

**ĐẠI HỌC QUỐC GIA TP.HCM**  
**TRƯỜNG ĐẠI HỌC BÁCH KHOA**  
**KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH**



**BÁO CÁO BÀI TẬP LỚN**  
**Môn : Lập trình hướng đối tượng**

**Giáo viên hướng dẫn :**

Thầy Nguyễn Văn Hiệp

Thầy Trịnh Văn Giang

**Sinh viên thực hiện :**

Nguyễn Thị Phụng                      1527032

Nguyễn Nhật Quang                      1527033

TP.HCM, THÁNG 11 NĂM 2016

# **BÁO CÁO**

## **BÀI TẬP LỚN**

**GVTH:** Nguyễn Văn Hiệp

Trịnh Văn Giang

**Giờ học:** Thứ 3 - Tiết 8,9

**Nhóm:** LO2

**Tên ứng dụng:** Learning Assistant

**Sinh viên:**

- |                      |         |
|----------------------|---------|
| 1. Nguyễn Thị Phụng  | 1527032 |
| 2. Nguyễn Nhật Quang | 1527033 |

## Mục Lục

<b>I.</b>	<b>GIỚI THIỆU VỀ LEARNING ASSISTANT .....</b>	<b>4</b>
<b>II.</b>	<b>ĐẶC TẢ CHƯƠNG TRÌNH .....</b>	<b>4</b>
<b>III.</b>	<b>PHÂN TÍCH .....</b>	<b>4</b>
<b>IV.</b>	<b>XÂY DỰNG CHƯƠNG TRÌNH.....</b>	<b>10</b>
<b>1.</b>	<b>Hiện thực các chức năng liên quan đến tài khoản người dùng.....</b>	<b>10</b>
<b>1.1.</b>	<b>Xử lý chức năng login .....</b>	<b>10</b>
<b>1.2.</b>	<b>Xử lý chức năng register.....</b>	<b>12</b>
<b>2.</b>	<b>Hiện thực các chức năng thêm gói câu hỏi mới.....</b>	<b>15</b>
<b>3.</b>	<b>Hiện thực các chức năng liên quan đến việc thực hiện bài test.....</b>	<b>16</b>
<b>3.1.</b>	<b>Xử lý chức năng làm bài test.....</b>	<b>16</b>
<b>3.2.</b>	<b>Xử lý chức năng thông báo kết quả đến người dùng .....</b>	<b>21</b>
<b>V.</b>	<b>TÀI LIỆU THAM KHẢO .....</b>	<b>22</b>
<b>VI.</b>	<b>KẾT LUẬN.....</b>	<b>22</b>

## Danh mục hình ảnh

Hình 1:	Giao diện đăng nhập vào phần mềm .....	4
Hình 2:	Giao diện thông báo login không thành công .....	5
Hình 3:	Giao diện đăng ký tài khoản người dùng .....	5
Hình 4:	Giao diện quản lý các gói câu hỏi của người dùng.....	6
Hình 5:	Giao diện thêm gói câu hỏi.....	6
Hình 6:	Giao diện thông tin chi tiết gói .....	7
Hình 7:	Giao diện thay đổi thông tin gói câu hỏi .....	7
Hình 8:	Giao diện thực hiện bài test .....	8
Hình 9:	Giao diện thông báo câu trả lời đúng/sai.....	8
Hình 10:	Giao diện thông báo kết quả .....	9
Hình 11:	Class diagram của chương trình .....	10
Hình 12:	Flow chart chức năng thực hiện bài test.....	18

## I. GIỚI THIỆU VỀ LEARNING ASSISTANT

Learning Assistant là phần mềm hỗ trợ người dùng học thuộc bài thông qua hình thức tạo các câu hỏi trắc nghiệm dựa trên dữ liệu input của người dùng. Phần mềm sẽ bỏ qua những câu người dùng đã trả lời đúng, và ưu tiên hỏi lại những câu người dùng trả lời chưa đúng. Phần mềm cho phép người dùng lựa chọn số lần muốn lặp lại của một câu hỏi trả lời đúng nhằm mục đích hỗ trợ việc thuộc bài tốt hơn.

## II. ĐẶC TẢ CHƯƠNG TRÌNH

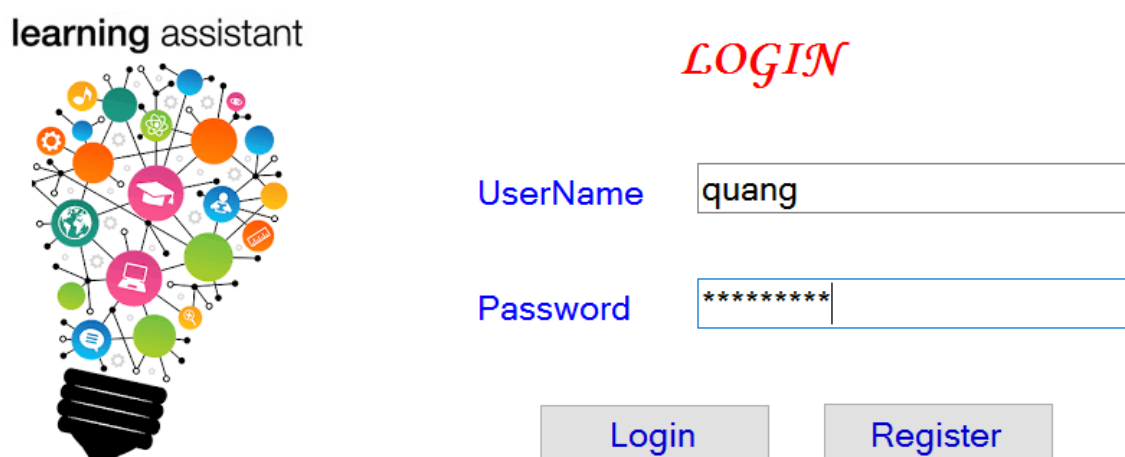
Learning Assistant cung cấp các chức năng chính như sau:

1. Người dùng đăng ký tài khoản và đăng nhập vào ứng dụng với tài khoản đã đăng ký.
2. Quản lý các gói câu hỏi theo chủ đề.
3. Thêm gói câu hỏi mới.
4. Thay đổi thông tin hoặc xóa gói câu hỏi đã có.
5. Thực hiện bài test với bộ câu hỏi đã chọn, thông báo kết quả đạt được cho người dùng.

## III. PHÂN TÍCH

Để thực hiện được các chức năng trên Learning Assistant cung cấp các giao diện với các chức năng cơ bản như sau:

### 1. Đăng nhập



learning assistant

**LOGIN**

UserName quang

Password \*\*\*\*\*

Login Register

*Hình 1: Giao diện đăng nhập vào phần mềm*

Người dùng sẽ đăng nhập vào phần mềm bằng cách điền user name và password vào giao diện Hình 1 và click Login.

Nếu thông tin đăng nhập sai phần mềm sẽ hiện lên giao diện thông báo như sau:

*Login fail*

Your user name or password were invalid, please try again

Try again

*Hình 2: Giao diện thông báo login không thành công*

Hoặc trường hợp chưa có tài khoản người dùng click vào Register để đăng ký một tài khoản mới.

## 2. Đăng ký tài khoản người dùng

**REGISTER**

UserName *	<input type="text" value="admin"/>
Password *	<input type="password" value="****"/>
Confirm Password *	<input type="password" value="****"/>
Email *	<input type="text" value="meo@kute.com"/>

*Hình 3: Giao diện đăng ký tài khoản người dùng*

Người dùng đăng ký tài khoản để truy cập vào ứng dụng cần cung cấp các thông tin: user name, password, email liên lạc. Đây là những thông tin bắt buộc, sau khi điền đầy đủ và đúng điều kiện phần mềm sẽ cho phép người dùng đăng ký tài khoản và đăng nhập với tài khoản đã tạo. Click Login để đăng nhập với giao diện Hình 1

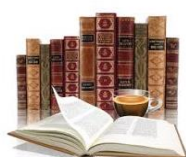
### 3. Quản lý các gói câu hỏi



Welcome: quang

*WELCOME TO LEARNING ASSISTANT*

Literature



History



Chemistry



Animal



NEW PACKAGE

*Hình 4: Giao diện quản lý các gói câu hỏi của người dùng*

Sau khi đăng nhập thành công, phần mềm sẽ hiển thị lên giao diện quản lý các gói câu hỏi như Hình 3. Tại giao diện này, người dùng có thể thêm gói câu hỏi mới bằng cách click vào New Package. Đồng thời, để xem chi tiết một gói câu hỏi bất kỳ, người dùng có thể click vào tên gói bất kỳ để xem.

### 4. Thêm gói câu hỏi mới

#### *Add New Package*

Import new file

Import

Topic

English

File name:

Name

English

Number of word:

You can get template at current program folder \template\import\_template.xls

SAVE

BACK

*Hình 5: Giao diện thêm gói câu hỏi*

Tại giao diện này, người dùng import một file gói câu hỏi mới, tùy biến topic và name cho gói mới thêm vào. Tất nhiên input gói câu hỏi tuân theo phải định dạng

mà chương trình quy định, người dùng lần đầu có thể lấy template theo đường link chỉ dẫn trên Hình 4. Click Save để lưu lại thay đổi và Back để ra lại giao diện Hình 3.

## 5. Xem thông tin gói câu hỏi



Welcome: quang

### Literature

Topic:	Literature	Nearly learning time:	11/4/2016
Number of word:	5	Completion time:	30 minutes

BEGIN

EDIT

BACK

Hình 6: Giao diện thông tin chi tiết gói

Các thông tin chi tiết về gói câu hỏi được hiện ra khi người dùng click vào một gói câu hỏi bất kỳ ở Hình 3. Tại đây người dùng có thể bắt đầu bài test gói câu hỏi bằng cách click vào BEGIN, thay đổi thông tin gói câu hỏi bằng cách click vào EDIT, hay trở lại giao diện Hình 3 khi click BACK.

## 6. Thay đổi thông tin gói



### Edit Package 4

Topic	<input type="text" value="English"/>	Name	<input type="text" value="Animal"/>
-------	--------------------------------------	------	-------------------------------------

DELETE

BACK

Hình 7: Giao diện thay đổi thông tin gói câu hỏi

Với giao diện này người dùng có thể thay đổi các thông tin về topic và tên của gói câu hỏi hoặc xóa gói câu hỏi này ra khỏi tài khoản của mình.

## 7. Thực hiện bài test



**English We Can**

*Literature*

**START** **EXIT**

Câu hỏi: "Qua đèo ngang" là tác phẩm của ai

☐ A Hồ Xuân Hương ☐ C Nguyễn Du

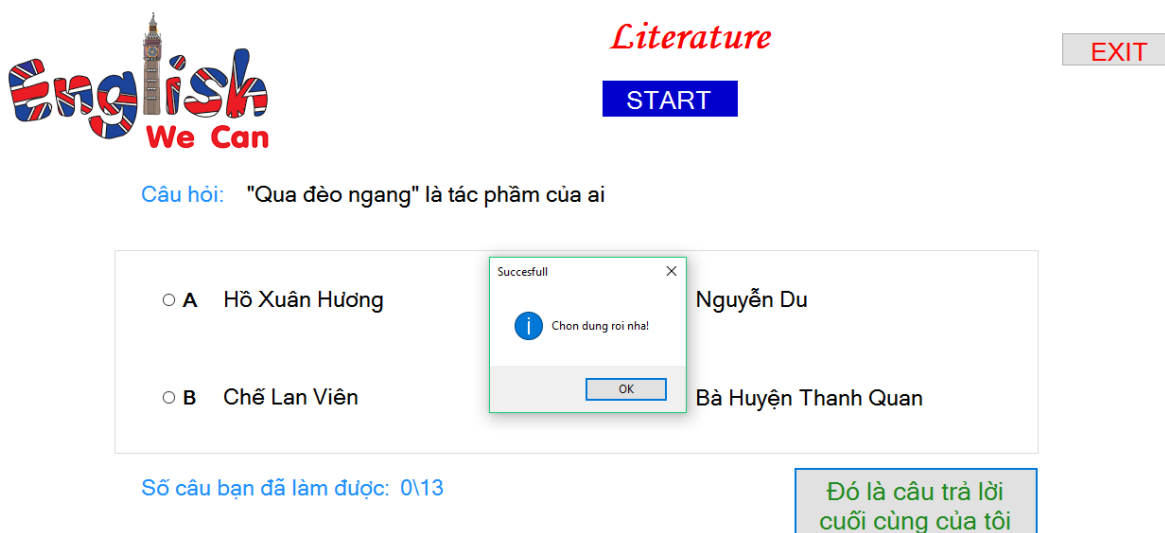
☐ B Chế Lan Viên ☐ D Bà Huyện Thanh Quan

Số câu bạn đã làm được: 0/13

Đó là câu trả lời cuối cùng của tôi

Hình 8: Giao diện thực hiện bài test

Khi click BEGIN ở giao diện Hình 5, giao diện Hình 7 sẽ xuất hiện. Tại đây, người dùng click START để bắt đầu bài test của mình, các câu hỏi và đáp án sẽ load lên từ input đưa vào, người dùng chọn đáp án của mình và click “Đó là câu trả lời cuối cùng của tôi” để hoàn thành câu trả lời. Lúc này màn hình giao diện sẽ thông báo câu trả lời đúng hoặc sai như Hình 8.



**English We Can**

*Literature*

**START** **EXIT**

Câu hỏi: "Qua đèo ngang" là tác phẩm của ai

☐ A Hồ Xuân Hương ☐ C Nguyễn Du

☐ B Chế Lan Viên ☐ D Bà Huyện Thanh Quan

Số câu bạn đã làm được: 0/13

Đó là câu trả lời cuối cùng của tôi

Hình 9: Giao diện thông báo câu trả lời đúng/sai



Gói câu hỏi sẽ chỉ kết thúc khi người dùng trả lời đúng hết tất cả các câu hỏi trong gói câu hỏi, hoặc click EXIT để thoát ra.

## 8. Thông báo kết quả

### *RESULT*

Completion Time: 00h:02m:43s \ toàn bộ câu hỏi

Time Per Question(TPQ): 00h:00m:12s \ câu trả lời đúng

Probability Wrong Answer(PWA): 0 %

Target: TPQ < 10s and PWA < 10%

Try again

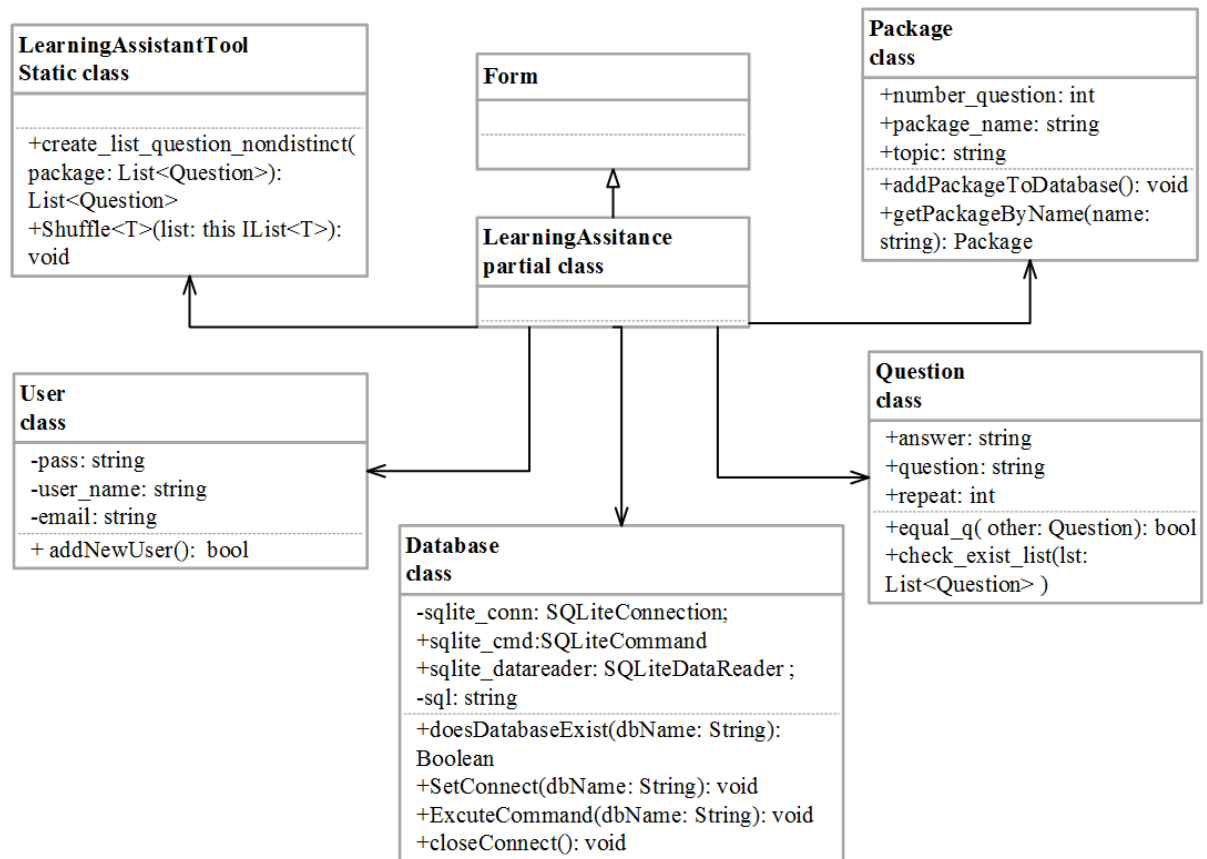
*Hình 10: Giao diện thông báo kết quả*

Sau khi hoàn thành bài test tại giao diện Hình 7, chương trình sẽ thông báo kết quả gồm tổng thời gian người dùng hoàn thành, thời gian trung bình trả lời một câu hỏi và tỷ lệ trả lời sai của người dùng. Tại đây, người dùng có thể thực hiện lại bài test bằng cách click Try again.

Sau khi phân tích giao diện và chức năng chính của chương trình, nhận thấy những công việc chính cần xử lý như sau:

- ✓ Từ input cần tạo thành bộ gói câu hỏi trắc nghiệm với số lần lặp lại các câu hỏi như input người dùng chọn.
- ✓ Cần một giải thuật để trộn câu hỏi cũng như các đáp án hiển thị random.
- ✓ Lưu lại các user account của người dùng đăng ký vào ứng dụng trên database.
- ✓ Quản lý dữ liệu các gói câu hỏi
- ✓ Kết nối với database chương trình
- ✓ Đổ dữ liệu từ input lên form giao diện.

Từ những phân tích nêu trên, chương trình xây dựng class diagram như sau:



Hình 11: Class diagram của chương trình

## IV. XÂY DỰNG CHƯƠNG TRÌNH

### 1. Hiện thực các chức năng liên quan đến tài khoản người dùng

#### 1.1. Xử lý chức năng login

Hàm xử lý sự kiện cho button Login tại giao diện Hình 1 được hiện thực như sau:

```

private void btnLogin_Click(object sender, EventArgs e)
{
    Database UserAccount = new Database();
    UserAccount.SetConnect("user.sqlite");
    List<string> userNameList = new List<string>();
    List<string> passList = new List<string>();
    UserAccount.Sql = "select user_name from user";
    UserAccount.ExcuteCommand("user.sqlite");
    while (UserAccount.sqlite_datareader.Read())
    {
        userNameList.Add(Convert.ToString(UserAccount.sqlite_datareader[
            "user_name"]));
    }
    UserAccount.Sql = "select pass from user";
    UserAccount.ExcuteCommand("user.sqlite");
    while (UserAccount.sqlite_datareader.Read())
    {
        passList.Add(Convert.ToString(UserAccount.sqlite_datareader["pas
s"]));
    }
}
  
```

```

        UserAccount.closeConnect();
        Boolean checkInfomation = false;
        for (int i = 0; i < userNameList.Count(); i++)
        {
            if (userNameList[i] == txtUserName.Text && passList[i] ==
                txtPassword.Text)
            {
                checkInfomation = true;
                break;
            }
        }
        if (checkInfomation == true)
        {
            welcome.Visible = true;
            changeButton(btnPack1);
            changeButton(btnPack2);
            changeButton(btnPack3);
            changeButton(btnPack4);
            login.Visible = false;
            register.Visible = false;
            label_welcom.Text += txtUserName.Text;
            label_welcome2.Text += txtUserName.Text;
        }
        else
        {
            invalid.Visible = true;
            login.Visible = false;
        }
    }
}

```

Khi người dùng click vào button Login: phần mềm sẽ tạo một kết nối đến database chứa dữ liệu quản lý các account user, kiểm tra user name và password mới nhập tại giao diện Hình 1 có trong dữ liệu hay không. Nếu user name và password phù hợp sẽ mở giao diện Hình 3 ngược lại sẽ mở giao diện Hình 2.

Để kết nối đến database ở đây có sử dụng đối tượng class Database, được hiện thực như sau:

```

class Database
{
    // Here you define properties: OK
    SQLiteConnection sqlite_conn;
    public SQLiteCommand sqlite_cmd;
    public SQLiteDataReader sqlite_datareader;
    string sql;

    public string Sql
    {
        get
        {
            return sql;
        }

        set
        {
            sql = value;
        }
    }
}

```

```

public Boolean doesDatabaseExist(String dbName)
{
    string curFile = @"data\" + dbName;
    if (File.Exists(curFile))
        return true;
    else
        return false;
}
public void SetConnect(String dbName)
{
    if (doesDatabaseExist(dbName) == false)
    {
        sqlite_conn = new SQLiteConnection("Data Source=" + "data\\" +
        dbName + ";Version=3; New=True;");
    }
    else
    {
        sqlite_conn = new SQLiteConnection("Data Source=" + "data\\" +
        dbName + ";Version=3;");
    }
    sqlite_conn.Open();
}
public void ExcuteCommand(String dbName)
{
    sqlite_cmd = new SQLiteCommand(sql, sqlite_conn);
    sqlite_datareader = sqlite_cmd.ExecuteReader();
}
public void closeConnect()
{
    sqlite_conn.Close();
}
}

```

## 1.2. Xử lý chức năng register

```

private void btnReg_Register_Click(object sender, EventArgs e)
{
    Regex reg = new Regex(@"[a-z0-9!#$%&'*/+=?^_`{|}~-]+(?:\.[a-z0-9!#$%&'*/+=?^_`{|}~-]+)*@(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?"); //Object initialization for Regex
    if (txt_reg_username.Text == "" || txt_reg_pass.Text == "" ||
        txt_reg_conf_pass.Text == ""
        || txt_reg_email.Text == "")
    {
        MessageBox.Show("Please fill all require information", "Erro",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else if (txt_reg_pass.TextLength < 8)
    {
        MessageBox.Show("Passwords must be at least 8 characters in
            length", "Erro",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else if (txt_reg_pass.Text != txt_reg_conf_pass.Text)
    {
        MessageBox.Show("Confirm Password was invalid", "Erro",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else if (!reg.IsMatch(txt_reg_email.Text))

```

```

    {
        MessageBox.Show("Email was invalid", "Error",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    else
    {
        User new_user = new User(txt_reg_username.Text,
            txt_reg_pass.Text, txt_reg_email.Text);
        bool add_user_success = new_user.addNewUser();
        if (add_user_success)
        {
            register.Visible = false;
            login.Visible = true;
        }
    }
}

```

Khi người dùng click button Register: phần mềm sẽ kiểm tra các input điền vào các text box đúng format quy định chưa, nếu chưa đúng sẽ hiện ra MesageBox thông báo đến người dùng để yêu cầu nhập lại, Khi tất cả các dữ liệu input phù hợp yêu cầu, phần mềm kết nối với dữ liệu database để kiểm tra user đã tồn tại hay chưa, nếu chưa tồn tại thì cho insert một user mới vào table database và thông báo đăng ký thành công, ngược lại sẽ thông báo user name tồn tại và yêu cầu đăng ký với tên khác.

Để kiểm tra các user đã tồn tại hay chưa, ở đây có sử dụng đối tượng của lớp User được hiện thực như sau:

```

class User
{
    private string user_name;
    private string pass;
    private string email;

    public string User_name {
        get
        {
            return user_name;
        }
        set
        {
            user_name = value;
        }
    }
    public string Pass {
        get
        {
            return pass;
        }
        set
        {
            pass = value;
        }
    }
    public string Email {
        get

```

```

        {
            return email;
        }
        set
        {
            email = value;
        }
    }

    public User(string user_name, string pass, string email)
    {
        this.user_name = user_name;
        this.pass = pass;
        this.email = email;
    }

    public bool addNewUser()
    {
        bool res = false;
        Database UserAccount = new Database();
        UserAccount.Sql = "create table if not exists user (user_name
        varchar(50), pass varchar(50), email varchar(50))";
        UserAccount.SetConnect("user.sqlite");
        UserAccount.ExcuteCommand("user.sqlite");

        //check user exited
        List<string> userList = new List<string>();
        UserAccount.Sql = "select user_name from user where user_name = '" +
        this.user_name + "'";
        UserAccount.ExcuteCommand("user.sqlite");
        while (UserAccount.sqlite_datareader.Read())
        {

            userList.Add(Convert.ToString(UserAccount.sqlite_datareader["user_n
            ame"]));
        }
        //UserAccount.sqlite_datareader.Close();
        if (userList.Count() == 0)
        {
            UserAccount.Sql = "insert into user (user_name, pass, email)
            values ('"
            + this.user_name + "','" + this.pass + "','" + this.email +
            "')";
            UserAccount.ExcuteCommand("user.sqlite");
            UserAccount.closeConnect();
            MessageBox.Show("You have successfully registered and click
            \\OK\\ to logged in", "Successfull",
            MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
            res = true;
            //register.Visible = false;
            //login.Visible = true;
        }
        else
        {
            MessageBox.Show("Username was exited, please try again",
            "Error",
            MessageBoxButtons.OK, MessageBoxIcon.Error);
        }
        return res;
    }
}

```

## 2. Hiện thức các chức năng thêm gói câu hỏi mới

Người dùng tạo gói câu hỏi mới bằng cách import từ file excel danh sách các từ mà họ muốn học. Dữ liệu trong file excel theo một template có sẵn gồm 3 cột như sau:

Key	Value	Repeat
-----	-------	--------

Trong đó:

- Key: Nội dung câu hỏi.
- Value: Nội dung câu trả lời
- Repeat: Số lần người dùng muốn hỏi lại câu hỏi tương ứng

Ngoài ra người dùng còn cần nhập vào các thông tin về: số câu hỏi muốn chọn trong file, tên của gói câu hỏi, tên của chủ đề.

Sau khi người dùng điền đầy đủ các thông tin thì có thể bấm SAVE để thực hiện tạo gói câu hỏi.

Chương trình dưới sự hỗ trợ của thư viện excel sẽ đọc file được người dùng import vào rồi tạo cơ sở dữ liệu để lưu lại nội dung cho các lần học sau.

Cụ thể được hiện thực như bên dưới:

```
private void button10_Click(object sender, EventArgs e)
{
    //store new package
    //store question list
    Package newPackage = new Package(label_package_name.Text,
    Int32.Parse(input_num_question.Text), label_topic.Text);
    newPackage.addPackageToDatabase();

    SQLiteConnection.CreateFile("data\\question\\data_" +
    label_package_name.Text + ".sqlite");

    m_dbConnection = new SQLiteConnection("Data Source=" +
    "data\\question\\data_" + label_package_name.Text +
    ".sqlite;Version=3;");
    m_dbConnection.Open();
    sql = "create table " + "data_" + label_package_name.Text + "_table
    (key varchar(20), value varchar(20), repeat int)";
    command = new SQLiteCommand(sql, m_dbConnection);
    command.ExecuteNonQuery();
    Excel.Application xlApp;
    Excel.Workbook xlWorkBook;
    Excel.Worksheet xlWorkSheet;

    xlApp = new Excel.Application();
    xlWorkBook = xlApp.Workbooks.Open(filePath, 0, true, 5, "", "",
    true,
    Excel.XlPlatform.xlWindows, "\t", false, false, 0, true, 1, 0);
}
```

```

xlWorkSheet = (Excel.Worksheet)xlWorkBook.Worksheets.get_Item(1);

int num_question = Int32.Parse(input_num_question.Text);

//one first in excel
for (int i = 2; i < num_question+2; i++)
{
    key = xlWorkSheet.Cells[i, 1].Value.ToString();
    value = xlWorkSheet.Cells[i, 2].Value.ToString();
    repeat = xlWorkSheet.Cells[i, 3].Value.ToString();
    sql = "insert into " + "data_" + label_package_name.Text +
    "_table (key, value , repeat) values ('" + key + "','" + value +
    "','" + repeat + "')";
    command = new SQLiteCommand(sql, m_dbConnection);
    command.ExecuteNonQuery();
}

xlWorkBook.Close(true, null, null);
xlApp.Quit();

Marshal.ReleaseComObject(xlWorkSheet);
Marshal.ReleaseComObject(xlWorkBook);
Marshal.ReleaseComObject(xlApp);
m_dbConnection.Close();
MessageBox.Show("Packet create succesfull", "Succesfull",
MessageBoxButtons.OK, MessageBoxIcon.Information);
}

```

### 3. Hiện thực các chức năng liên quan đến việc thực hiện bài test

#### 3.1. Xử lý chức năng làm bài test

```

private void chosed_butoon_Click(object sender, EventArgs e)
{
    total_question_asked++;
    //y
    right_ans = answer.FindIndex(a => a == question_chose.answer);
    if (radio_A.Checked) index_chosed = 0;
    if (radio_B.Checked) index_chosed = 1;
    if (radio_C.Checked) index_chosed = 2;
    if (radio_D.Checked) index_chosed = 3;

    if(index_chosed == right_ans)
    {
        MessageBox.Show("Đúng rồi nha!", "Succesfull",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        list_question_nondistinct.Remove(question_chose);
        current_right_answer++;
        current_complete_question.Text = current_right_answer.ToString()
        + "\\\" + total_point.ToString();
    } else
    {
        MessageBox.Show("Sai mất rồi!", "Succesfull",
        MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }
    //first_question = false;
    if (list_question_nondistinct.Count == 0)
    {
        MessageBox.Show("You was clean package!", "Succesfull",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        result.Visible = true;
    }
}

```



```

begin.Visible = false;
stopWatch.Stop();
TimeSpan ts = stopWatch.Elapsed;
result_completion_time.Text =
String.Format("{0:00}h:{1:00}m:{2:00}s", ts.Hours, ts.Minutes,
ts.Seconds) + result_completion_time.Text;

TimeSpan ts_per_question = new TimeSpan(ts.Ticks / total_point);
result_sec_per_question.Text =
String.Format("{0:00}h:{1:00}m:{2:00}s", ts_per_question.Hours,
ts_per_question.Minutes, ts_per_question.Seconds) +
result_sec_per_question.Text;

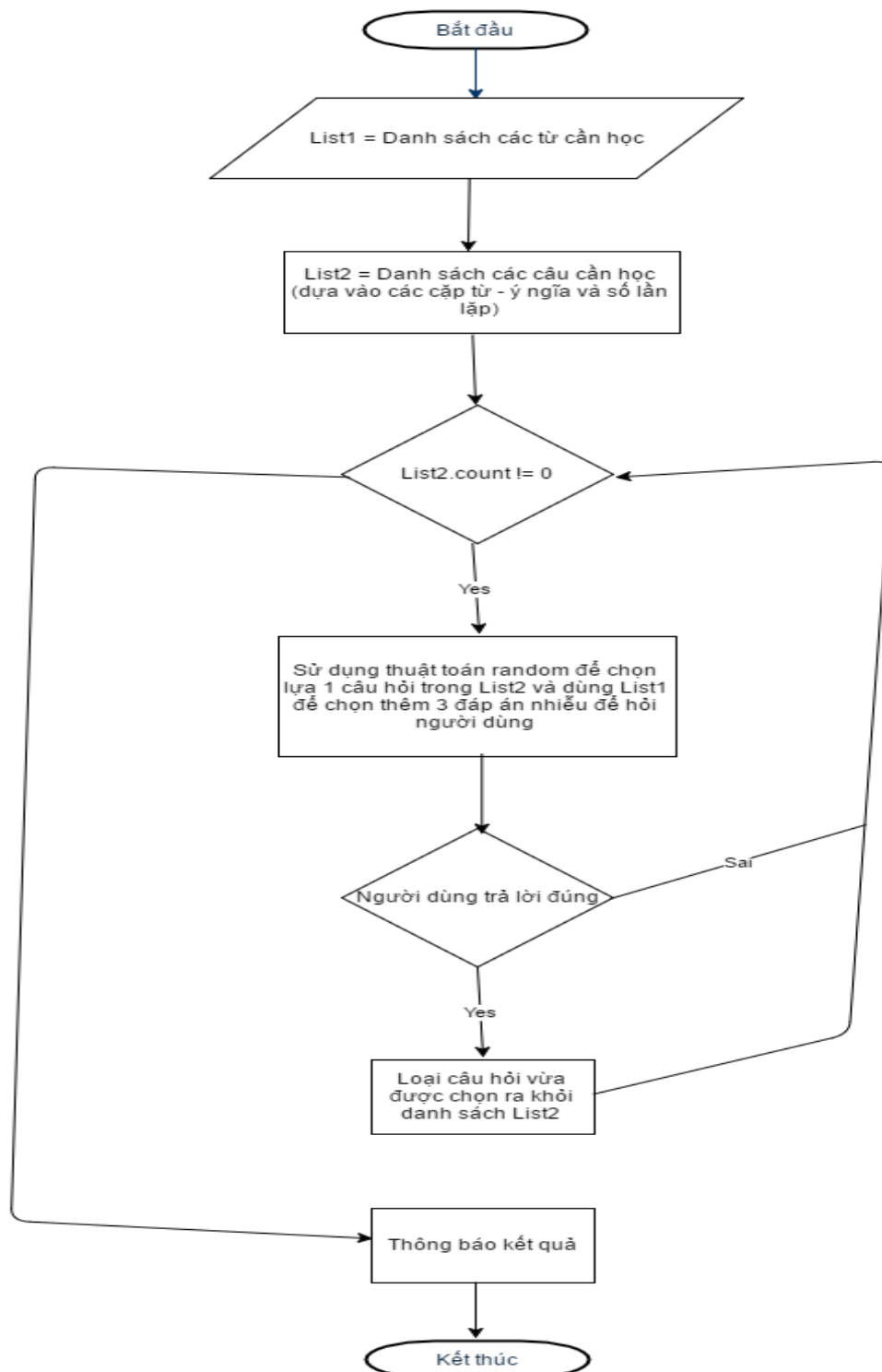
double p_res = (total_question_asked - total_point) * 100 /
total_question_asked;
p_wrong.Text = Math.Round(p_res) + p_wrong.Text;
total_question_asked = 0;
return;
}

while (true)
{
    list_question_nondistinct.Shuffle();
    question_chose = list_question_nondistinct.ElementAt(0);
    if (list_question_nondistinct.Distinct().Count() == 1) break;
    if (!question_chose.equal_q(last_question_chose)) break;
    //kiem tra truong hop list chi con 1 loai cau hoi
}
last_question_chose = list_question_nondistinct.ElementAt(0);
package.Shuffle();
//create list answer
answer = new List<string>();
answer.Add(package.ElementAt(0).answer);
answer.Add(package.ElementAt(1).answer);
answer.Add(package.ElementAt(2).answer);
if (answer.Exists(item => item.Equals(question_chose.answer,
StringComparison.Ordinal)))
    answer.Add(package.ElementAt(3).answer);
else
    answer.Add(question_chose.answer);

answer.Shuffle();
question_content.Text = question_chose.question;
ans_A.Text = answer.ElementAt(0);
ans_B.Text = answer.ElementAt(1);
ans_C.Text = answer.ElementAt(2);
ans_D.Text = answer.ElementAt(3);
}

```

Chức năng làm bài kiểm tra được thực hiện ở 2 điểm khi người dùng bấm nút START và nút chọn câu trả lời cuối cùng. Thuật toán chủ yếu của phần này như hình vẽ bên dưới:



Hình 12: Flow chart chức năng thực hiện bài test

Các class được sử dụng trong phần hiện thực chức năng này bao gồm: class Package, Question và LearningAssistantTool được hiện thực như sau:

```
class Package
{
    public string package_name;
    public int number_question;
    public string topic;

    public Package(string package_name, int number_question, string topic)
    {
        this.package_name = package_name;
        this.number_question = number_question;
        this.topic = topic;
    }

    public void addPackageToDatabase()
    {
        SQLiteConnection m_dbConnection = new SQLiteConnection("Data
Source=data\\list_package.sqlite;Version=3;");
        m_dbConnection.Open();
        string sql = string.Format("insert into list_package values ('{0}',
'{1}', '{2}')" , package_name, number_question, topic);
        SQLiteCommand command = new SQLiteCommand(sql, m_dbConnection);
        command.ExecuteNonQuery();
    }

    public static Package getPackageByName(string name)
    {
        SQLiteConnection m_dbConnection = new SQLiteConnection("Data
Source=data\\list_package.sqlite;Version=3;");
        m_dbConnection.Open();
        string sql = string.Format("select * from list_package where
package_name = '{0}'", name);
        SQLiteCommand command = new SQLiteCommand(sql, m_dbConnection);
        SQLiteDataReader reader = command.ExecuteReader();
        string pac_name = "";
        string topic = "";
        int num_ques = 0;
        while (reader.Read())
        {
            pac_name = reader["package_name"].ToString();
            topic = reader["topic"].ToString();
            num_ques = Convert.ToInt32(reader["number_question"]);
        }

        return new Package(pac_name, num_ques, topic);
    }
}
```

```

class Question
{
    public string question;
    public string answer;
    public int repeat;
    public Question(string question, string answer, int repeat)
    {
        this.question = question;
        this.answer = answer;
        this.repeat = repeat;
    }
    public Question()
    {
        question = "";
        answer = "";
        repeat = 0;
    }
    public bool equal_q(Question other)
    {
        return other.question.Equals(question, StringComparison.Ordinal)
            && other.answer.Equals(answer, StringComparison.Ordinal)
            && other.repeat == repeat;
    }
    public bool check_exist_list(List<Question> lst)
    {
        bool res = false;
        foreach (var item in lst)
        {
            if (equal_q(item)) { res = true; break; }
        }
        return res;
    }
}

static class LearningAssistantTool
{
    public static void Shuffle<T>(this IList<T> list)
    {
        RNGCryptoServiceProvider provider = new RNGCryptoServiceProvider();
        int n = list.Count;
        while (n > 1)
        {
            byte[] box = new byte[1];
            do provider.GetBytes(box);
            while (!(box[0] < n * (Byte.MaxValue / n)));
            int k = (box[0] % n);
            n--;
            T value = list[k];
            list[k] = list[n];
            list[n] = value;
        }
    }
    public static List<Question>
    create_list_question_nondistinct(List<Question> package)
    {
        List<Question> res = new List<Question>();
        foreach (var item in package)
        {
            for (int i = 0; i < item.repeat; i++) res.Add(item);
        }
        return res;
    }
}

```

### 3.2. Xử lý chức năng thông báo kết quả đến người dùng

```
private void chosed_butoon_Click(object sender, EventArgs e)
{
    total_question_asked++;
    ...
    if (list_question_nondistinct.Count == 0)
    {
        MessageBox.Show("You was clean package!", "Succesfull",
            MessageBoxButtons.OK, MessageBoxIcon.Information);
        result.Visible = true;
        begin.Visible = false;
        stopWatch.Stop();
        TimeSpan ts = stopWatch.Elapsed;
        result_completion_time.Text =
            String.Format("{0:00}h:{1:00}m:{2:00}s", ts.Hours, ts.Minutes,
                ts.Seconds) + result_completion_time.Text;

        TimeSpan ts_per_question = new TimeSpan(ts.Ticks / total_point);
        result_sec_per_question.Text =
            String.Format("{0:00}h:{1:00}m:{2:00}s", ts_per_question.Hours,
                ts_per_question.Minutes, ts_per_question.Seconds) +
            result_sec_per_question.Text;

        double p_res = (total_question_asked - total_point) * 100 /
            total_question_asked;
        p_wrong.Text = Math.Round(p_res) + p_wrong.Text;
        total_question_asked = 0;
        return;
    }
}
```

Phần này có chức năng thông báo kết quả cho người dùng, sau khi hoàn thành số câu trả lời đúng đã đặt ra. Kết quả gồm 3 phần:

1. Thời gian làm bài. Được tính bằng cách kích hoạt stopWatch khi bắt đầu nhất nút START, và dừng stopWatch khi không còn câu hỏi nào cần hoàn thành nữa. Biến `TimeSpan ts = stopWatch.Elapsed`; giữ giá trị này, sau đó trình bày ra cho người dùng.
2. Thời gian làm bài trên mỗi câu trả lời đúng. Chỉ số này được đo bằng tổng thời gian làm bài chia cho số câu hỏi mục tiêu.
3. Xác suất trả lời đúng. Chỉ số này được đo lường bằng tổng số câu hỏi được hỏi trừ đi tổng số câu hỏi mục tiêu rồi tất cả chia cho tổng số câu hỏi.

Các chỉ số ở phần 2 và 3 có thể giúp người dùng theo dõi việc học của mình. Nếu suy nghĩ quá lâu thì chỉ số thời gian làm trên câu sẽ cao. Nếu đánh không suy nghĩ thì chỉ số xác suất trả lời đúng sẽ cao. Vì thế ta có thể dùng 2 chỉ số này để đặt mục tiêu cho bản thân, hoặc cho người khác (giáo viên đặt mục tiêu cho học sinh).

## **V. TÀI LIỆU THAM KHẢO**

1. <http://stackoverflow.com/>
2. <http://www.ibm.com/developerworks/rational/library/content/RationalEdge/sep04/bell/>
3. Giáo trình Lập trình hướng đối tượng thầy Nguyễn Văn Hiệp đại học Bách khoa Hồ Chí Minh

## **VI. KẾT LUẬN**

Chương trình còn một số chức năng chưa hoàn thiện như việc quản lý nhóm câu hỏi, thay đổi thông tin tài khoản người dùng..., nhóm phát triển sẽ cố gắng hiện thực những chức năng còn thiếu sót trong tương lai. Với những ý tưởng và những chức năng cơ bản đã trình bày như trên, phần mềm này sẽ giúp người dùng học bài trắc nghiệm hiệu quả hơn từ chính những bộ câu hỏi mà họ cung cấp. Để hiểu rõ hơn về cách sử dụng phần mềm, xin mời xem clip demo đính kèm.