Bài tập tuần 04

Quản lý dự án phần mềm &

Lập trình với giao diện đồ hoạ 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 hoạ người dùng:
  + Làm quen với mô hình MVC các thành phần giao diện người dùng
  + Thiết kế giao diện GUI trên công cụ Netbeans
  + 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 hoạ người dùng

Phần I:

Bài 1.1

1. Dự án kết thúc khi? (chọn nhiều)
2. Thiếu kinh phí (có thể điều chỉnh)
3. Hết kinh phí trước thời hạn (Kết thúc thất bại)
4. Không khả thi (ko được thực hiện luôn nên ko có kết thúc)
5. Quá hạn dự kiến (có làm tiếp cũng không có ý nghĩa gì)
6. Hoàn thành mục tiêu đề ra, nghiệm thu (kết thúc tốt đẹp) trước thời hạn
7. Người quản lý kém (có thể điều chỉnh)
8. Một dự án phát triển phầm mềm là thành công khi?
9. Sản phẩm đáp ứng yêu cầu chất lượng
10. Không vượt quá kinh phí dự kiến
11. Hoàn thành trong thời gian dự kiến
12. Tất cả các phương án trên
13. … 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.
14. Rủi ro dự án
15. Lập kế hoạch dự án
16. Quản lý nhân sự
17. Yêu cầu người dùng

Phạm vi, tài nguyên và ngân sách là những ràng buộc của thực hiện dự án

Bài 1.2

1. 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?

Trả lời:

Theo em, với một người làm công, tham gia vào một dự án mà không biết về toàn bộ chức năng của dự án đó có thể hợp lý trong trường hợp người đó được phân chia công việc, hiểu rõ về những gì cần làm và có đủ kiến thức liên quan, hoàn thành trong kế hoạch và lịch trình được giao.

Nói chung cần xác định tùy theo vai trò của người tham gia dự án.

1. Nếu không áp dụng các mô hình vòng đời phần mềm thì có phát triển được phần mềm không? Tại sao?

Trả lời:

Có thể phát triển phần mềm mà không áp dụng các mô hình vòng đời phần mềm, nhưng việc sẽ trở nên khó khăn hơn và có thể dẫn đến các vấn đề trong quá trình phát triển. Mô hình vòng đời phần mềm cung cấp một khuôn khổ cơ bản cho việc quản lý và tổ chức dự án phần mềm. Tuy nhiên, có nhiều yếu tố cần xem xét để xác định việc áp dụng mô hình nào là phù hợp nhất cho dự án cụ thể.

1. Trong phương pháp Agile, việc luôn có đại diện của khác hàng trong nhóm phát triển thì có ưu điểm gì?

Trả lời:

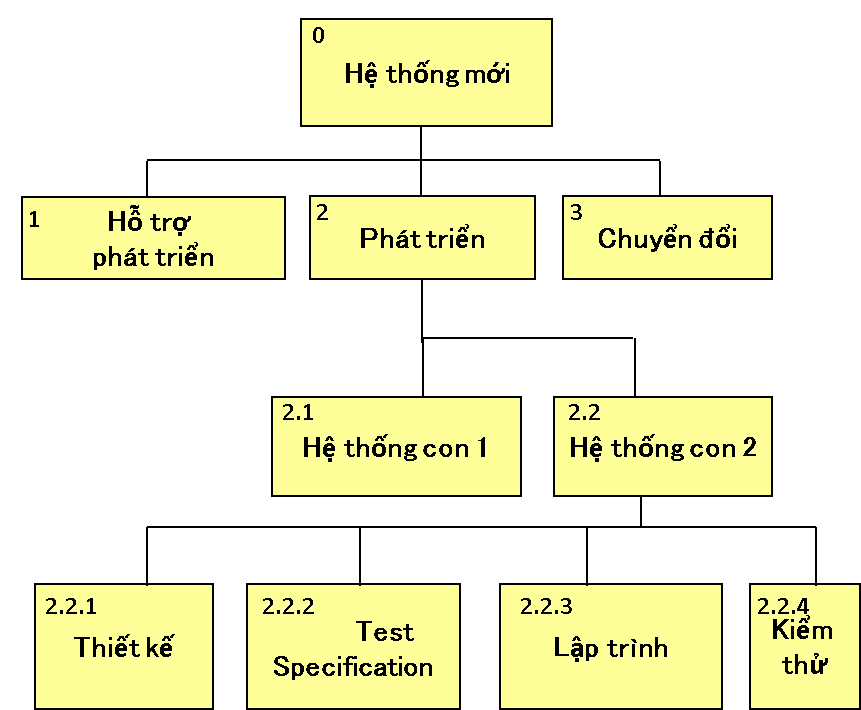
Việc có đại diện của khách hàng hoặc người sử dụng cuối trong nhóm phát triển là một trong các nguyên tắc quan trọng của phương pháp Agile và đem lại nhiều lợi ích quan trọng, bao gồm:

* Lập trình theo yêu cầu thực tế: Đại diện của khách hàng có thể cung cấp thông tin chi tiết về yêu cầu thực tế và ưu tiên của họ. Điều này giúp đảm bảo rằng sản phẩm phát triển thực sự đáp ứng được nhu cầu của người dùng cuối, và không phát triển những tính năng không cần thiết.
* Phản hồi nhanh chóng: Khách hàng có thể cung cấp phản hồi ngay khi sản phẩm đang phát triển. Điều này giúp điều chỉnh sản phẩm một cách nhanh chóng để đảm bảo rằng nó đáp ứng được yêu cầu và mong muốn của khách hàng.
* Đảm bảo sự hiểu biết: Đại diện của khách hàng có thể giúp đảm bảo sự hiểu biết đúng đắn về yêu cầu và mục tiêu của dự án trong toàn bộ nhóm phát triển. Điều này giúp tránh hiểu sai và hạn chế việc phát triển các tính năng không cần thiết.
* Tạo sự cam kết: Sự tham gia của đại diện của khách hàng giúp tạo ra sự cam kết đối với dự án và sản phẩm. Họ có trách nhiệm đưa ra quyết định liên quan đến ưu tiên hóa công việc và sự phê duyệt sản phẩm.
* Xác định sớm các vấn đề: Khách hàng thường có khả năng xác định sớm các vấn đề và thay đổi cần thiết. Điều này giúp tránh việc phát triển sản phẩm theo hướng sai hoặc phát sinh các vấn đề lớn sau này.
* Tích hợp dịch vụ: Khách hàng có thể giúp đảm bảo rằng sản phẩm phát triển có thể dễ dàng tích hợp với hệ thống hiện có và đáp ứng các yêu cầu kỹ thuật và bảo mật của họ.

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
  + 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:



1. **Yêu cầu: Nhóm sinh viên thảo luận và 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  … |  |  |
| … |  |  |

1. **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:

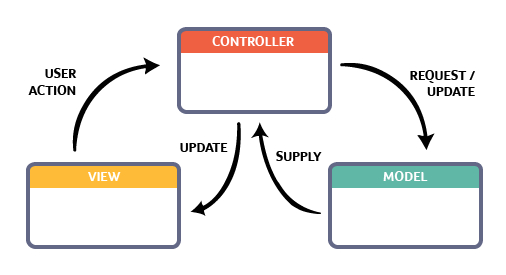
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Công việc / Hoạt động | Xác định rủi ro | | | Quản lý rủi ro | |
| **Mối nguy** | **Rủi ro** | **Mức độ** | **Chiến lược** | **Biện pháp** |
| Mua hàng | Hàng bị hư hại khi vận chuyển | Không có nguyên liệu sản xuất | Trung bình | Giảm thiểu | 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 hoạ người dùng GUI

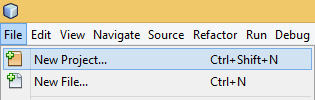
**Background:** **Model – View – Controller (**MVC) là một khuôn mẫu kiến trúc phần mềm. Mẫu thiết kế này cho phép phân tách giữa logic ứng dụng và giao diện người dùng. MVC giúp cho người phát triển phần mềm cô lập các xử lý nghiệp vụ và giao diện người dùng một cách rõ ràng hơn. Phần mềm phát triển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



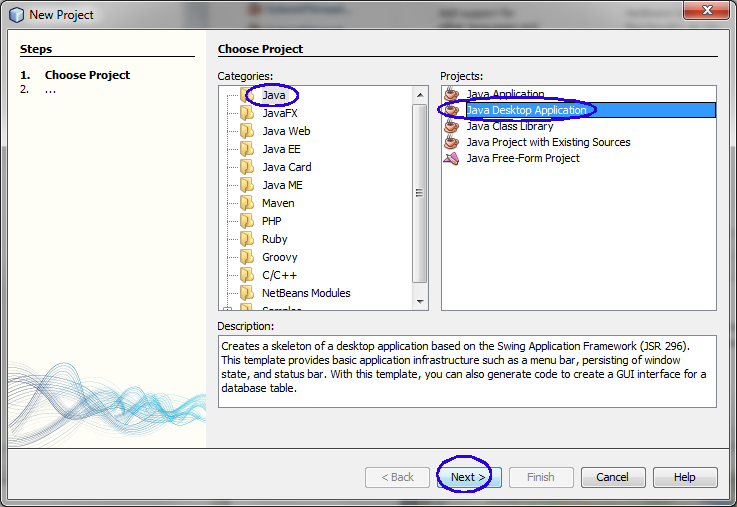
MVC chia một ứng dụng thành ba phần tương tác được với nhau để tách biệt giữa cách thức mà thông tin được xử lý nội hàm và phần thông tin được trình bày và tiếp nhận từ phía người dùng:

* **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 …**

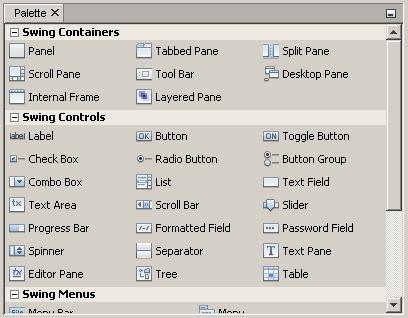


* Chọn **Java** trong mục **Categories**và**Java Desktop**
* Nhấn **Next**

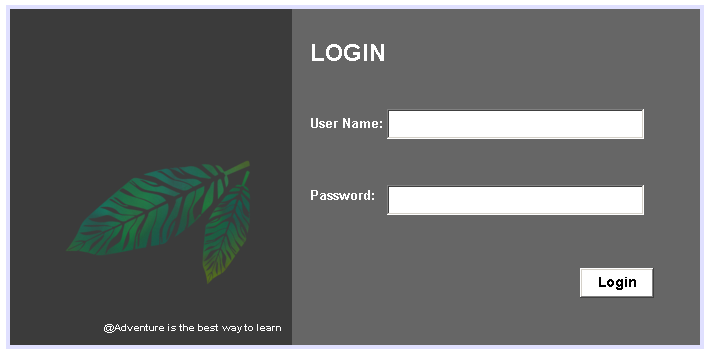


**Bước 2:** Tạo JFrame Form

* 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:

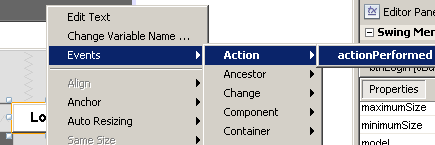


* Chú ý đặt tên các điều khiển cho phù hợp để viết mã xử lý dữ liệu trong form. Trong giao diện này có 3 điều khiển: 2 ô textbox và một button.

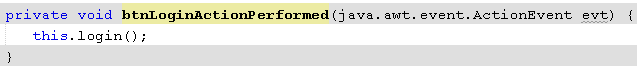
**txtUserName, txtPasswd và btnLogin**

**Bước 3:** Đăng ký sự kiện

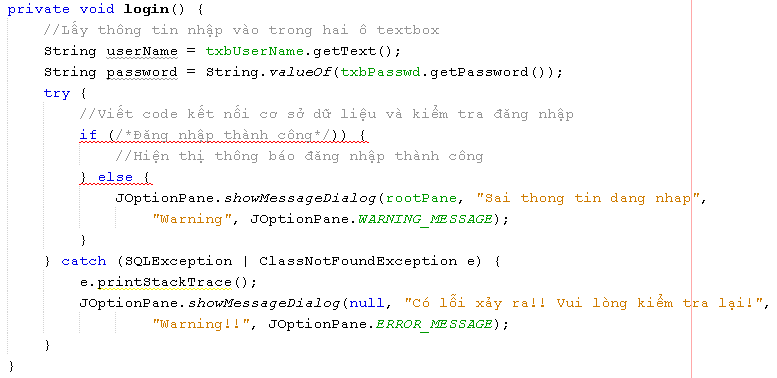
* Nhấn chuột phải vào nút **Login** -> chọn **Events** -> chọn **Action** -> chọn **actionPerformed**



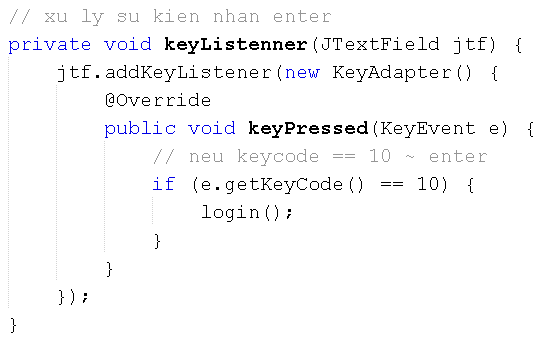
* Viết code để xử lý sự kiện



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



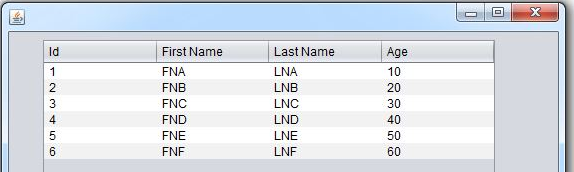
* Bổ sung thêm xử lý khi nhấn phím Enter cũng kiểm tra đăng nhập:



**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:



**HẾT**