Bài tập tuần 02

Vòng đời phần mềm &

Lập trình với cơ sở dữ liệu

Mục tiêu

* Thực hiện các bài tập (câu hỏi) về nội dung Vòng đời phần mềm
* Xác định các đầu vào (input) và kết quả (output) cho nhóm yêu cầu nghiệp vụ của bài toán (case study)
* Lập trình với cơ sở dữ liệu:
  + - Xây dựng ứng dụng Java trên công cụ Netbeans
    - Thiết kế, xây dựng cơ sở dữ liệu với phpMyAdmin trên MySQL
    - Lập trình ứng dụng Java kết nối MySQL, thực thi các truy vấn SQL

Đánh giá

* + Hoàn thành các bài tập về nội dung Vòng đời phần mềm, nắm được đặc điểm chính của các mô hình phát triển phần mềm khác nhau.
  + Các nhóm sinh viên xác định được thông tin cơ bản (input / output) cho nghiệp vụ mà nhóm đã lựa chọn để phát triển
  + Hoàn thành bài thực hành lập trình với cơ sở dữ liệu

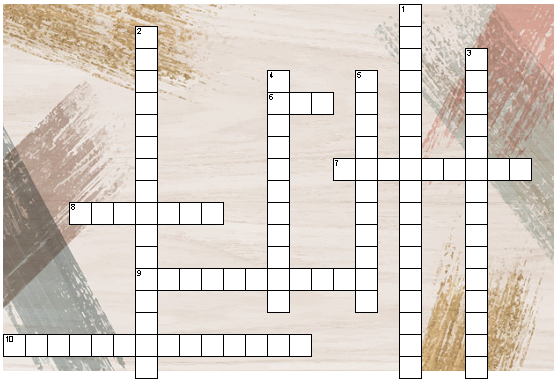
Phần I:

Bài 1.1

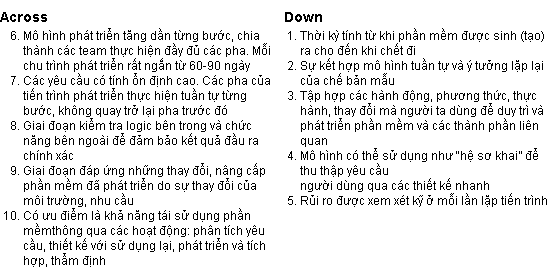
1. Mô hình bản mẫu (prototyping model) của phát triển phần mềm là …
2. Một cách tiếp cận hợp lý khi yêu cầu được định nghĩa rõ ràng
3. Một cách tiếp cận hữu ích khi khách hàng không thể định nghĩa yêu cầu rõ ràng
4. Cách tiếp cận tốt nhất cho những dự án có đội phát triển lớn
5. Tất cả các phương án trên đều sai
6. Bước đầu tiên trong vòng đời phát triển phần mềm (Software Development Life Cycle) là?
7. Xác định các nhu cầu và ràng buộc
8. Viết phần mềm
9. Vận hành hệ thống để loại bỏ các khiếm khuyết
10. Nâng cao sản phẩm sau khi đã triển khai
11. Tình huống nào không phù hợp để có thể áp dụng mô hình thác nước? (chọn nhiều)
12. Khó khăn trong việc bổ sung yêu cầu mới trong các pha sau của tiến trình
13. Các yêu cầu được xác định rõ ràng, đầy đủ ngay từ đầu
14. Khách hàng muốn có sản phẩm vận hành được sớm
15. Khó thu thập đầy đủ yêu cầu ngay ở giai đoạn đầu của dự án

Bài 1.2

Hãy giải ô chữ dưới đây với các gợi ý kèm theo?



Các gợi ý cho ô chữ:



**Trả lời**

1. Software Lifecycle
2. Incremental model
3. Software Process
4. Prototyping
5. Spiral model
6. RAD
7. Waterfall
8. Testing
9. Maintenance
10. Component based

Bài 1.3

Hãy so sánh các mô hình phát triển phần mềm: Thác nước, Chế thử, Gia tăng, Xoắn ốc, Phát triển dựa trên thành phần

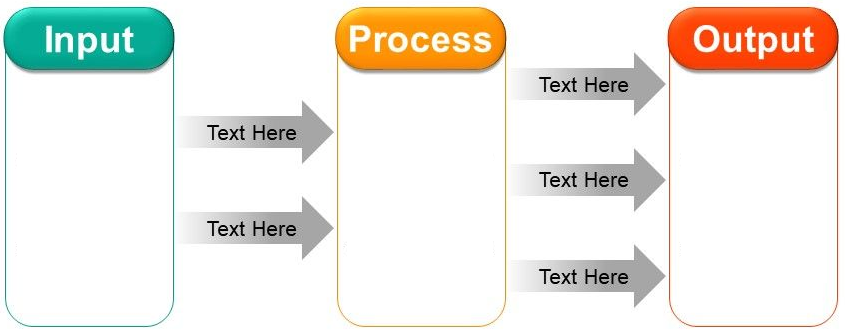
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Đặc điểm chính | Ưu điểm | Nhược điểm | Tình huống áp dụng phù hợp |
| Mô hình Thác nước | * Chuỗi các hoạt động qua các giai đoạn của vòng đời “xuống dốc”: phân tích, yêu cầu, đặc tả, thiết kế, cài đặt, kiểm thử, bảo trì | * Hoàn thành một giai đoạn trước khi tiếp tục giai đoạn tiếp * Nhấn mạnh việc lập kế hoạch sớm, đầu vào của khách hàng và thiết kế * Nhấn mạnh kiểm tra như một phần không thể thiếu của vòng đời • Cung cấp các chất lượng ở mỗi giai đoạn vòng đời | * Phụ thuộc vào các yêu cầu được xác định sớm từ đầu * Phụ thuộc vào việc tách các yêu cầu khỏi thiết kế * Không khả thi trong một số trường hợp đòi hỏi có nhiều thay đổi * Nhấn mạnh vào sản phẩm hơn là quy trình | * Mô hình thác nước phù hợp với các dự án quy mô lớn, yêu cầu duy trì các giai đoạn và thời hạn theo đúng kế hoạch đã lập ra. Hoặc các dự án đã được thực hiện nhiều lần mà ít xảy ra phát sinh trong quá trình thực thi. Đặc biệt phù hợp trong dự án sản xuất và xây dựng các sản phẩm vật lý và theo các đơn đặt hàng, có thể dễ dàng tham khảo áp dụng các quy trình quản lý từ các dự án trước đó vào công việc hiện tại với rất ít hoặc không cần điều chỉnh. |
| Mô hình Chế thử | * Prototype là một mô hình phát triển phần mềm được phát triển dựa trên các yêu cầu hệ thống. Dựa vào bản prototype mà khách hàng có cái nhìn tổng quan về hệ thống thực tế. * Prototype là một ý tưởng hay cho các hệ thống phức tạp và lớn mà không có quy trình thủ công để giúp xác định các yêu cầu. * Prototype thường không phải là hệ thống hoàn chỉnh và nhiều chi tiết không được xây dựng trong bản prototype. Mục tiêu là cung cấp một hệ thống với chức năng tổng thể. | * Người dùng tích cực tham gia vào quá trình phát triển phần mềm từ đó sẽ hiểu rõ hơn về hệ thống đang được phát triển * Lỗi có thể được phát hiện sớm và được fix sớm hơn * Người dùng sẽ dễ dàng nhận thấy những điểm bất cập và đưa ra các giải pháp để giải quyết sớm hơn * Tránh tình trạng bị thiếu mất chức năng của hệ thống * Những chức năng phức tập gây khó hiểu được xác định và giải thích sớm hơn * Xác thực được yêu cầu của hệ thống từ đó thực hiện nhanh hơn nhưng không đầy đủ chức năng của ứng dụng | * Trên thực tế bản prototype có thể làm tăng sự phức tạp của hệ thống vì phạm vi của hệ thống có thể mở rộng hơn các kế hoạch ban đầu * Bản mẫu này được chính khách hàng nhận xét đánh giá, nếu bản prototype mà khách hàng không hài lòng phải làm lại gây tốn thời gian và công sức. | * Mô hình prototype nên được sử dụng khi hệ thống cần có nhiều tương tác với người dùng cuối * Thông thường, các hệ thống trực tuyến, giao diện web có lượng tương tác rất cao với người dùng cuối, phù hợp nhất với mô hình Prototype. |
| Mô hình Gia tăng | * Kết hợp mô hình tuần tự và ý tưởng lặp lại của chế bản mẫu * Sản phẩm với những yêu cầu cơ bản nhất của hệ thống được phát triển * Các chức năng với những yêu cầu khác được phát triển thêm sau (gia tăng) * Lặp lại quy trình để hoàn thiện dần | * Có thể sớm tạo ra nguyên mẫu của sản phẩm trong vòng đời phát triển của nó. * Độ linh hoạt cao hơn và khi thay đổi yêu cầu dự án thì chi phí sẽ ít hơn nhiều, vì những thay đổi thuộc về module nào thì module đó sẽ thay đổi mà các module khác không hề bị ảnh hưởng. * Việc phân chia thành các module cũng sẽ làm cho việc test nhẹ nhàng hơn, những module đơn giản thì test cũng đơn giản, sớm kết thúc. * Giảm chi phí cho lần đầu giao sản phẩm. * Dễ dàng quản lý các rủi ro có thể phát sinh. | * Cần phải có những khả năng thiết kế tốt và phương pháp tốt, để có thể hiểu rõ được yêu cầu và biết cách phân chi nó ra như thế nào cho hợp lý. * Chi phí để phát triển theo phương pháp này là rất cao, cao hơn hẳn waterfall. | * Áp dụng mô hình này khi yêu cầu của dự án là rõ ràng, đầy đủ, và nắm rõ được các yêu cầu của dự án. * Khi sớm cần có một nguyên mẫu phần mềm để quáng bá, giới thiệu hoặc thử nghiệm. * Sử dụng mô hình này khi một công nghệ mới được áp dụng. * Tài nguyên và kỹ năng chuyên môn luôn sẵn sàng. * Khi có một tính năng hay các mục tiêu có nguy cơ lỗi cao. |
| Mô hình Xoắn ốc | * Mô hình xoắn ốc (Spiral-Model) là mô hình có sự kết hợp giữa mô hình thác nước (Waterfall-Model) và mô hình tiếp cận lặp (Iterative-Model) và nó có nhiều điểm giống nhau với mô hình gia tăng (Incremental-Model). * Chú trọng vào phân tích rủi ro dự án. Mỗi giai đoạn trong mô hình được bắt đầu với yêu cầu/mục tiêu thiết kế và kết thúc với việc khách hàng kiểm tra tiến độ của từng giai đoạn. | * Tốt cho các hệ phần mềm quy mô lớn * Dễ kiểm soát các mạo hiểm ở từng mức tiến hóa | * Khó thuyết phục khách hàng là phương pháp tiến hóa xoắn ốc có thể kiểm soát được * Chưa được dùng rộng rãi như các mô hình tuyến tính hoặc chế thử | * Mô hình xoắn ốc là một cách tiếp cận thực tế để phát triển các sản phẩm phần mềm quy mô lớn bởi vì phần mềm phát triển khi quá trình tiến triển (the software evolves as the process progresses). Ngoài ra, nhà phát triển và khách hàng hiểu rõ hơn và phản ứng với các rủi ro ở mỗi cấp độ (level) phát triển. |
| Mô hình dựa thành phần | * Là quy trình phát triển phần mềm gia tăng, tăng dần từng bước (Incremental software development) với mỗi chu trình phát triển rất ngắn (60-90 ngày) * Xây dựng dựa trên hướng thành phần (Component-based construction) với khả năng tái sử dụng (reuse) * Gồm một số nhóm (teams), mỗi nhóm làm 1 RAD theo các pha: Mô hình nghiệp vụ, Mô hình dữ liệu, Mô hình xử lý, Tạo ứng dụng, Kiểm thử và đánh giá (Business, Data, Process, Appl. Generation, Test) | * Giảm được thời gian phát triển * Dễ hiểu * Giảm rủi ro * Ít nỗ lực hơn * Tăng khả năng sử dụng lại của các thành phần. * Giảm thiểu các lỗi trong quá trình phát triển hệ thống. * Dễ dàng thu thập phản hồi từ khách hàng để cải thiện * Được chứng minh là tốt nhất cho các dự án quy mô vừa và nhỏ. | * Không lý tưởng cho ngân sách thấp * Yêu cầu các nhà phát triển có kỹ năng cao * Không thích hợp cho các dự án kỹ thuật phần mềm lớn * Khó theo dõi tiến trình và các vấn đề * Hoạt động trên hệ thống module hóa * Phụ thuộc nhiều vào hiệu suất của nhóm * triển song song. | * Mô hình này là một mô hình đặc biệt, nên nó chỉ được sử dụng khi có một sự cần thiết để tạo ra các hệ thống mà có thể module hóa 2 hoặc trong 3 tháng. * Chỉ được sử dụng khi mô hình thiết kế là sẵn có và bên cạnh đó là ngân sách phải có nhiều để có thể xây dựng được nhiều team cùng phát |

Phần II: Phân tích thông tin cơ bản (input / output) cho nghiệp vụ bài toán (case study)

Bài 1.4

* Các nhóm thảo luận để xác định đầu vào (input) và kết quả (output) cho nhóm yêu cầu nghiệp vụ của bài toán (case study) mà nhóm đã lựa chọn.
* Phân rã mô tả của nhóm yêu cầu nghiệp vụ thành các nghiệp vụ nhỏ hơn.
* Với mỗi nghiệp vụ nhỏ này thực hiện phân tích các thông tin cụ thể

- Có thể vẽ sơ đồ thể hiện kết quả phân tích:

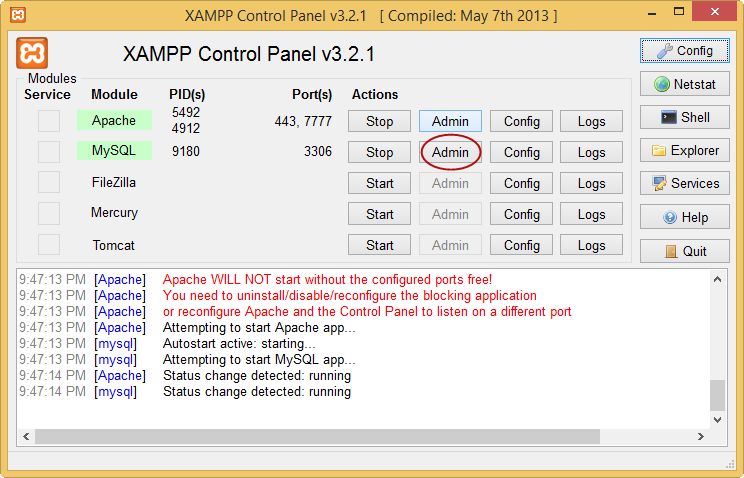


Hoặc thể hiện dưới dạng bảng:

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
|  |  |  |
|  |  |  |

Phần III: Thực hành lập trình với cơ sở dữ liệu

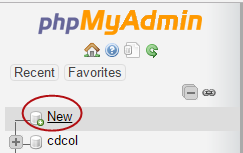
**Bước 1:**Khởi động **XAMPP Control Panel** à **Start** các dịch vụ Apache và MySQL à chọn **Admin** để mở công cụ quản trị phpMyAdmin cho cơ sở dữ liệu MySQL



* Hoặc có thể mở trình duyệt và truy cập vào: <http://localhost/phpmyadmin>

**Bước 2:** Tạo cơ sở dữ liệu

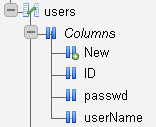
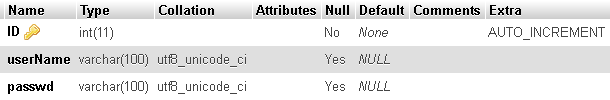
* Tại trang quản trị  phpMyAdmin à chọn **New**



* Nhập tên database và chọn Create (nhớ chọn Collation là **utf8\_unicode\_ci**)

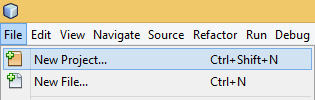


* Tạo một bảng dữ liệu và nhập các dữ liệu mẫu vào bảng

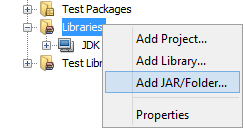
 

**Bước 3:** Xây dựng ứng dụng Java kết nối với MySQL

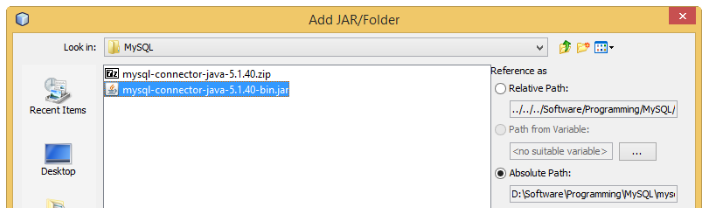
* Khởi động **Netbeans -> File -> New Project …**



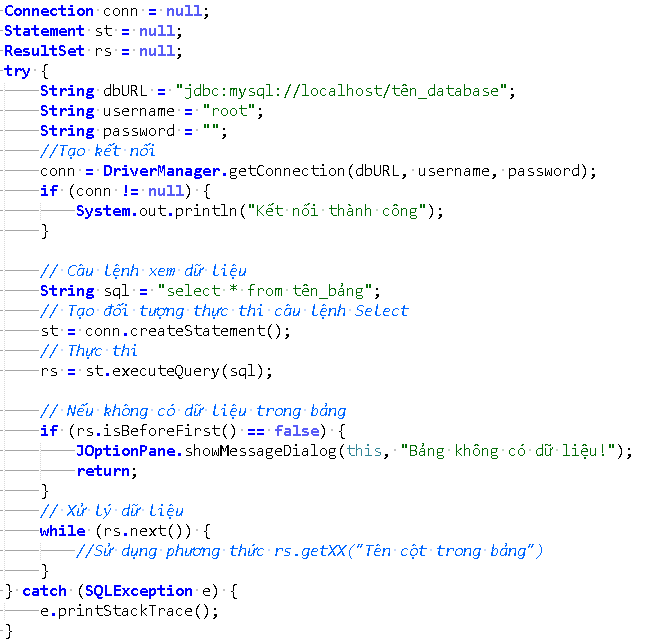
* Tải **JDBC Driver cho MySQL** và thêm vào project (<https://dev.mysql.com/downloads/file/?id=465644> )



* Chọn tập tin jar **->** chọn **Open**



* Viết code để truy vấn dữ liệu từ bảng trong cơ sở dữ liệu và hiển thị ra console (chú ý import các package cần thiết và sử dụng try/catch để bắt ngoại lệ khi thực thi các thực hiện các thao tác với CSDL / truy vấn SQL)



Nội dung bài tập tự làm

* Thực hành xây dựng chương trình java với các lệnh SQL cơ bản (SELECT, INSERT, UPDATE, DELETE)

**HẾT**