**TRƯỜNG ĐẠI HỌC PHENIKAA**

**KHOA CÔNG NGHỆ THÔNG TIN**

**--------------------**



**BÁO CÁO BÀI TẬP LỚN**

**HỌC PHẦN “*CÔNG NGHỆ .NET”***

**Đề tài:**

**Lập trình xây dựng trò chơi ghép tranh**

**sử dụng công nghệ Dot Net**

**GVHD: ThS. Nguyễn Thành Trung**

**Sinh viên thực hiện: 19010044 – Đỗ Minh Vượng**

**19010016 – Nguyễn Trung Kiên**

**19010029 – Nguyễn Thị Thanh**

Hà Nội, 24/07/2021

MỤC LỤC:

**I. Lời nói đầu**

**II. Tổng quan về công nghệ .NET và framework WPF**

1. Tổng quan về .NET

a. .Net là gì

b. Ưu điểm của .NET

c. Các ngôn ngữ lập trình trên .NET

2. Tổng quan về WPF

a. WPF là gì

b. Một số tính năng nổi trội của WPF

c. Thành phần của WPF

**III. Trò chơi Ghép tranh**

1. Mục tiêu

2. Nền tảng

3. Các bước thực hiện dự án

4. Xây dựng chương trình

**IV. Kết luận**

**V. Tài liệu tham khảo**

**I. Lời nói đầu**

Trong thế giới công nghệ hiện nay, nền công nghệ máy tính phát triển rất nhanh kéo theo sự phát triển của các ngôn ngữ lập trình hiện đại giúp cho các chương trình, phần mềm máy tính của chúng ta ngày càng trở lên mạnh mẽ hơn. Song song với đó là sự tồn tại của các nền tảng lập trình hiện đại, tối ưu nhằm giúp người lập trình viên tiết kiệm thời gian xây dựng, triển khai và chạy các ứng dụng, chương trình của mình.

Việc nắm được nền tảng lập trình cho bản thân rất quan trọng, nó giúp chúng ta định hướng chính xác cách thức tổ chức xây dựng chương trình và sử dụng các ngôn ngữ lập trình thích hợp. Chính vì vậy nên việc lựa chọn công nghệ, nền tảng của người lập trình viên sẽ quyết định mực độ thành công trong công việc của bản thân sau này. Vậy nên trong học phần “Công nghệ .NET” lần này, để nâng cao hiểu biết, kinh nghiệm bản thân cũng như để rèn luyện kỹ năng lập trình và làm việc nhóm, chúng em xin chọn chủ đề “Xây dựng chương trình trò chơi Ghép tranh” để thực hiện.

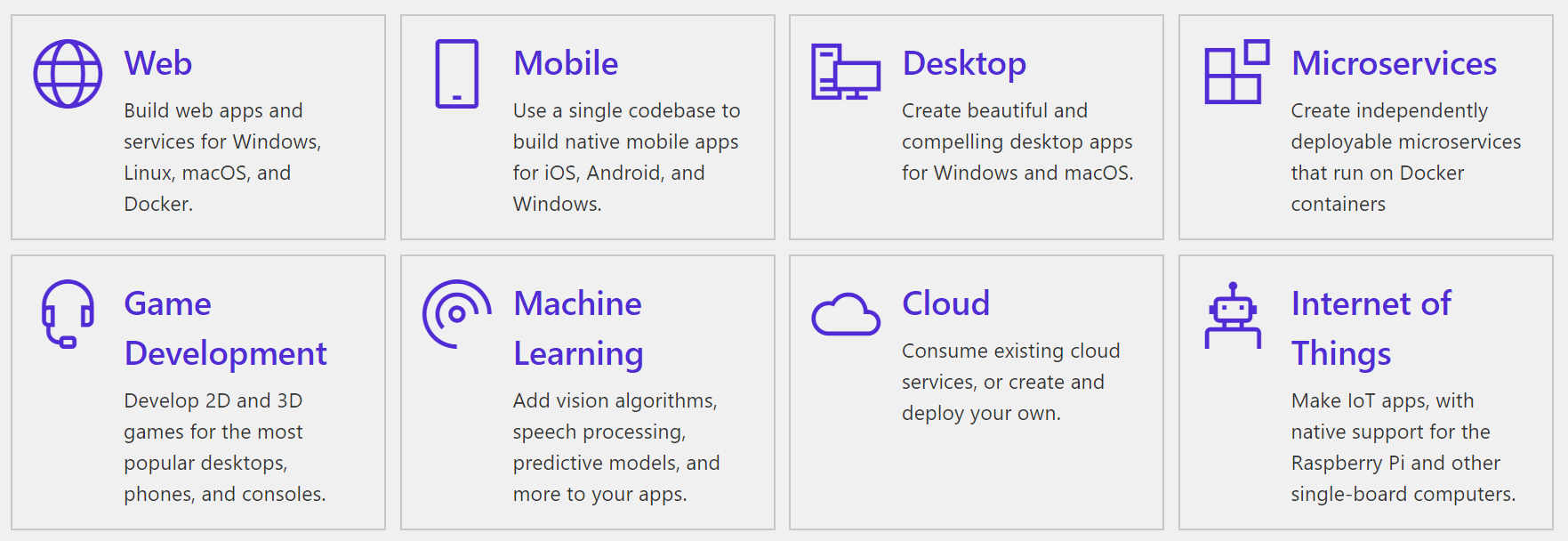
Để hoàn thành được bài tập lớn lần này, chúng em xin chân thành cảm ơn thày giáo đã giảng dạy học phần này là thày Nguyễn Thành Trung, Giảng viên khoa Công Nghệ Thông Tin trường Đại học PHENIKAA – thày đã rất nhiệt tình giảng dạy và hướng dẫn, chỉ bảo chi tiết mặc cho điều kiện giảng dạy còn nhiều khó khăn và bất tiện, chúng em xin cảm ơn thày.

**II. Tổng quan về công nghệ .NET và framework WPF**

**1. Tổng quan về .NET**

**a. .NET là gì?**

Công nghệ .NET (dot NET) là tên gọi chung của một loạt các công nghệ phát triển ứng dụng của Microsoft hiện đang được sử dụng vô cùng rộng rãi trên thế giới. Nó được dùng để xây dựng các ứng dụng cho desktop, web, cloud, các ứng dụng điện thoại, video game, IoT và các chương trình AI.



***.NET có thể được dùng để xây dựng ứng dụng cho hầu hết mọi thứ***

1. **Ưu điểm của công nghệ .NET**

Nền tảng này có nhiều ưu điểm nổi bật, sau đây là một số các ưu điểm có thể kể đến:

* **Thư viện lập trình lớn:** .NET sở hữu những thư viện lập trình rất lớn, có khả năng hỗ trợ tối đa cho việc tạo lập, xây dựng các ứng dụng web, truy cập, kết nối các cơ sở dữ liệu, cấu trúc dữ liệu, lập trình giao diện,...
* **Năng suất làm việc cao:** Lập trình, thiết kế ứng dụng với .NET tiết kiệm rất nhiều thời gian bởi nó cung cấp sẵn khá nhiều thành phần dùng trong thiết kế.
* **Biến đổi linh hoạt nhờ Loose Coupling:** .NET được thiết kế, xây dựng với khả năng biến đổi linh hoạt nhờ cấu trúc loose coupling. Điều này đem lại nhiều lợi thế về năng suất.
* **Đa ngôn ngữ:** .NET là nền tảng hỗ trợ cho đa ngôn ngữ. NGười lập trình viên có thể sử dụng nền tảng này để xây dựng các ứng dụng web bằng nhiều ngôn ngữ lập trình khác nhau mà vẫn đảm bảo khả năng tích hợp của nó.
* **Bảo mật cao:** .NET có phần kiến trúc bảo mật được thiết kế theo dạng từ dưới lên. Điều này giúp bảo vệ dữ liệu cũng như các ứng dụng khỏi các mối đe dọa thông qua mô hình bảo mật tân tiến evidence-based.
* **Tận dụng các dịch vụ có sẵn trong hệ điều hành:** Windows sở hữu rất nhiều dịch vụ có khả năng hoạt động trên mọi nền tảng như truy cập dữ liệu, mô hình dạng đối tượng thành phần, giao diện người dùng tương tác, bảo mật tích hợp và cả giám sát giao dịch. .NET tận dụng những dịch vụ này để đơn giản hóa cách sử dụng, giúp lập trình trên nền tảng này dễ dàng hơn.



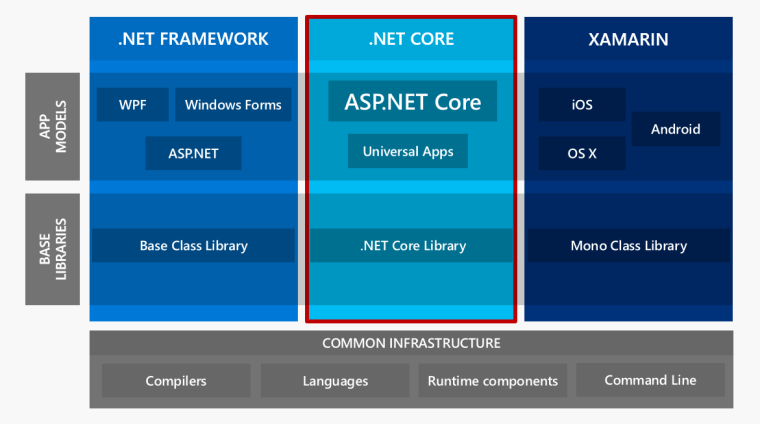
**c. Các ngôn ngữ lập trình trên .NET**

.NET hỗ trợ nhiều ngôn ngữ lập trình khác nhau, có thể kể đến các ngôn ngữ cần chú ý sau:

* **C#**: là một ngôn ngữ lập trình đơn giản, hiện đại, hướng đối tượng và thuộc loại an toàn.
* **F#**: là ngôn ngữ lập trình đa nền tảng, mã nguồn mở và cung cấp các chức năng lập trình cho .NET. Ngôn ngữ này cũng bao gồm lập trình hướng đối tượng và lập trình mệnh lệnh.
* **Visual Basic**: là một ngôn ngữ dễ tiếp cận với cú pháp đơn giản. Thường được dùng để xây dựng các ứng dụng loại an toàn và hướng đối tượng.

.NET sẽ giúp bạn xử lý phần công việc nặng nhọc thông qua các triển khai đi kèm:

* **.NET Core** là triển khai đa nền tảng của .NET, nó dùng cho các trang web, máy chủ và console app trên Windows, Linux và MacOS.
* **.NET Framework** hỗ trợ các trang web, dịch vụ và ứng dụng dành cho desktop và trên Windows.
* **Xamarin/Mono** là triển khai của .NET được dùng để chạy các ứng dụng trên tất cả các hệ điều hành di động chính.

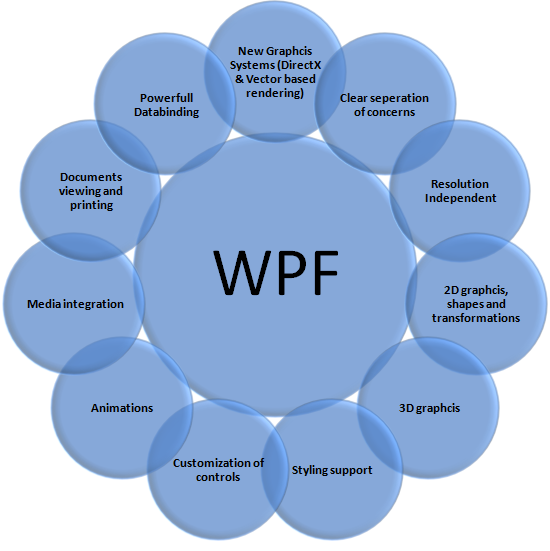


**2. Tổng quan về WPF**

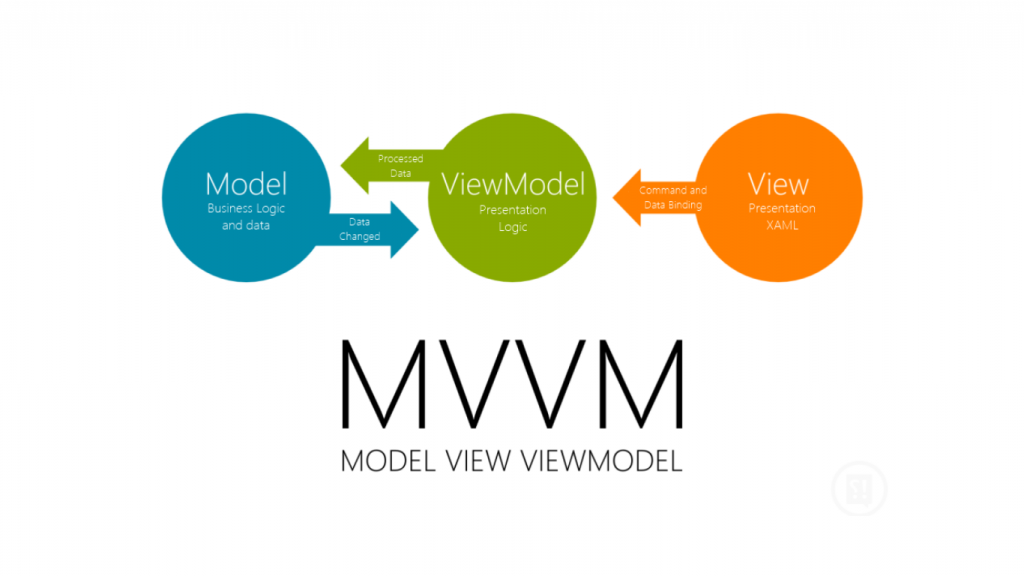
**a.WPF là gì?**

Windows Presentation Foundation (WPF) ra đời cùng với .NET 3.5 là một framework mới và rất hiện đại dành cho phát triển ứng dụng desktop cho Windows trên .NET Framework. Nó được ưa chuộng và sử dụng rất rộng rãi ngày nay. WPF hoàn toàn đơn giản hóa việc lập trình GUI (giao diện đồ họa), hỗ trợ thiết kế giao diện trực quan, đồng thời nhận được sự hỗ trợ rất tốt từ các hãng thứ ba bên ngoài (Devexpress, Syncfusion, ...) và cộng đồng người sử dụng.

Framework này sử dụng DirectX để tạo ra giao diện với khả năng xử lý đồ họa rất mạnh cho phép tạo ra những giao diện hiện đại, đẹp, mượt mà và những hiệu ứng tân tiến, hiện đại. Ngoài ra, WPF còn sử dụng ngôn ngữ riêng, XAML, để biểu diễn giao diện, cho phép các UI/UX designer hoạt động độc lập với các coder.



Ngoài các mẫu kiến trúc đã quen thuộc từ trước, WPF còn có một mẫu kiến trúc riêng là Model – View – Viewmodel (MVVM).



***Mẫu kiến trúc mới MVVM***

Cơ chế Data binding của WPF mạnh hơn nhiều so với Winforms, giúp giải quyết vấn đề hiển thị dữ liệu rất hiệu quả, ổn định và an toàn.

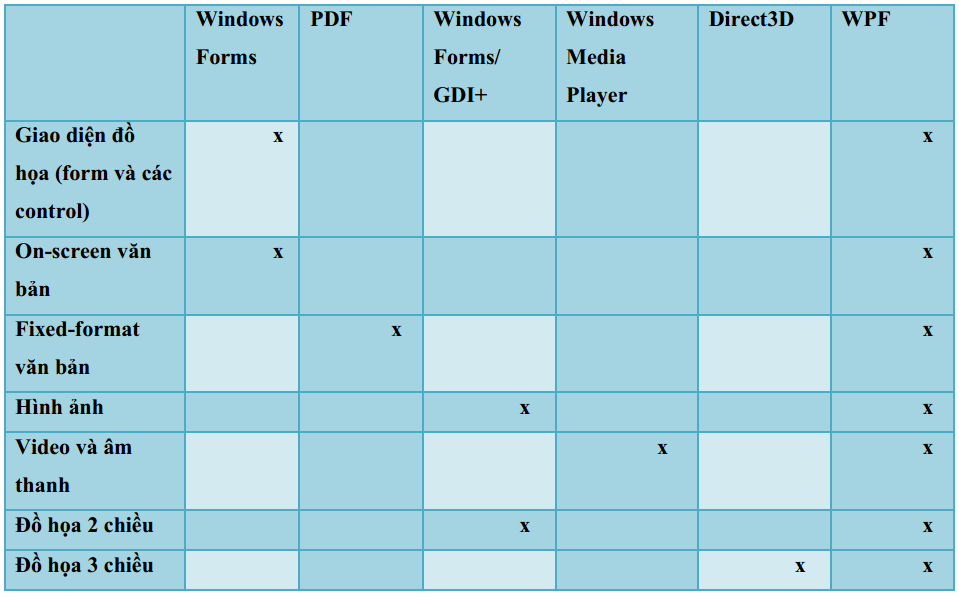
**b.Một số tính năng nổi trội của WPF**

WPF mang đến nhiều ưu việt, giúp cho việc lập trình trở lên đơn giản hơn từ thiết kế giao diện, xử lý dữ liệu giúp cho việc lập trình dễ dàng và mạnh mẽ hơn rất nhiều. Cùng với nền tảng .NET 4.5 hỗ trợ Web API (trước đó, .NET 3.5 dùng WCF, hay Web Service cho .NET trước đó nữa), giúp chúng ta dễ dàng xây dựng các ứng dụng điện toán đám mây.

WPF hỗ trợ hiệu ứng dựa trên thời gian thực thi của chương trình. Các hiệu ứng đơn giản có thể xử lý bằng việc quản lý thời gian chạy, còn các xử lý phức tạp hơn cần đến sự hỗ trợ của lớp Animation.

* Tất cả các thuộc tính của đối tượng trong WPF đều có thể được xử lý để trở nên sinh động hơn.
* Các lớp quản lý hiệu ứng tùy theo loại của thuộc tính được xử lý.

Các tính năng nổi trội của WPF có thể thể hiện thông qua bảng so sánh đơn giản sau:



***Tính năng nổi trội của WPF***

Tuy nhiên WPF vẫn tồn tại một số hạn chế nhất định, nhược điểm lớn nhất của WPF là phức tạp, khó học, khó thành thạo (nếu so với Windows Forms). Để phát huy hiệu quả thực sự của WPF đòi hỏi người người lập trình phải sử dụng mẫu thiết kế MVVM xây dựng riêng cho WPF, thành thạo XAML, hiểu cơ chế Data Binding, có khả năng thiết kế, và rất nhiều kỹ thuật nâng cao khác.

**b.Thành phần của WPF**

WPF tổ chức các chức năng theo một nhóm namespace cùng trực thuộc namespace System.Windows.

Cấu trúc cơ bản của mọi ứng dụng WPF đều gần như nhau. Là ứng dụng Windows độc lập hay là một XBAP, một ứng dụng WPF điển hình bao giờ cũng gồm một tập các trang XAML và phần code tương ứng được viết bằng C# hoặc Visual Basic, còn gọi là các file code-behind.

Mặc dù WPF cung cấp một nền tảng thống nhất để tạo giao diện người dùng, những công nghệ mà WPF chứa đựng vẫn có thể phân chia thành những thành phần độc lập.

**III. Trò chơi Ghép tranh**

**1. Mục tiêu**

Tìm hiểu cách xây dựng và chạy một chương trình trò chơi đơn giản sử dụng .NET Framework và áp dụng phương pháp lập trình hướng đối tượng OOP. Qua đó nâng cao hiểu biết và kỹ năng lập trình cho bản thân.

Ta sẽ sử dụng ngôn ngữ C#, chạy và lập trình trên Visual Studio. Lưu ý, ta cũng có thể sử dụng các ngôn ngữ lập trình trong .NET và hỗ trợ OOP khác.

**2. Nền tảng**

Hệ điều hành: Windows / MAC / Linux

.NET Framework 4.7 trở lên

Microsoft Visual Studio 2017 hoặc 2019

SQLite hoặc các hệ quản trị cơ sở dữ liệu khác.

Ngôn ngữ C#

Tải Visual Studio [tại đây](https://visualstudio.microsoft.com/)

Tải SQLite [tại đây](https://www.sqlite.org/index.html)

Tải .NET Framework [tại đây](https://dotnet.microsoft.com/download/dotnet-framework)

**3. Các bước thực hiện dự án**

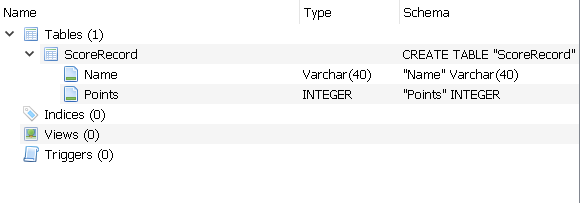
* Tạo một project mới sử dụng .NET Framework
* Tùy chỉnh SQL Server (nếu sử dụng, tuy nhiên trong bài này sẽ sử dụng SQLite):
  + Chuyển chế độ mặc định sang SQL Server Authentication
  + Chuyển permission người dùng của bạn sang Grant, server role để những mục public và serveradmin
  + Tùy chỉnh kết nối server trong SQL Server Configuration Manager
* Tạo database cho chương trình

Tên database: Playerdata

* + - Bảng ScoreRecord: dùng để lưu tên người chơi và số điểm đạt được

-Name (varchar)

-Points (Integer)

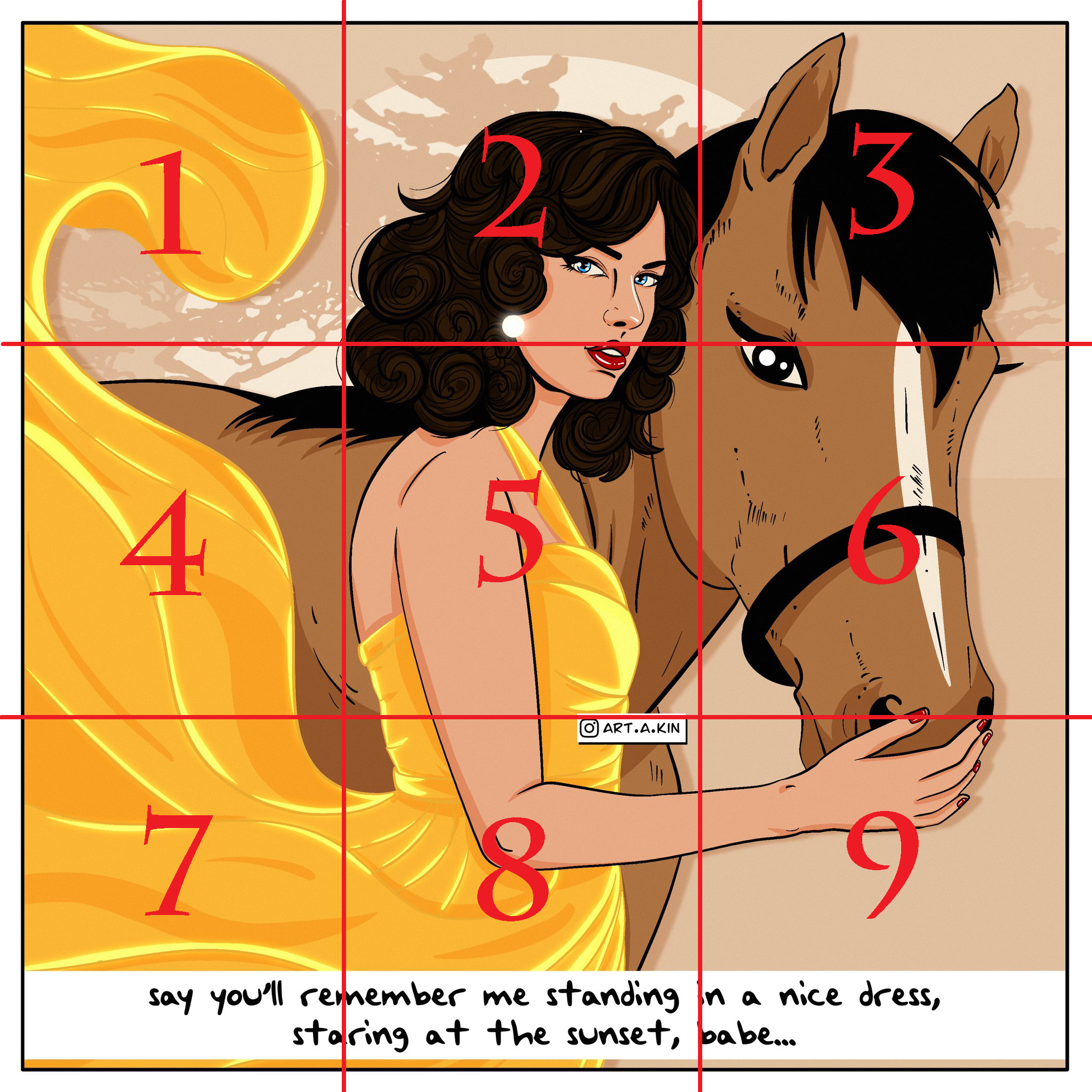


* Xuất databse vừa tạo ra một file .db để đưa vào project
* Chuẩn bị 3 hình ảnh để sử dụng cho 3 màn chơi

**4. Chương trình quản lý thông tin nhân viên**

**Tổng quan:**

* Giao diện gọn gàng, khoa học, dễ nhìn.
* Cho phép người chơi chọn độ khó (beginner, advance, expert).
* Chia hình ảnh thành các phần bằng nhau: chế độ dễ 3x3, trung bình 4x4 và đối với khó là 5x5. Các phần sau khi tách sẽ được đặt tên theo thứ tự.



***Minh họa mức độ dễ 3x3***

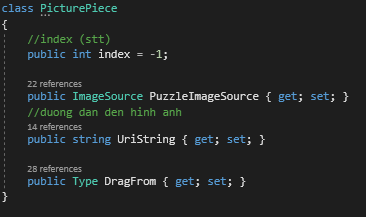
* Người chơi có thể chọn một ảnh từ máy tính, hệ thống sẽ tự chia ảnh và đặt tên, tạo một màn chơi mới.
* Màn chơi sẽ gồm 2 phần chính: 1 listbox để chứa các mảnh hình ảnh đã được xáo chộn, các canvas để chứa các mảnh người chơi kéo vào.
* Sau khi hoàn thành sẽ tính điểm dựa vào số lần di chuyển của người chơi sau đó lưu tên và điểm người chơi vào file database, các bản ghi này có thể xuất hiện tại mục điểm cao.

**Các class quan trọng:**

* **PicturePiece: mảnh ảnh (tranh):**

Thuộc tính và đối tượng:

* index: index của 1 mảnh ảnh (đã cắt)
* PuzzleImageSource: chứa mảnh ảnh
* Uristring: đường dẫn
* DragFrom: để xác định người chơi di chuyển ảnh từ đâu (listbox hoặc canvas)



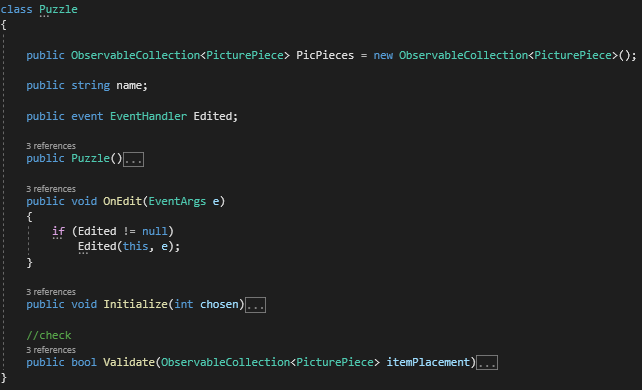
* **Puzzle: class màn chơi:**

Thuộc tính và đối tượng:

* PicPieces: một collection các mảnh ảnh (đã cắt)
* Name:

Phương thức:

* + - Puzzle: constructor
    - OnEdit: dùng để xác định mỗi khi người chơi di chuyển
    - Initialize: khởi tạo và xáo chộn các mảnh ảnh
    - Validate: kiểm tra xem người chơi đã hoàn thành màn chơi chưa
    - Edited: event mỗi khi người chơi di chuyển



* **Tạo một class *conn* để khởi tạo kết nối tới SQL Server**
* import java.sql.Connection;  
  import java.sql.DriverManager;  
  import java.sql.SQLException;  
  import java.sql.Statement;  
    
  public class conn {  
   public Connection conn;  
   public Statement stmt;  
    
   public conn(){  
   try{  
   Class.*forName*("com.microsoft.sqlserver.jdbc.SQLServerDriver");  
    
   conn = DriverManager.*getConnection*("jdbc:sqlserver://DESKTOP-693S9N0\\SQLEXPRESS:1433;databaseName=Employee","VuongDAneil", "Vuongdaneil88");  
    
   stmt = conn.createStatement();  
   } catch (ClassNotFoundException | SQLException e) {  
   e.printStackTrace();  
   }  
   }  
  }
* **Một class *Login* để đăng nhập**
  + 3 button Đăng nhập, Thoát hoặc Quên mật khẩu
  + Tên đăng nhập và mật khẩu có trong database -> thành công -> chuyển đến giao diện chính phần mềm
  + Quên mật khẩu -> chuyển đến màn hình đặt lại mật khẩu
* public void actionPerformed(ActionEvent e){  
    
   if (e.getSource() == logbt) {  
   try {  
   conn c1 = new conn();  
   String username = t1.getText();  
   String pass = t2.getText();  
    
   String query = "SELECT \* FROM account WHERE username = '" + username + "' AND password = '" + pass + "'";  
    
   ResultSet rs = c1.stmt.executeQuery(query);  
   if (rs.next()) {  
   new Functions().f.setVisible(true);  
   f.setVisible(false);  
   } else {  
   JOptionPane.*showMessageDialog*(null, "Invalid login, try again");  
   f.setVisible(true);  
   }  
   } catch (SQLException throwables) {  
   throwables.printStackTrace();  
   }  
   }  
    
   if (e.getSource() == forgetbt){  
   new Forget\_pass();  
   }  
    
   if (e.getSource() == cancelbt){  
   f.setVisible(false);  
   }  
  }
* ***Forget\_pass* để đặt lại mật khẩu sử dụng adminID**
  + Sử dụng mã admin để xác thực -> nếu đúng thì đặt lại mật khẩu mới
  + Cập nhật mật khẩu mới vào database của admin đó
* public void actionPerformed(ActionEvent e) {  
    
   conn con = new conn();  
    
   if (e.getSource() == searchbt){  
   try {  
   String username = tuser.getText();  
   String id = tadid.getText();  
    
   String query = "SELECT \* FROM account WHERE username = '"+ username +"' AND adminID = '"+ id +"'";  
    
   ResultSet rs = con.stmt.executeQuery(query);  
   if(rs.next()){  
    
   lbnewpass.setVisible(true);  
   lbconfirm.setVisible(true);  
   tnewpass.setVisible(true);  
   tconfirm.setVisible(true);  
    
   submitbt.setVisible(true);  
   cancelbt.setVisible(true);  
    
   lbadminame.setText("admin " + username);  
   lbadminame.setVisible(true);  
    
    
   }else{  
   JOptionPane.*showMessageDialog*(null, "Invalid, try again");  
   f.setVisible(true);  
   }  
   } catch (SQLException throwables) {  
   throwables.printStackTrace();  
   }  
   }  
    
   //submit password  
   if (e.getSource() == submitbt){  
   try {  
   String id = tadid.getText();  
   String pass = tnewpass.getText();  
   String confirmpass = tconfirm.getText();  
    
   if (pass.equalsIgnoreCase(confirmpass)) {  
   String newquery = "UPDATE account SET password = '" + pass + "' WHERE adminID = '" + id + "'";  
   con.stmt.executeUpdate(newquery);  
    
   JOptionPane.*showMessageDialog*(null,"Done!");  
   f.setVisible(false);  
    
   } else JOptionPane.*showMessageDialog*(null, "Error, try again!");  
   } catch (SQLException throwables) {  
   throwables.printStackTrace();  
   }  
   }  
    
   if (e.getSource() == cancelbt) f.setVisible(false);  
  }
* **Tạo một class *Functions* - giao diện menu chính của phần mềm**
  + Gồm 4 button chính: tra cứu, thêm, xóa, sửa
* import javax.swing.\*;  
  import java.awt.\*;  
  import java.awt.event.ActionEvent;  
  import java.awt.event.ActionListener;  
    
  public class Functions implements ActionListener {  
    
   JFrame f;  
   JLabel l1,l2;  
   JButton Addbt, Viewbt, Rmbt, b4;  
    
   Functions(){  
   f = new JFrame("EMPLOYEE CENTER");  
   f.setBackground(Color.*WHITE*);  
   f.setLayout(null);  
    
   ImageIcon i1 = new ImageIcon(ClassLoader.*getSystemResource*("Asset/Funtionbg.png"));  
   Image bg = i1.getImage().getScaledInstance(700,500,1);  
   ImageIcon img2 = new ImageIcon(bg);  
   l1 = new JLabel(img2);  
   l1.setBounds(0,0,700,500);  
   l1.setLayout(null);  
   f.add(l1);  
    
   l2 = new JLabel("Employee Management");  
   l2.setBounds(420,20,300,40);  
   l2.setFont(new Font("serif", Font.*BOLD*, 25));  
   l2.setForeground(Color.*black*);  
   l1.add(l2);  
    
   //Add employee  
   Addbt= new JButton("New");  
   Addbt.setBounds(430,80,100,40);  
   Addbt.setFont(new Font("serif",Font.*BOLD*,15));  
   Addbt.addActionListener(this);  
   l1.add(Addbt);  
    
   //Show employee  
   Viewbt = new JButton("Info");  
   Viewbt.setBounds(550,80,100,40);  
   Viewbt.setFont(new Font("serif", Font.*BOLD*, 15));  
   Viewbt.addActionListener(this);  
   l1.add(Viewbt);  
    
   //Remove employee  
   Rmbt = new JButton("Remove");  
   Rmbt.setBounds(430,140,100,40);  
   Rmbt.setFont(new Font("serif",Font.*BOLD*,15));  
   Rmbt.addActionListener(this);  
   l1.add(Rmbt);  
    
   //Update employee  
   b4 = new JButton("Update");  
   b4.setBounds(550,140,100,40);  
   b4.setFont(new Font("serif",Font.*BOLD*,15));  
   b4.addActionListener(this);  
   l1.add(b4);  
    
   f.setVisible(true);  
   f.setSize(700, 500);  
   f.setLocation(330, 110);  
   }  
   public void actionPerformed(ActionEvent e) {  
    
   //New Employee  
   if (e.getSource() == Addbt){  
   new Add\_employee();  
   }  
    
   //Show employee  
   if (e.getSource() == Viewbt){  
   new View\_info();  
   }  
    
   //Remove employee  
   if (e.getSource() == Rmbt){  
   new Remove\_employee();  
   }  
    
   //Update employee  
   if (e.getSource() == b4){  
   new View\_info();  
   }  
   }
* ***View\_info để* tìm kiếm một nhân viên**
  + Một textfield để nhập id nhân viên cần tìm
  + 2 Button để tìm và cập nhật (b1 & b2)
  + Tìm thấy -> hiển thị cửa sổ thông tin
* public void actionPerformed(ActionEvent e) {  
   if (e.getSource() == b1){  
   f.setVisible(false);  
   new Employee\_info(t.getText());  
   }  
   if (e.getSource() == b2){  
   f.setVisible(false);  
   new Update\_employee(t.getText());  
   }  
  }
* ***Employee\_info* để hiển thị thông tin của một nhân viên**
  + Các label để hiển thị các thông tin của nhân viên
  + Id nhân viên lấy từ cửa sổ tìm kiếm trước đó (View\_info)

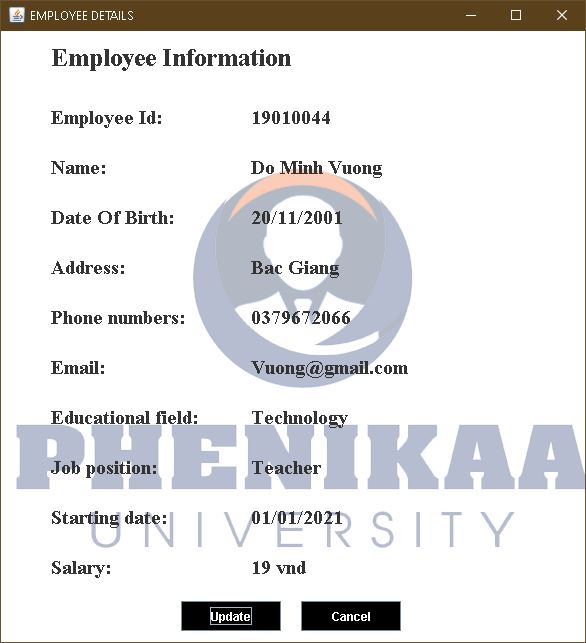
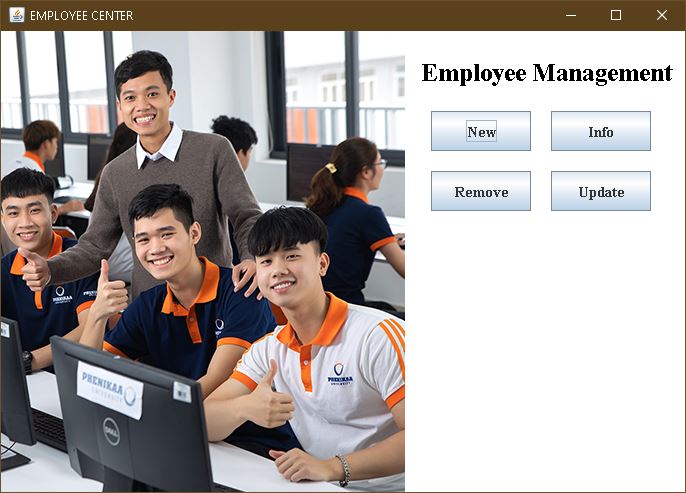
Employee\_info(String id){  
 try{  
 conn con = new conn();  
 String query = "SELECT \* FROM employee WHERE id ='"+ id +"'";  
 ResultSet rs = con.stmt.executeQuery(query);  
  
 while (rs.next()){  
  
 name = rs.getString("name");  
 salary = rs.getString("salary) + " vnd";  
 dob = rs.getString("dateofbirth");  
 address = rs.getString("address");  
 phone = rs.getString("phone");  
 mail = rs.getString("email");  
 edu = rs.getString("education");  
 pos = rs.getString("position");  
 sid = rs.getString("id");  
 start = rs.getString("startingdate");  
  
 }  
 } catch (SQLException throwables) {  
 throwables.printStackTrace();  
 }

* ***Add\_employee* để thêm nhân viên**
  + Gồm các label và textfield để điền thông tin nhân viên
  + 2 button để thêm hoặc hủy thêm (addbt & cancelbt)
  + Nếu thông tin hợp lệ -> thêm vào database
  + Nếu thông tin không hợp lệ -> yêu cầu nhập lại
* public void actionPerformed(ActionEvent e) {  
    
   String name = tname.getText();  
   String salary = tsalary.getText();  
   String dob = tdob.getText();  
   String addr = taddress.getText();  
   String phone = tphone.getText();  
   String mail = tmail.getText();  
   String edu = tedu.getText();  
   String pos = tpos.getText();  
   String id = tid.getText();  
   String date = tdate.getText();  
    
   if (e.getSource() == addbtn){  
   try {  
   conn conn = new conn();  
   String query = "INSERT INTO employee VALUES('" + name +"','"+ salary +"','"+ dob +"','"+ addr +"','"+ phone +"','"+ mail +"','"+ edu +"','"+ pos +"','"+ id +"','"+ date +"')";  
   conn.stmt.executeUpdate(query);  
   JOptionPane.*showMessageDialog*(null,"Done!");  
    
   f.setVisible(false);  
   new Functions();  
    
   } catch (SQLException throwables) {  
   throwables.printStackTrace();  
   JOptionPane.*showMessageDialog*(null,"Cant add, check the information again!");  
   }  
   }  
    
   else if (e.getSource() == cancelbtn){  
   f.setVisible(false);  
   }  
  }
* ***Remove\_employee* để xóa một nhân viên**
  + Một texfield để nhập id nhân viên -> tìm thấy nhân viên thì hiển thị thông tin ra màn hình
  + 3 button: tìm, xóa và hủy xóa (searchbt, rmbt, cancelbt)
* public void actionPerformed(ActionEvent e) {  
    
   conn con = new conn();  
    
   if (e.getSource() == searchbt){  
    
   id = t.getText();  
    
   try{  
   String query = "SELECT name,phone,email FROM employee WHERE id = '"+ id +"'";  
   ResultSet rs = con.stmt.executeQuery(query);  
    
   int i = 0;  
   if(rs.next()){  
   i = 1;  
    
   String name = rs.getString(1);  
   String mob = rs.getString(2);  
   String email=rs.getString(3);  
    
   lbname.setVisible(true);  
   lbphone.setVisible(true);  
   lbmail.setVisible(true);  
   rmbt.setVisible(true);  
   backbt.setVisible(true);  
   cancelbt.setVisible(true);  
    
   ename.setText(name);  
   enumber.setText(mob);  
   eemail.setText(email);  
   }  
    
   if(i == 0) JOptionPane.*showMessageDialog*(null,"Cant found employee, check the ID again!");  
    
   } catch (SQLException throwables) {  
   throwables.printStackTrace();  
   }  
   }  
    
   if (e.getSource() == rmbt){  
   try{  
   String query = "DELETE FROM employee WHERE id = '"+ id +"'";  
   con.stmt.executeUpdate(query);  
   JOptionPane.*showMessageDialog*(null,"Removed successfully!");  
    
   lbname.setVisible(false);  
   lbname.setVisible(false);  
   lbphone.setVisible(false);  
   lbmail.setVisible(false);  
    
   ename.setVisible(false);  
   enumber.setVisible(false);  
   eemail.setVisible(false);  
   rmbt.setVisible(false);  
   backbt.setVisible(false);  
    
    
   } catch (SQLException throwables) {  
   throwables.printStackTrace();  
   }  
   }  
    
   if(e.getSource() == cancelbt){  
   f.setVisible(false);  
   }  
    
   if(e.getSource() == backbt){  
   f.setVisible(false);  
   }  
    
  }
* ***Update-employee* để cập nhật thông tin một nhân viên**
  + Lấy id nhân viên từ cửa sổ tìm kiếm trước đó (View\_info)
  + Các label và texfield để nhập thông tin cần cập nhật
  + 2 button để cập nhật và hủy cập nhật (updatebt & cancelbt)
  + Phương thức hiển thị thông tin nhân viên hiện tại
* void Show\_detail(String agrid){  
   try {  
    
   String query = "SELECT \* FROM employee WHERE id = '" + agrid + "'";  
   ResultSet rs = con.stmt.executeQuery(query);  
    
   if (rs.next()) {  
   f.setVisible(true);  
   i = 1;  
    
   txname.setText(rs.getString(1));  
   txsalary.setText(rs.getString(2));  
   txdob.setText(rs.getString(3));  
   txaddress.setText(rs.getString(4));  
   txphone.setText(rs.getString(5));  
   txmail.setText(rs.getString(6));  
   txedu.setText(rs.getString(7));  
   txpos.setText(rs.getString(8));  
   txid.setText(rs.getString(9));  
   txdate.setText(rs.getString(10));  
   }  
    
   if (i == 0) {  
   JOptionPane.*showMessageDialog*(null, "ID not found, check again!");  
   }  
   } catch (SQLException e) {  
   e.printStackTrace();  
   }  
   f.setVisible(true);  
  }
  + Cập nhật hoặc hủy cập nhật
* public void actionPerformed(ActionEvent e) {  
   if (e.getSource() == updatebt && i == 1){  
   try{  
   String query = "UPDATE employee SET name='"+ txname.getText() +"',salary='"+ txsalary.getText() +"',dateofbirth='"+ txdob.getText() +"',address='"+ txaddress.getText() +"',phone='"+ txphone.getText() +"',email='"+ txmail.getText() +"',education='"+ txedu.getText() +"',position='"+ txpos.getText() +"',id='"+ txid.getText() +"',startingdate='"+ txdate.getText() +"' WHERE id='"+ eid +"'";  
   con.stmt.executeUpdate(query);  
   JOptionPane.*showMessageDialog*(null, "Update done!");  
    
   f.setVisible(false);  
   new View\_info();  
   } catch (SQLException throwables) {  
   throwables.printStackTrace();  
   }  
   }  
    
   if (e.getSource() == cancelbt){  
   f.setVisible(false);  
   }

**5. Đóng gói và chạy phần mềm**

* **Đóng gói**
  + Vào **Project Setting** -> **Artifacts** -> **Thêm** -> **JAR from modules** -> tích chọn **copy to the output directory and link via manifes ->** chọn mainclass là *Welcome\_page* (trang nền đầu tiên của ứng dụng) -> **OK**
  + Chọn **Build -> Build Artifacts -> Build**
* **Chạy chương trình**
* Mở **Command Prompt** (trên windows) -> di chuyển đến thư mục chứa file jar vừa build.
* Chạy **java –jar PhenikaaEmployeeManagement.jar**

**Một số hình ảnh chạy chương trình**



**IV. Kết luận**

Nhóm chúng em đã hoàn thành được bài tập lớn “Chương trình quản lý nhân viên sử dụng OOP”. Mặc dù chương trình còn đơn giản nhưng về cơ bản chúng em đã hoàn thành được phần thiết yếu của yêu cầu.

Trong quá trình thực hiện, do mới làm quen và chưa có tư duy cũng như kinh nghiệm lập trình tốt nên bài của nhóm vẫn còn nhiều hạn chế và thiếu xót. Rất mong nhận được những lời nhận xét và góp ý của thày và các bạn để chúng em lỗ lực hơn.

Sau cùng, chúng em xin chân thành cảm ơn thày Trần Đăng Hoan, Giảng viên khoa Công Nghệ Thông Tin đã nhiệt tình hướng dẫn, giảng dạy giúp chúng em hoàn thành được bài bào cáo này.

**IV. Tài liệu tham khảo**

1. **Object-Oriented Software Development Using UML** - Zhiming Liu
2. **Database Design and Implementation -** Edward Sciore