

# Bài tập 10

## Quản lý dự án phần mềm & Lập trình với giao diện đồ họa người dùng (GUI)

### Mục tiêu

- Thực hiện các bài tập (câu hỏi) về nội dung Quản lý dự án phần mềm
- Phân rã các chức năng đã xác định thành các nhiệm vụ (task) để xây dựng một kế hoạch dự án phần mềm đơn giản
- Lập trình với giao diện đồ họa người dùng:
  - o Làm quen với mô hình MVC các thành phần giao diện người dùng o Thiết kế giao diện GUI trên công cụ Netbeans
  - o Viết mã xử lý sự kiện

### Đánh giá

- Hoàn thành các bài tập về nội dung Quản lý dự án phần mềm, cơ bản nắm được nguyên lý, quy trình và các kỹ năng / kỹ thuật trong quản lý dự án
- Các nhóm sinh viên xây dựng được biểu đồ Cấu trúc phân chia công việc (Work Breakdown Structure - WBS)
- Hoàn thành bài thực hành lập trình với giao diện đồ họa người dùng

---

### Phần I:

#### Bài 1.1

- a) Dự án kết thúc khi? (chọn nhiều)
  1. Thiếu kinh phí
  2. Hết kinh phí trước thời hạn (Kết thúc thất bại)
  3. Không khả thi
  4. Quá hạn dự kiến (có làm tiếp cũng không có ý nghĩa gì)
  5. Hoàn thành mục tiêu đề ra, nghiệm thu (kết thúc tốt đẹp) trước thời hạn
  6. Người quản lý kém
- b) Một dự án phát triển phần mềm là thành công khi?
  1. Sản phẩm đáp ứng yêu cầu chất lượng
  2. Không vượt quá kinh phí dự kiến
  3. Hoàn thành trong thời gian dự kiến
  4. Tất cả các phương án trên
- c) ... là một yếu tố không biết trước mà khi nó xảy ra thì có thể ảnh hưởng tiêu cực hoặc tích cực đến việc hoàn thành các mục tiêu của dự án.
  1. Rủi ro dự án
  2. Lập kế hoạch dự án

- 
3. Quản lý nhân sự
  4. Yêu cầu người dùng

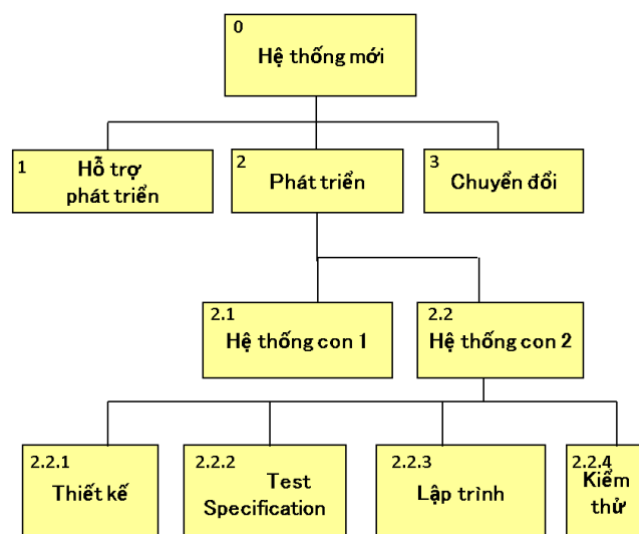
## Bài 1.2

- a) Chúng ta thấy rằng các hệ thống phần mềm lớn, phức tạp thường được phát triển bởi rất nhiều cá nhân, rất ít người có được bức tranh toàn cảnh về toàn bộ dự án. Vậy, đối với một người làm công, tham gia vào một dự án mà không biết về toàn bộ chức năng của dự án đó thì có hợp lý không? Vì sao?
- b) Nếu không áp dụng các mô hình vòng đời phần mềm thì có phát triển được phần mềm không? Tại sao?
- c) Trong phương pháp Agile, việc luôn có đại diện của khách hàng trong nhóm phát triển thì có ưu điểm gì?

## Phần II: Xây dựng biểu đồ WBS

---

- WBS là một phân rã phân cấp (hierarchical decomposition) của toàn bộ phạm vi công việc sẽ được nhóm dự án thực hiện để hoàn thành các mục tiêu của dự án và tạo ra các sản phẩm bàn giao được yêu cầu.
- Các thành phần phân rã ở mức cuối cùng – mức lá nên thoả mãn các tiêu chí:
  - Tình trạng / tính hoàn tất của công việc có thể đo được hoặc có sản phẩm cụ thể
  - Thời gian, tài nguyên / chi phí có thể ước lượng được o Thời gian hoàn thành công việc trong giới hạn
  - Công việc được phân công độc lập (nghĩa là công việc không bị ngừng giữa chừng để chờ kết quả của công việc khác) - Ví dụ:



- a) **Yêu cầu:** Xây dựng WBS đối với dự án phát triển phần mềm trong bài tập môn học. Có thể phân tích theo các chức năng nghiệp vụ hoặc theo các pha trong quá trình phát triển.

Trên cơ sở biểu đồ WBS phía trên hãy xây dựng một bản kế hoạch đơn giản như sau:

Công việc	Thời gian (số giờ làm việc)	Số người
1. Công việc 1 1.1 Công việc 1.1 1.2 Công việc 1.2 ...		
2. Công việc 2 2.1 Công việc 2.1 2.2 Công việc 2.2 ...		
...		

- b) **Yêu cầu:** Xây dựng một bảng quản lý đơn giản các rủi ro với dự án phát triển phần mềm trong bài tập môn học.

Ví dụ:

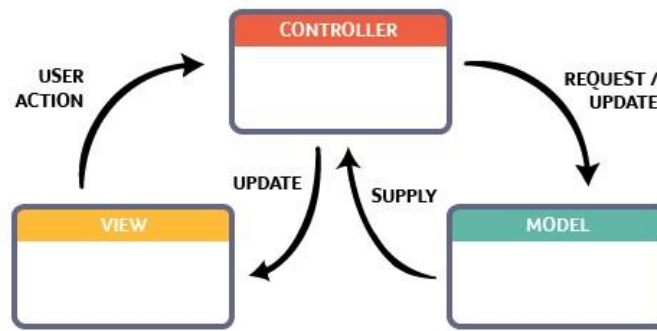
Công việc / Hoạt động	Xác định rủi ro			Quản lý rủi ro	
	Mối nguy	Rủi ro	Mức độ	Chiến lược	Biện pháp
Mua hàng	Hàng bị hư hại khi vận chuyển	Không có nguyên liệu sản xuất	Trung bình	Giảm thiểu	Xây dựng tồn kho tối thiểu

(Mức độ có thể đánh giá theo: thấp / trung bình / cao)

(Chiến lược có thể lựa chọn: Giảm thiểu rủi ro / Chuyển rủi ro / Loại bỏ rủi ro)

### Phần III: Thực hành lập trình giao diện đồ họa người dùng GUI

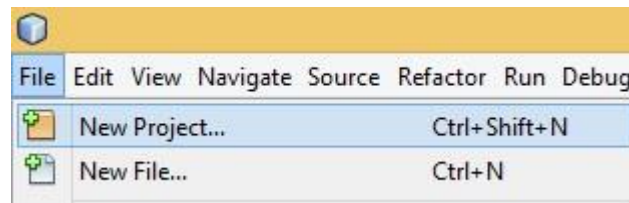
**Background: Model – View – Controller (MVC)** là một khuôn mẫu kiến trúc phần mềm. Mẫu thiết kế này cho phép phân tách giữa logic ứng dụng và giao diện người dùng. MVC giúp cho người phát triển phần mềm cô lập các xử lý nghiệp vụ và giao diện người dùng một cách rõ ràng hơn. Phần mềm phát triển theo mẫu MVC tạo nhiều thuận lợi cho việc bảo trì.



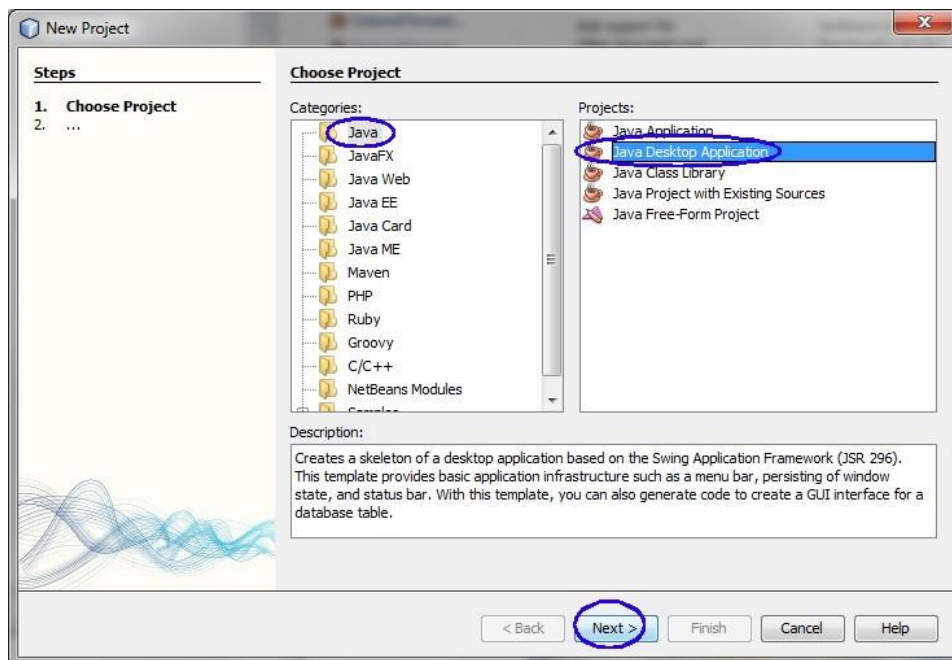
MVC chia một ứng dụng thành ba phần tương tác được với nhau để tách biệt giữa cách thức mà thông tin được xử lý nội hàm và phần thông tin được trình bày và tiếp nhận từ phía người dùng:

- **model** thành phần dữ liệu của chương trình
- **view** bao gồm các thành phần của giao diện người dùng
- **controller** quản lý sự trao đổi giữa dữ liệu và các xử lý nghiệp vụ trong các thao tác liên quan đến mô hình

### Bước 1: Khởi động Netbeans -> File -> New Project ...

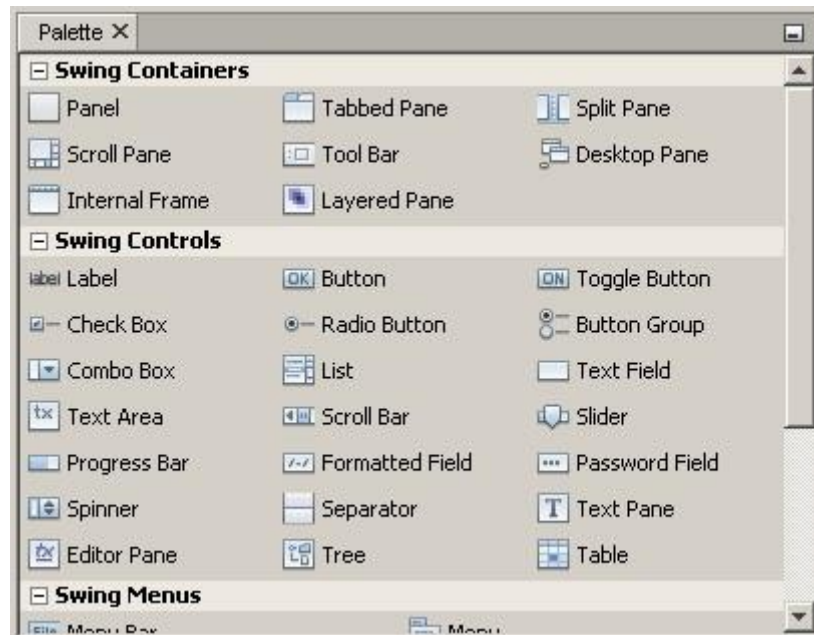


- Chọn **Java** trong mục **Categories** và **Java Desktop** • Nhấn **Next**

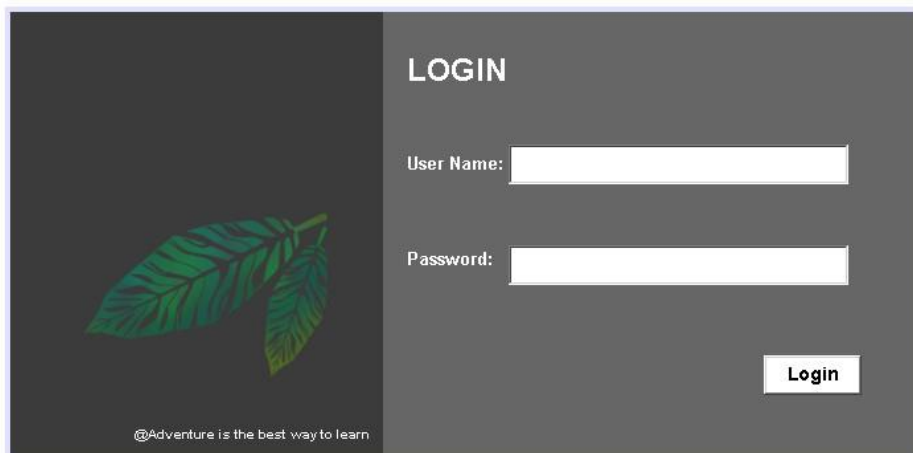


### Bước 2: Tạo JFrame Form

- Nhấn chuột phải **Project** -> chọn **New** -> chọn **JFrame Form** -> nhập **LoginUI** tại **Class Name** -> chọn **Finish**
- Netbeans cung cấp các điều khiển UI trong cửa sổ **Palette**, kéo thả các điều khiển này vào khung thiết kế để xây dựng giao diện cho ứng dụng



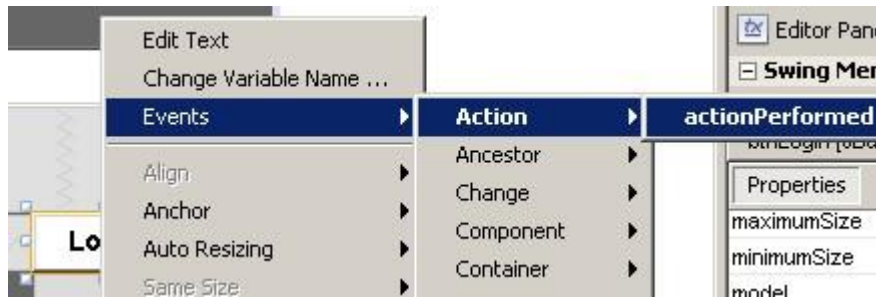
- Trong bài tập này chúng ta sẽ thiết kế một form đăng nhập đơn giản:



- Chú ý đặt tên các điều khiển cho phù hợp để viết mã xử lý dữ liệu trong form. Trong giao diện này có 3 điều khiển: 2 ô textbox và một button.  
**txtUserName, txtPasswd và btnLogin**

### Bước 3: Đăng ký sự kiện

- Nhấn chuột phải vào nút **Login** -> chọn **Events** -> chọn **Action** -> chọn **actionPerformed**



- Viết code để xử lý sự kiện

```
private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) {
    this.login();
}
```

- Xây dựng phương thức **login()** kết nối cơ sở dữ liệu và kiểm tra đăng nhập

```
private void login() {
    //Lấy thông tin nhập vào trong hai ô textbox
    String userName = txbUserName.getText();
    String password = String.valueOf(txbPasswd.getPassword());
    try {
        //Viết code kết nối cơ sở dữ liệu và kiểm tra đăng nhập
        if (/*Đăng nhập thành công*/)) {
            //Hiện thị thông báo đăng nhập thành công
        } else {
            JOptionPane.showMessageDialog(rootPane, "Sai thông tin đang nhập",
                "Warning", JOptionPane.WARNING_MESSAGE);
        }
    } catch (SQLException | ClassNotFoundException e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(null, "Có lỗi xảy ra!! Vui lòng kiểm tra lại!",
            "Warning!!", JOptionPane.ERROR_MESSAGE);
    }
}
```

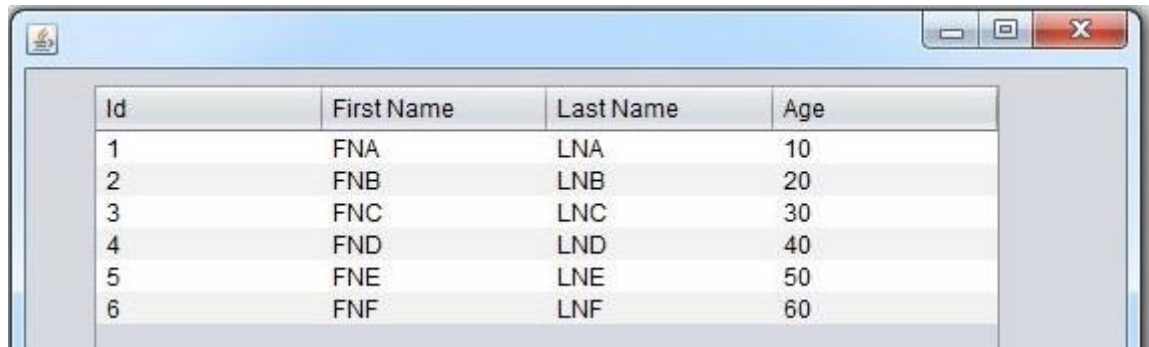
- Bổ sung thêm xử lý khi nhấn phím Enter cũng kiểm tra đăng nhập:

```
// xử lý sự kiện nhấn enter
private void keyListenerner(JTextField jtf) {
    jtf.addKeyListener(new KeyAdapter() {
        @Override
        public void keyPressed(KeyEvent e) {
            // nếu keycode == 10 ~ enter
            if (e.getKeyCode() == 10) {
                login();
            }
        }
    });
}
```

**Chạy thử chương trình và kiểm tra kết quả.**

## Nội dung bài tập tự làm

- Thực hành xây dựng chương trình java với giao diện đồ hoạ: Liệt kê danh sách người dùng sau khi đăng nhập thành công:

A screenshot of a Java Swing window with a standard Windows-style title bar (minimize, maximize, close buttons). The window contains a table with four columns: 'Id', 'First Name', 'Last Name', and 'Age'. The table has six rows of data. The background of the window is light blue.

Id	First Name	Last Name	Age
1	FNA	LNA	10
2	FNB	LNB	20
3	FNC	LNC	30
4	FND	LND	40
5	FNE	LNE	50
6	FNF	LNF	60

**HẾT**