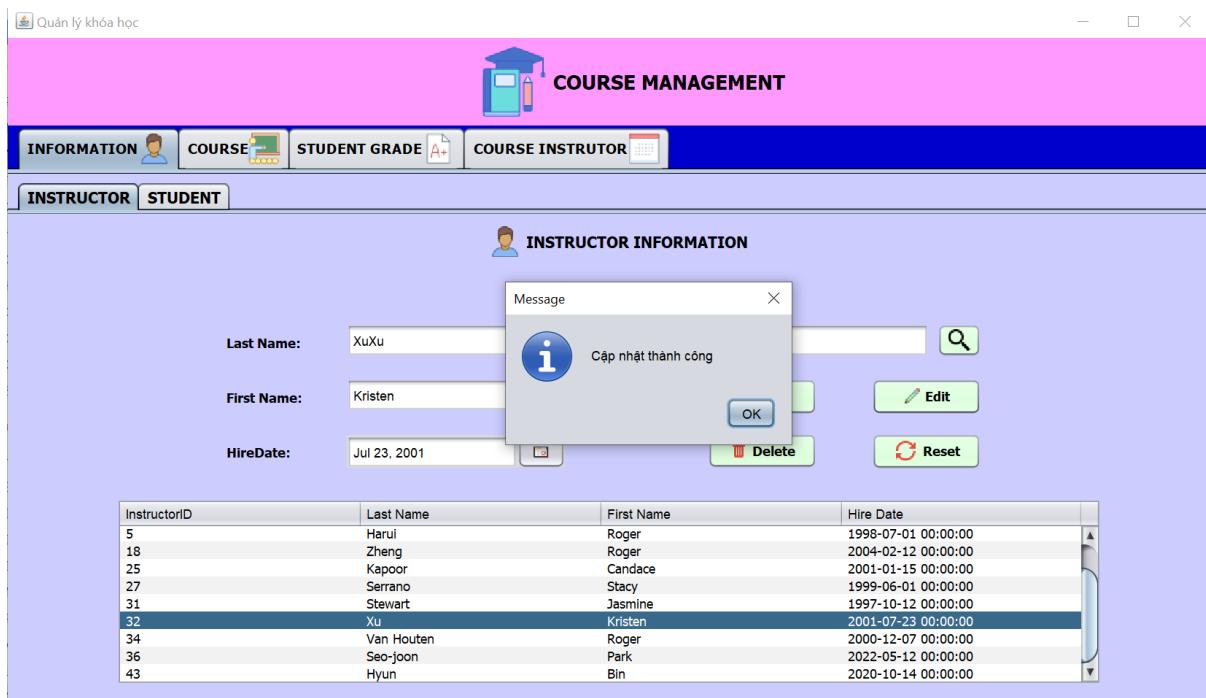
#### 3.1. Sơ đồ tuần tự



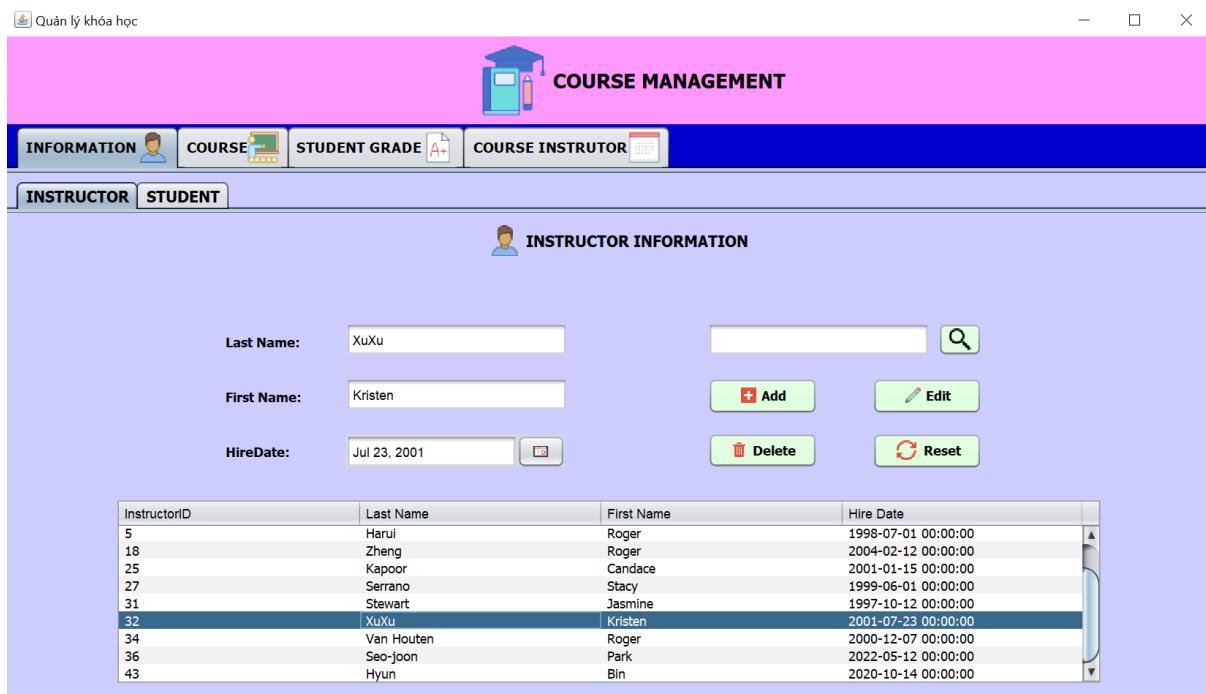
### 3.2. Giao diện



Giao diện xác nhận cập nhật thông tin người dạy mới sau khi bấm nút Edit



Giao diện cập nhật thông tin người dạy thành công



Giao diện người dạy sau khi cập nhật thông tin

### 3.3. Code 3 class:

❖ DAO

```

public boolean update(PersonDTO p) {
    String sql;
    if (p.getHireDate() == null) {
        LocalDate enrollmentDate = LocalDate.parse(p.getEnrollmentDate(), dtf);
        sql = String.format("UPDATE `person`"
            + "SET LastName='%s', Firstname='%s', EnrollmentDate='%s'"
            + "WHERE PersonID='%s'", p.getLastName(), p.getFirstName(), enrollmentDate, p.getPersonID());
    } else {
        LocalDate hireDate = LocalDate.parse(p.getHireDate(), dtf);
        sql = String.format("UPDATE `person`"
            + "SET LastName='%s', Firstname='%s', HireDate='%s'"
            + "WHERE PersonID='%s'", p.getLastName(), p.getFirstName(), hireDate, p.getPersonID());
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean UpdatePerson(PersonDTO p) {
    for (int i = 0; i < PersonList.size(); i++) {
        if (p.getPersonID() == PersonList.get(i).getPersonID()) {
            PersonList.set(i, p);
            break;
        }
    }
    return pDAO.update(p);
}

```

## ❖ GUI

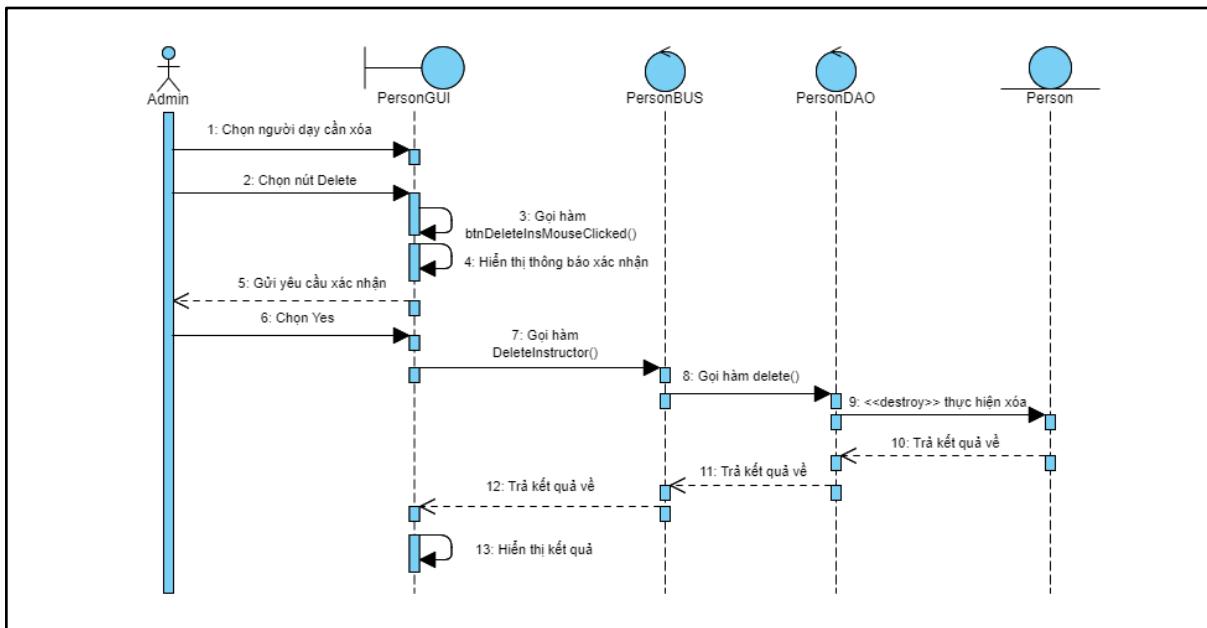
```

private void btnEditInsMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int i = tblInstructor.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn giảng viên cần cập nhật");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn cập nhật giảng viên không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            PersonDTO pDTO = new PersonDTO();
            pDTO.setPersonID(Integer.parseInt(modelTableInstructor.getValueAt(i, 0).toString()));
            pDTO.setFirstName(txtFirstNameIns.getText());
            pDTO.setLastName(txtLastNameIns.getText());
            pDTO.setHireDate(formatType.format(dateHire.getDate()));
            if (pBUS.UpdatePerson(pDTO)) {
                JOptionPane.showMessageDialog(null, "Cập nhật thành công");
                txtFirstNameIns.setText("");
                txtLastNameIns.setText("");
                dateHire.setDate(null);
                readTableInstructor();
                cbPersonID.setModel(new DefaultComboBoxModel(ComboBoxPersonID()));
                cbPersonIdCouIns.setModel(new DefaultComboBoxModel(ComboBoxPersonInstructorID()));
            } else {
                JOptionPane.showMessageDialog(null, "Cập nhật thất bại");
            }
        }
    }
}

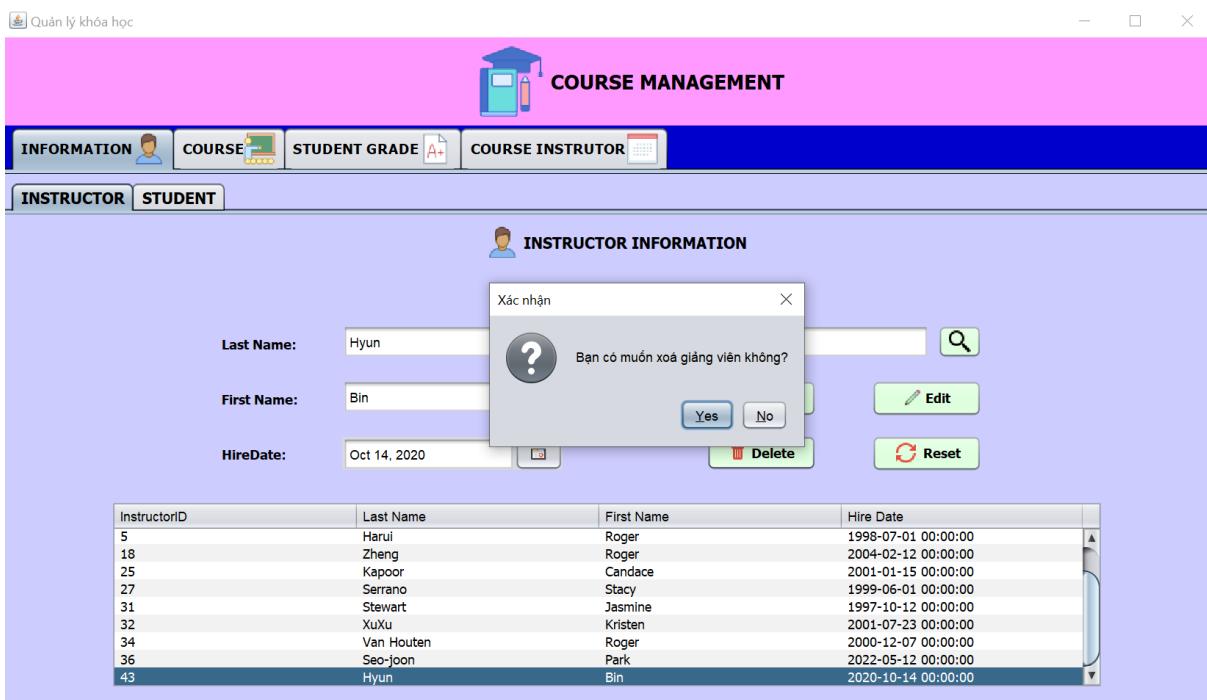
```

## 4. Xử lý 4: Xóa thông tin người dạy

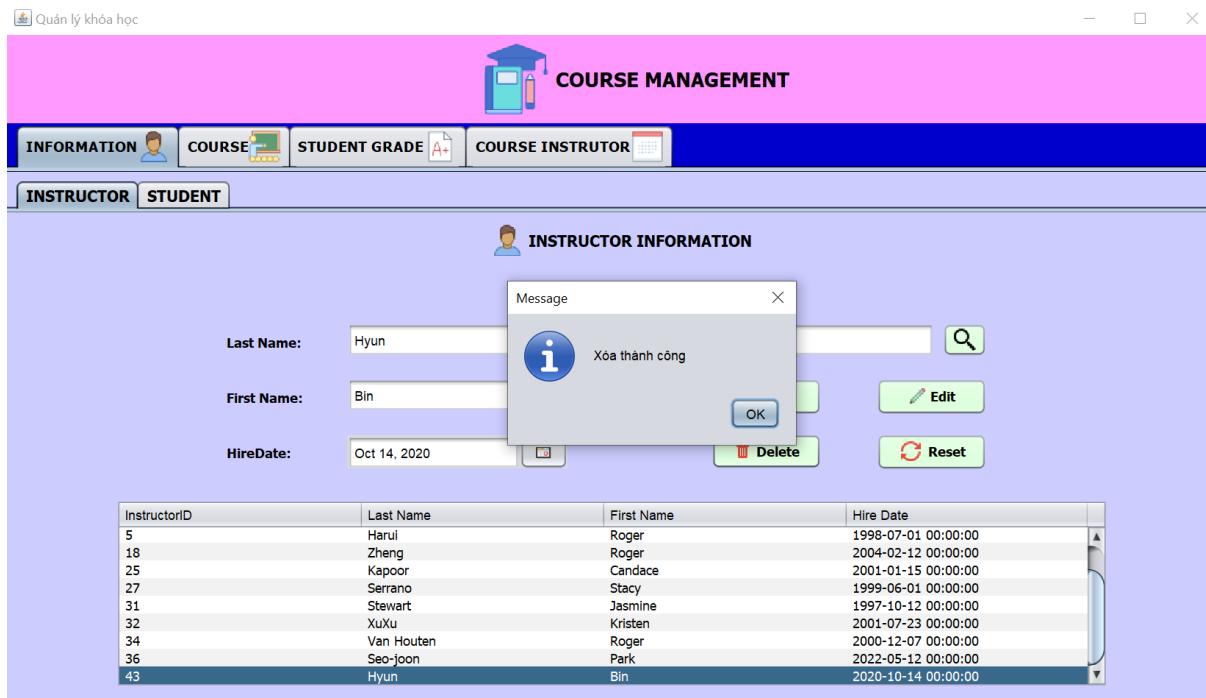
### 5.1. Sơ đồ tuần tự



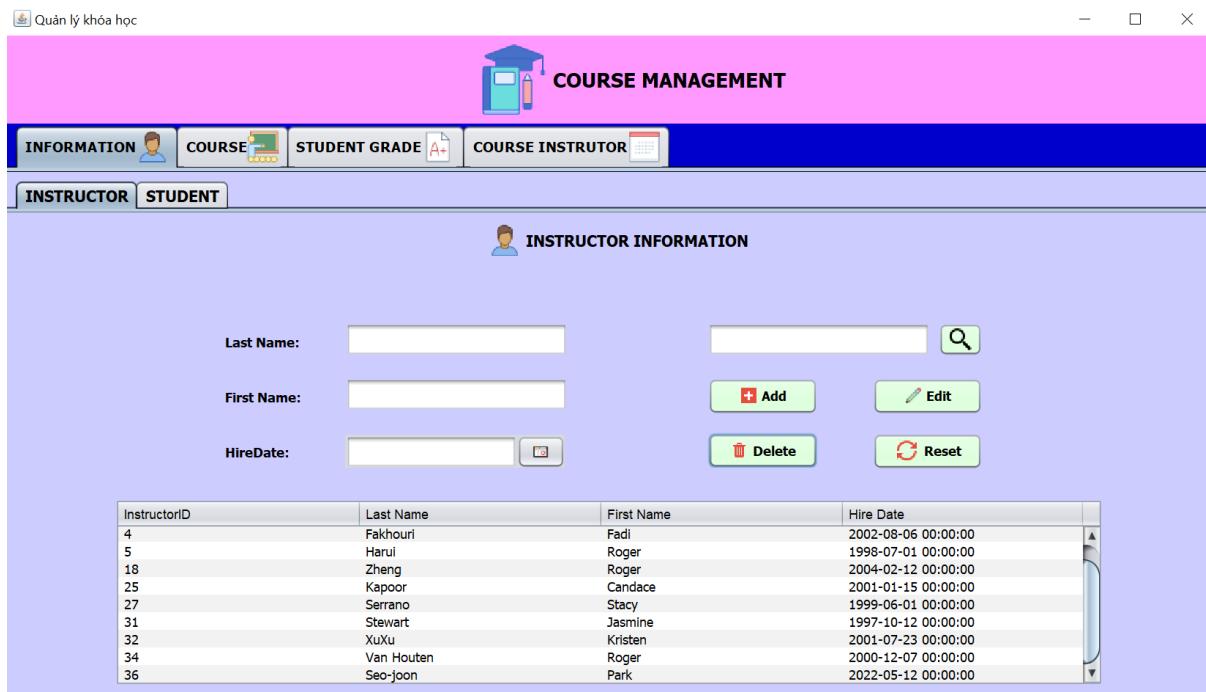
### 5.2. Giao diện



Giao diện xác nhận xóa thông tin người dạy đã chọn



Giao diện xóa thông tin người dạy thành công



Giao diện hiển thị danh sách thông tin người dạy sau khi xóa

### 5.3. Code 3 class:

❖ DAO

```

public boolean delete(int id) {
    String sql = String.format("DELETE FROM `person` WHERE PersonID='%s'", id);
    String result = ConnectDatabase.GetInstance().ExecuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean DeleteInstructor(int id){
    for (CourseInstructorDTO ciDTO : CourseInstructorBUS.CourseInstructorList) {
        if (ciDTO.getPersonID() == id) {
            return false;
        }
    }
    for (PersonDTO person : PersonList) {
        if (person.getPersonID() == id) {
            PersonList.remove(person);
            break;
        }
    }
    return pDAO.delete(id);
}

```

## ❖ GUI

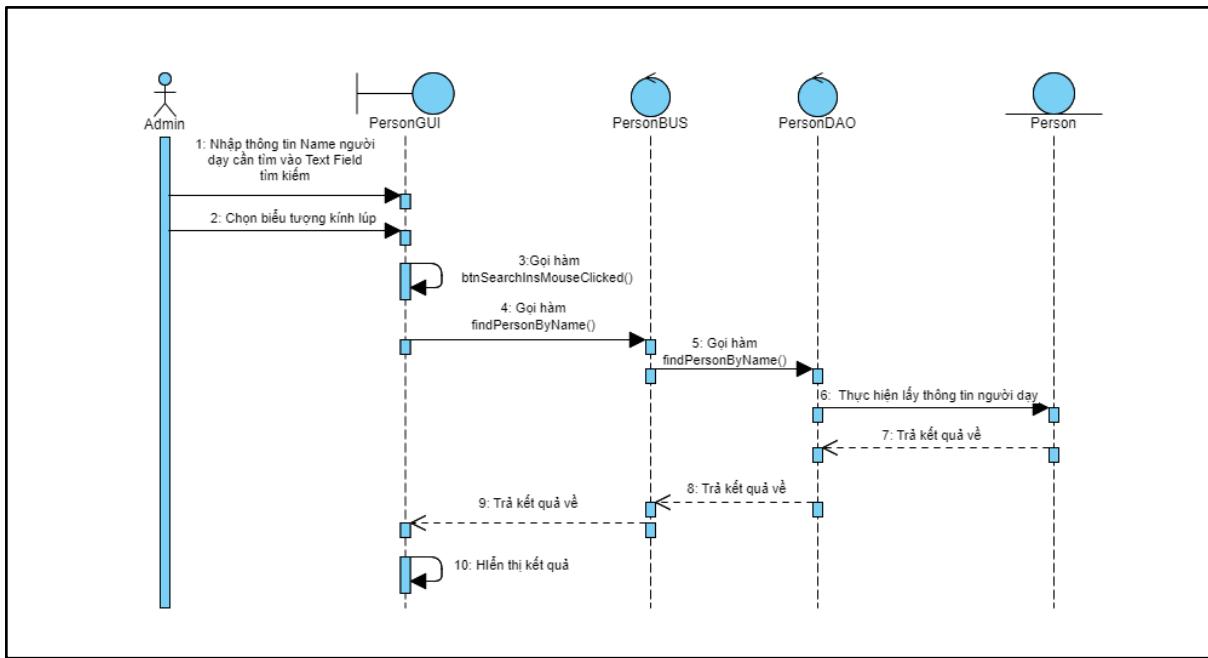
```

private void btnDeleteInsMouseClicked(java.awt.event.MouseEvent evt) {
    int i = tbInstructor.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn giảng viên cần xoá");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn xoá giảng viên không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            int idInstructor = Integer.parseInt(modelTableInstructor.getValueAt(i, 0).toString());
            if (PBUS.DeleteInstructor(idInstructor)) {
                JOptionPane.showMessageDialog(null, "Xóa thành công");
                txtFirstNameIns.setText("");
                txtLastNameIns.setText("");
                dateHire.setDate(null);
                readTableInstructor();
                cbPersonID.setModel(new DefaultComboBoxModel(ComboBoxPersonID()));
                cbPersonidCouIns.setModel(new DefaultComboBoxModel(ComboBoxPersonInstructorID()));
            } else {
                JOptionPane.showMessageDialog(null, "Xóa thất bại");
            }
        }
    }
}

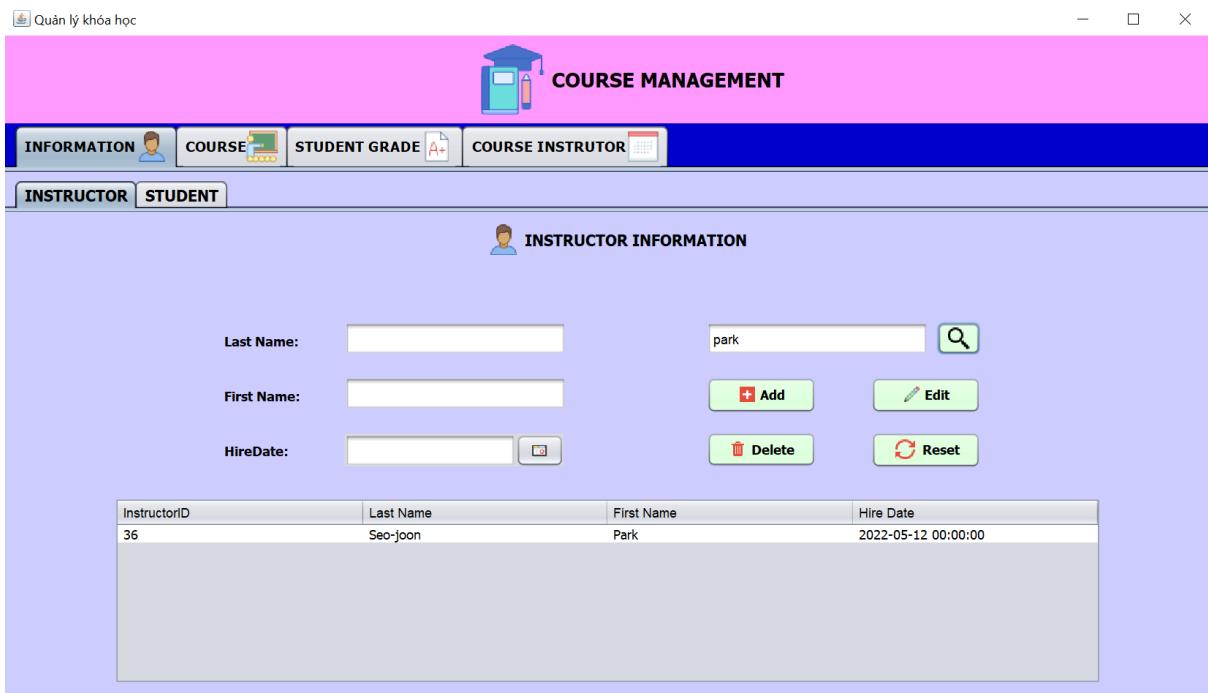
```

## 5. Xử lý 5: Tìm kiếm thông tin người dạy theo tên

### 5.1. Sơ đồ tuần tự



## 5.2. Giao diện



*Giao diện tìm kiếm thông tin người dạy theo tên*

## 5.3. Code 3 class:

❖ DAO

```

public ArrayList<PersonDTO> findPersonByName(String name) throws SQLException {
    ArrayList<PersonDTO> personList = new ArrayList<>();
    if (name.contains(" ")) {
        String[] word = name.split("\\s+");
        String sql = String.format("SELECT * FROM `person` WHERE (`Firstname` LIKE '%s%' AND Lastname LIKE '%s%')",
            word[0], word[1]);
        ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
        while (rs.next()) {
            PersonDTO pDTO = new PersonDTO();
            pDTO.setPersonID(rs.getInt("PersonID"));
            pDTO.setLastName(rs.getString("Lastname"));
            pDTO.setFirstName(rs.getString("Firstname"));
            pDTO.setHireDate(rs.getString("HireDate"));
            pDTO.setEnrollmentDate(rs.getString("EnrollmentDate"));
            personList.add(pDTO);
        }
    } else {
        String sql = "SELECT * FROM `person` WHERE (`Firstname` LIKE '%" + name + "%' OR Lastname LIKE '%" + name + "%')";
        ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
        while (rs.next()) {
            PersonDTO pDTO = new PersonDTO();
            pDTO.setPersonID(rs.getInt("PersonID"));
            pDTO.setLastName(rs.getString("Lastname"));
            pDTO.setFirstName(rs.getString("Firstname"));
            pDTO.setHireDate(rs.getString("HireDate"));
            pDTO.setEnrollmentDate(rs.getString("EnrollmentDate"));
            personList.add(pDTO);
        }
    }
    return personList;
}

```

## ❖ BUS

```

public ArrayList<PersonDTO> findPersonByName(String name) throws SQLException {
    return pDAO.findPersonByName(name);
}

```

## ❖ GUI

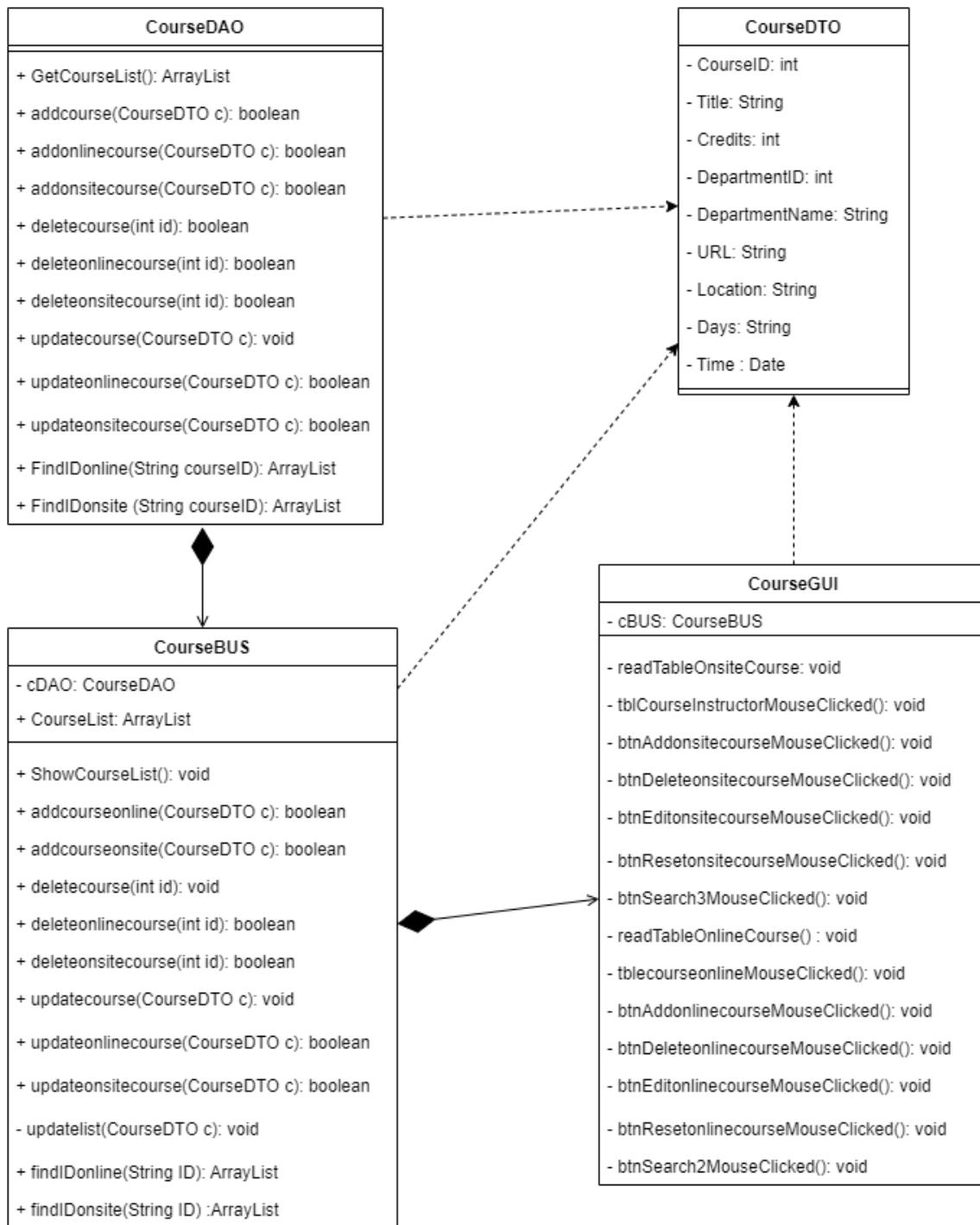
```

private void btnSearchInsMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    String input = txtSearchIns.getText();
    System.out.println(input);
    if (!input.isEmpty()) {
        try {
            ArrayList<PersonDTO> listPerson = pBUS.findPersonByName(input);
            if (!listPerson.isEmpty()) {
                modelTableInstructor.setRowCount(0);
                for (PersonDTO p : listPerson) {
                    if (p.getEnrollmentDate() == null) {
                        modelTableInstructor.addRow(new Object[]{
                            p.getPersonID(), p.getLastName(), p.getFirstName(), p.getHireDate()
                        });
                    }
                }
                JOptionPane.showMessageDialog(null, "Tìm thấy rồi");
            } else {
                JOptionPane.showMessageDialog(null, "Không tìm thấy");
            }
        } catch (SQLException ex) {
            Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
        }
    } else {
        JOptionPane.showMessageDialog(null, "Thiếu Thông Tin");
    }
}

```

## Phần 2: Chức năng quản lý thông tin các khóa học online, onsite

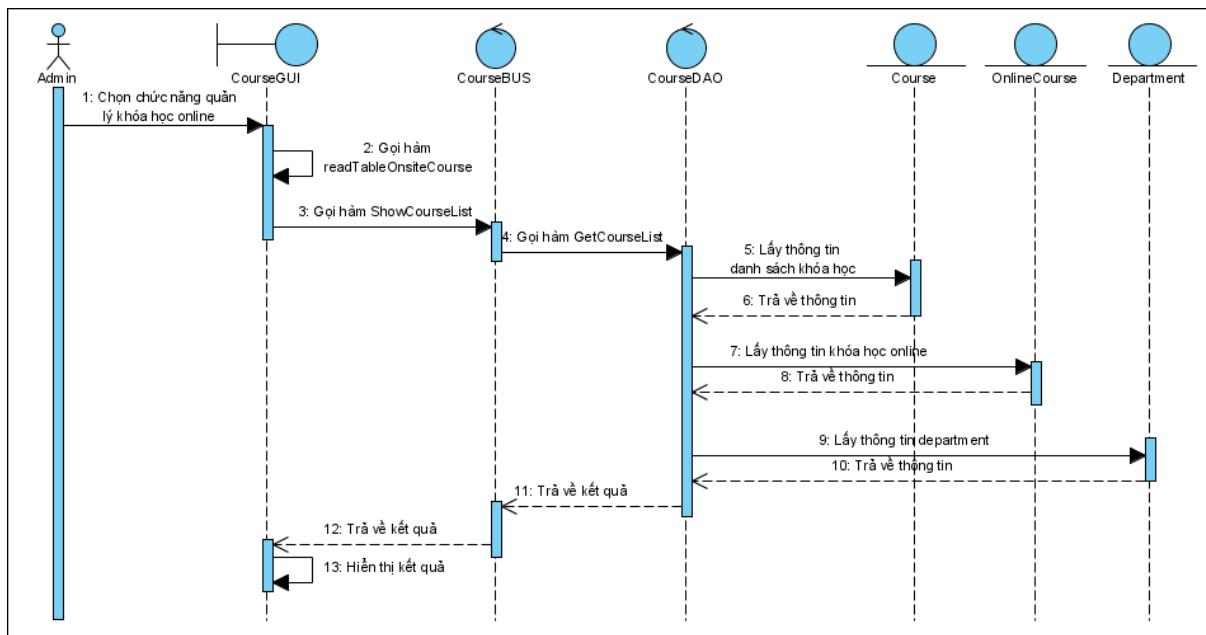
### I. Sơ đồ class chung



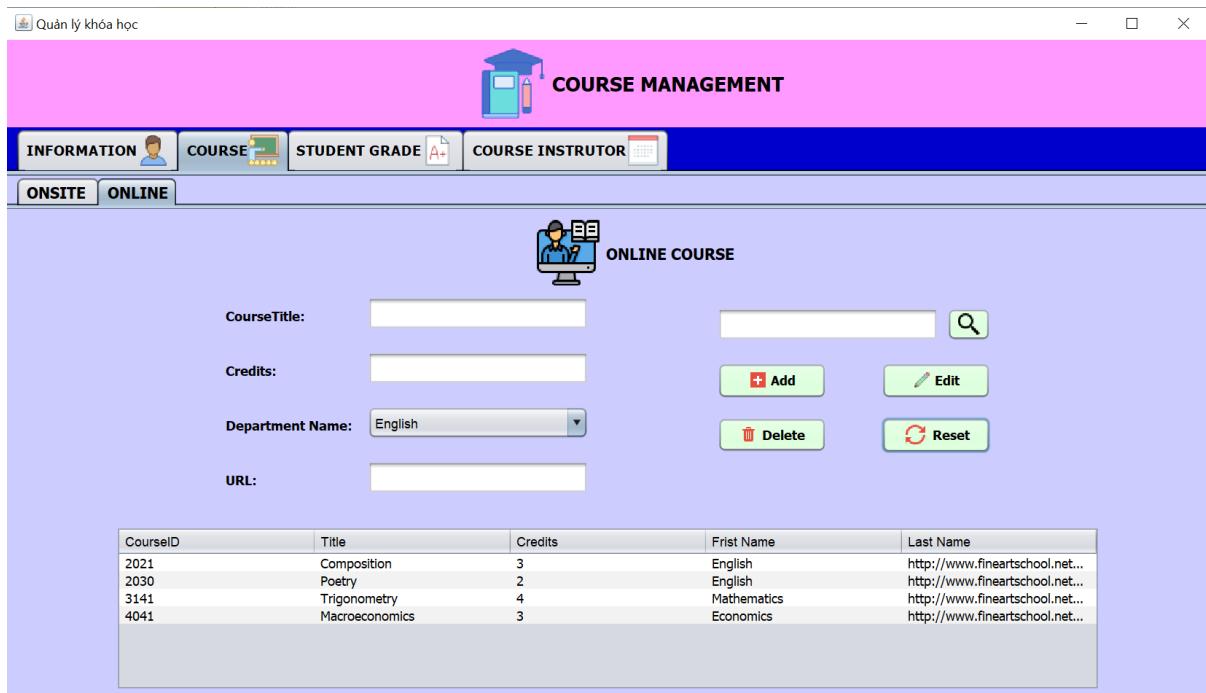
### II. Quản lý thông tin khóa học online

#### 1. Xử lý 1: Hiển thị danh sách thông tin khóa học online

##### 1.1 Sơ đồ tuần tự



## 1.2 Giao diện



Giao diện hiển thị danh sách thông tin khóa học online

## 1.3 Code 3 class:

❖ DAO

```

public ArrayList<CourseDTO> GetCourseList() throws SQLException {
    ArrayList<CourseDTO> courseList = new ArrayList<>();
    String sql = "select * from course "
        + "left JOIN onlinecourse on onlinecourse.CourseID = course.CourseID "
        + "LEFT JOIN onsitecourse on course.CourseID = onsitecourse.CourseID";
    ResultSet rs = ConnectDatabase.GetInstance().ExcuteSELECT(sql);
    while (rs.next()) {
        String sql2 = "select * from department where DepartmentID =" + rs.getInt("DepartmentID");
        ResultSet rs2 = ConnectDatabase.GetInstance().ExcuteSELECT(sql2);
        CourseDTO cDTO = new CourseDTO();
        cDTO.setCourseID(rs.getInt("course.CourseID"));
        cDTO.setTitle(rs.getString("course.Title"));
        cDTO.setCredits(rs.getInt("course.Credits"));
        cDTO.setDepartmentID(rs.getInt("DepartmentID"));
        while (rs2.next()) {
            cDTO.setDepartmentName(rs2.getString("Name"));
        }
        cDTO.setURL(rs.getString("url"));
        cDTO.setLocation(rs.getString("Location"));
        cDTO.setDays(rs.getString("Days"));
        cDTO.setTime(rs.getTime("Time"));
        courseList.add(cDTO);
    }
    return courseList;
}

```

## ❖ BUS

```

public void ShowCourseList() throws SQLException {
    CourseList = cDAO.GetCourseList();
}

```

## ❖ GUI

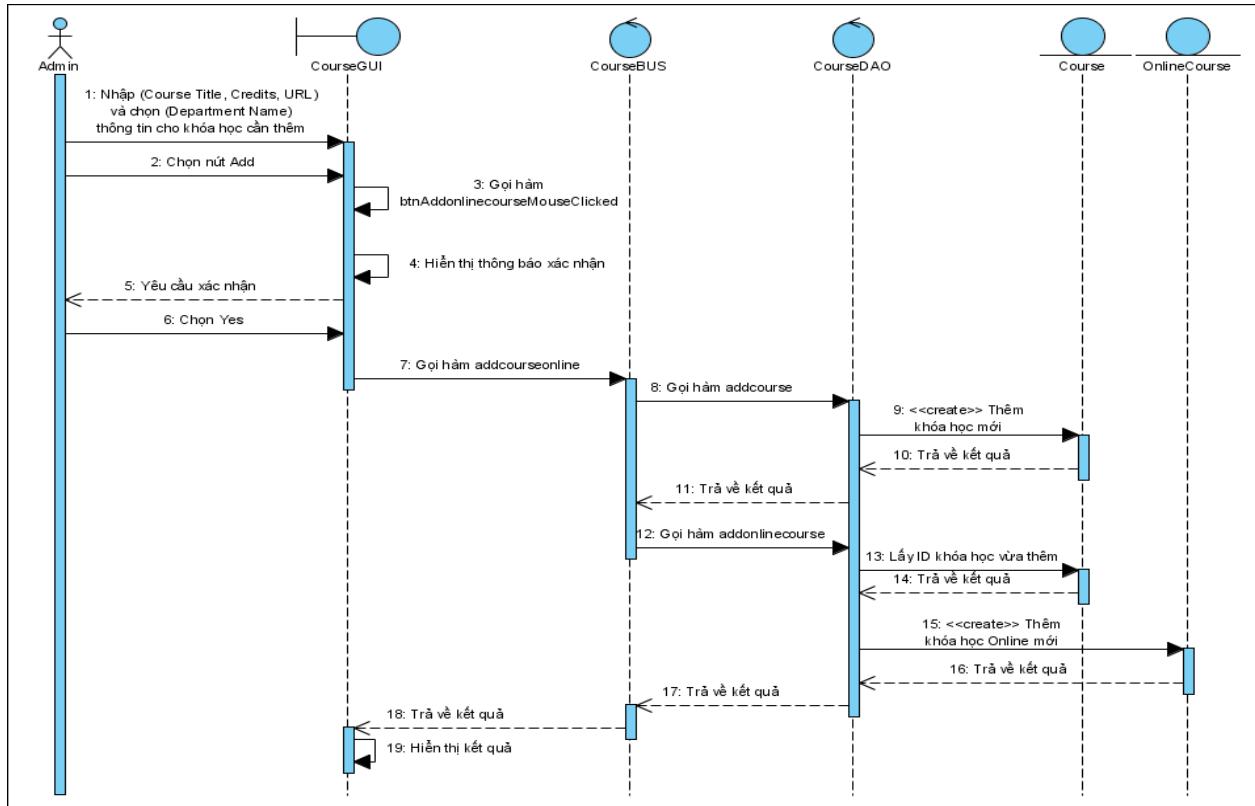
```

private void readTableOnlineCourse() {
    try {
        cBUS.ShowCourseList();
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
    modelTableCourseOnline.setRowCount(0);
    for (CourseDTO cdto : CourseBUS.CourseList) {
        if (cdto.getURL() != null) {
            modelTableCourseOnline.addRow(new Object[]{
                cdto.getCourseID(), cdto.getTitle(), cdto.getCredits(),
                cdto.getDepartmentName(), cdto.getURL()
            });
        }
    }
}

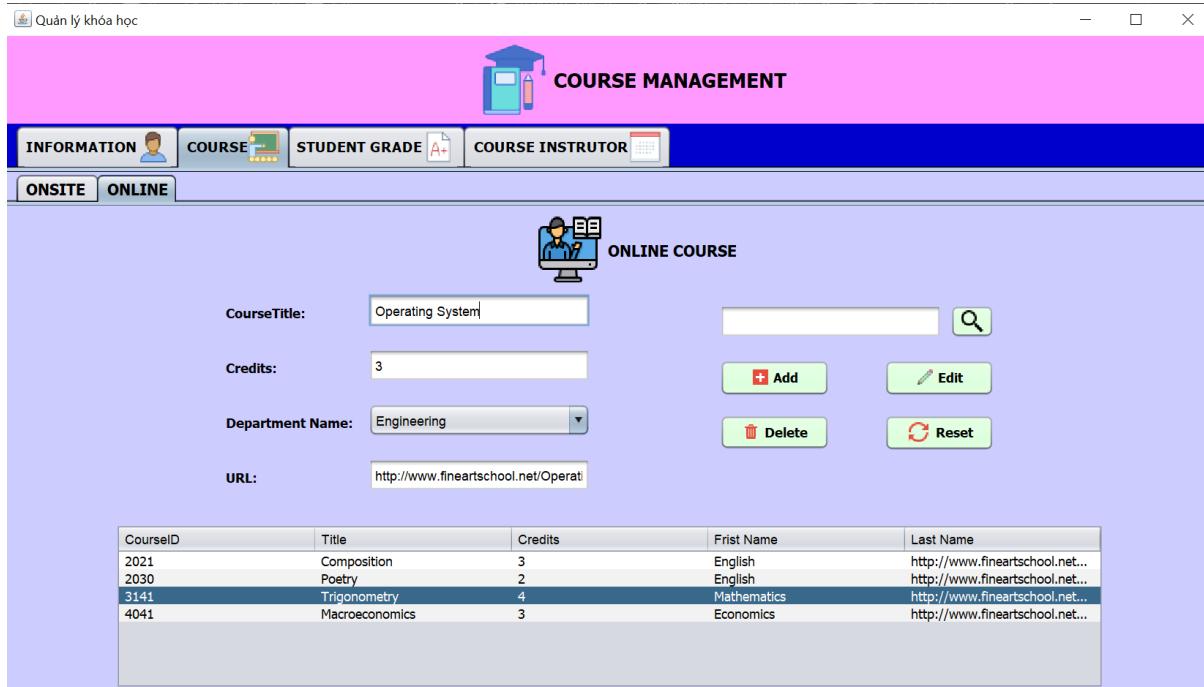
```

## 2. Xử lý 2: Thêm thông tin khóa học online

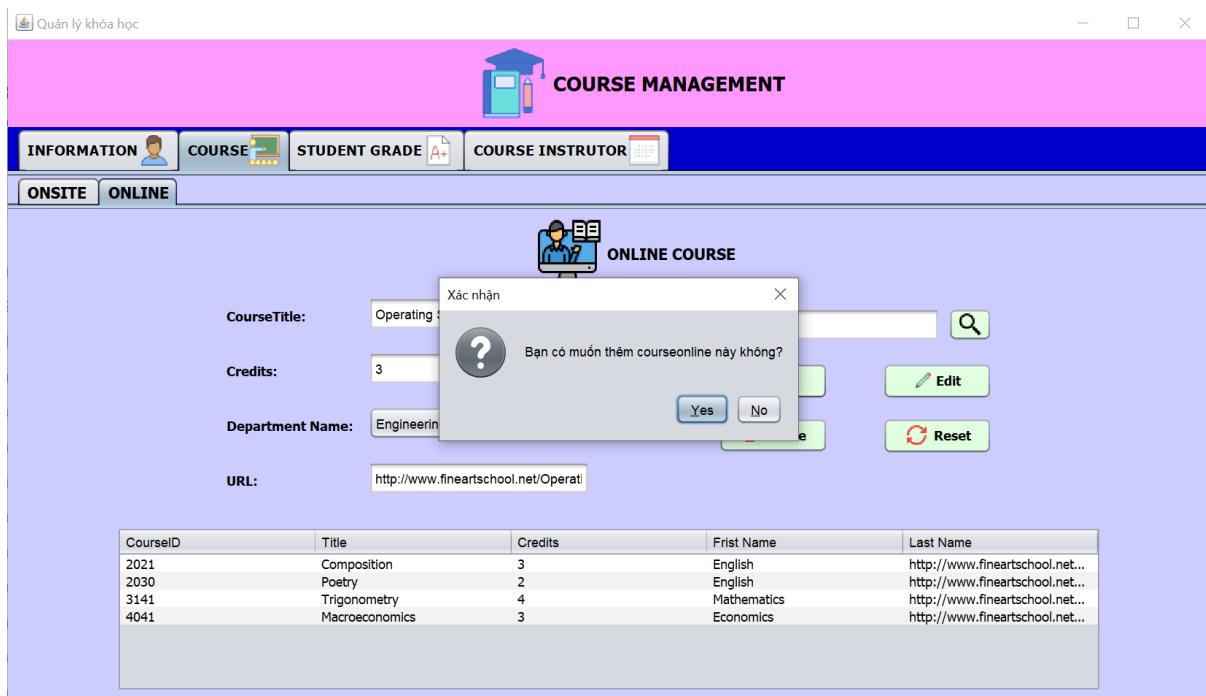
### 2.1 Sơ đồ tuần tự



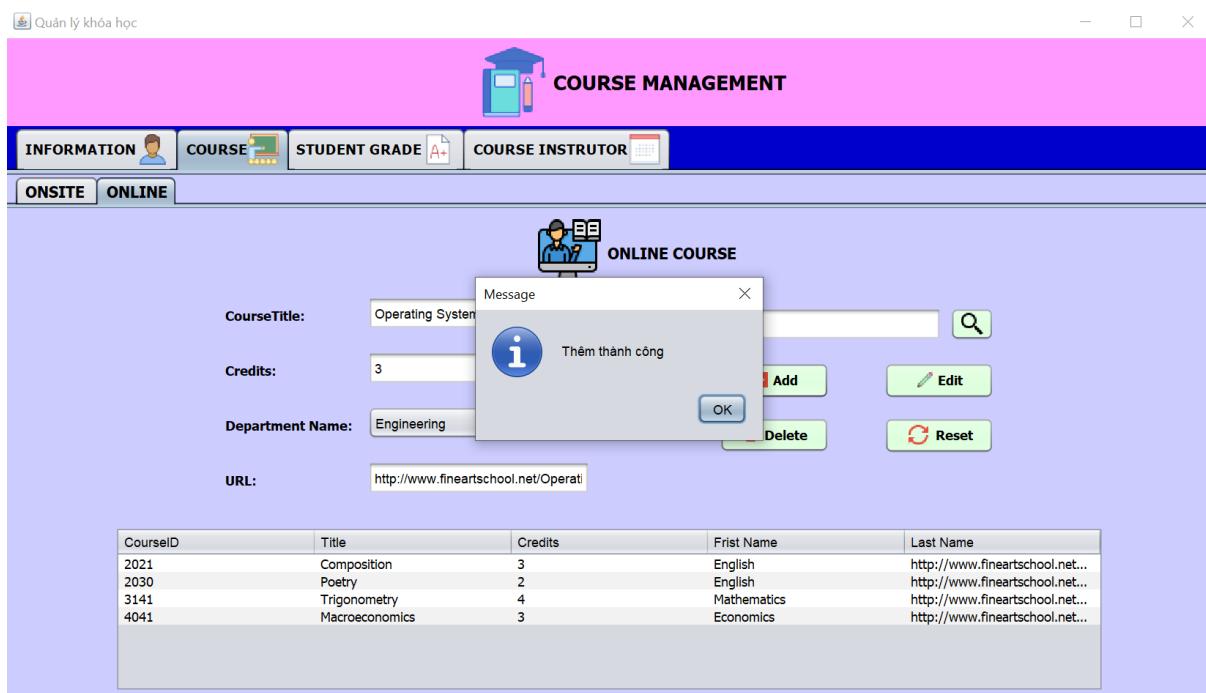
### 2.2 Giao diện



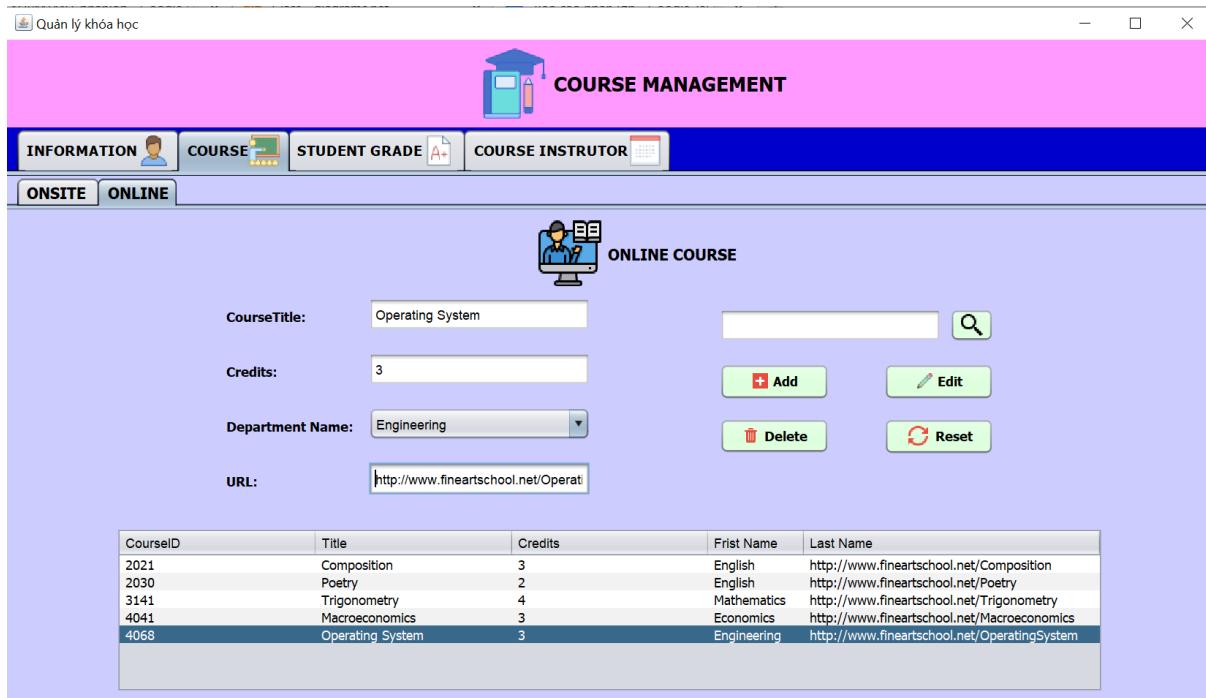
Giao diện nhập thông tin khóa học online mới



Giao diện thêm khóa học online mới sau khi bấm nút Add



Giao diện thông báo thêm khóa học online mới thành công



*Giao diện danh sách thông tin khóa học online sau khi thêm*

## 2.3 Code 3 class:

### ❖ DAO

```
public boolean addcourse(CourseDTO c) {
    String sql = null;
    if (c != null) {
        sql = String.format("INSERT INTO `course`(`Title`, `Credits`, `DepartmentID`) VALUES ('%s', '%d', '%d')",
            c.getTitle(), c.getCredits(), c.getDepartmentID());
    }
    String result = ConnectDatabase.GetInstance().ExecuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}
```

```
public boolean addonlinecourse(CourseDTO c) throws SQLException{
    String sql = null;
    String sql2 = String.format("SELECT * FROM `course` WHERE Title = '%s'", c.getTitle());
    ResultSet rs2 = ConnectDatabase.GetInstance().ExecuteSELECT(sql2);
    int temp = 0;
    while(rs2.next()){
        temp = rs2.getInt("CourseID");
    }
    if(c != null){
        sql =String.format("INSERT INTO `onlinecourse`(`CourseID`, `url`) VALUES ('%d', '%s')",temp,c.getURL());
    }
    String result = ConnectDatabase.GetInstance().ExecuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    }else
        return false;
}
```

### ❖ BUS

```

public boolean addcourseonline(CourseDTO c) throws SQLException {
    cDAO.addcourse(c);
    return cDAO.addonlinecourse(c);
}

```

## ❖ GUI

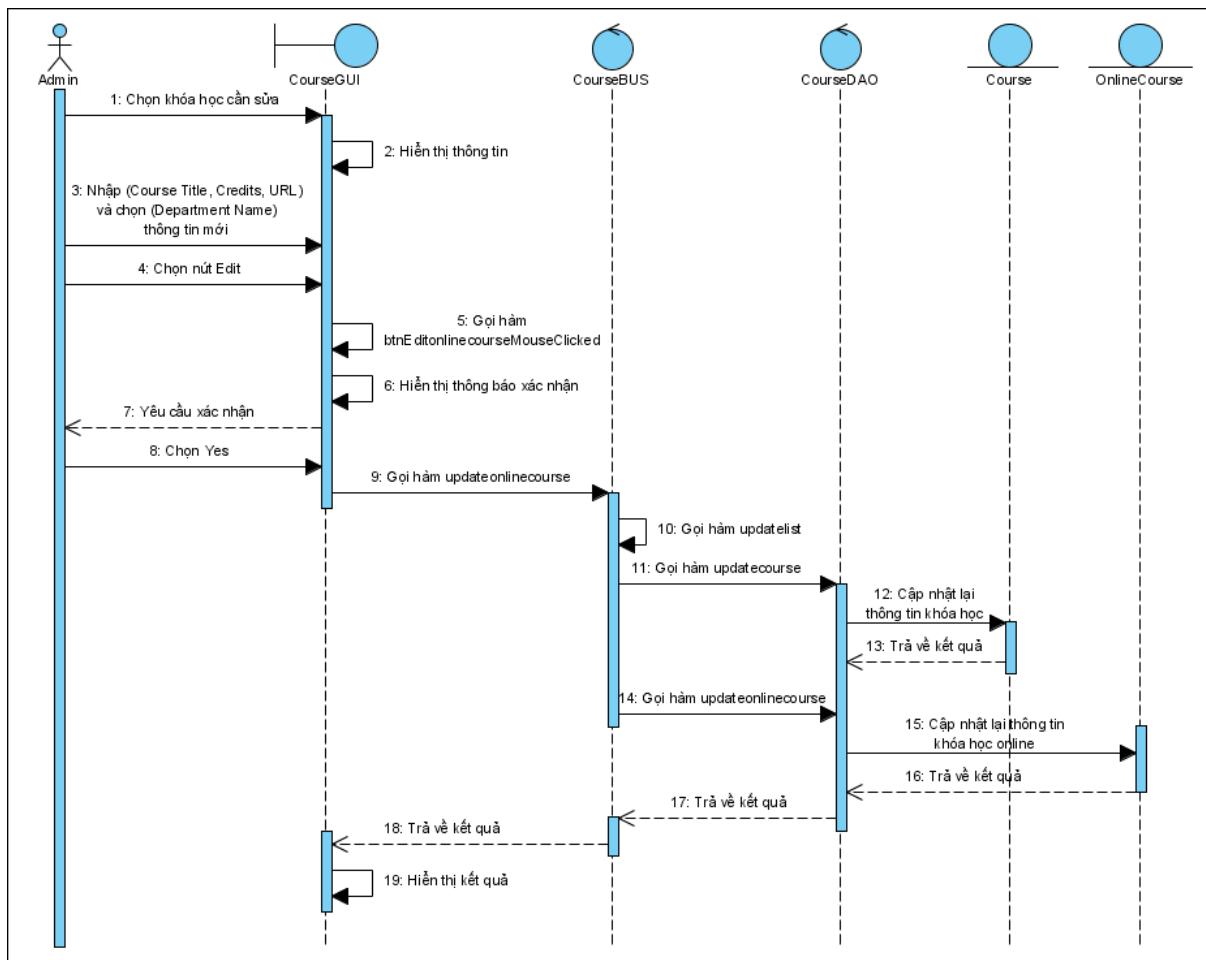
```

private void btnAddonlinecourseMouseClicked(java.awt.event.MouseEvent evt) {
    int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn thêm courseonline này không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
    if (result == JOptionPane.NO_OPTION) {
        return;
    }
    CourseDTO cDTO = new CourseDTO();
    int temp = 0;
    for (DepartmentDTO dpDTO : DepartmentBUS.dpList) {
        if (dpDTO.getName() == jdnameonline.getSelectedItem()) {
            temp = dpDTO.getDepartmentID();
            break;
        }
    }
    cDTO.settitle(txtcoursetitle2.getText());
    cDTO.setCredits(Integer.parseInt(txtcredits2.getText()));
    cDTO.setDepartmentID(temp);
    cDTO.setURL(txturl.getText());
    try {
        if (cBUS.addcourseonline(cDTO)) {
            JOptionPane.showMessageDialog(null, "Thêm thành công");
            readTableOnlineCourse();
            cbCourseTitle.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
            cbCourseTitleCouIns.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
        } else {
            JOptionPane.showMessageDialog(null, "Thêm thất bại");
        }
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
}

```

### 3. Xử lý 3: Sửa thông tin khóa học online

#### 4.1 Sơ đồ tuần tự



## 4.2 Giao diện

Quản lý khóa học

**COURSE MANAGEMENT**

**INFORMATION** **COURSE** **STUDENT GRADE** **COURSE INSTRUTOR**

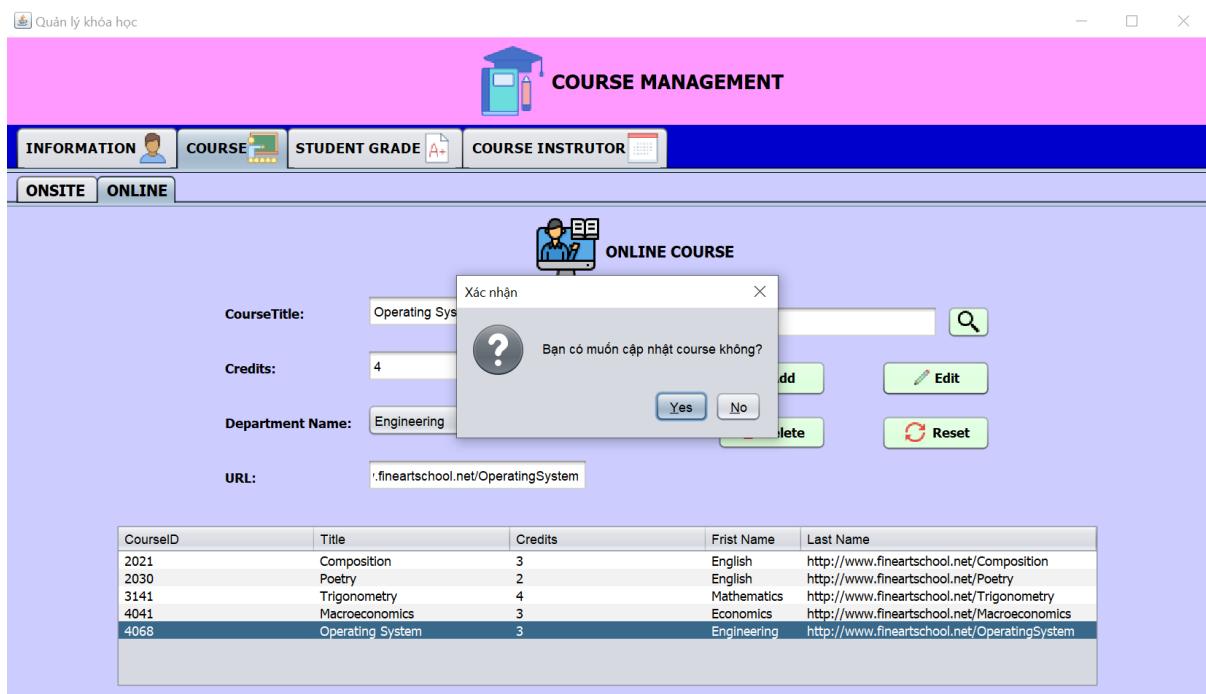
**ONSITE** **ONLINE**

**ONLINE COURSE**

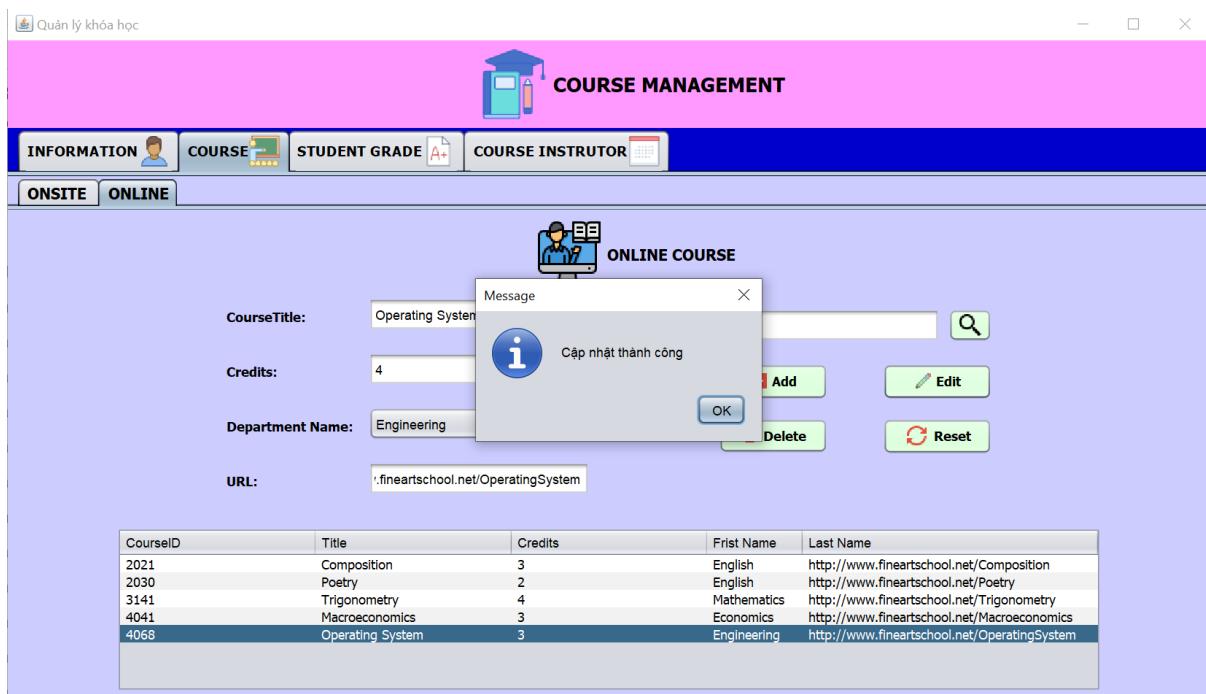
CourseTitle:	Operating System	<input type="button" value="Search"/>
Credits:	3	<input type="button" value="Add"/> <input type="button" value="Edit"/>
Department Name:	Engineering	<input type="button" value="Delete"/> <input type="button" value="Reset"/>
URL:	<a href="http://www.fineartschool.net/OperatingSystem">http://www.fineartschool.net/OperatingSystem</a>	

CourseID	Title	Credits	First Name	Last Name
2021	Composition	3	English	<a href="http://www.fineartschool.net/Composition">http://www.fineartschool.net/Composition</a>
2030	Poetry	2	English	<a href="http://www.fineartschool.net/Poetry">http://www.fineartschool.net/Poetry</a>
3141	Trigonometry	4	Mathematics	<a href="http://www.fineartschool.net/Trigonometry">http://www.fineartschool.net/Trigonometry</a>
4041	Macroeconomics	3	Economics	<a href="http://www.fineartschool.net/Macroeconomics">http://www.fineartschool.net/Macroeconomics</a>
4068	Operating System	3	Engineering	<a href="http://www.fineartschool.net/OperatingSystem">http://www.fineartschool.net/OperatingSystem</a>

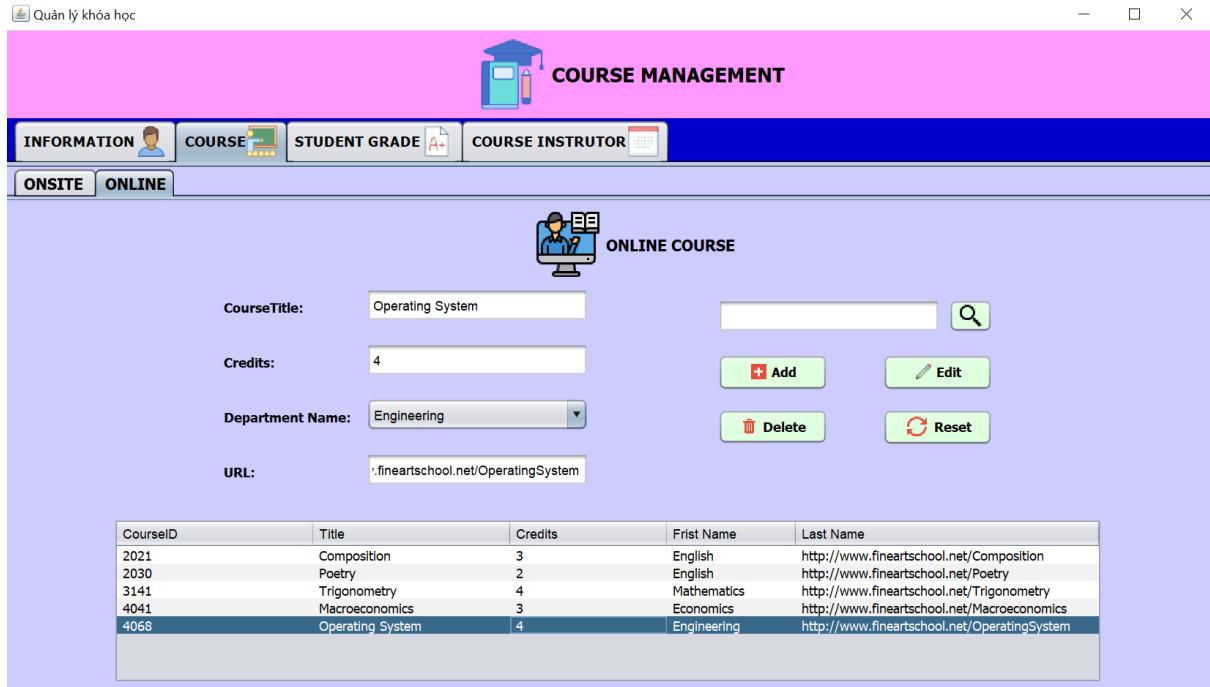
Giao diện thông tin khóa học online trước khi sửa có Credits: 3



Giao diện thông báo xác nhận sau khi nhập Credits: 4 và chọn nút Edit



Giao diện thông báo cập nhật thành công



*Giao diện hiển thị thông tin khóa học online đã được sửa*

### 4.3 Code 3 class:

#### ❖ DAO

```
public void updatecourse(CourseDTO c) {
    String sql = null;
    if (c != null) {
        sql = String.format("UPDATE `course` SET `Title`='%s', `Credits`=%d, `DepartmentID`=%d"
            + "WHERE CourseID = '%d'", c.getTitle(), c.getCredits(), c.getDepartmentID(), c.getCourseID());
    }
    if (sql != null) {
        String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    }
}
```

```
public boolean updateonsitecourse(CourseDTO c) {
    String sql = null;
    if (c != null) {
        sql = String.format("UPDATE `onsitecourse` SET `Location`=%s, `Days`=%s, `Time`=%s"
            + "WHERE CourseID = '%d'", c.getLocation(), c.getDays(), c.getTime(), c.getCourseID());
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}
```

## ❖ BUS

```
public boolean updateonlinecourse(CourseDTO c) {  
    updatelist(c);  
    cDAO.updatecourse(c);  
    return cDAO.updateonlinecourse(c);  
}
```

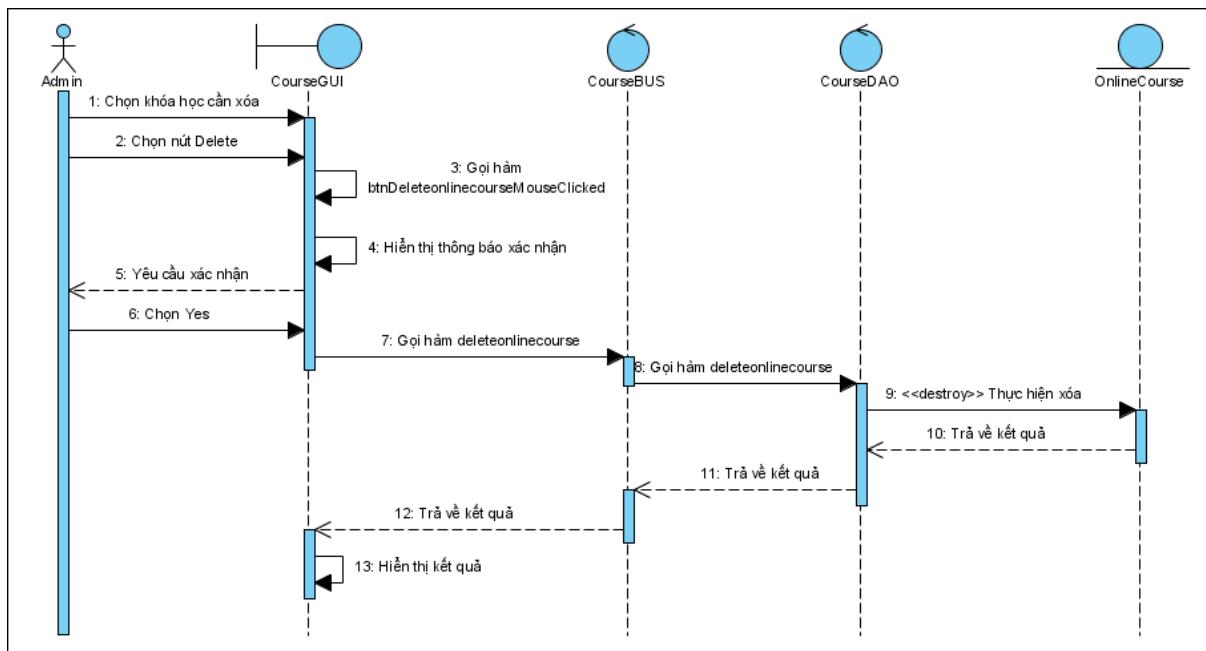
```
private void updatelist(CourseDTO c) {  
    for (int i = 0; i < CourseList.size(); i++) {  
        if (c.getCourseID() == CourseList.get(i).getCourseID()) {  
            CourseList.set(i, c);  
        }  
    }  
}
```

## ❖ GUI

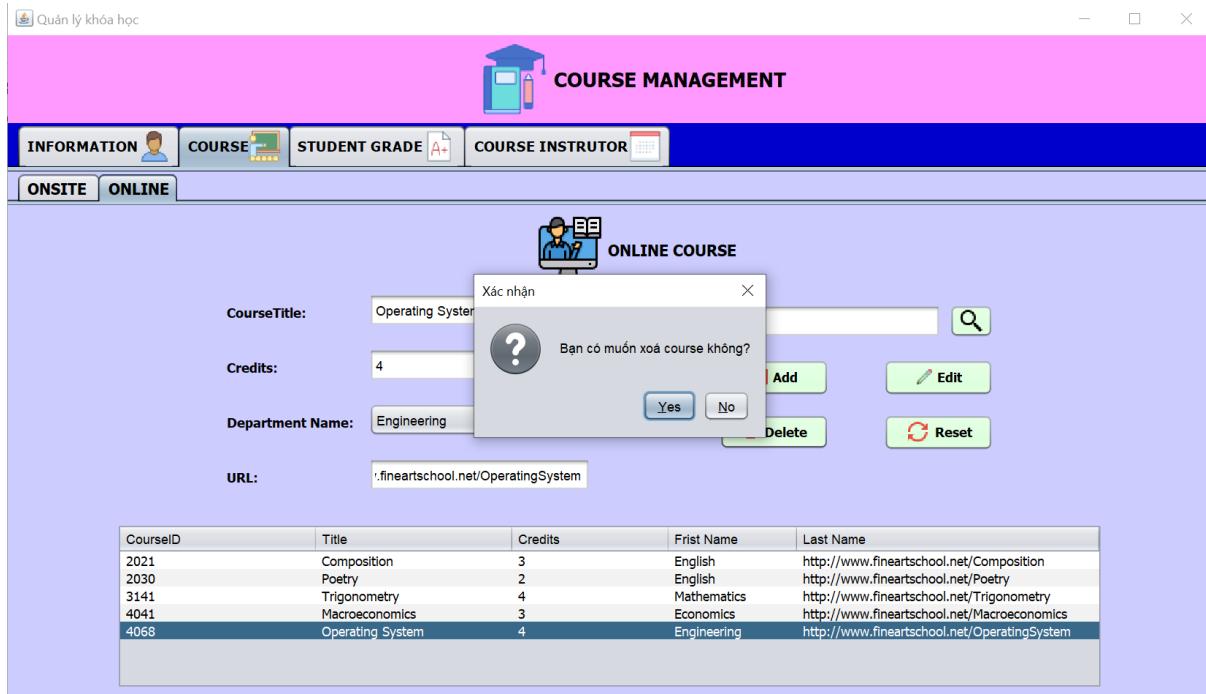
```
private void btnEditonlinecourseMouseClicked(java.awt.event.MouseEvent evt) {  
    int i = tbecauseonline.getSelectedRow();  
    if (i == -1) {  
        JOptionPane.showMessageDialog(null, "Hãy chọn course cần cập nhật");  
    } else {  
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn cập nhật course không?", "Xác nhận", JOptionPane.YES_NO_OPTION);  
        if (result == JOptionPane.YES_OPTION) {  
            CourseDTO cDTO = new CourseDTO();  
            int temp = 0;  
            for (DepartmentDTO dpDTO : DepartmentBUS.dpList) {  
                if (dpDTO.getName() == jdnameonline.getSelectedItem()) {  
                    temp = dpDTO.getDepartmentID();  
                    break;  
                }  
            }  
            cDTO.setCourseID(Integer.parseInt(tbecauseonline.getValueAt(i, 0).toString()));  
            cDTO.setTitle(txtcoursetitle2.getText());  
            cDTO.setCredits(Integer.parseInt(txtcredits2.getText()));  
            cDTO.setDepartmentID(temp);  
            cDTO.setURL(txturl.getText());  
            if (cBUS.updateonlinecourse(cDTO)) {  
                JOptionPane.showMessageDialog(null, "Cập nhật thành công");  
                txtcoursetitle2.setText("");  
                txtcredits2.setText("");  
                txturl.setText("");  
                readTableOnlineCourse();  
                cbCourseTitle.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));  
                cbCourseTitleCouIns.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));  
            } else {  
                JOptionPane.showMessageDialog(null, "Cập nhật thất bại");  
            }  
        }  
    }  
}
```

## 4. Xử lý 4: Xóa thông tin khóa học online

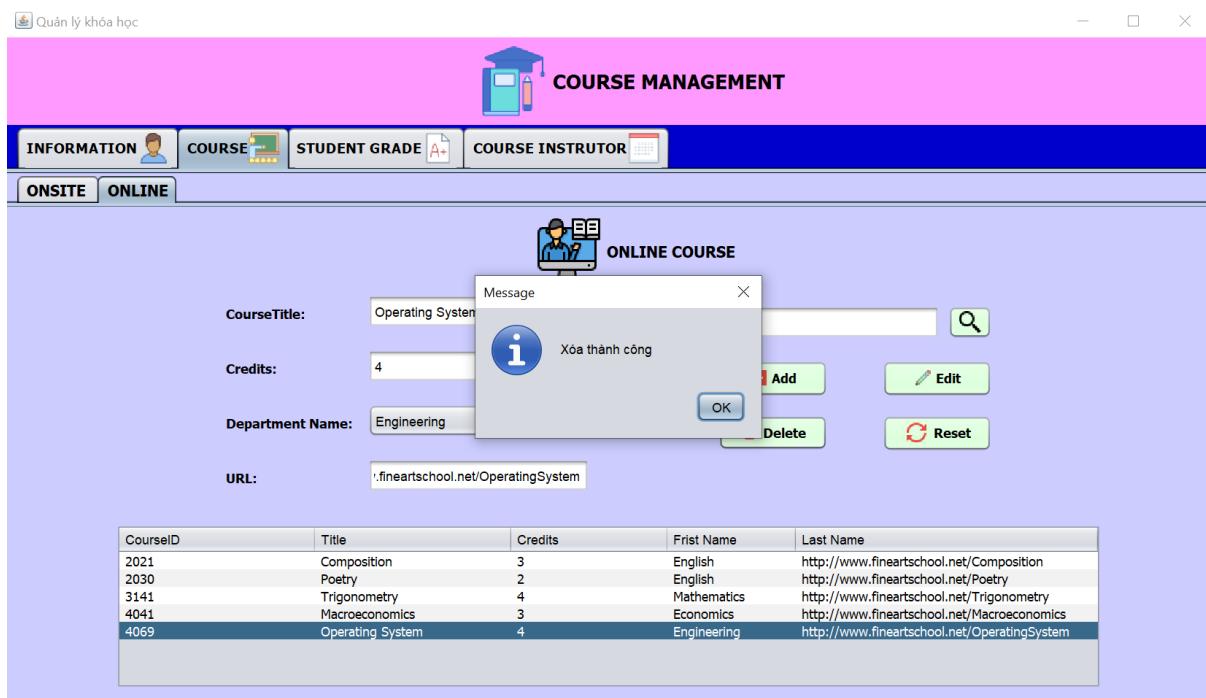
### 3.1 Sơ đồ tuần tự



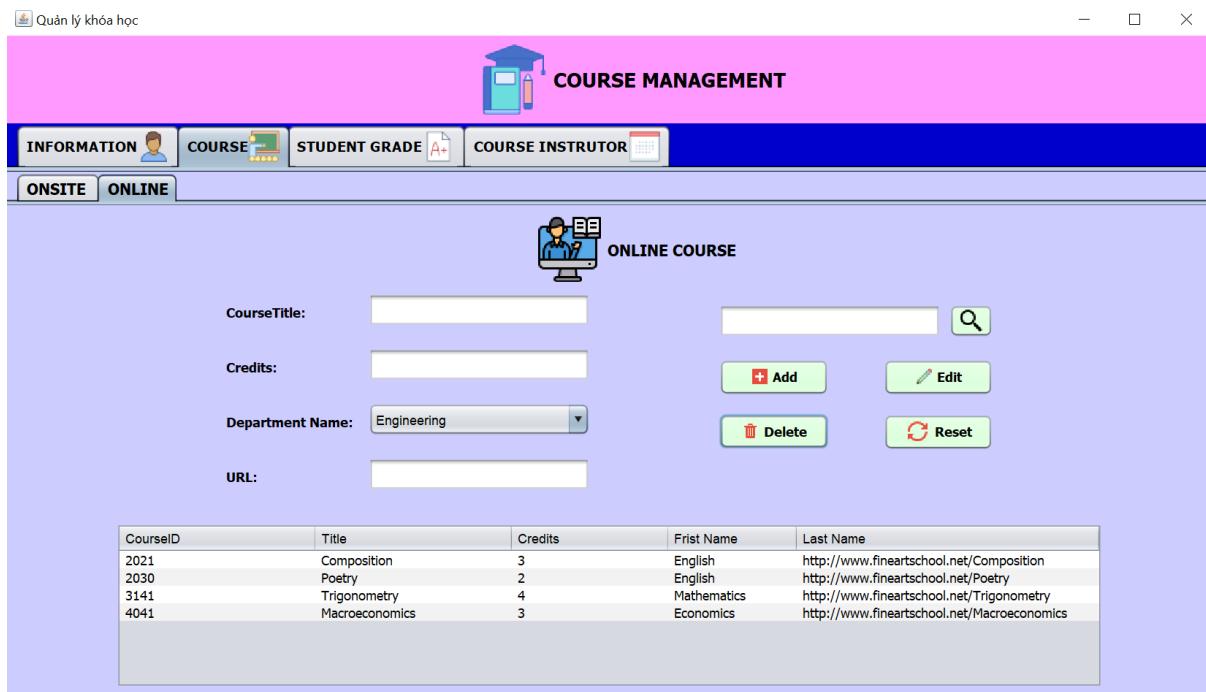
### 3.2 Giao diện



Giao diện thông báo xác nhận xóa



*Giao diện thông báo xóa thành công*



*Giao diện hiển thị danh sách khóa học sau khi xóa*

### 3.3 Code 3 class:

- ❖ DAO

```

public boolean deleteonlinecourse(int id) {
    String sql = String.format("DELETE FROM `onlinecourse` WHERE CourseID='%d'", id);
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean deleteonlinecourse(int id) throws SQLException {
    return cDAO.deleteonlinecourse(id);
}

```

## ❖ GUI

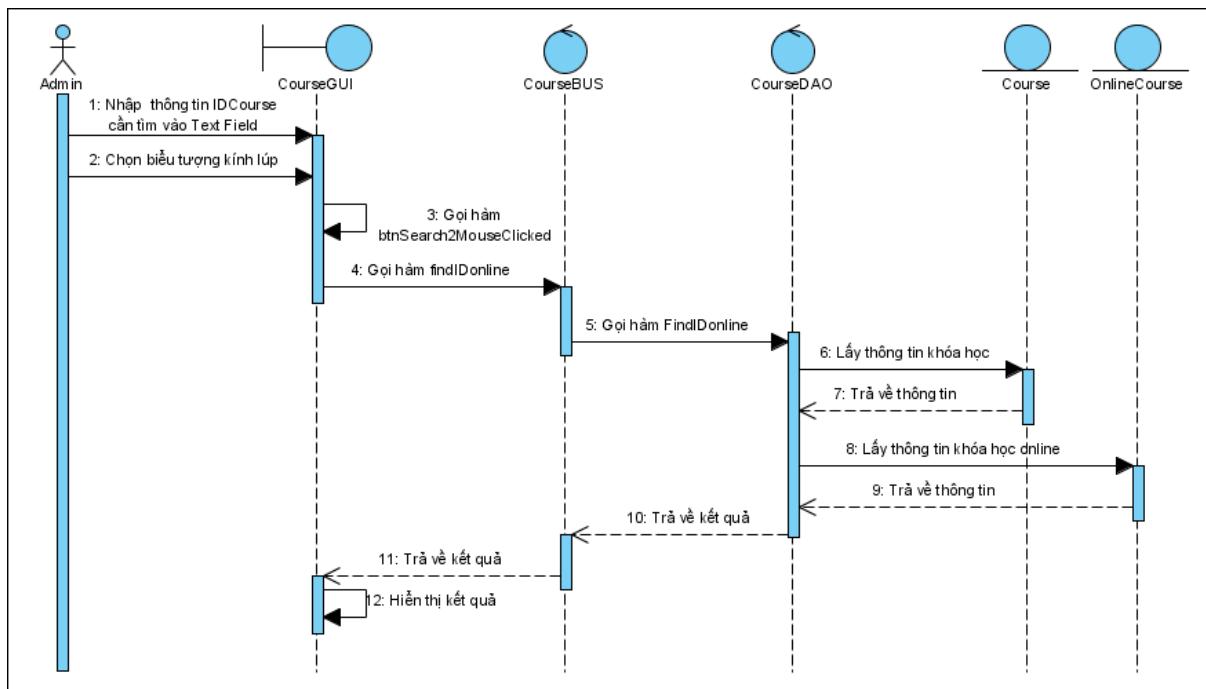
```

private void btnDeleteonlinecourseMouseClicked(java.awt.event.MouseEvent evt) {
    int i = tblecourseonline.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn course cần xoá");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn xoá course không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            int idcourse = Integer.parseInt(tblecourseonline.getValueAt(i, 0).toString());
            try {
                if (cBUS.deleteonlinecourse(idcourse)) {
                    JOptionPane.showMessageDialog(null, "Xóa thành công");
                    txtcoursetitle2.setText("");
                    txtcredits2.setText("");
                    txturl.setText("");
                    readTableOnlineCourse();
                    cbCourseTitle.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
                    cbCourseTitleCouIns.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
                } else {
                    JOptionPane.showMessageDialog(null, "Xóa thất bại");
                }
            } catch (SQLException ex) {
                Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}

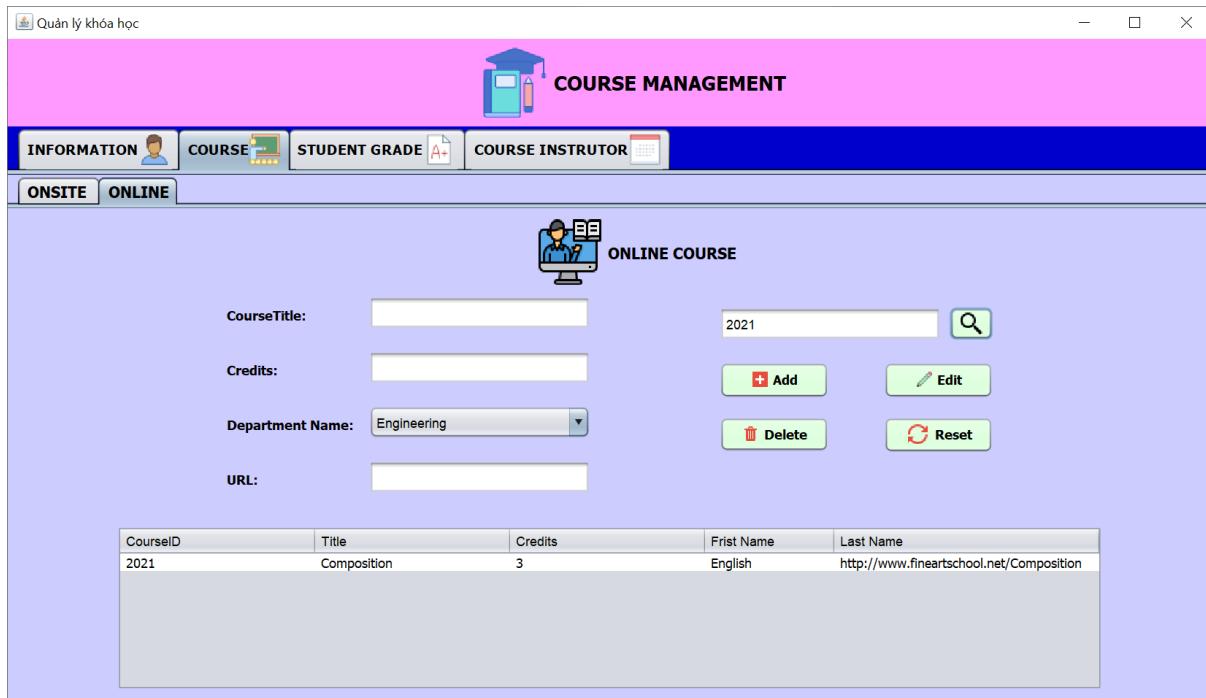
```

## 5. Xử lý 5: Tìm kiếm thông tin khóa học online theo mã khóa học

### 5.1 Sơ đồ tuần tự



## 5.2 Giao diện



Giao diện hiển thị kết quả tìm kiếm

## 5.3 Code 3 class:

❖ DAO

```

public ArrayList<CourseDTO> FindIDonline(String courseID) throws SQLException {
    ArrayList<CourseDTO> searchList = new ArrayList<>();
    int courseID2 = Integer.parseInt(courseID);
    String sql = String.format("select * from course left JOIN onlinecourse on onlinecourse.CourseID = course.CourseID "
        + "WHERE course.CourseID = '%d'", courseID2);
    ResultSet rs = ConnectDatabase.GetInstance().ExcutesSELECT(sql);
    while (rs.next()) {
        CourseDTO cDTO = new CourseDTO();
        cDTO.setCourseID(rs.getInt("CourseID"));
        cDTO.setTitle(rs.getString("Title"));
        cDTO.setCredits(rs.getInt("Credits"));
        String sql2 = "select * from Department where DepartmentID = " + rs.getInt("DepartmentID") + "'";
        ResultSet rs2 = ConnectDatabase.GetInstance().ExcuteSELECT(sql2);
        while (rs2.next()) {
            cDTO.setDepartmentName(rs2.getString("Name"));
        }
        cDTO.setURL(rs.getString("URL"));
        searchList.add(cDTO);
    }
    return searchList;
}

```

## ❖ BUS

```

public ArrayList<CourseDTO> findIDonline(String ID) throws SQLException {
    return cDAO.FindIDonline(ID);
}

```

## ❖ GUI

```

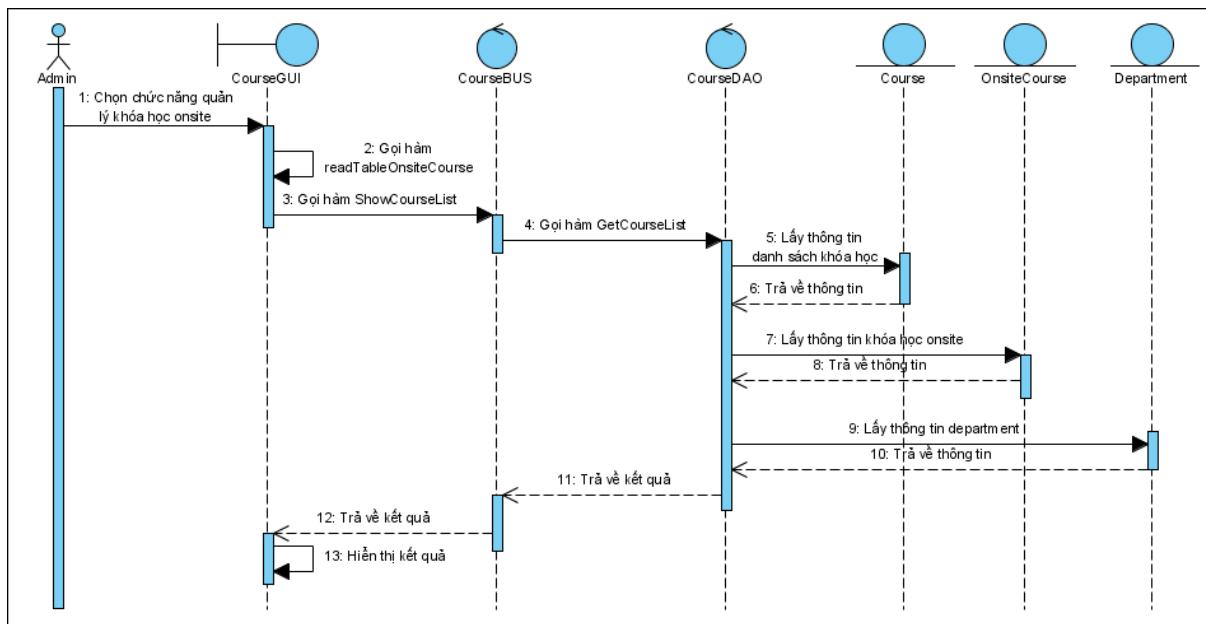
private void btnSearch2MouseClicked(java.awt.event.MouseEvent evt) {
    String inputsearch = txtSearchononlinecourse.getText();
    ArrayList<CourseDTO> rssearch = new ArrayList<>();
    try {
        rssearch = cBUS.findIDonline(inputsearch);
        modelTableCourseOnline.setRowCount(0);
        for (CourseDTO cdto : rssearch) {
            if (cdto.getURL() != null) {
                modelTableCourseOnline.addRow(new Object[]{
                    cdto.getCourseID(), cdto.getTitle(), cdto.getCredits(), cdto.getDepartmentName(), cdto.getURL()
                });
            }
        }
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
}

```

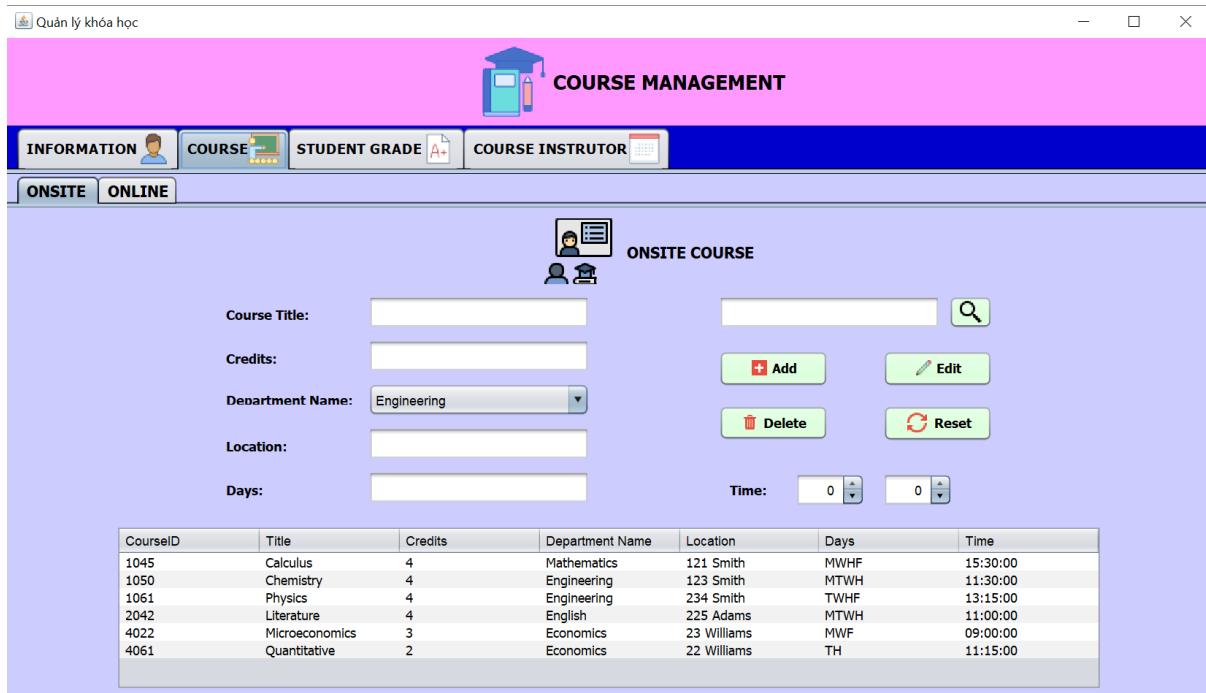
## III. Quản lý thông tin khóa học onsite

### 1. Xử lý 1: Hiển thị danh sách thông tin khóa học onsite

#### 1.1 Sơ đồ tuần tự



## 1.2 Giao diện



Giao diện quản lý khóa học onsite

## 1.3 Code 3 class:

❖ DAO

```

public ArrayList<CourseDTO> GetCourseList() throws SQLException {
    ArrayList<CourseDTO> courseList = new ArrayList<>();
    String sql = "select * from course "
        + "left JOIN onlinecourse on onlinecourse.CourseID = course.CourseID "
        + "LEFT JOIN onsitecourse on course.CourseID = onsitecourse.CourseID";
    ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
    while (rs.next()) {
        String sql2 = "select * from department where DepartmentID =" + rs.getInt("DepartmentID");
        ResultSet rs2 = ConnectDatabase.GetInstance().ExecuteSELECT(sql2);
        CourseDTO cDTO = new CourseDTO();
        cDTO.setCourseID(rs.getInt("course.CourseID"));
        cDTO.setTitle(rs.getString("course.Title"));
        cDTO.setCredits(rs.getInt("course.Credits"));
        cDTO.setDepartmentID(rs.getInt("DepartmentID"));
        while (rs2.next()) {
            cDTO.setDepartmentName(rs2.getString("Name"));
        }
        cDTO.setURL(rs.getString("url"));
        cDTO.setLocation(rs.getString("Location"));
        cDTO.setDays(rs.getString("Days"));
        cDTO.setTime(rs.getTime("Time"));
        courseList.add(cDTO);
    }
    return courseList;
}

```

## ❖ BUS

```

public void ShowCourseList() throws SQLException {
    CourseList = cDAO.GetCourseList();
}

```

## ❖ GUI

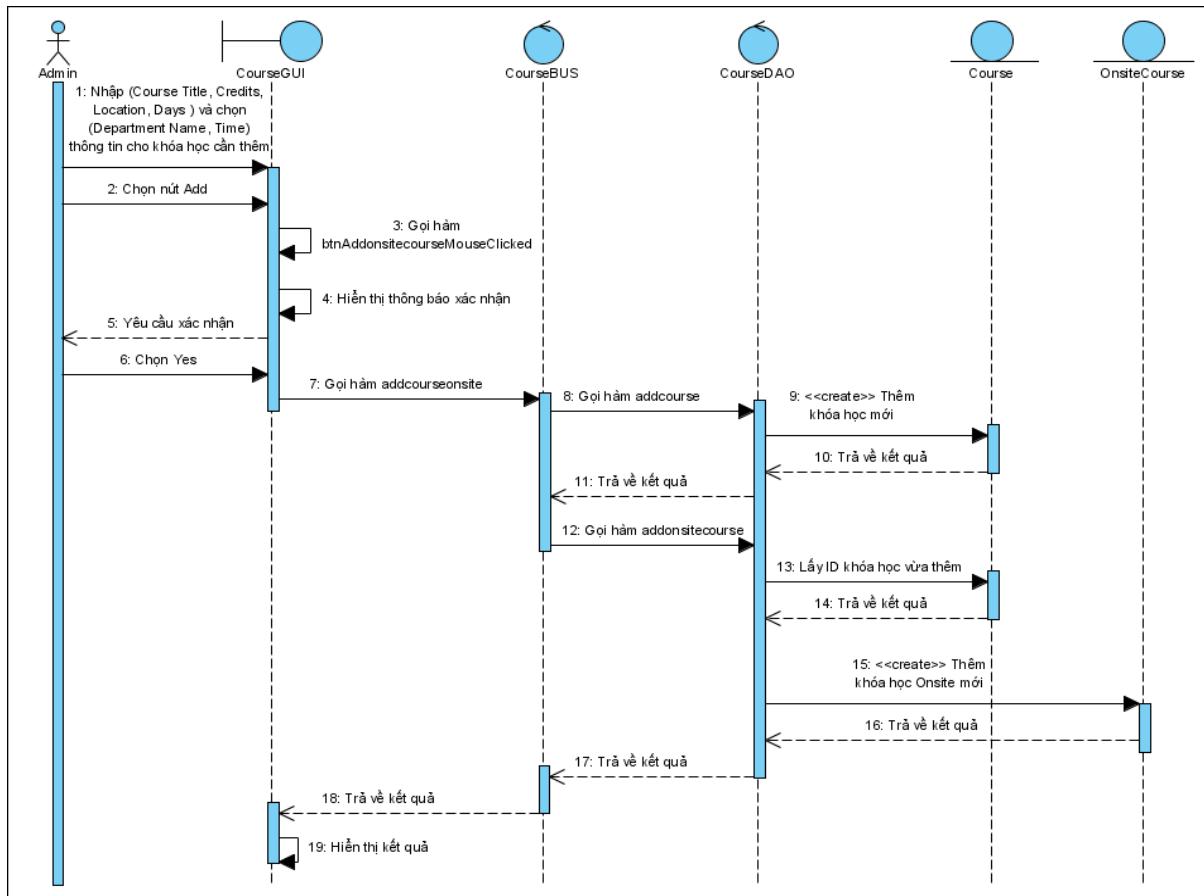
```

private void readTableOnsiteCourse() {
    try {
        cBUS.ShowCourseList();
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
    modelTableCourseOnsite.setRowCount(0);
    for (CourseDTO cdto : CourseBUS.CourseList) {
        if (cdto.getLocation() != null) {
            modelTableCourseOnsite.addRow(new Object[]{
                cdto.getCourseID(), cdto.getTitle(),
                cdto.getCredits(), cdto.getDepartmentName(),
                cdto.getLocation(), cdto.getDays(), cdto.getTime()
            });
        }
    }
}

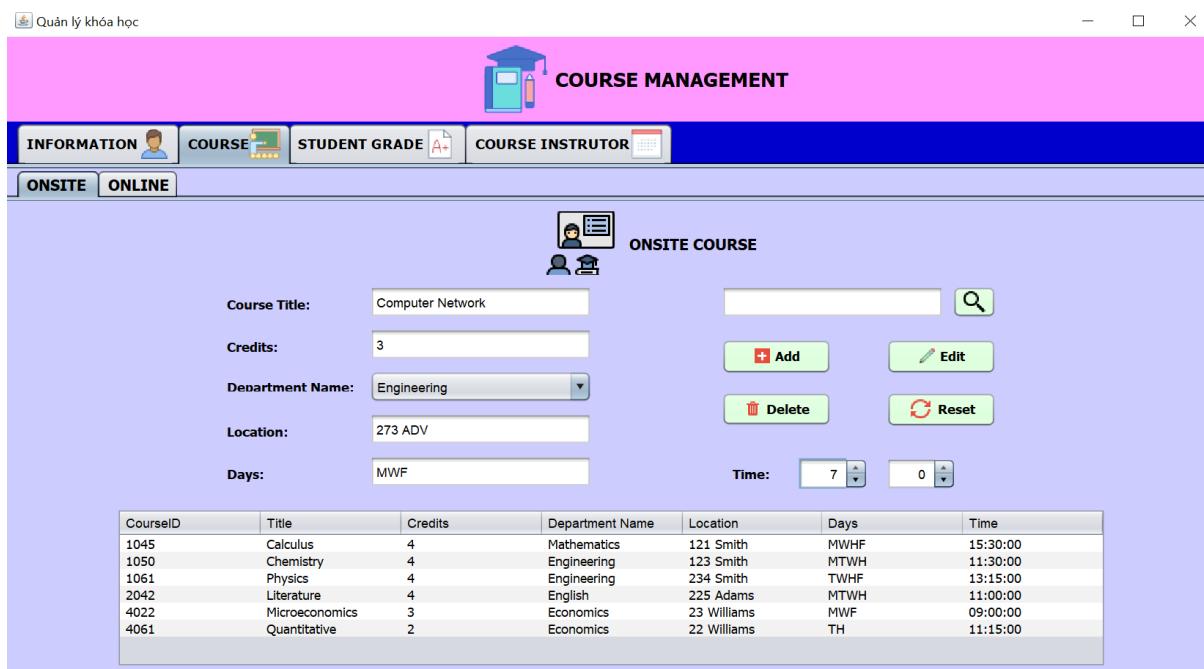
```

## 2. Xử lý 2: Thêm thông tin khóa học onsite

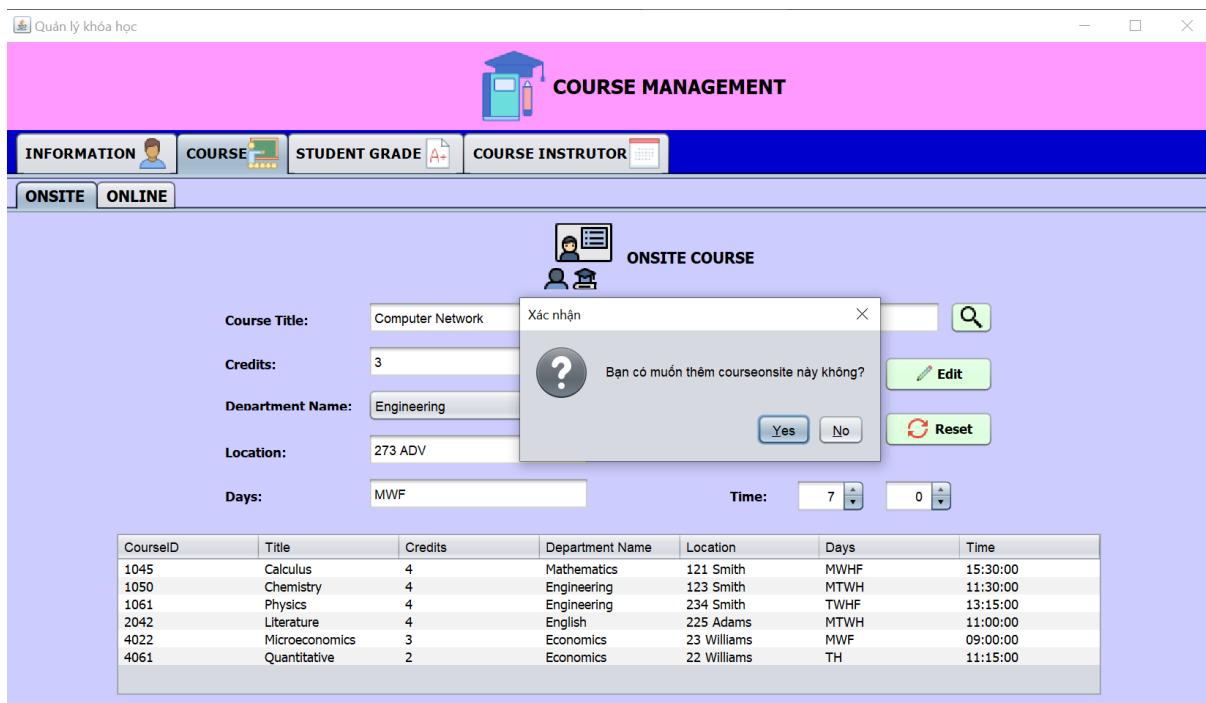
### 2.1 Sơ đồ tuần tự



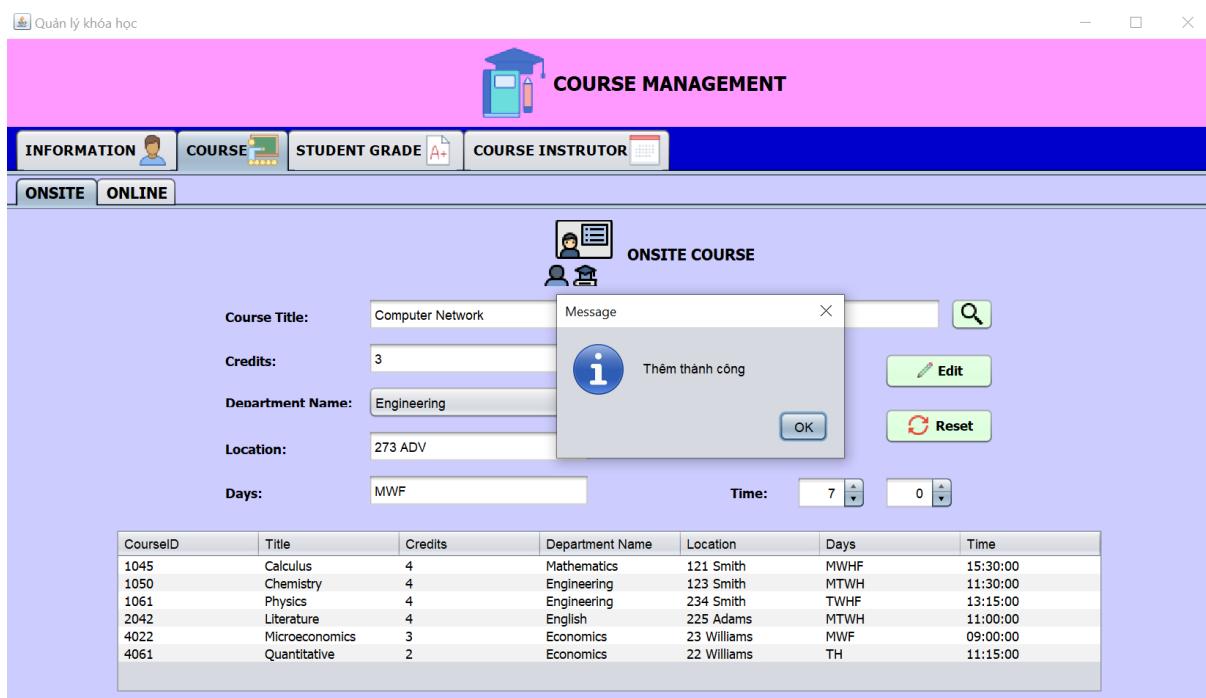
### 2.2 Giao diện



Giao diện nhập thông tin khóa học onsite mới



Giao diện thông báo xác nhận



Giao diện thông báo thêm thành công

Quản lý khóa học

**COURSE MANAGEMENT**

INFORMATION COURSE STUDENT GRADE COURSE INSTRUCTOR

ONSITE ONLINE

**ONSITE COURSE**

CourseID	Title	Credits	Department Name	Location	Days	Time
1045	Calculus	4	Mathematics	121 Smith	MWHF	15:30:00
1050	Chemistry	4	Engineering	123 Smith	MTWH	11:30:00
1061	Physics	4	Engineering	234 Smith	TWHF	13:15:00
2042	Literature	4	English	225 Adams	MTWH	11:00:00
4022	Microeconomics	3	Economics	23 Williams	MWF	09:00:00
4061	Quantitative	2	Economics	22 Williams	TH	11:15:00
4070	Computer Network	3	Engineering	273 ADV	MWF	07:00:00

Giao diện hiển thị thông tin khóa học đã được thêm

### 2.3 Code 3 class:

#### ❖ DAO

```
public boolean addcourse(CourseDTO c) {
    String sql = null;
    if (c != null) {
        sql = String.format("INSERT INTO `course`(`Title`, `Credits`, `DepartmentID`) VALUES ('%s', '%d', '%d')",
            c.getTitle(), c.getCredits(), c.getDepartmentID());
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}
```

```

public boolean addonsitecourse(CourseDTO c) throws SQLException{
    String sql = null;
    String sql2 = String.format("SELECT * FROM `course` WHERE Title = '%s'", c.getTitle());
    ResultSet rs2 = ConnectDatabase.GetInstance().ExcuteSELECT(sql2);
    int temp = 0;
    while(rs2.next()){
        temp = rs2.getInt("CourseID");
    }
    if(c != null){
        sql = String.format("INSERT INTO `onsitecourse`(`CourseID`, `Location`, `Days`, `Time`) "
            + "VALUES ('%d','%s','%s','%tT')",
            temp,c.getLocation(),c.getDays(),c.getTime());
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    }else
        return false;
}

```

## ❖ BUS

```

public boolean addcourseonsite(CourseDTO c) throws SQLException {
    cDAO.addcourse(c);
    return cDAO.addonsitecourse(c);
}

```

## ❖ GUI

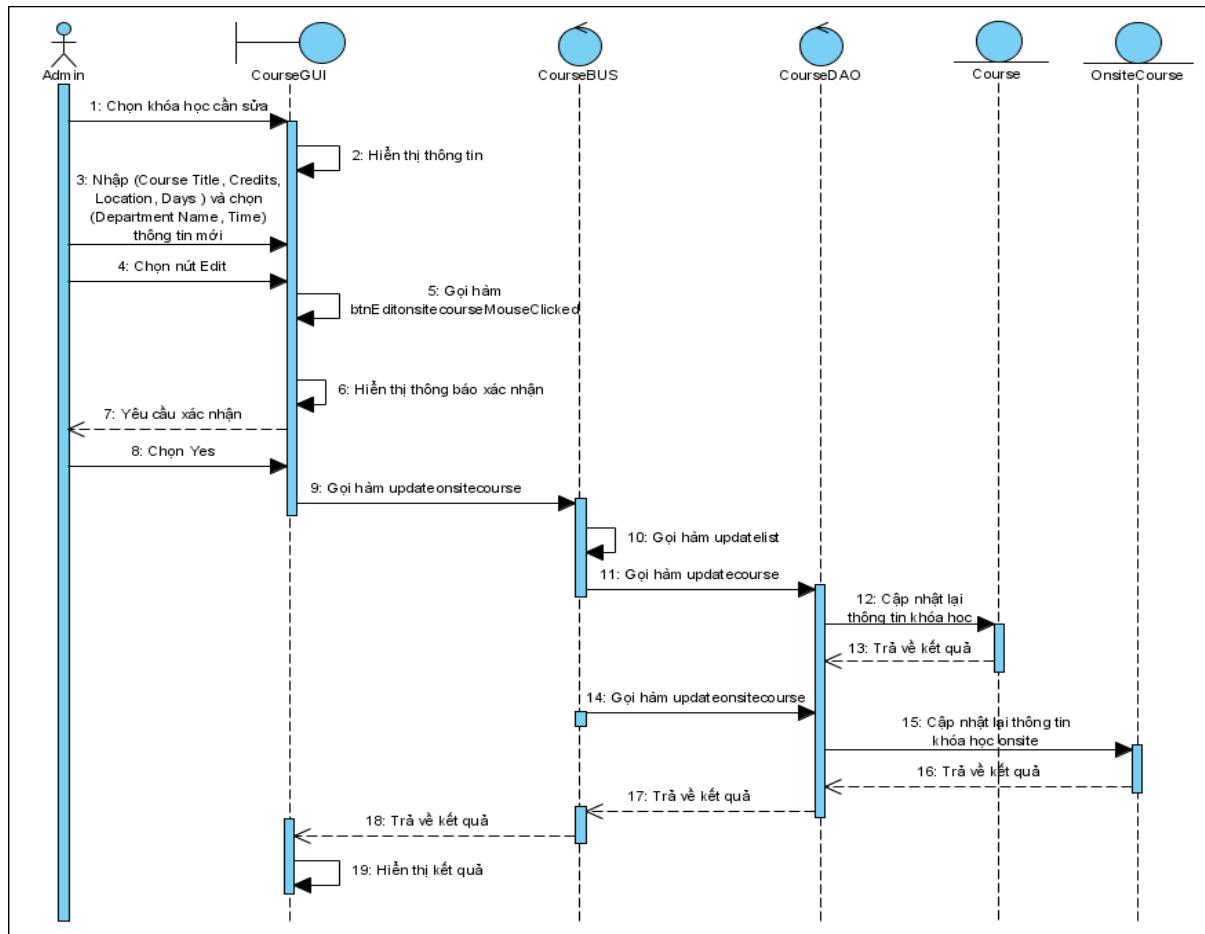
```

private void btnAddonsitecourseMouseClicked(java.awt.event.MouseEvent evt) {
    int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn thêm courseonsite này không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
    if (result == JOptionPane.NO_OPTION) {return; }
    CourseDTO cDTO = new CourseDTO();
    int temp = 0;
    for (DepartmentDTO dpDTO : DepartmentBUS.dpList) {
        if (dpDTO.getName().equals(jdepartmentname.getSelectedItem())) {
            temp = dpDTO.getDepartmentID();
            break;
        }
    }
    String timecourse = jspinnerHour.getValue().toString() + ":" + jspinnerMinutes.getValue().toString() + ":" + "00";
    DateFormat dateFormat = new SimpleDateFormat("HH:mm:ss");
    Date temp2 = null;
    try { temp2 = dateFormat.parse(timecourse);
    } catch (ParseException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
    cDTO.setTitle(txtcoursetitle.getText().toString());
    cDTO.setCredits(Integer.parseInt(txtcredits.getText()));
    cDTO.setDepartmentID(temp);
    cDTO.setLocation(txtlocation.getText());
    cDTO.setDays(txtdays.getText());
    cDTO.setTime(temp2);
    try {
        if (CBUS.addcourseonsite(cDTO)) {
            JOptionPane.showMessageDialog(null, "Thêm thành công");
            readTableOnsiteCourse();
            cbCourseTitle.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
            cbCourseTitleCouIns.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
        } else {
            JOptionPane.showMessageDialog(null, "Thêm thất bại");
        }
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
}

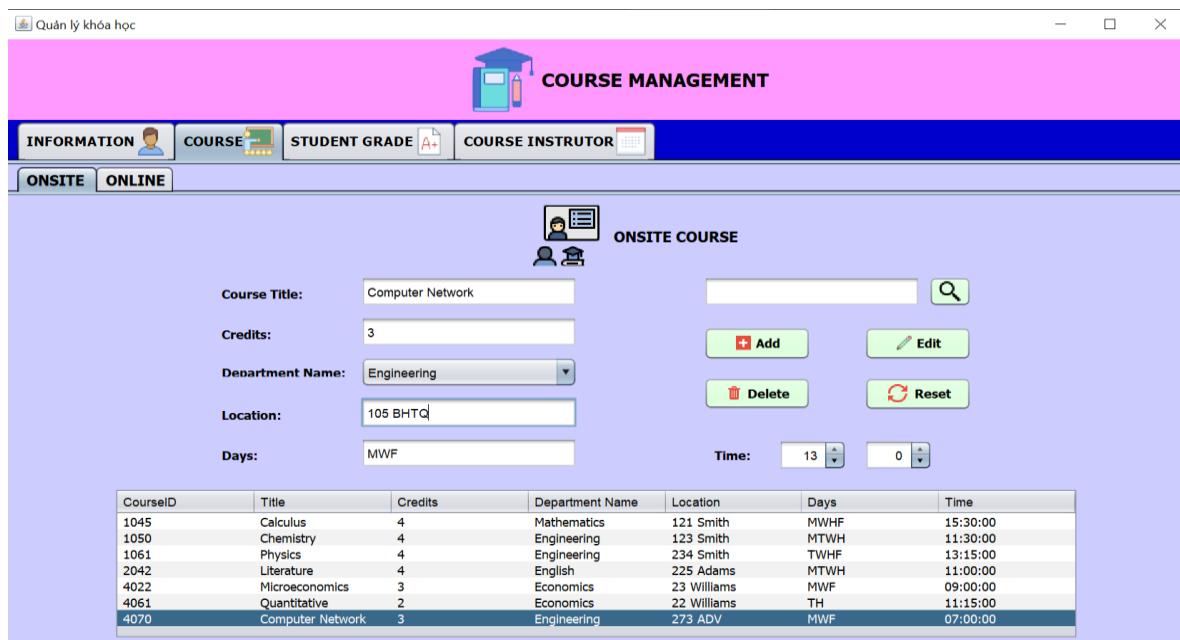
```

### 3. Xử lý 3: Sửa thông tin khóa học onsite

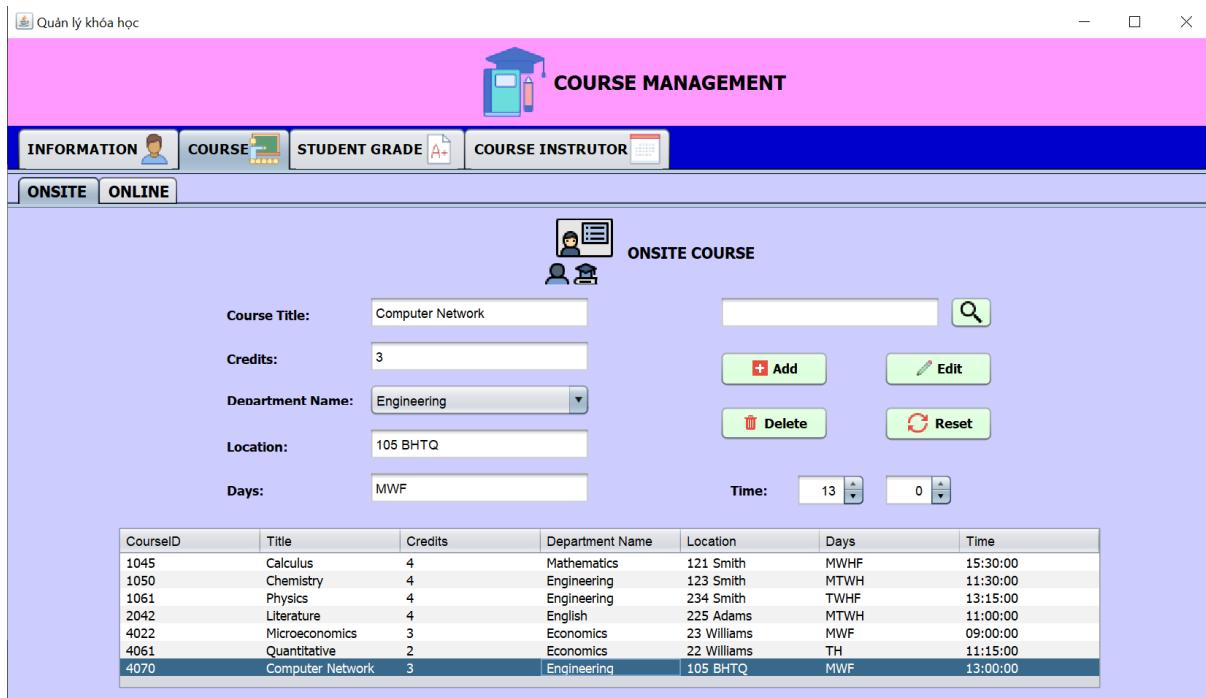
#### 4.1 Sơ đồ tuần tự



#### 4.2 Giao diện



Giao diện nhập thông tin cập nhật Location: 105 BHTQ và Time: 13



*Giao diện thông tin khóa học đã được cập nhật*

### 4.3 Code 3 class:

#### ❖ DAO

```
public boolean updateonsitecourse(CourseDTO c) {
    String sql = null;
    if (c != null) {
        sql = String.format("UPDATE `onsitecourse` SET `Location`='%s', `Days`='%s', `Time`='%tT' "
            + "WHERE CourseID = '%d'", c.getLocation(), c.getDays(), c.getTime(), c.getCourseID());
    }
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}
```

#### ❖ BUS

```
public boolean updateonsitecourse(CourseDTO c) {
    updatelist(c);
    cDAO.updatecourse(c);
    return cDAO.updateonsitecourse(c);
}
```

```

private void updateList(CourseDTO c) {
    for (int i = 0; i < CourseList.size(); i++) {
        if (c.getCourseID() == CourseList.get(i).getCourseID()) {
            CourseList.set(i, c);
        }
    }
}

```

## ❖ GUI

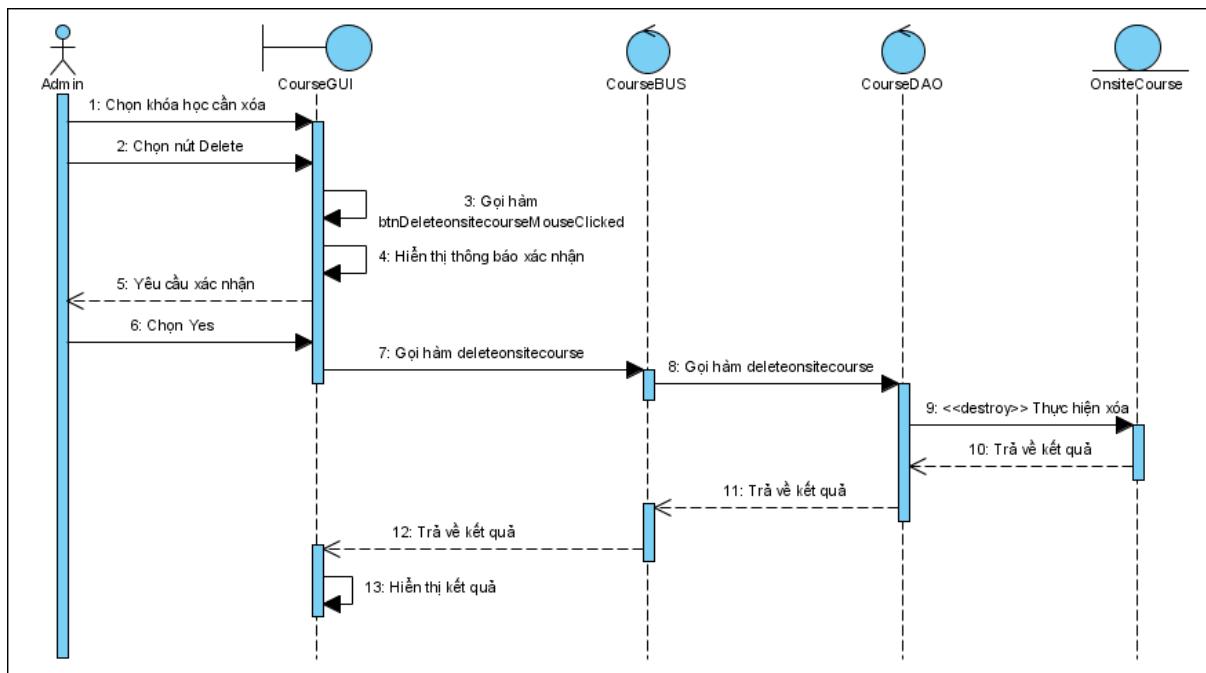
```

private void btnEditOnsiteCourseMouseClicked(java.awt.event.MouseEvent evt) {
    int i = tblcourseonsite.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn course cần cập nhật");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn cập nhật course không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            CourseDTO cDTO = new CourseDTO();
            int temp = 0;
            for (DepartmentDTO dpDTO : DepartmentBUS.dpList) {
                if (dpDTO.getName().equals(jdepartmentname.getSelectedItem())) {
                    temp = dpDTO.getDepartmentID();
                    break;
                }
            }
            String timecourse = jSpinnerHour.getValue().toString() + ":" + jSpinnerMinutes.getValue().toString() + ":" + "00";
            DateFormat dateFormat = new SimpleDateFormat("HH:mm:ss");
            Date temp2 = null;
            try {
                temp2 = dateFormat.parse(timecourse);
            } catch (ParseException ex) {
                Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
            }
            cDTO.setCourseID(Integer.parseInt(tblcourseonsite.getValueAt(i, 0).toString()));
            cDTO.setTitle(txtcourseTitle.getText());
            cDTO.setCredits(Integer.parseInt(txtcredits.getText()));
            cDTO.setDepartmentID(temp);
            cDTO.setLocation(txtlocation.getText());
            cDTO.setDays(txtdays.getText());
            cDTO.setTime(temp2);
            if (CBUS.updateonsitecourse(cDTO)) {
                JOptionPane.showMessageDialog(null, "Cập nhật thành công");
                txtcourseTitle.setText("");
                txtcredits.setText("");
                txtlocation.setText("");
                txtdays.setText("");
                jSpinnerHour.setValue(0);
                jSpinnerMinutes.setValue(0);
                readTableOnsiteCourse();
                cbCourseTitle.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
                cbCourseTitleCouIns.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
            } else {
                JOptionPane.showMessageDialog(null, "Cập nhật thất bại");
            }
        }
    }
}

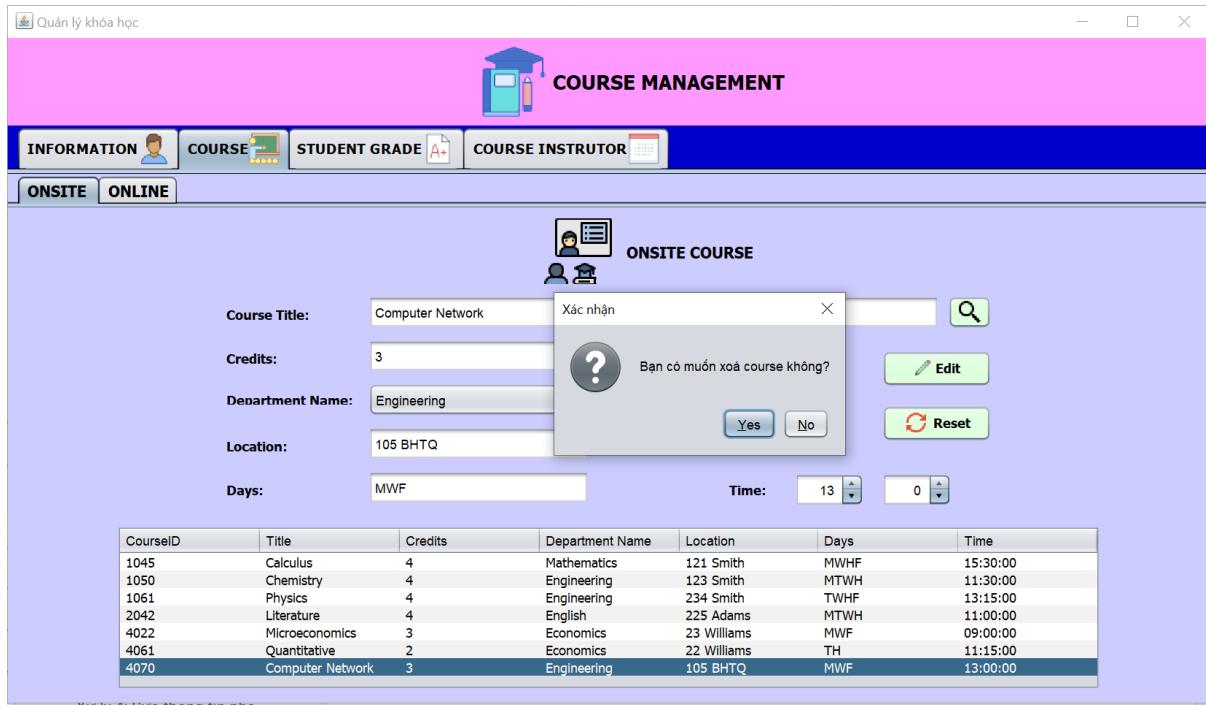
```

## 4. Xử lý 4: Xóa thông tin khóa học onsite

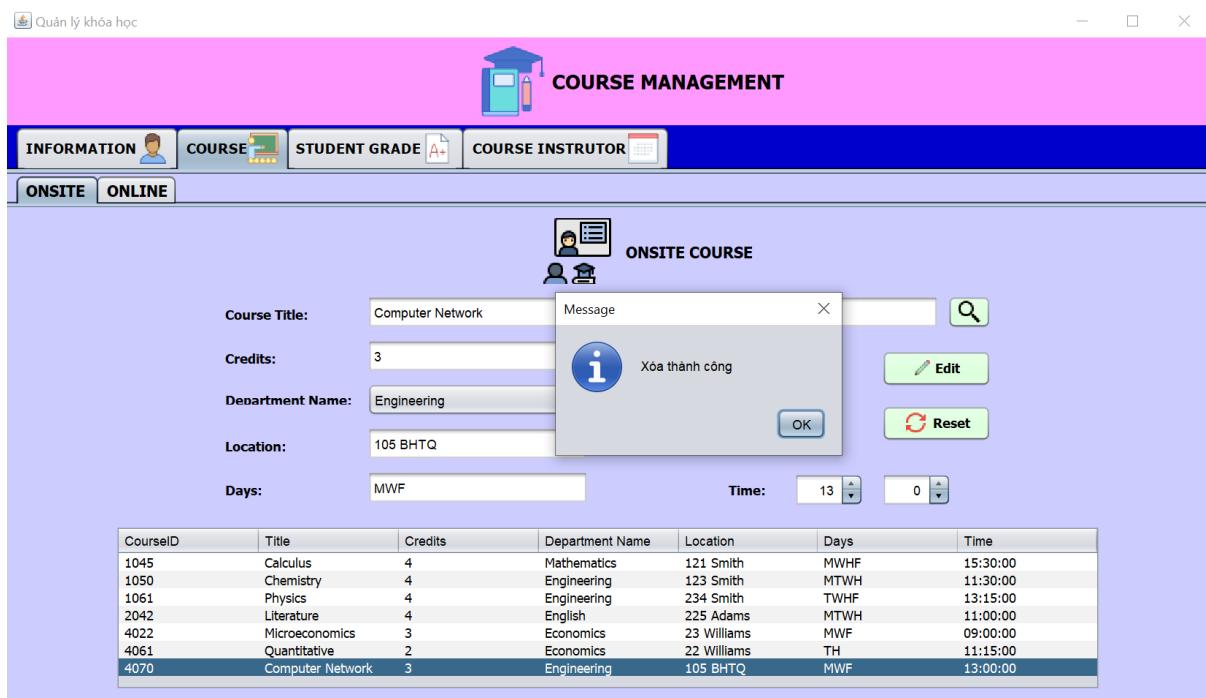
### 3.1 Sơ đồ tuần tự



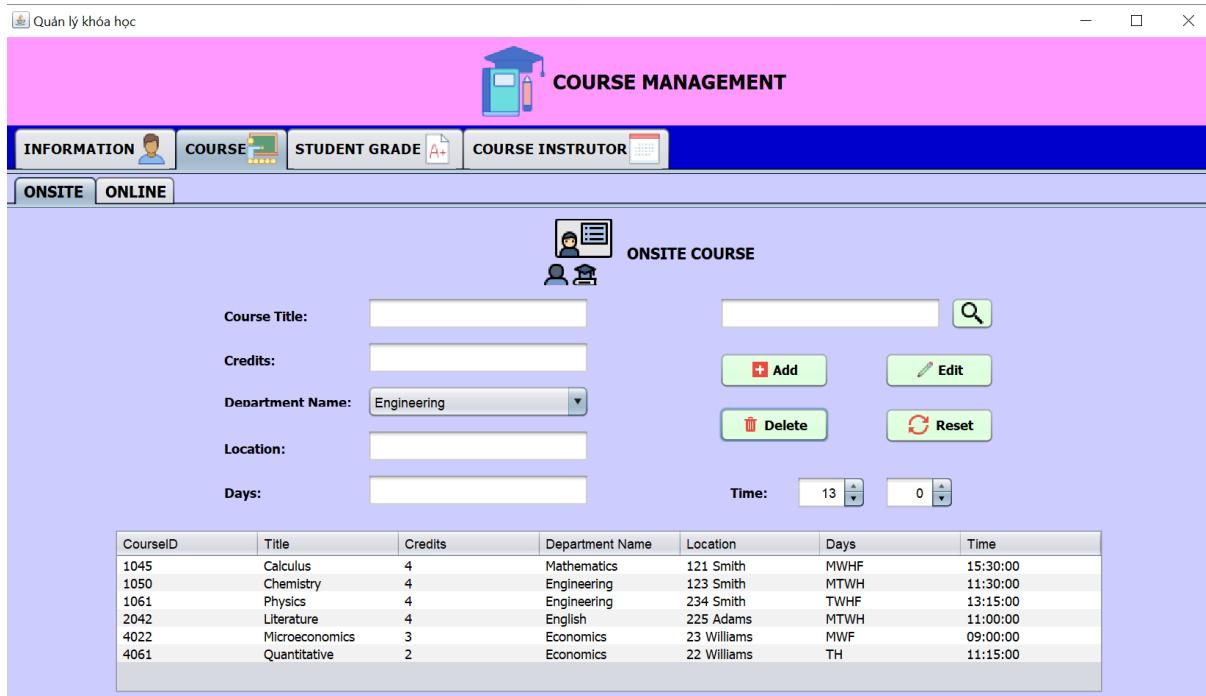
### 3.2 Giao diện



Giao diện thông báo xác nhận xóa



Giao diện thông báo xóa thành công



Giao diện hiển thị danh sách khóa học sau khi xóa

### 3.3 Code 3 class:

- ❖ DAO

```

public boolean deleteonsitecourse(int id){
    String sql = String.format("DELETE FROM `onsitecourse` WHERE CourseID=%d", id);
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    }else
        return false;
}

```

## ❖ BUS

```

public boolean deleteonsitecourse(int id) throws SQLException {
    ...
    return cDAO.deleteonsitecourse(id);
}

```

## ❖ GUI

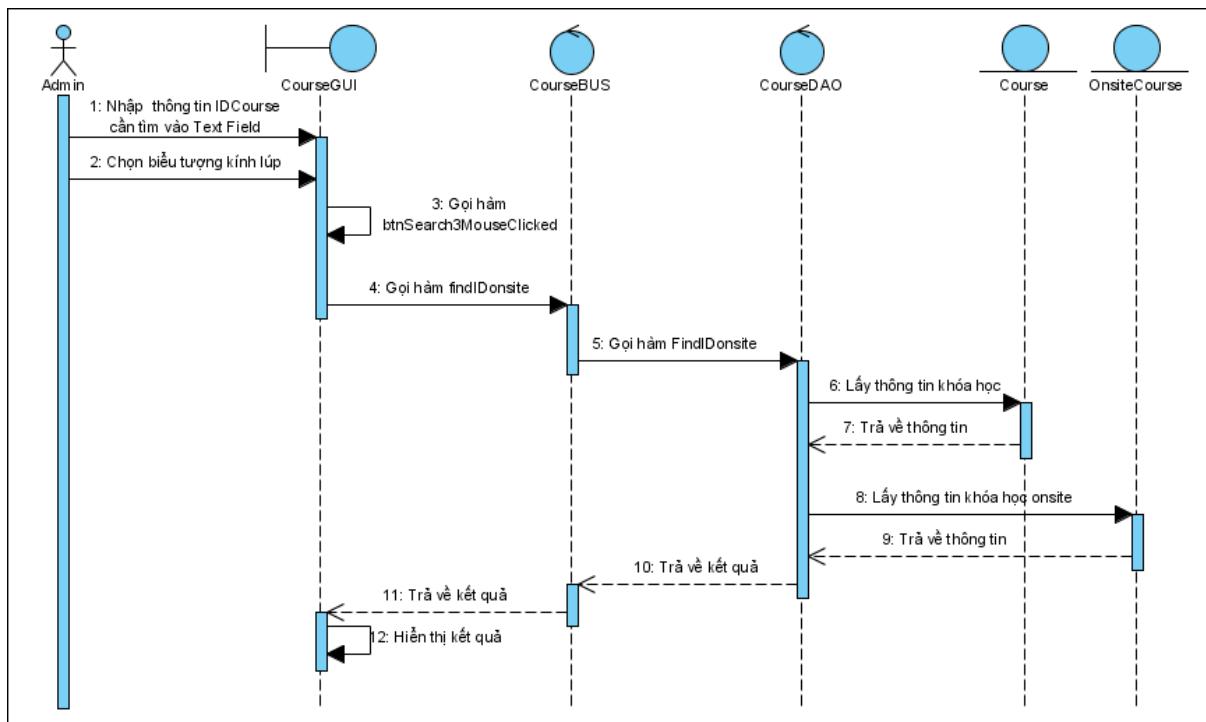
```

private void btnDeleteonsitecourseMouseClicked(java.awt.event.MouseEvent evt) {
    int i = tblcourseonsite.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn course cần xoá");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn xoá course không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            int idcourse = Integer.parseInt(tblcourseonsite.getValueAt(i, 0).toString());
            try {
                if (cBUS.deleteonsitecourse(idcourse)) {
                    JOptionPane.showMessageDialog(null, "Xóa thành công");
                    txtcoursetitle.setText("");
                    txtcredits.setText("");
                    txtlocation.setText("");
                    txtdays.setText("");
                    readTableOnsiteCourse();
                    cbCourseTitle.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
                    cbCourseTitleCouIns.setModel(new DefaultComboBoxModel(ComboBoxCourseTitle()));
                } else {
                    JOptionPane.showMessageDialog(null, "Xóa thất bại");
                }
            } catch (SQLException ex) {
                Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}

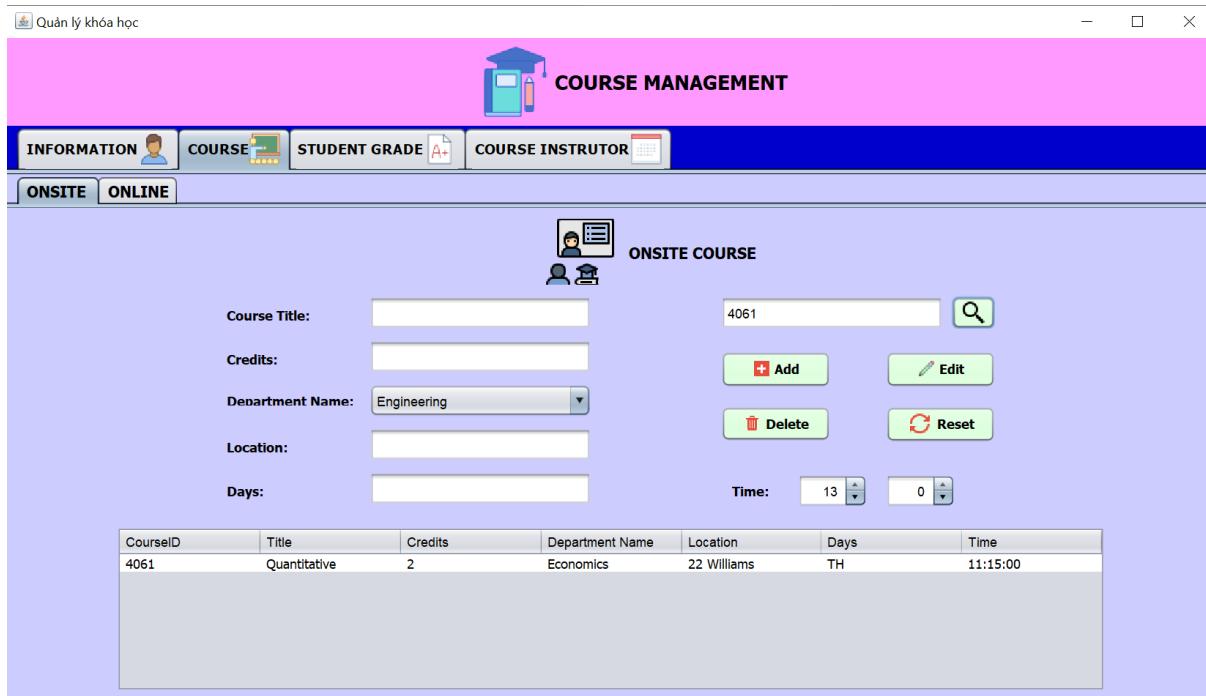
```

## 5. Xử lý 5: Tìm kiếm thông tin khóa học onsite theo mã khóa học

### 5.1 Sơ đồ tuần tự



## 5.2 Giao diện



*Giao diện kết quả tìm kiếm*

## 5.3 Code 3 class:

❖ DAO

```

public ArrayList<CourseDTO> FindIDonsite(String courseID) throws SQLException {
    ArrayList<CourseDTO> searchList = new ArrayList<>();
    int courseID2 = Integer.parseInt(courseID);
    String sql = String.format("select * from course left JOIN onsitecourse on onsitecourse.CourseID = course.CourseID "
        + "WHERE course.CourseID = '%d'", courseID2);
    ResultSet rs = ConnectDatabase.GetInstance().ExcuteSELECT(sql);
    while (rs.next()) {
        CourseDTO cDTO = new CourseDTO();
        cDTO.setCourseID(rs.getInt("CourseID"));
        cDTO.setTitle(rs.getString("Title"));
        cDTO.setCredits(rs.getInt("Credits"));
        String sql2 = "select * from Department where DepartmentID = '" + rs.getInt("DepartmentID") + "'";
        ResultSet rs2 = ConnectDatabase.GetInstance().ExcuteSELECT(sql2);
        while (rs2.next()) {
            cDTO.setDepartmentName(rs2.getString("Name"));
        }
        cDTO.setLocation(rs.getString("Location"));
        cDTO.setDays(rs.getString("Days"));
        cDTO.setTime(rs.getTime("Time"));
        searchList.add(cDTO);
    }
    return searchList;
}

```

## ❖ BUS

```

public ArrayList<CourseDTO> findIDonsite(String ID) throws SQLException {
    return cDAO.FindIDonsite(ID);
}

```

## ❖ GUI

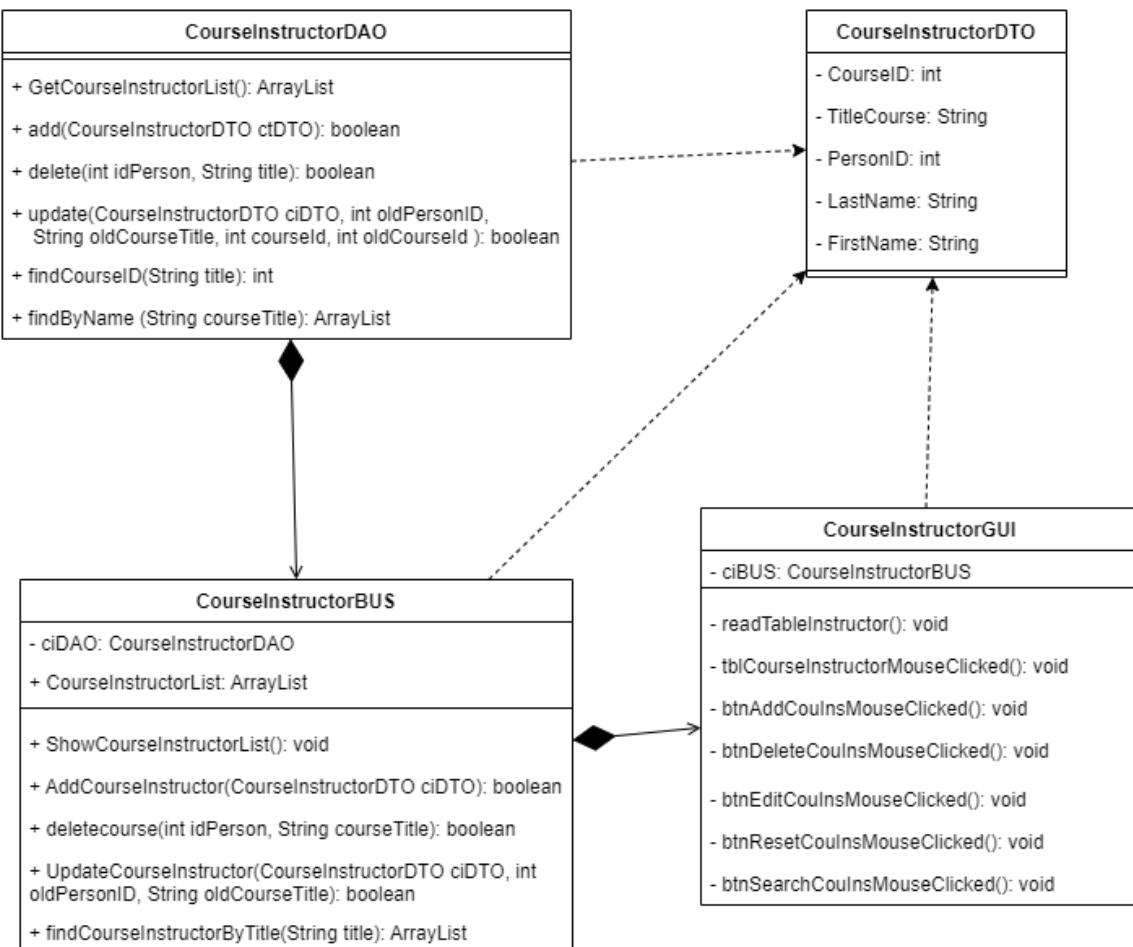
```

private void btnSearch3MouseClicked(java.awt.event.MouseEvent evt) {
    String inputsearch = txtSearchonsitecourse.getText();
    ArrayList<CourseDTO> rssearch = new ArrayList<>();
    try {
        rssearch = cBUS.findIDonsite(inputsearch);
        modelTableCourseOnsite.setRowCount(0);
        for (CourseDTO cdto : rssearch) {
            if (cdto.getLocation() != null) {
                modelTableCourseOnsite.addRow(new Object[]{
                    cdto.getCourseID(), cdto.getTitle(),
                    cdto.getCredits(), cdto.getDepartmentName(),
                    cdto.getLocation(), cdto.getDays(), cdto.getTime()
                });
            }
        }
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
}

```

## Phần 3: Chức năng quản lý thông tin phân công giảng dạy

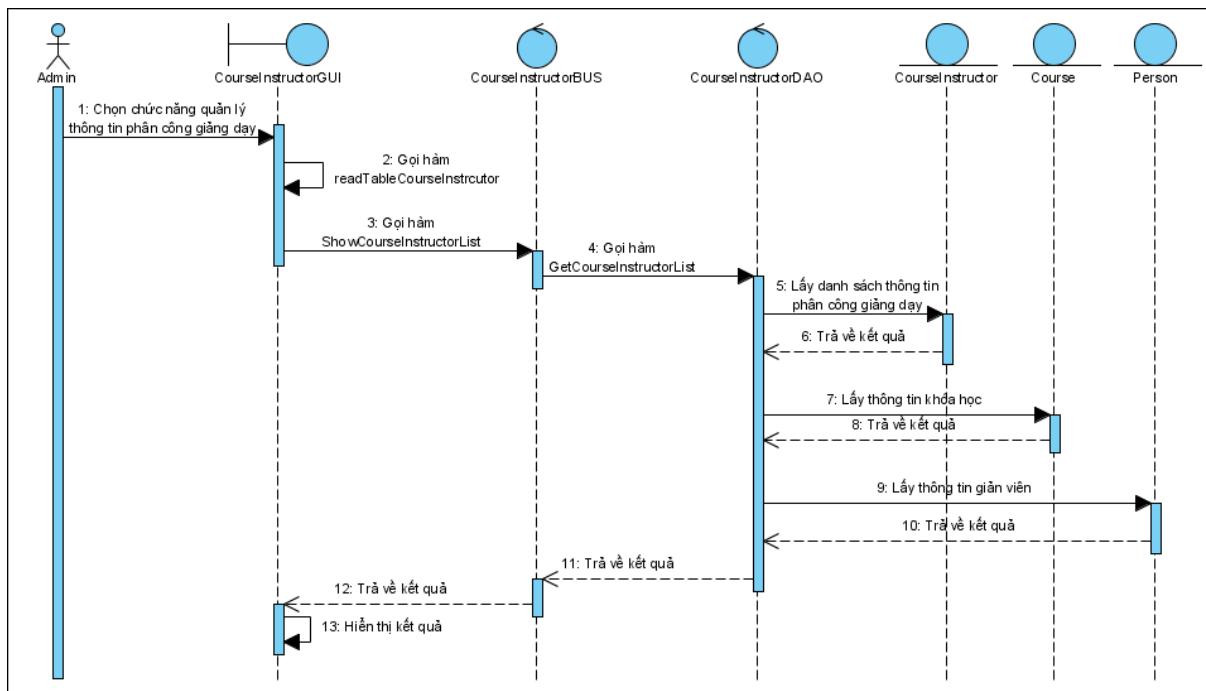
### I. Sơ đồ class chung



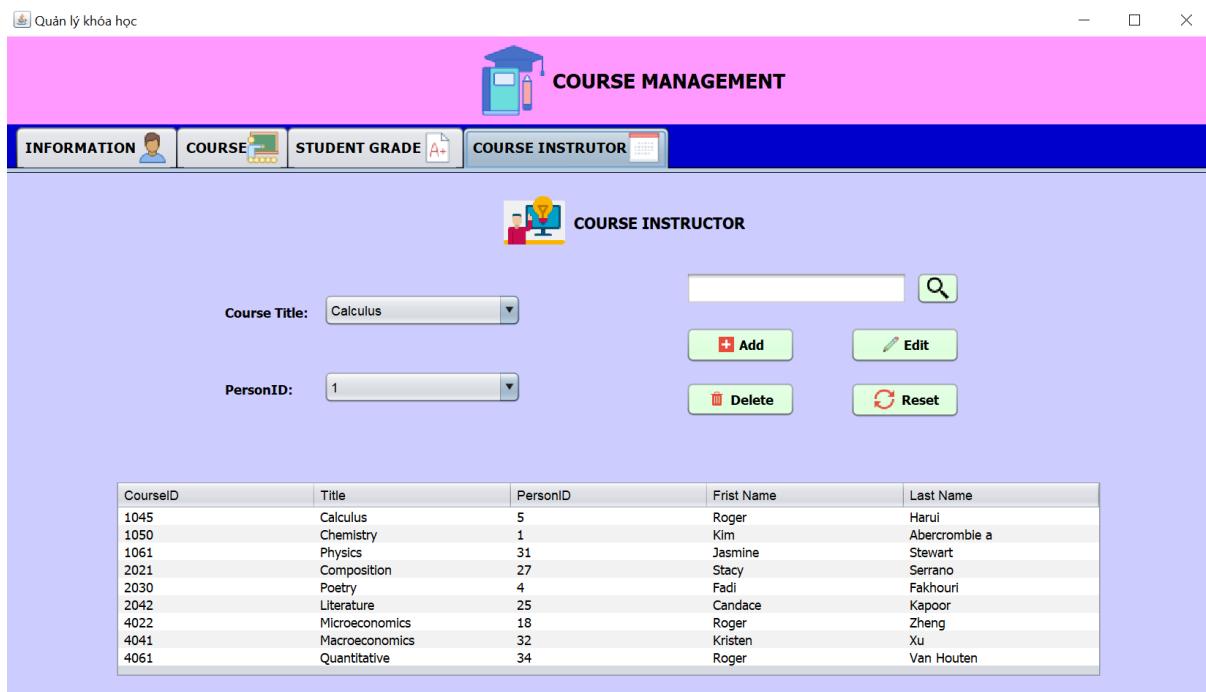
### II. Quản lý thông tin phân công giảng dạy

#### 1. Xử lý 1: Hiển thị danh sách thông tin phân công giảng dạy

##### 1.1 Sơ đồ tuần tự



## 1.2 Giao diện



Giao diện quản lý phân công giảng dạy

## 1.3 Code 3 class:

❖ DAO

```

public ArrayList<CourseInstructorDTO> GetCourseInstructorList() throws SQLException {
    ArrayList<CourseInstructorDTO> courseInstructorList = new ArrayList<>();
    String sql = "select * from courseinstructor left JOIN course on courseinstructor.CourseID = course.CourseID"
        + " left join person on person.PersonID = courseinstructor.PersonID";
    ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
    while (rs.next()) {
        CourseInstructorDTO ciDTO = new CourseInstructorDTO();
        ciDTO.setPersonID(rs.getInt("person.PersonID"));
        ciDTO.setFirstName(rs.getString("person.Firstname"));
        ciDTO.setLastName(rs.getString("person.Lastname"));
        ciDTO.setCourseID(rs.getInt("courseinstructor.CourseID"));
        ciDTO.setTitleCourse(rs.getString("course.Title"));
        courseInstructorList.add(ciDTO);
    }
    return courseInstructorList;
}

```

## ❖ BUS

```

public void ShowCourseInstructorList() throws SQLException {
    CourseInstructorList = ciDAO.GetCourseInstructorList();
}

```

## ❖ GUI

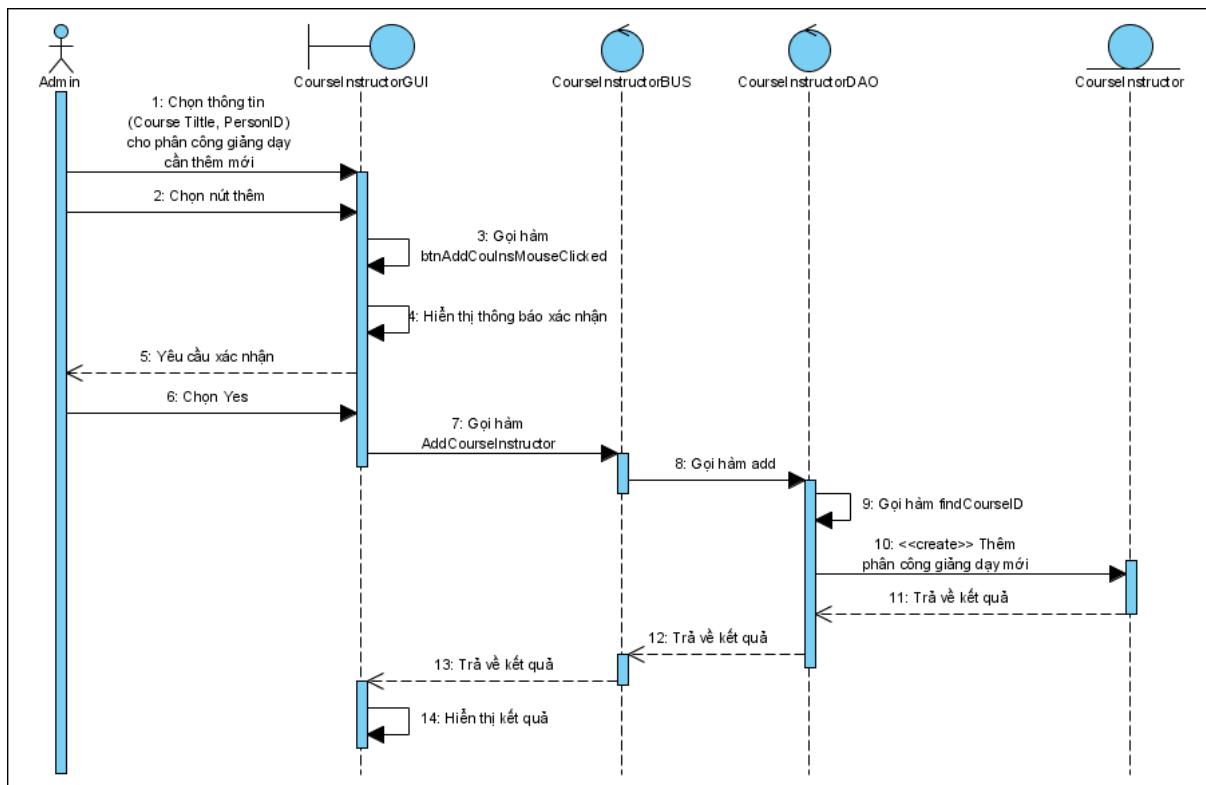
```

private void readTableCourseInstrcutor() {
    try {
        ciBUS.ShowCourseInstructorList();
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
    modelTableCourseInstructor.setRowCount(0);
    for (CourseInstructorDTO ctDTO : CourseInstructorBUS.CourseInstructorList) {
        modelTableCourseInstructor.addRow(new Object[]{
            ctDTO.getCourseID(), ctDTO.getTitleCourse(),
            ctDTO.getPersonID(), ctDTO.getFirstName(), ctDTO.getLastName()
        });
    }
}

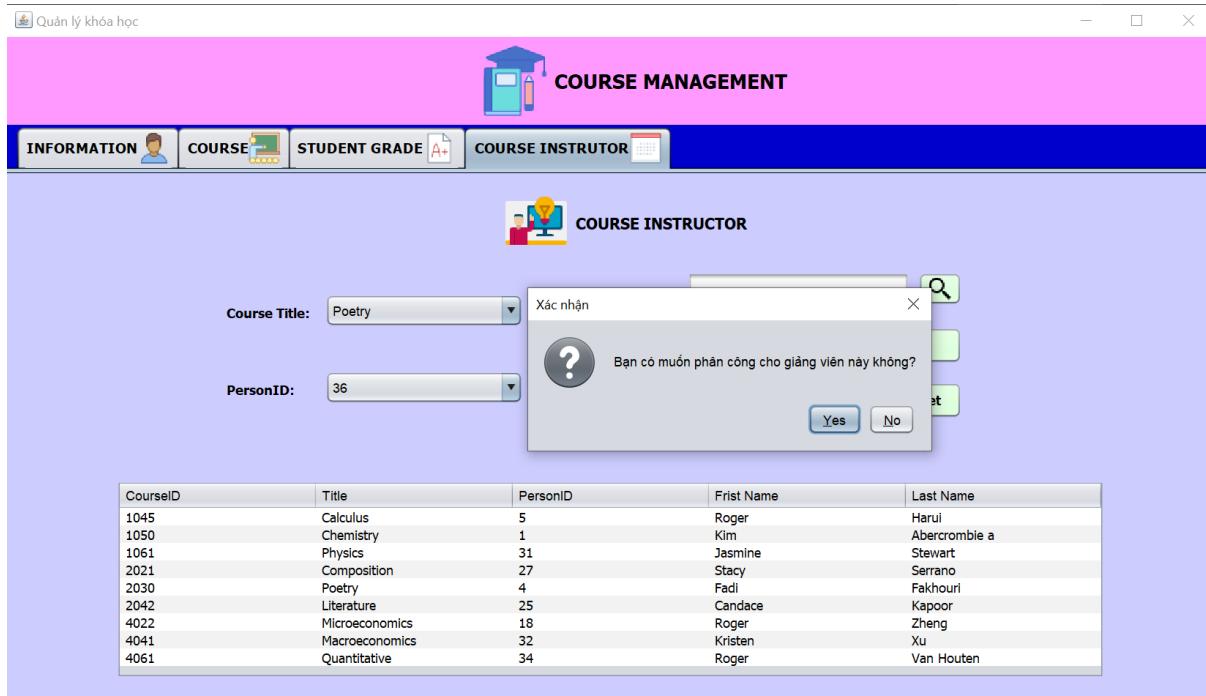
```

## 2. Xử lý 2: Thêm thông tin phân công giảng dạy

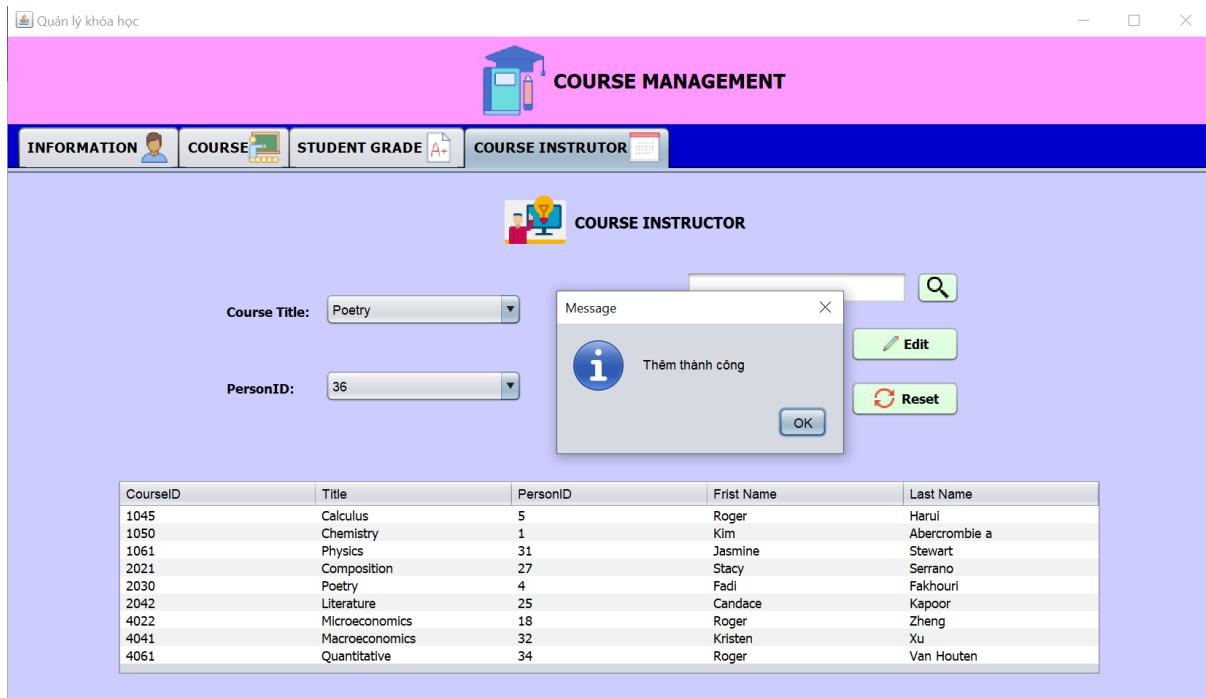
### 2.1 Sơ đồ tuần tự



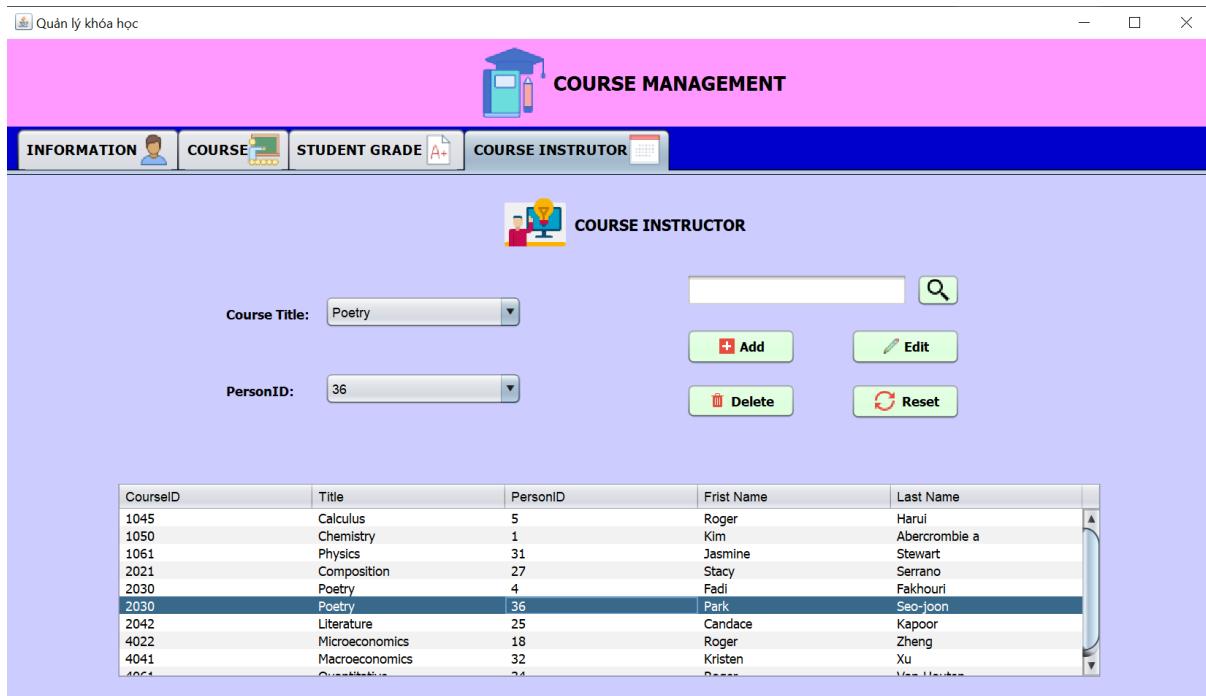
## 2.2 Giao diện



Giao diện thông báo xác nhận thêm phân công giảng dạy



Giao diện thông báo thêm thành công



Giao diện hiển thị thông tin phân công giảng dạy vừa được thêm

### 2.3 Code 3 class:

- ❖ DAO

```

public boolean add(CourseInstructorDTO ctDTO) throws SQLException {
    int courseId = findCourseID(ctDTO.getTitleCourse());
    String sql2 = String.format("INSERT INTO `courseinstructor`(`CourseID`, `PersonID`) VALUES (%d, %d)",
        courseId, ctDTO.getPersonID());
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql2);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean AddCourseInstructor(CourseInstructorDTO ciDTO) throws SQLException {
    CourseInstructorList.add(ciDTO);
    return ciDAO.add(ciDTO);
}

```

## ❖ GUI

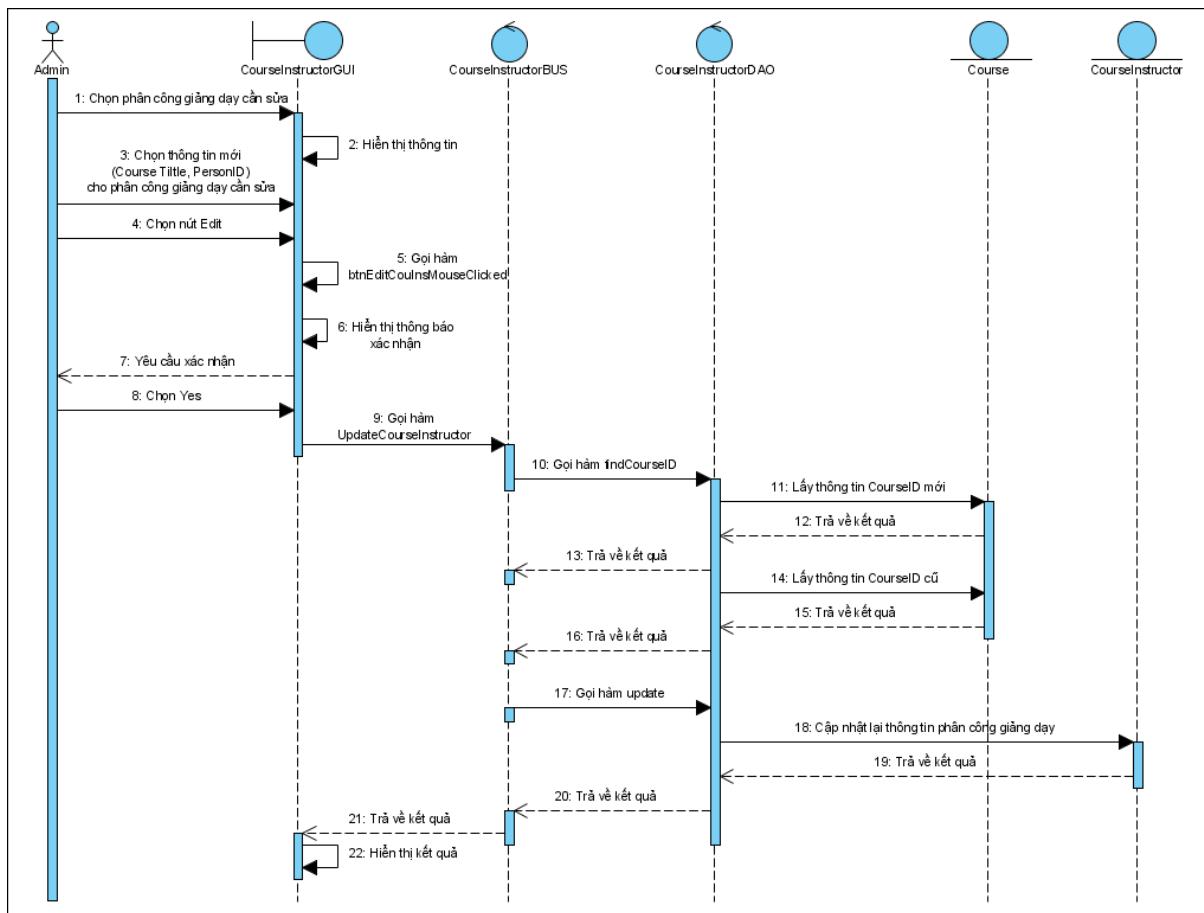
```

private void btnAddCouInsMouseClicked(java.awt.event.MouseEvent evt) {
    int result = JOptionPane.showConfirmDialog(null,
        "Bạn có muốn phân công cho giảng viên này không?",
        "Xác nhận", JOptionPane.YES_NO_OPTION);
    if (result == JOptionPane.NO_OPTION) {
        return;
    }
    CourseInstructorDTO ciDTO = new CourseInstructorDTO();
    ciDTO.setPersonID(Integer.parseInt(cbPersonIdCouIns.getSelectedItem().toString()));
    ciDTO.setTitleCourse(cbCourseTitleCouIns.getSelectedItem().toString());
    ciDTO.setFirstName("");
    ciDTO.setLastName("");
    ciDTO.setCourseID(0);
    try {
        if (ciBUS.AddCourseInstructor(ciDTO)) {
            JOptionPane.showMessageDialog(null, "Thêm thành công");
            readTableCourseInstrcutor();
        } else {
            JOptionPane.showMessageDialog(null, "Thêm thất bại");
        }
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
}

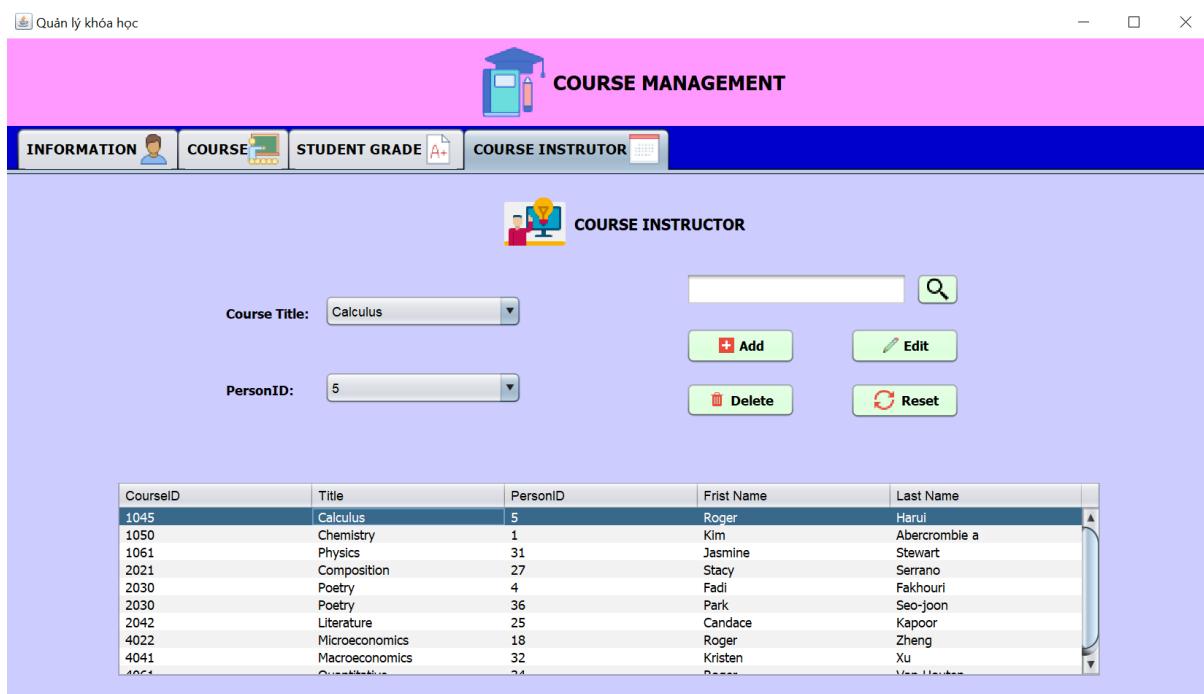
```

### 3. Xử lý 3: Sửa thông tin phân công giảng dạy

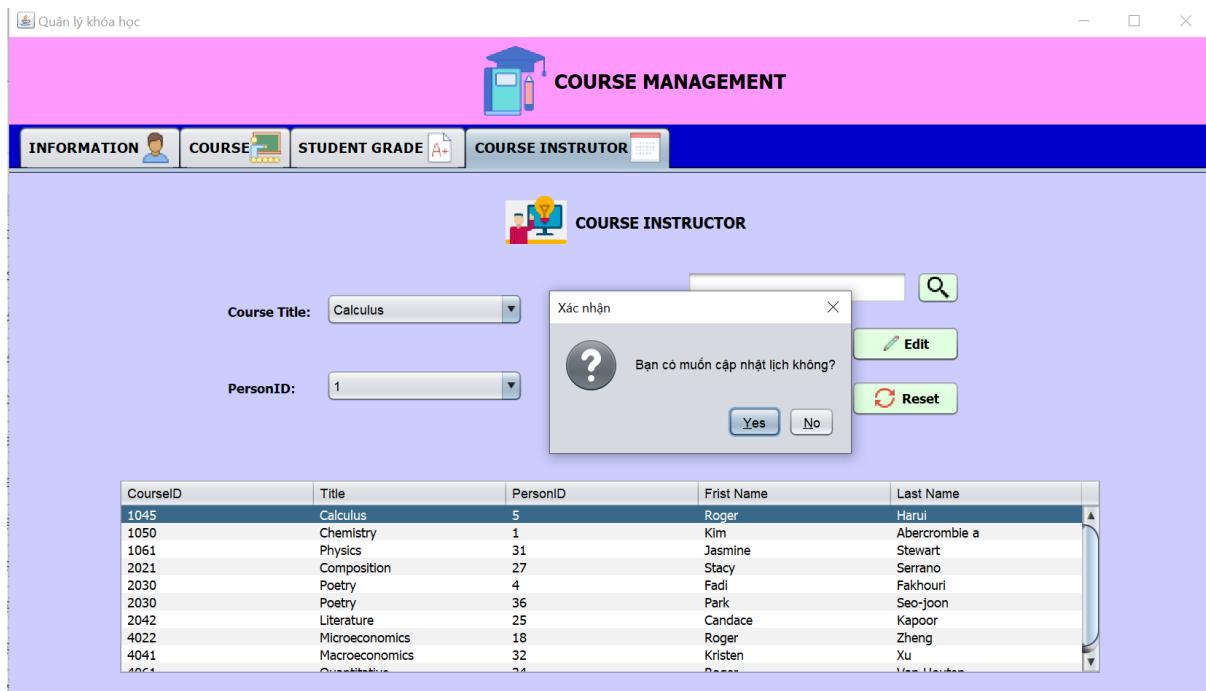
#### 4.1 Sơ đồ tuần tự



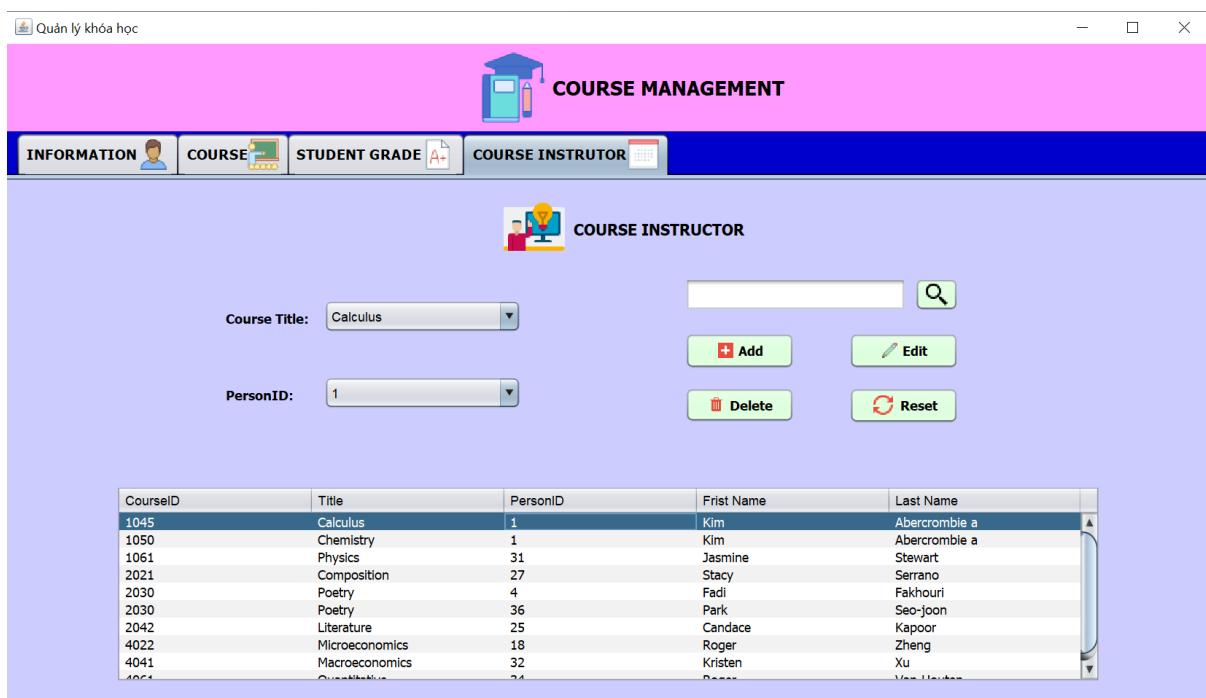
## 4.2 Giao diện



Giao diện hiển thị thông tin phân công Course Title: Calculus và PersonId: 5



Giao diện cập nhật thông tin Course Title: Calculus do PersonId: 1 giảng dạy



Giao diện hiển thị phân công sau khi cập nhật thành công

### 4.3 Code 3 class:

- ❖ DAO

```

public int findCourseID(String title) throws SQLException {
    String sql1 = "SELECT courseinstructor.CourseID, `PersonID` FROM `courseinstructor`, `course` "
    + "WHERE course.Title = '" + title + "' AND courseinstructor.CourseID = course.CourseID ";
    ResultSet rs1 = ConnectDatabase.GetInstance().ExcuteSELECT(sql1);
    if (rs1.next()) {
        return rs1.getInt("CourseID");
    } else {
        System.out.println("Can't find the course");
        return 0;
    }
}

```

```

public boolean update(CourseInstructorDTO ciDTO, int oldPersonID, String oldCourseTitle, int courseId, int oldCourseId) throws SQLException {
    String sql = String.format("UPDATE `courseinstructor` "
        + "SET PersonID=%d, CourseID=%d "
        + "WHERE PersonID=%d AND CourseID=%d",
        ciDTO.getPersonID(), courseId, oldPersonID, oldCourseId);
    System.out.println(sql);
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}

```

## ❖ BUS

```

public boolean UpdateCourseInstructor(CourseInstructorDTO ciDTO, int oldPersonID, String oldCourseTitle) throws SQLException {
    int courseId = ciDAO.findCourseID(ciDTO.getTitleCourse());
    int oldCourseId = ciDAO.findCourseID(oldCourseTitle);
    for (int i = 0; i < courseInstructorList.size(); i++) {
        if (oldPersonID == courseInstructorList.get(i).getPersonID() && oldCourseId == courseInstructorList.get(i).getCourseID()) {
            courseInstructorList.set(i, ciDTO);
            break;
        }
    }
    return ciDAO.update(ciDTO, oldPersonID, oldCourseTitle, courseId, oldCourseId );
}

```

## ❖ GUI

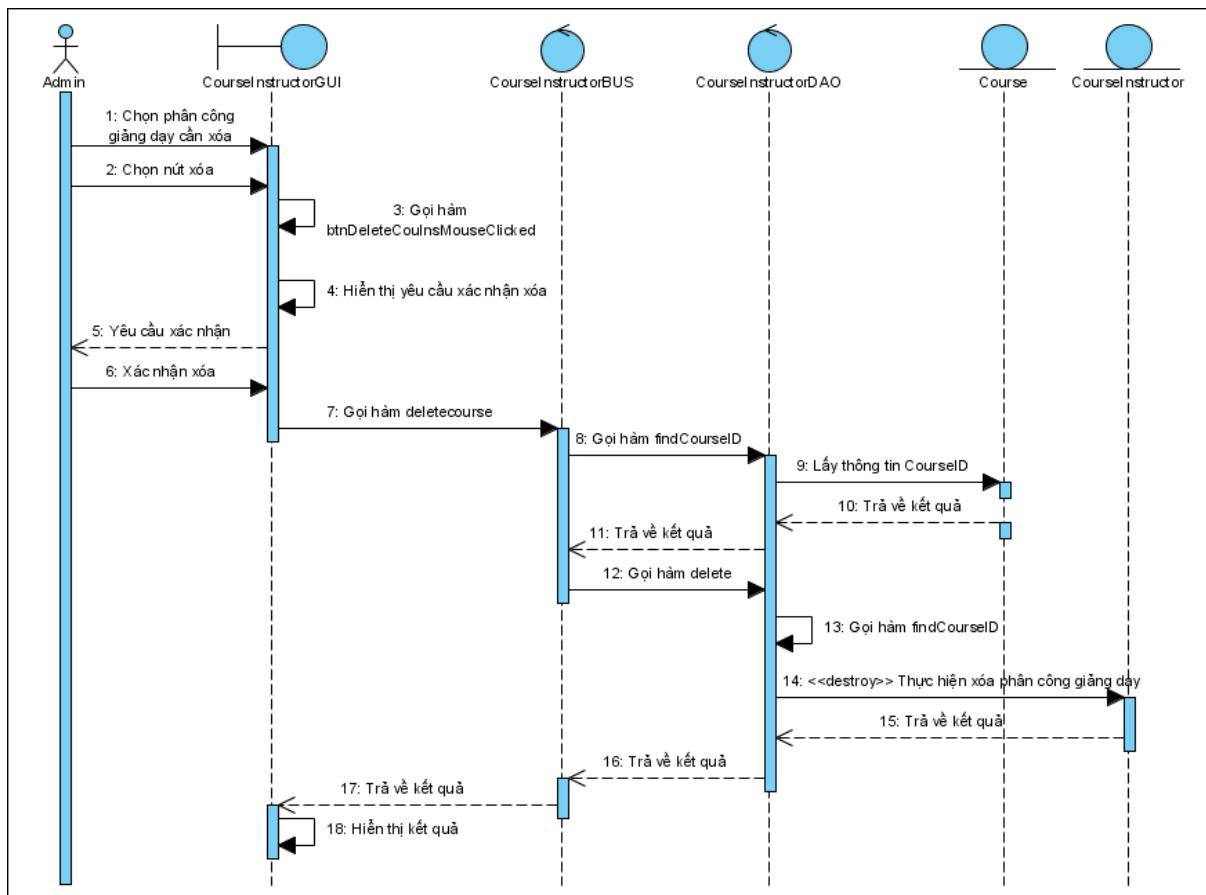
```

private void btnEditCouInsMouseClicked(java.awt.event.MouseEvent evt) {
    int i = tblCourseInstructor.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn lịch cần cập nhật");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn cập nhật lịch không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            CourseInstructorDTO ciDTO = new CourseInstructorDTO();
            String oldCourseTitle = tblCourseInstructor.getModel().getValueAt(i, 1).toString();
            int oldIdPerson = Integer.parseInt(tblCourseInstructor.getModel().getValueAt(i, 2).toString());
            ciDTO.setCourseID(0);
            ciDTO.setFirstName("");
            ciDTO.setLastName("");
            ciDTO.setTitleCourse(cbCourseTitleCouIns.getSelectedItem().toString());
            ciDTO.setPersonID(Integer.parseInt(cbPersonIdCouIns.getSelectedItem().toString()));
            try {
                if (ciBUS.UpdateCourseInstructor(ciDTO, oldIdPerson, oldCourseTitle)) {
                    JOptionPane.showMessageDialog(null, "Cập nhật thành công");
                    readTableCourseInstructor();
                } else {
                    JOptionPane.showMessageDialog(null, "Cập nhật thất bại");
                }
            } catch (SQLException ex) {
                Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}

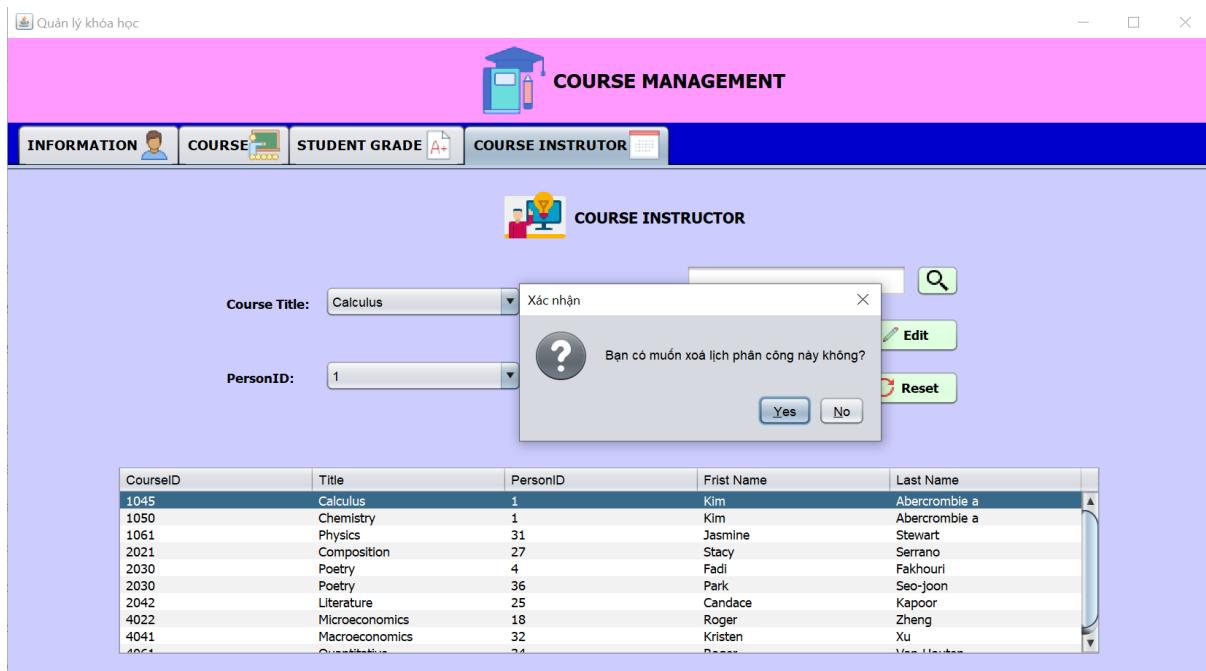
```

## 4. Xử lý 4: Xóa thông tin phân công giảng dạy

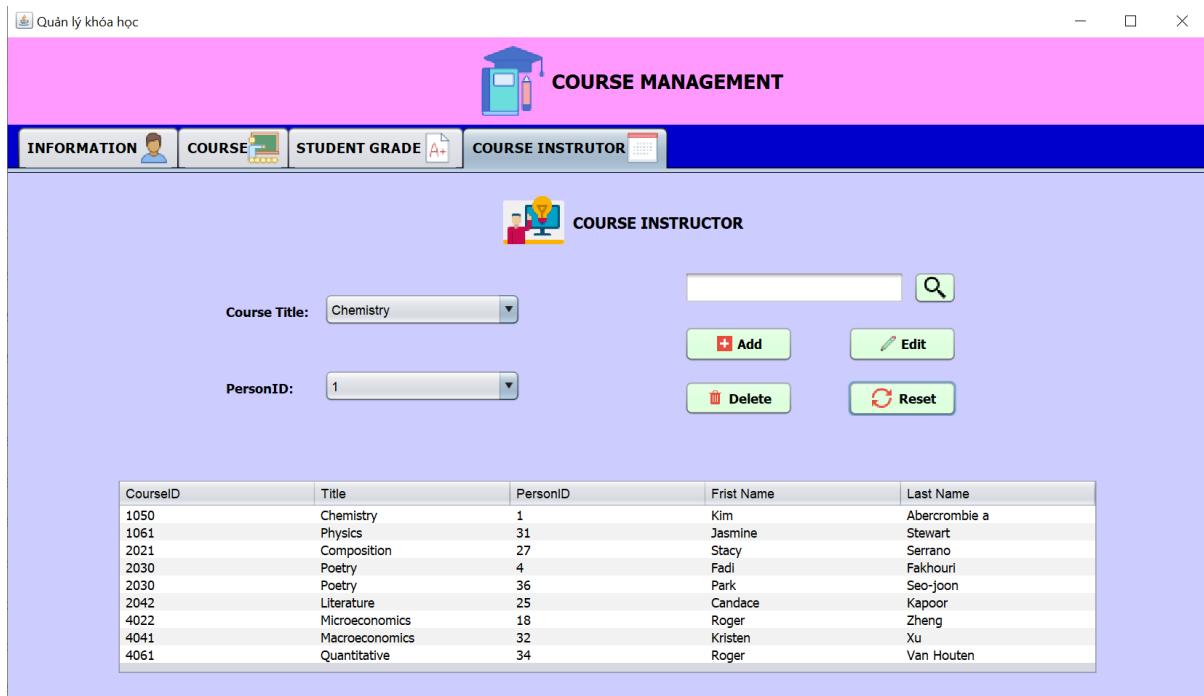
### 3.1 Sơ đồ tuần tự



### 3.2 Giao diện



Giao diện thông báo xác nhận xóa lịch phân công



*Giao diện hiển thị danh sách phân công sau khi xóa*

### 3.3 Code 3 class:

#### ❖ DAO

```
public int findCourseID(String title) throws SQLException {
    String sql1 = "SELECT courseinstructor.CourseID, `PersonID` FROM `courseinstructor`, `course` "
    + "WHERE course.Title = '" + title + "' AND courseinstructor.CourseID = course.CourseID ";
    ResultSet rs1 = ConnectDatabase.GetInstance().ExcuteSELECT(sql1);
    if (rs1.next()) {
        return rs1.getInt("CourseID");
    } else {
        System.err.println("Can't find the course");
        return 0;
    }
}
```

```
public boolean delete(int idPerson, String title) throws SQLException {
    int courseId = findCourseID(title);
    String sql = String.format("DELETE FROM `courseinstructor` WHERE CourseID=%d AND PersonID=%d", courseId, idPerson);
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result.equals("Thành công")) {
        return true;
    } else {
        return false;
    }
}
```

## ❖ BUS

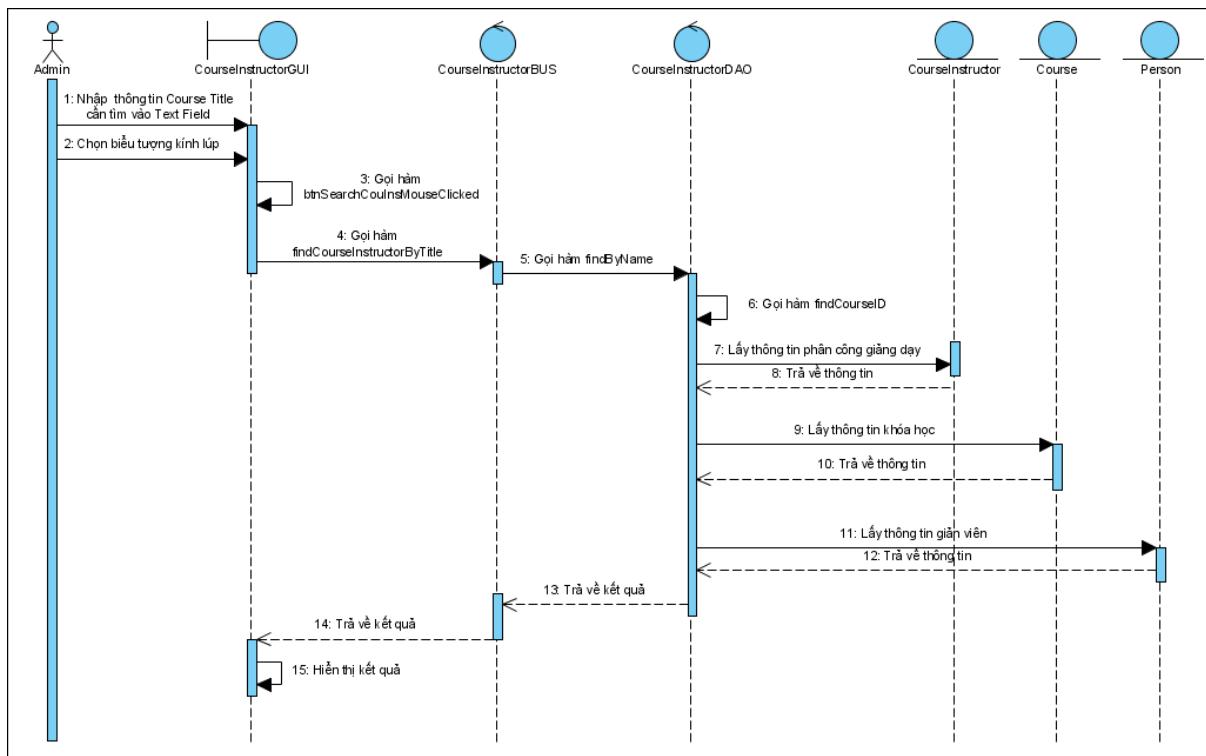
```
public boolean deletecourse(int idPerson, String courseTitle) throws SQLException {
    int courseId = ciDAO.findCourseID(courseTitle);
    for (CourseInstructorDTO courseInstructor : CourseInstructorList) {
        if (courseInstructor.getPersonID() == idPerson && courseInstructor.getCourseID() == courseId) {
            CourseInstructorList.remove(courseInstructor);
            break;
        }
    }
    return ciDAO.delete(idPerson, courseTitle);
}
```

## ❖ GUI

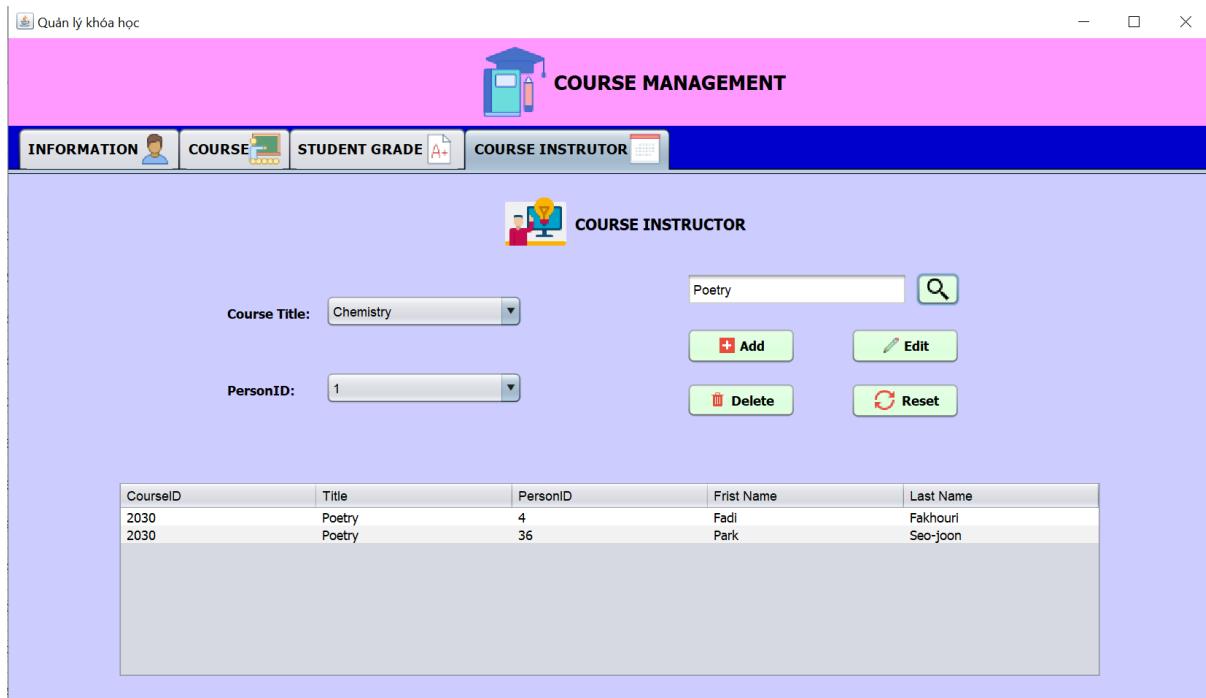
```
private void btnDeleteCouInsMouseClicked(java.awt.event.MouseEvent evt) {
    int i = tbCourseInstructor.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn course cần xoá");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn xoá lịch phân công này không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            int idPerson = Integer.parseInt(cbPersonIdCouIns.getSelectedItem().toString());
            String courseTitle = cbCourseTitleCouIns.getSelectedItem().toString();
            try {
                if (ciBUS.deletecourse(idPerson, courseTitle)) {
                    JOptionPane.showMessageDialog(null, "Xóa thành công");
                    readTableCourseInstructor();
                } else {
                    JOptionPane.showMessageDialog(null, "Xóa thất bại");
                }
            } catch (SQLException ex) {
                Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}
```

## 5. Xử lý 5: Tìm kiếm thông tin phân công giảng dạy theo tên khóa học

### 5.1 Sơ đồ tuần tự



## 5.2 Giao diện



*Giao diện kết quả tìm kiếm*

## 5.3 Code 3 class:

- ❖ DAO

```

public ArrayList<CourseInstructorDTO> findByName (String courseTitle) throws SQLException{
    ArrayList<CourseInstructorDTO> searchList = new ArrayList<>();
    int courseId = findCourseID(courseTitle);
    String sql = String.format("select * from courseinstructor "
        + "left JOIN course on courseinstructor.CourseID = course.CourseID"
        + " left join person on person.PersonID = courseinstructor.PersonID "
        + "WHERE courseinstructor.CourseID = '%d'", courseId);
    ResultSet rs = ConnectDatabase.GetInstance().ExcuteSELECT(sql);
    while (rs.next()) {
        CourseInstructorDTO ciDTO = new CourseInstructorDTO();
        ciDTO.setPersonID(rs.getInt("person.PersonID"));
        ciDTO.setFirstName(rs.getString("person.Firstname"));
        ciDTO.setLastName(rs.getString("person.Lastname"));
        ciDTO.setCourseID(rs.getInt("courseinstructor.CourseID"));
        ciDTO.setTitleCourse(rs.getString("course.Title"));
        searchList.add(ciDTO);
    }
    return searchList;
}

```

```

public int findCourseID(String title) throws SQLException {
    String sql1 = "SELECT courseinstructor.CourseID, `PersonID` FROM `courseinstructor`, `course` "
        + "WHERE course.Title = '" + title + "' AND courseinstructor.CourseID = course.CourseID ";
    ResultSet rs1 = ConnectDatabase.GetInstance().ExcuteSELECT(sql1);
    if (rs1.next()) {
        return rs1.getInt("CourseID");
    } else {
        System.err.println("Can't find the course");
        return 0;
    }
}

```

## ❖ BUS

```

public ArrayList<CourseInstructorDTO> findCourseInstructorByTitle(String title) throws SQLException {
    return ciDAO.findByName(title);
}

```

## ❖ GUI

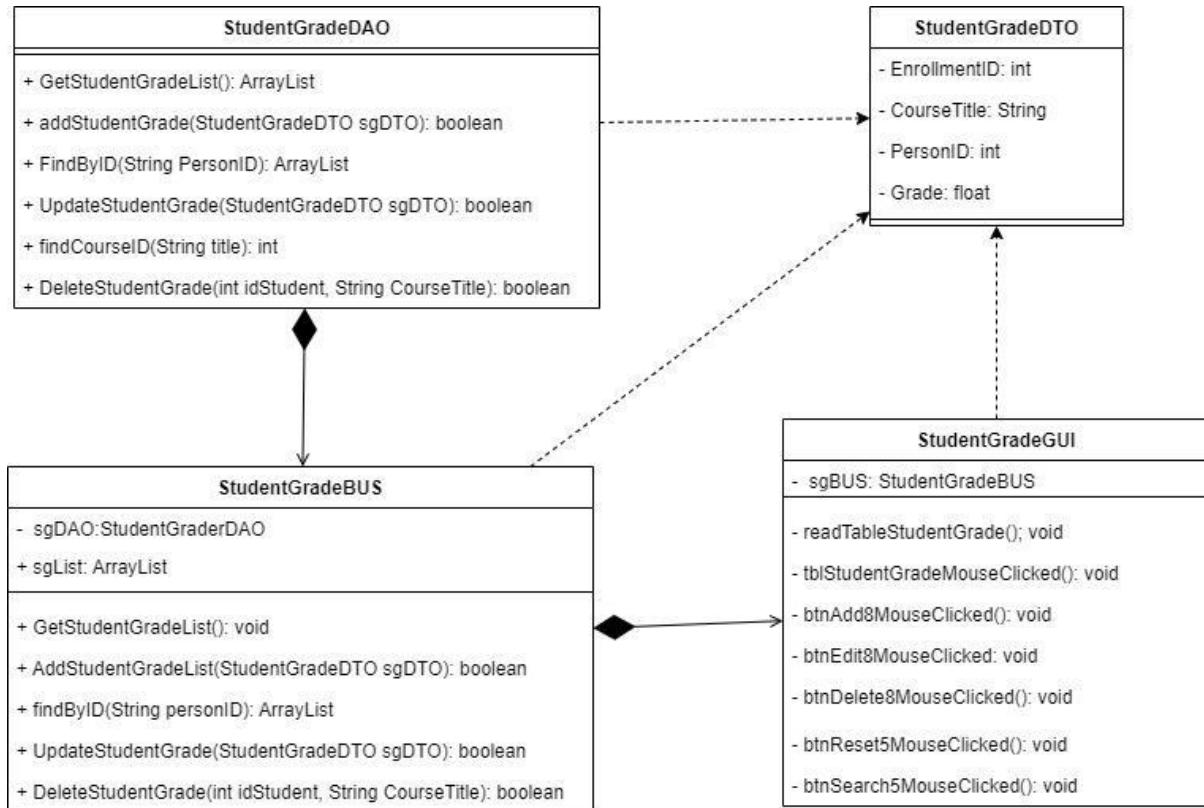
```

private void btnSearchCouInsMouseClicked(java.awt.event.MouseEvent evt) {
    String input = txtSearchCouIns.getText();
    if (!input.isEmpty()) {
        try {
            ArrayList<CourseInstructorDTO> listCourseInstructor = ciBUS.findCourseInstructorByTitle(input);
            if (!listCourseInstructor.isEmpty()) {
                modelTableCourseInstructor.setRowCount(0);
                for (CourseInstructorDTO ciDTO : listCourseInstructor) {
                    modelTableCourseInstructor.addRow(new Object[]{
                        ciDTO.getCourseID(), ciDTO.getTitleCourse(), ciDTO.getPersonID(), ciDTO.getFirstName(), ciDTO.getLastName()
                    });
                }
            } else {
                JOptionPane.showMessageDialog(null, "Không tìm thấy");
            }
        } catch (SQLException ex) {
            Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
        }
    } else {
        JOptionPane.showMessageDialog(null, "Thiếu Thông Tin");
    }
}

```

## Phần 4: Chức năng quản lý kết quả khóa học

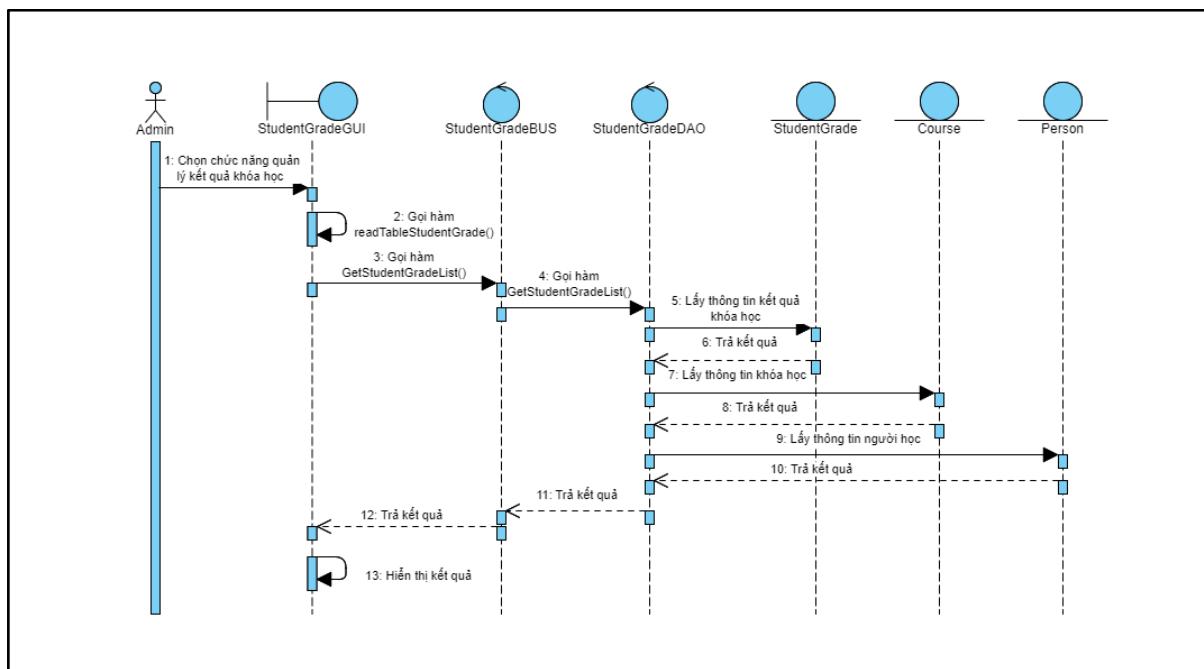
### I. Sơ đồ class chung



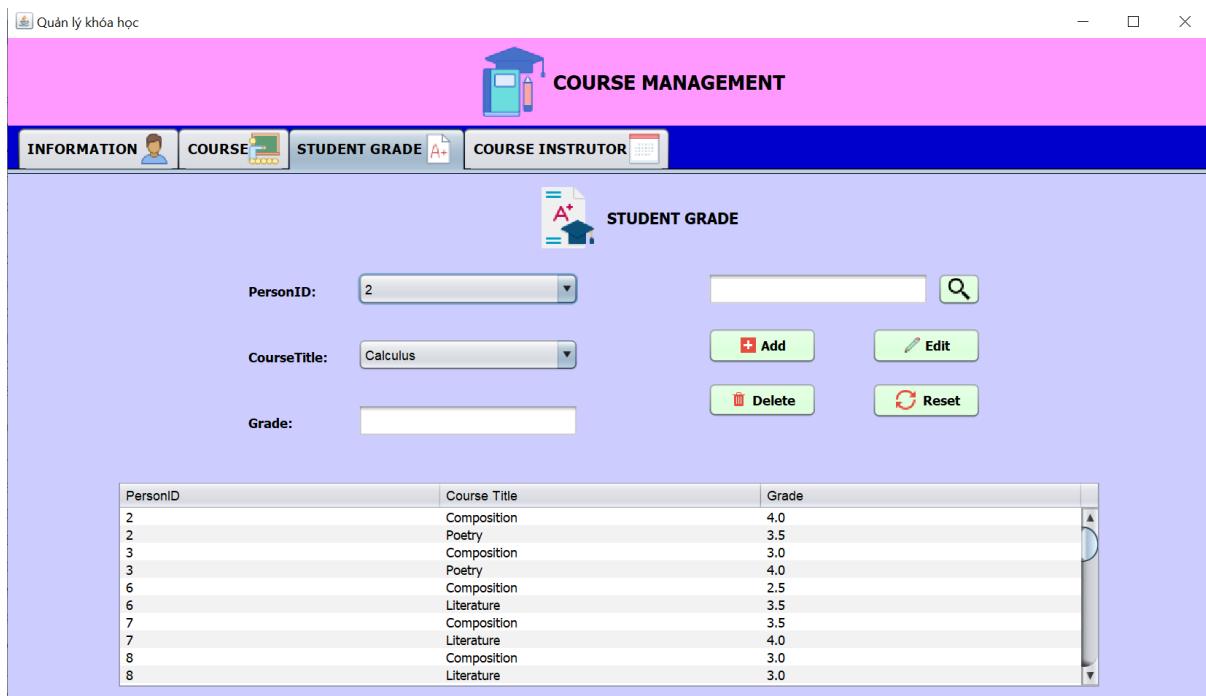
### II. Quản lý kết quả khóa học

#### 1. Xử lý 1: Hiển thị danh sách kết quả khóa học

##### 1.1. Sơ đồ tuần tự



##### 1.2. Giao diện



Giao diện hiển thị danh sách kết quả khóa học

### 1.3. Code 3 class:

#### ❖ DAO

```

public ArrayList<StudentGradeDTO> GetStudentGradeList() throws SQLException {
    ArrayList<StudentGradeDTO> sgList = new ArrayList<>();

    String sql = "select * from studentgrade";
    ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
    while (rs.next()) {
        StudentGradeDTO sgDTO = new StudentGradeDTO();
        String sql2 = "select course.Title,studentgrade.EnrollmentID  from studentgrade left join course "
            + "on studentgrade.courseID = course.courseID where studentgrade.EnrollmentID ="
            + rs.getInt("EnrollmentID");
        ResultSet rs2 = ConnectDatabase.GetInstance().ExecuteSELECT(sql2);
        sgDTO.setEnrollmentID(rs.getInt("EnrollmentID"));
        while (rs2.next()) {
            sgDTO.setCourseTitle(rs2.getString("Title"));
        }
        sgDTO.setPersonID(rs.getInt("StudentID"));
        sgDTO.setGrade(rs.getFloat("Grade"));
        sgList.add(sgDTO);
    }
    return sgList;
}

```

#### ❖ BUS

```

public void GetStudentGradeList() throws SQLException{
    sgList = sgDAO.GetStudentGradeList();
}

```

#### ❖ GUI

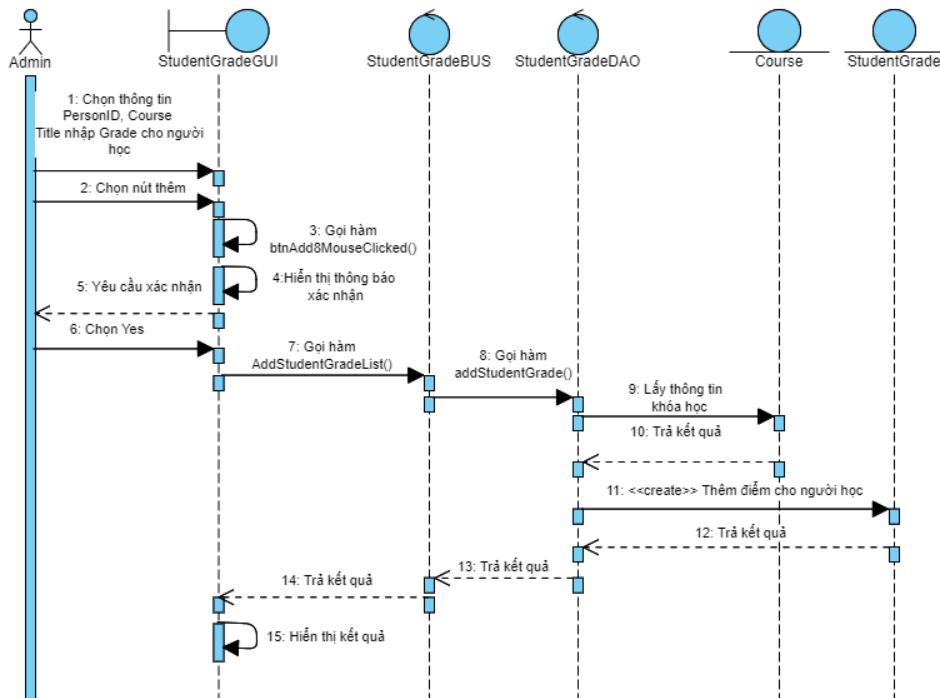
```

private void readTableStudentGrade() {
    try {
        sgBUS.GetStudentGradeList();
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
    modelTableStudentGrade.setRowCount(0);
    for (StudentGradeDTO stDTO : StudentGradeBUS.sgList) {
        modelTableStudentGrade.addRow(new Object[]{
            stDTO.getPersonID(), stDTO.getCourseTitle(), stDTO.getGrade()
        });
    }
}

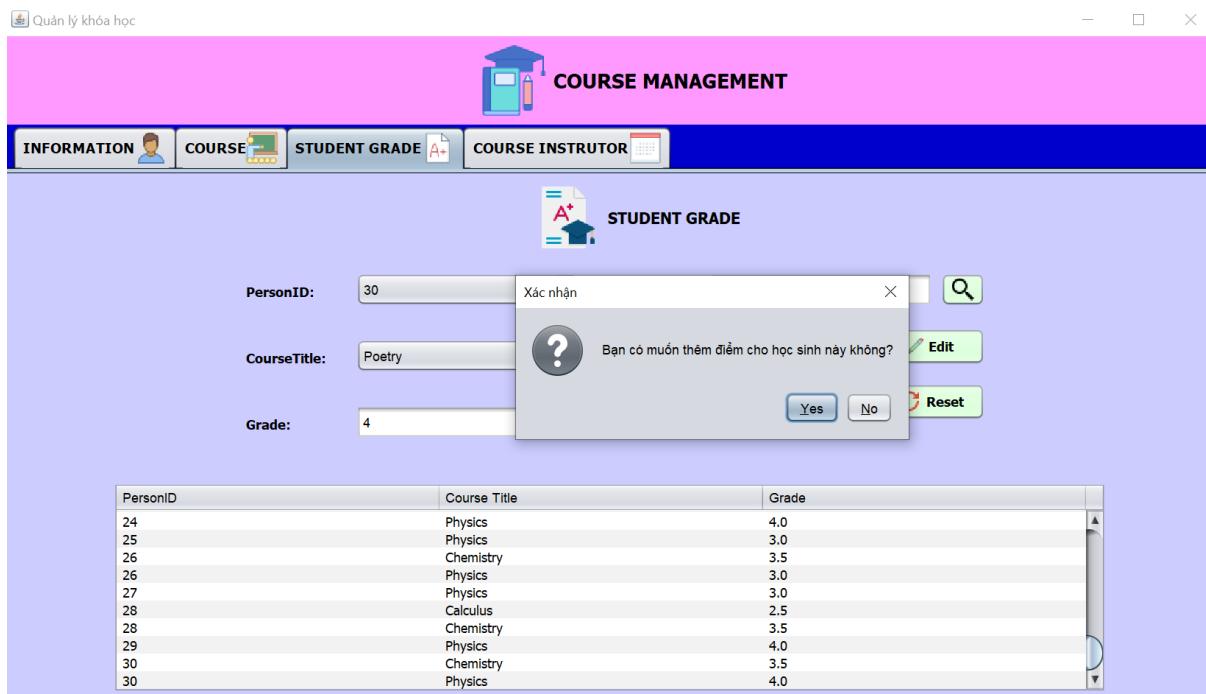
```

## 2. Xử lý 2: Thêm điểm cho người học

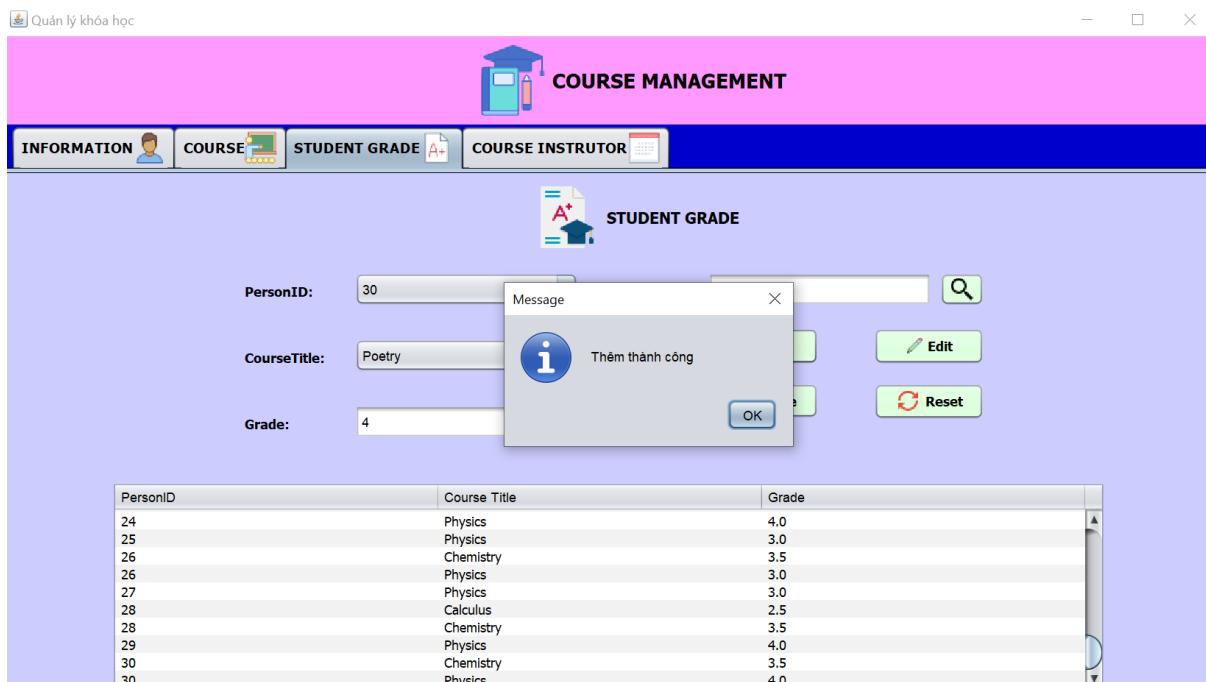
### 2.1. Sơ đồ tuần tự



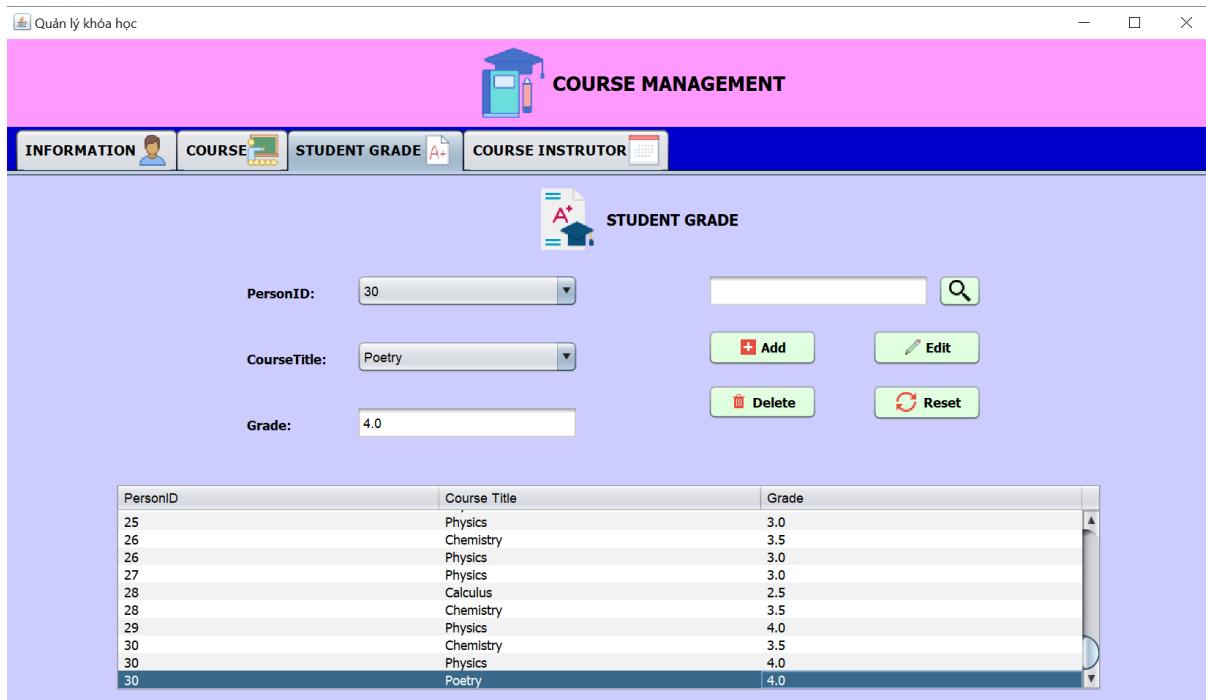
### 2.2. Giao diện



Giao diện thông báo xác nhận thêm điểm



Giao diện thông báo thêm điểm thành công



Giao diện hiển thị điểm người học sau khi thêm

### 2.3. Code 3 class:

#### ❖ DAO

```
public boolean addStudentGrade(StudentGradeDTO sgDTO) throws SQLException {
    String sql = "select CourseID from course where Title = '" + sgDTO.getCourseTitle() + "'";
    ResultSet rsl = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
    int titleCourse = 0;
    if (rsl.next()) {
        titleCourse = rsl.getInt("CourseID");
    } else {
        System.out.println("Can't find the course");
        return false;
    }
    String sql2 = String.format("INSERT INTO `StudentGrade`(`CourseID`, `StudentID`, `Grade`) VALUES (%d, %d, %f)"
        , titleCourse, sgDTO.getPersonID(), sgDTO.getGrade());
    String result = ConnectDatabase.GetInstance().ExecuteINSERTDELETEUPDATE(sql2);
    System.out.println(result);
    if (result == "Thành công") {
        return true;
    } else
        return false;
}
```

#### ❖ BUS

```
public boolean AddStudentGradeList(StudentGradeDTO sgDTO) throws SQLException {
    sgList.add(sgDTO);
    return sgDAO.addStudentGrade(sgDTO);
}
```

#### ❖ GUI

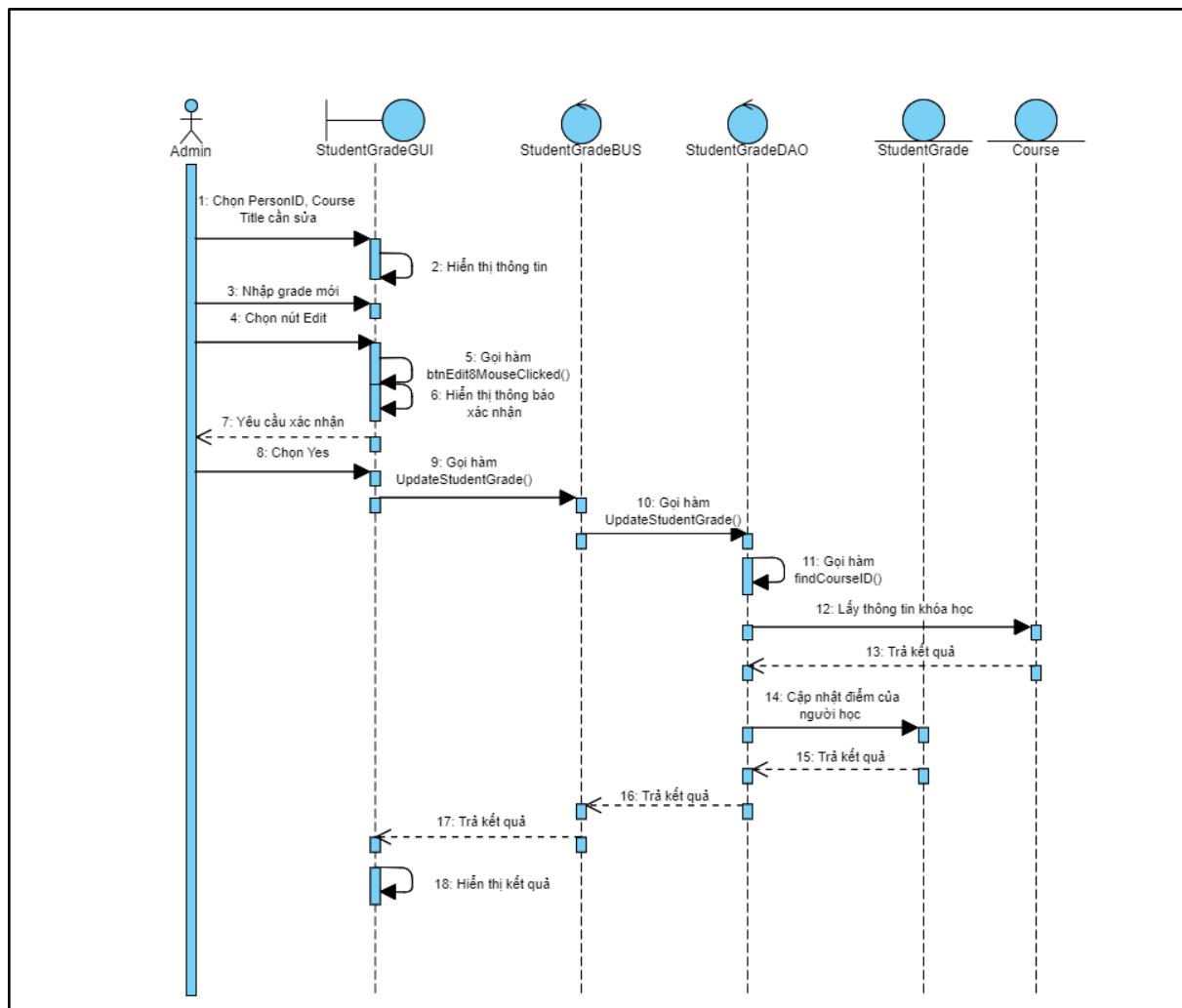
```

private void btnAdd8MouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    // comboBox_tenhang.getSelectedItem().toString()
    int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn thêm điểm cho học sinh này không?", "Xác nhận",
        JOptionPane.YES_NO_OPTION);
    if (result == JOptionPane.NO_OPTION) {
        return;
    }
    StudentGradeDTO stDTO = new StudentGradeDTO();
    stDTO.setPersonID(Integer.parseInt(cbPersonID.getSelectedItem().toString()));
    stDTO.setCourseTitle(cbCourseTitle.getSelectedItem().toString());
    stDTO.setGrade(Float.parseFloat(txtGrade.getText()));
    try {
        if (sgBUS.AddStudentGradeList(stDTO)) {
            JOptionPane.showMessageDialog(null, "Thêm thành công");
            readTableStudentGrade();
        } else {
            JOptionPane.showMessageDialog(null, "Thêm thất bại");
        }
    } catch (SQLException ex) {
        Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
    }
}

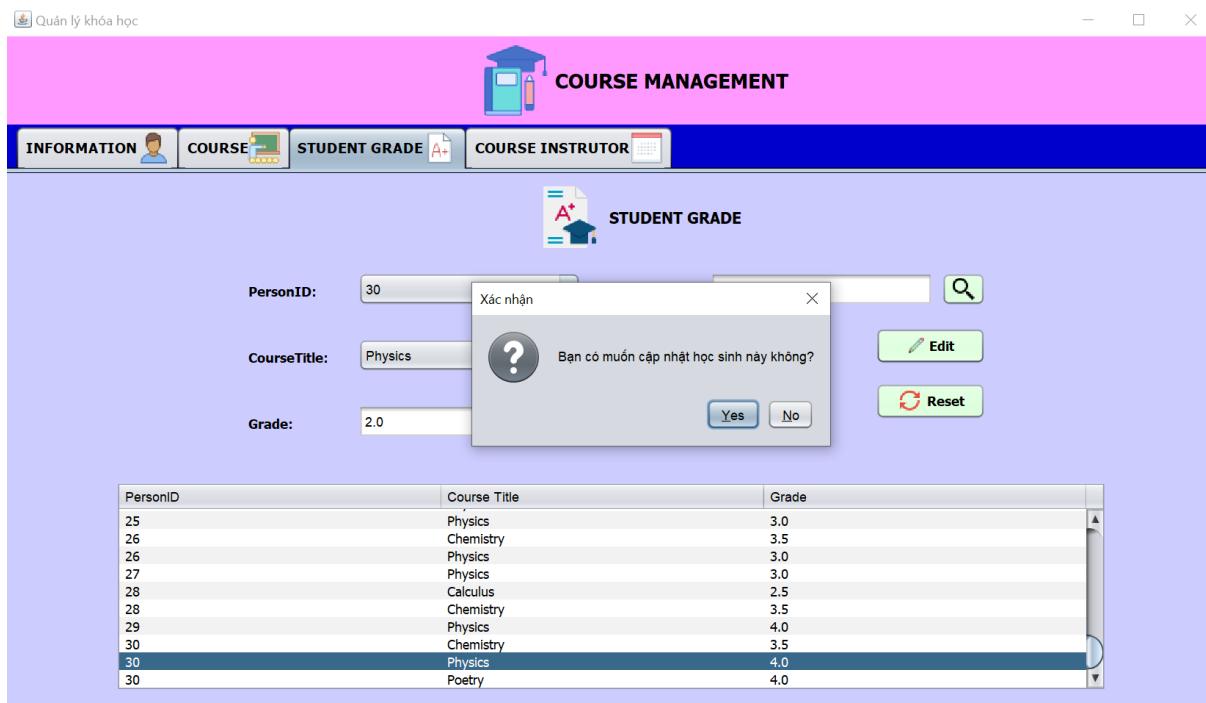
```

### 3. Xử lý 3: Cập nhập điểm của người học

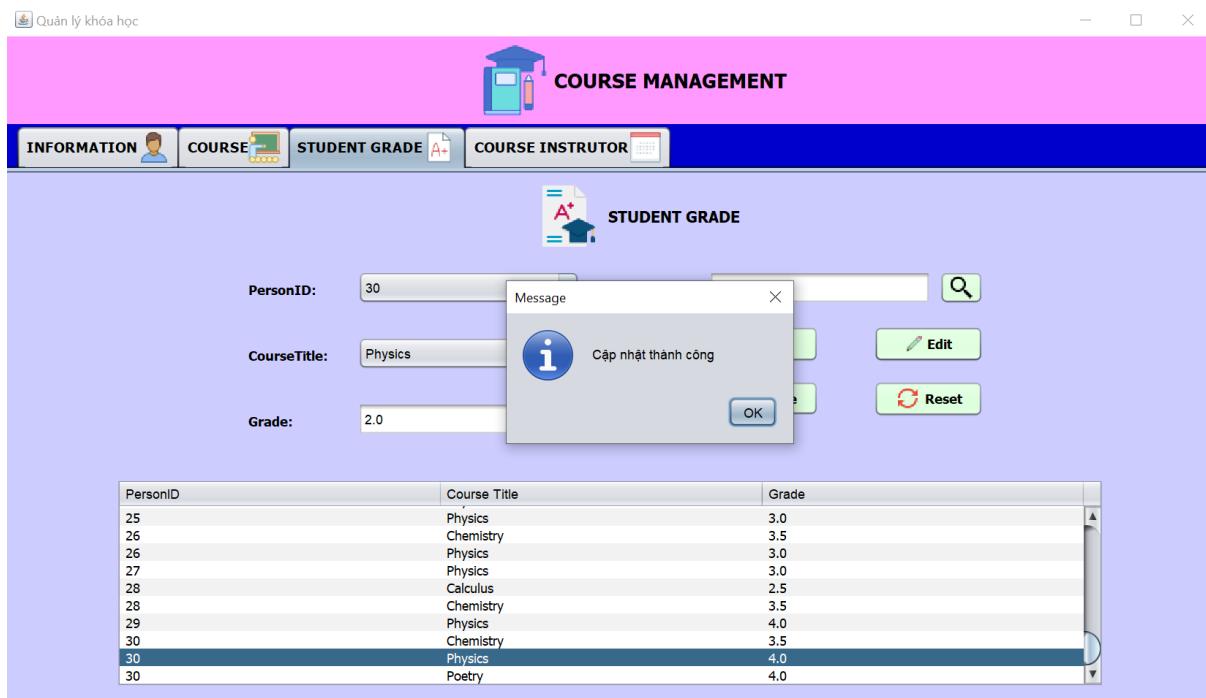
#### 3.1. Sơ đồ tuần tự



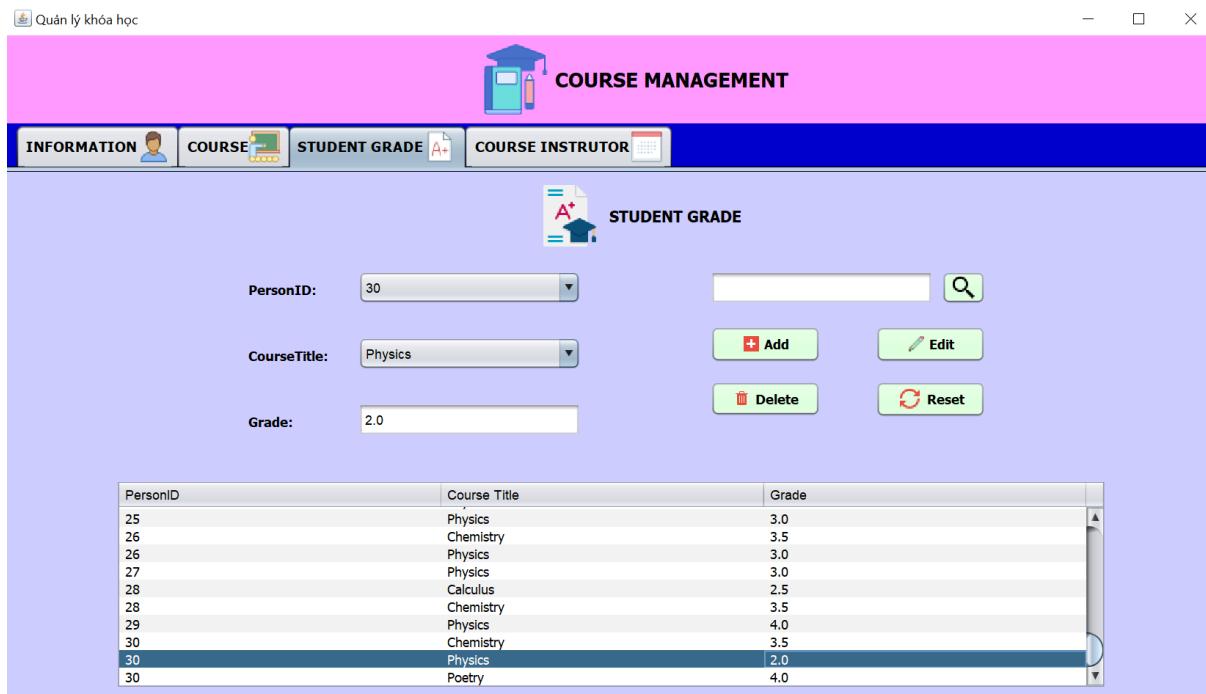
#### 3.2. Giao diện



Giao diện thông báo xác nhận cập nhật điểm



Giao diện thông báo cập nhật điểm thành công



Giao diện điểm người học sau khi cập nhật

### 3.3. Code 3 class:

#### ❖ DAO

```
public boolean UpdateStudentGrade(StudentGradeDTO sgDTO) throws SQLException{
    int courseId = findCourseID(sgDTO.getCourseTitle());
    String sql = String.format("UPDATE `studentgrade`"
        + "SET grade=%f"
        + "WHERE StudentID=%d AND CourseID=%d",
        sgDTO.getGrade(), sgDTO.getPersonID(), courseId);
    System.out.println(sql);
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}
```

```
public int findCourseID(String title) throws SQLException {
    String sql1 = "SELECT CourseID FROM `course` WHERE Title = '" + title + "'";
    ResultSet rsl = ConnectDatabase.GetInstance().ExcuteSELECT(sql1);
    if (rsl.next()) {
        return rsl.getInt("CourseID");
    } else {
        System.err.println("Can't find the course");
        return 0;
    }
}
```

#### ❖ BUS

```

public boolean UpdateStudentGrade(StudentGradeDTO sgDTO) throws SQLException{
    for (int i = 0; i < sgList.size(); i++) {
        if (sgDTO.getPersonID() == sgList.get(i).getPersonID() && sgDTO.getCourseTitle()
            == sgList.get(i).getCourseTitle()) {
            sgList.set(i, sgDTO);
        }
    }
    return sgDAO.UpdateStudentGrade(sgDTO);
}

```

## ❖ GUI

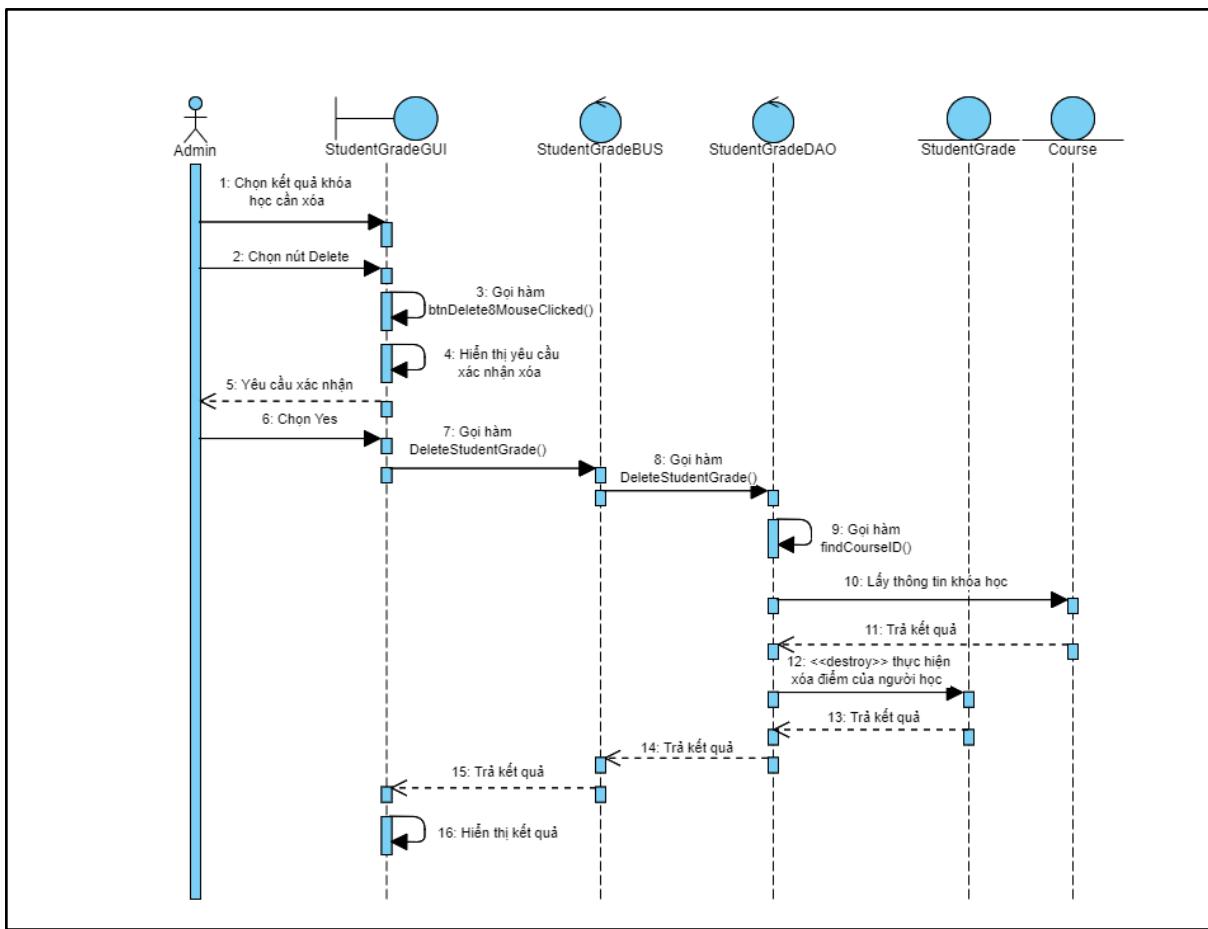
```

private void btnEdit8MouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int i = tblStudentGrade.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn học sinh cần cập nhật");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn cập nhật học sinh này không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            StudentGradeDTO sgDTO = new StudentGradeDTO();
            String oldCourseTitle = tblStudentGrade.getModel().getValueAt(i, 1).toString();
            int oldIdPerson = Integer.parseInt(tblStudentGrade.getModel().getValueAt(i, 0).toString());
            sgDTO.setCourseTitle(oldCourseTitle);
            sgDTO.setPersonID(oldIdPerson);
            sgDTO.setGrade(Float.parseFloat(txtGrade.getText()));
            try {
                if (sgBUS.UpdateStudentGrade(sgDTO)) {
                    JOptionPane.showMessageDialog(null, "Cập nhật thành công");
                    readTableStudentGrade();
                } else {
                    JOptionPane.showMessageDialog(null, "Cập nhật thất bại");
                }
            } catch (SQLException ex) {
                Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}

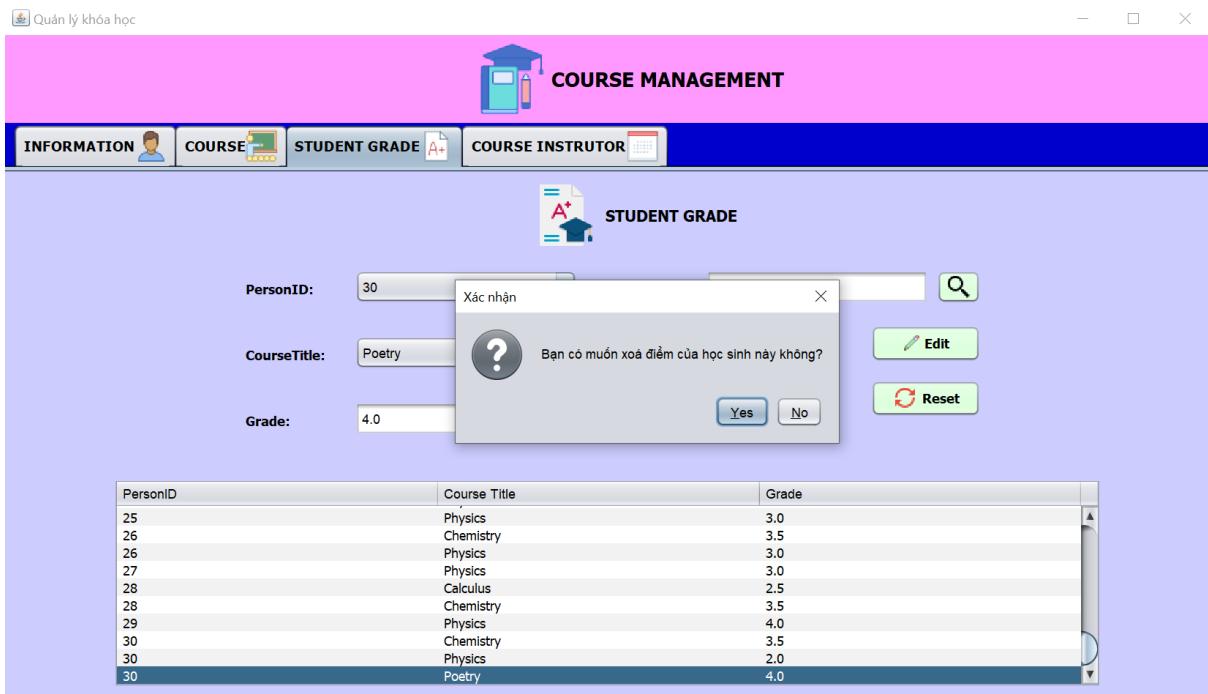
```

## 4. Xử lý 4: Xóa điểm của người học

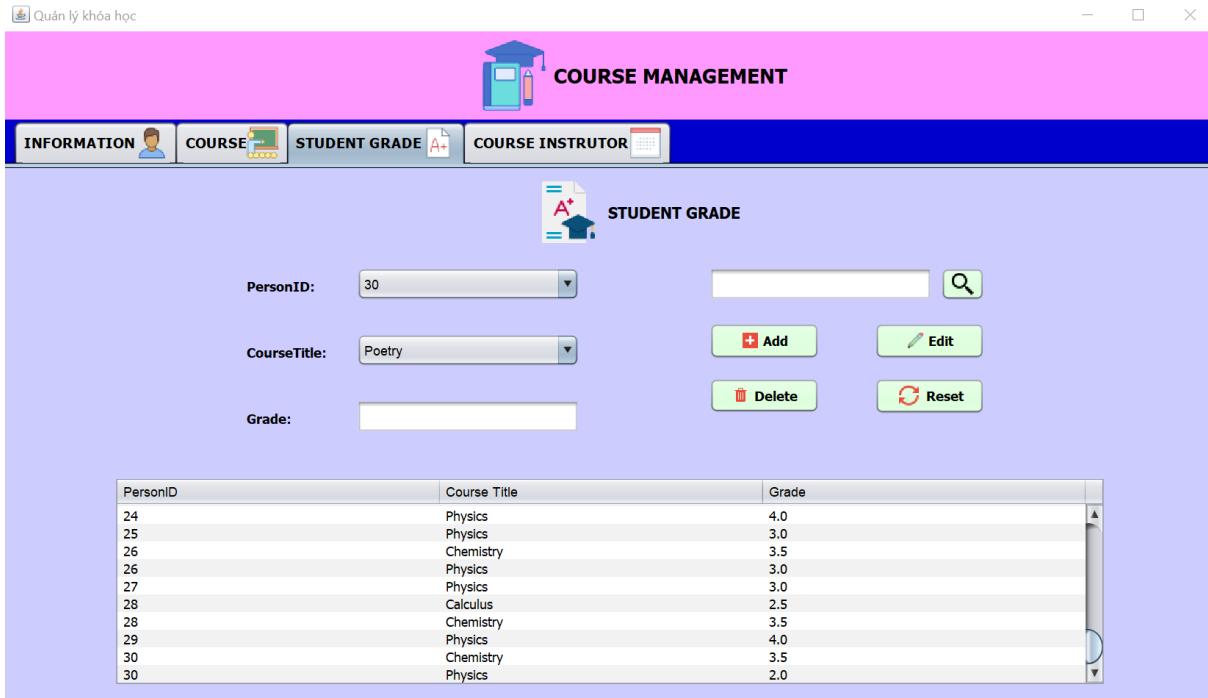
### 4.1. Sơ đồ tuần tự



## 4.2. Giao diện



Giao diện thông báo xác nhận xóa điểm của người học



Giao diện sau khi xóa điểm thành công

### 4.3. Code 3 class:

#### ❖ DAO

```

public boolean DeleteStudentGrade(int idStudent, String CourseTitle) throws SQLException{
    int courseId = findCourseID(CourseTitle);

    String sql = String.format("DELETE FROM `studentgrade` WHERE CourseID='%d' AND StudentID='%d'", courseId, idStudent);
    String result = ConnectDatabase.GetInstance().ExcuteINSERTDELETEUPDATE(sql);
    if (result == "Thành công") {
        return true;
    } else {
        return false;
    }
}

```

```

public int findCourseID(String title) throws SQLException {
    String sql1 = "SELECT CourseID  FROM `course` WHERE Title = '" + title + "'";
    ResultSet rs1 = ConnectDatabase.GetInstance().ExcuteSELECT(sql1);
    if (rs1.next()) {
        return rs1.getInt("CourseID");
    } else {
        System.out.println("Can't find the course");
        return 0;
    }
}

```

#### ❖ BUS

```

public boolean DeleteStudentGrade(int idStudent, String CourseTitle) throws SQLException{
    for (StudentGradeDTO sgDTO : sgList) {
        if (sgDTO.getPersonID() == idStudent && sgDTO.getCourseTitle().equals(CourseTitle)) {
            sgList.remove(sgDTO);
            break;
        }
    }
    return sgDAO.DeleteStudentGrade(idStudent, CourseTitle);
}

```

## ❖ GUI

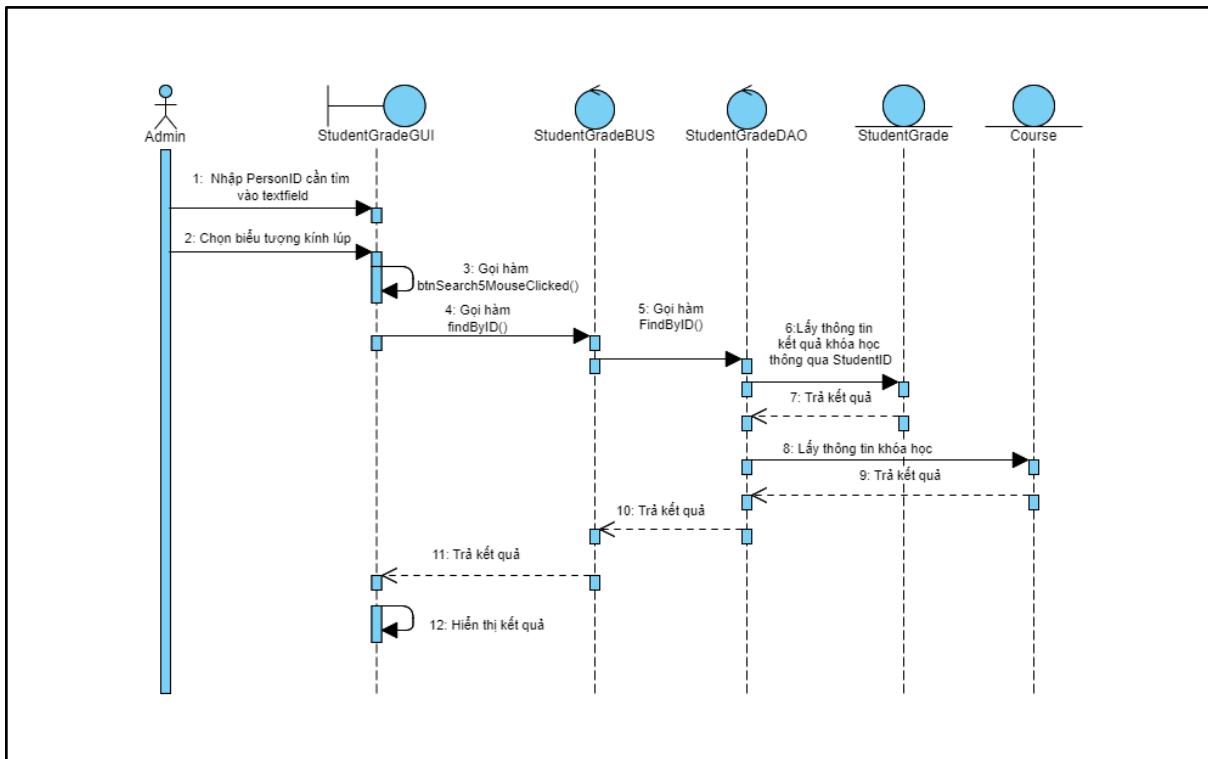
```

private void btnDelete8MouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int i = tblStudentGrade.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(null, "Hãy chọn điểm của học sinh cần xoá");
    } else {
        int result = JOptionPane.showConfirmDialog(null, "Bạn có muốn xoá điểm của học sinh này không?", "Xác nhận", JOptionPane.YES_NO_OPTION);
        if (result == JOptionPane.YES_OPTION) {
            String CourseTitle = tblStudentGrade.getModel().getValueAt(i, 1).toString();
            int IdPerson = Integer.parseInt(tblStudentGrade.getModel().getValueAt(i, 0).toString());
            try {
                if (sgBUS.DeleteStudentGrade(IdPerson, CourseTitle)) {
                    JOptionPane.showMessageDialog(null, "Xóa thành công");
                    readTableStudentGrade();
                } else {
                    JOptionPane.showMessageDialog(null, "Xóa thất bại");
                }
            } catch (SQLException ex) {
                Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
            }
        }
    }
}

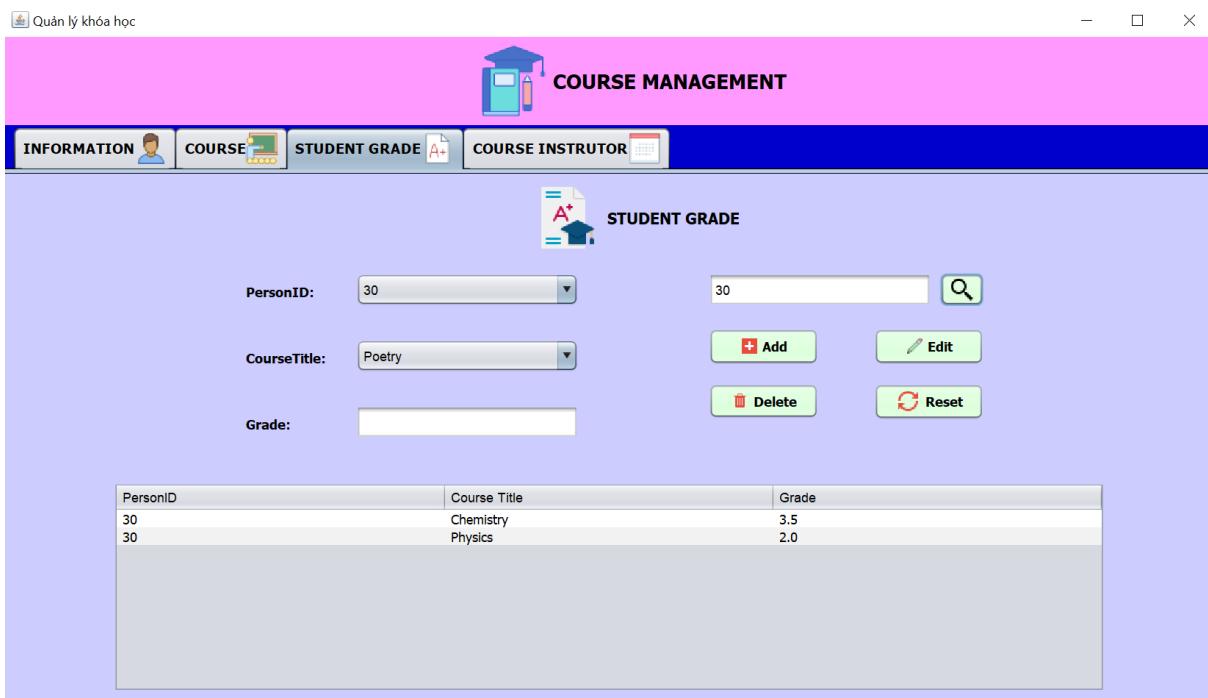
```

## 5. Xử lý 5: Tìm kiếm điểm bằng mã người học

### 5.1. Sơ đồ tuần tự



## 5.2. Giao diện



*Giao diện tìm kiếm điểm người học bằng mã người học*

## 5.3. Code 3 class:

❖ DAO

```

public ArrayList<StudentGradeDTO> FindByID(String PersonID) throws SQLException{
    ArrayList<StudentGradeDTO> StudentGradeList = new ArrayList<>();
    int personID = Integer.parseInt(PersonID);
    String sql = String.format("SELECT * FROM `studentgrade` WHERE (`StudentID` = '%d')", personID);
    System.out.println(sql);
    ResultSet rs = ConnectDatabase.GetInstance().ExecuteSELECT(sql);
    while (rs.next()) {
        StudentGradeDTO sgDTO = new StudentGradeDTO();
        sgDTO.setPersonID(rs.getInt("StudentID"));
        String sql2 = "select Title from course where CourseID = '" + rs.getInt("CourseID") + "'";
        System.out.println(sql2);
        ResultSet rs2 = ConnectDatabase.GetInstance().ExecuteSELECT(sql2);
        while (rs2.next()) {
            sgDTO.setCourseTitle(rs2.getString("Title"));
        }
        sgDTO.setGrade(rs.getFloat("Grade"));
        StudentGradeList.add(sgDTO);
    }
    return StudentGradeList;
}

```

## ❖ BUS

```

public ArrayList findByID(String personID) throws SQLException{
    return sgDAO.FindByID(personID);
}

```

## ❖ GUI

```

private void btnSearch5MouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    String input = txtSearchStudentGrade.getText();
    System.out.println(input);
    if (!input.isEmpty()) {
        try {
            ArrayList<StudentGradeDTO> listStudentGrade = sgBUS.findByID(input);
            if (!listStudentGrade.isEmpty()) {
                modelTableStudentGrade.setRowCount(0);
                for (StudentGradeDTO sgDTO : listStudentGrade) {
                    modelTableStudentGrade.addRow(new Object[]{
                        sgDTO.getPersonID(), sgDTO.getCourseTitle(), sgDTO.getGrade()
                    });
                }
                JOptionPane.showMessageDialog(null, "Tim thấy rồi");
            } else {
                JOptionPane.showMessageDialog(null, "Không thể tìm thấy");
            }
        } catch (SQLException ex) {
            Logger.getLogger(Index.class.getName()).log(Level.SEVERE, null, ex);
        }
    } else {
        JOptionPane.showMessageDialog(null, "Thiếu Thông Tin");
    }
}

```