**Phần 1: Lý thuyết**

# Câu 0. Giới thiệu giấy phép BSD, LGPL, GPL, MPL, MIT

| **BSD** | **Tên**: Berkeley Software Distribution License  **Lịch sử**: giấy phép BSD được thiết kế bởi Đại học California tại Berkeley năm 1980 cho dự án BSD (Berkeley Source Distribution) được biết đến với cái tên BSD Unix, một phiên bản mở rộng của hệ điều hành Unix nguyên mẫu được viết vào năm 1969 bởi Ken Thompson tại Bell Labs.  **Nội dung chính:**   * Quyền lợi   + Giấy phép BSD cho phép sử dụng và phân phối lại mã nguồn và sản phẩm, có hoặc không có sửa đổi, miễn là tuân thủ các yêu cầu sau. * Nghĩa vụ   + Phải giữ nguyên thông báo bản quyền của sản phẩm. Yêu cầu này nhằm đảm bảo một người dùng bất kì không thể tuyên bố anh ta đã viết ra một phần mềm nếu thực sự anh ta không viết ra nó.   + Phải kèm theo 2 thông báo: Danh sách các điều kiện và từ chối trách nhiệm.   + Không được sử dụng tên dự án hay tên nhà phân phối vào mục đích quảng bá bản thân nếu không được cho phép.   **Ví dụ:** Giấy phép BSD sử dụng bởi hệ điều hành FreeBSD:, giấy phép BSD sử dụng cho tiện ích Sudo (một chương trình sử dụng trong các hệ điều hành Unix để chuyển tạm thời sang quyền root):  - Hệ thống Window Xfree86: nền tảng của hầu hết các giao diện với người sử dụng trong các hệ thống phần mềm nguồn mở.  - FreeBSD, NetBSD và OpenBSD: các biến thể của phiên bản Unix gốc được cấp giấy phép BSD, cả ba đều được sử dụng rộng rãi trên mạng Internet, đặc biệt là FreeBSD, chương trình điều hành Yahoo Mail và dịch vụ Hotmail.  **Tóm tắt:**   * Cho phép các nhà phát triển phần mềm có thể thương mại hóa (kiếm tiền) một cách thực sự các sản phẩm phần mềm có sử dụng mã nguồn mở dùng giấy phép BSD. * Cho phép các nhà phát triển thay thế, bổ sung thêm các điều khoản vào trong giấy phép cho phù hợp với mình, hoặc thậm chí sử dụng một giấy phép khác. |
| --- | --- |
| **LGPL** | **Tên**: GNU Lesser General Public License  **Lịch sử**: GNU Lesser General Public License (LGPL). Phiên bản 2.1 phát hành tháng 2 năm 1999.  Lesser GNU General Public License (LGPLv3) phát hành tháng 6 năm 2007. Giấy phép này là một tập các điều khoản bổ sung thêm vào phiên bản 3 của GNU General Public License.  **Nội dung chính:**   * Giấy phép LGPL rất gần với GPL, nhưng cho phép gọi các hàm từ một chương trình khác, không cần các chương trình sử dụng chương trình dưới LGPL này phải là nguồn mở. * Được tạo ra để cho phép liên kết động mã nguồn không phát hành dưới dạng GPL hoặc LGPL vào mã nguồn LGPL * Dàn xếp việc sử dụng các thư viện tự do vào mục đích thương mại, ví dụ thư viện GNU C * Hầu hết các điều khoản và điều kiện tương tự GPL * Nếu bạn thay đổi và phân phối một thư viện LGPL   + Thư viện và những thay đổi phải được công bố (mã thực thi và mã nguồn cùng với chú thích về những sửa đổi)   + Bằng sáng chế được gắn với sự phân phối những sửa đổi   **Ví dụ:** Crush em T |
| **GPL** | **Tên**: GNU General Public License  **Lịch sử:** ban đầu được thiết kê bới Richard Stallman, dành cho dự án GNU. Phiên bản 1 của GNU GPL, phát hành vào tháng 1 năm 1989. Phiên bản 2 của giấy phép này được phát hành năm 1991(với 12 điều). Phiên bản 3 của giấy phép đã được phát hành trong năm 2007 (với 17 điều) và đang được triển khai dần.  **Nội dung chính:**   * **Quyền lợi:**   + Quyền được sao chép và phân phối chương trình, quyền được yêu cầu trả phí cho việc phân phối đó.   + Quyền được thay đổi chương trình để sử dụng cho mục đích cá nhân.   + Quyền được phân phối bản đã được thay đổi đó. * **Nghĩa vụ**   + Khi sao chép và phân phối chương trình, phải đính kèm các thông báo về bản quyền gốc và không nhận bảo hành (trừ trường hợp có văn bản thêm về quy định bảo hành.)   + Khi phân phối bản đã được thay đổi bởi bản thân, phải chú thích rõ đó là bản đã được thay đổi, các thành phần được thay đổi, và áp dụng giấy phép GNU cho bản đã được thay đổi đó.   + Khi phát hành chương trình phải công khai mã nguồn của chương trình của nó, đồng thời phải công bố mã nguồn của chương trình trong tối thiểu 3 năm mà không được đòi một khoản phí nào từ những người yêu cầu mã nguồn trừ chi phí vận chuyển hay tương đương.   **Ví dụ:** RedHat Enterprise Linux, Ubuntu, GIMP, Drupal, WordPress, Joomla…  **Tóm tắt:**   * đặc tính virus: Một người nhận sản phẩm từ người mang giấy phép GNU thì ngay lập tức người đó cũng mang giấy phép GNU * Tác giả gốc giữ bản quyền, và cho người dùng các quyền hợp pháp trong việc: sao chép, chỉnh sửa, phân phối sản phẩm. * cho phép người phân phối có thể kinh doanh với sản phẩm nhờ đưa ra các chính sách về bảo hành, tính chi phí phân phối sản phẩm, đào tạo sử dụng …   **\* Ý tưởng của GPL:**  + Phần mềm GPL phải là phần mềm tự do:Tức là người sử dụng có 4 quyền sau với phần mềm GPL:  - Tự do chạy chương trình, cho bất cứ mục đích nào.  - Tự do tìm hiểu cách hoạt động của chương trình, và tự do sửa đổi nó (Quyền truy cập mã nguồn là điều kiện tiên quyết cho quyền tự do này).  - Tự do tái phân phối bản sao.  - Tự do cải tiến chương trình, và phát hành những gì cải tiến ra công cộng (Quyền truy cập mã nguồn là điều kiện tiên quyết cho quyền tự do này). |
| **MPL** | **Tên**: Mozilla Public License  **Lịch sử:** một giấy phép phần mềm tự do và mã nguồn mở. Phiên bản 1.0 do Mitchell Baker phát triển khi bà là luật sư tại Tập đoàn Thông tin liên lạc Netscape và phiên bản 1.1 tại Quỹ Mozilla.  **Nội dung chính:**   * là lai giữa giấy phép BSD có chỉnh sửa và Giấy phép Công cộng GNU * MPL là giấy phép dùng cho Mozilla Application Suite, Mozilla Firefox, Mozilla Thunderbird và các phần mềm khác của Mozilla. * Giấy phép được xem là một copyleft yếu. Cụ thể hơn, mã nguồn nếu được sao chép hoặc thay đổi theo giấy phép MPL phải giữ nguyên giấy phép MPL * MPL vừa được Sáng kiến Mã nguồn mở chứng nhận vừa là giấy phép phần mềm mã nguồn mở vừa được Quỹ Phần mềm Tự do chứng nhận là một giấy phép phần mềm tự do. * Tương thích với các giấy phép khác.   **Ví dụ:** Drupal, GIMP, CakePHP, Mozilla Firefox, Apache Server… |
| **MIT** | **Tên**: Massachusetts Institute of Technology License  **Lịch sử:** Theo Tổ chức phần mềm tự do, giấy phép MIT nên được gọi chính xác hơn là giấy phép X11, do Học viện Massachusetts đã từng dùng nhiều loại giấy phép khác nữa, và do giấy phép này ban đầu được soạn thảo cho X Window System (dự án tạo giao diện đồ họa cho người dùng ở các máy tính kết nối mạng vào năm 1984).  **Nội dung chính:**   * MIT là một giấy phép dạng “permissive”, nghĩa là nó cho phép tái sử dụng các phần mềm độc quyền nhưng với điều kiện giấy phép MIT đã được phân phối kèm phần mềm đó * Giấy phép MIT cũng tương thích GLP, nghĩa là GPL cho phép tổng hợp và tái phân phối các phần mềm sử dụng giấy phép MIT. * Các quyền lợi của người sử dụng giấy phép vẫn tương tự như ở giấy phép GNU. Tuy nhiên MIT loại bỏ thuộc tính virus khỏi giấy phép, tức không bắt buộc các chương trình sử dụng tài nguyên có dùng giấy phép MIT cũng phải sử dụng giấy phép MIT. * Giấy phép MIT không bắt buộc phải công khai mã nguồn. * Người sử dụng phải kèm theo giấy phép MIT vào bản chỉnh sửa của mình, tuy nhiên không bị bắt buộc phải sử dụng giấy phép MIT cho toàn bộ bản đó. * Giấy phép MIT cũng có thể được chỉnh sửa lại cho phù hợp với thực tế. * Ngoài ra, giấy phép MIT không cấm sử dụng tên của người giữ bản quyền vào mục đích quảng bá, và cũng không bắt buộc phải hiện danh sách tất cả những người từng tham gia thực hiện dự án trong phần About của chương trình.   **Ví dụ:** Các gói phần mềm sử dụng giấy phép MIT bao gồm Expat, PuTTY, Ruby on Rails, Lua 5.0 và X Window System. Cũng có một số sản phẩm sử dụng chung giấy phép MIT với một giấy phép khác, mà nổi tiếng nhất là thư viện jQuery của JavaScript, sản phẩm này sử dụng cả giấy phép MIT và GNU. |

# Câu 1. Phân biệt các giấy phép BSD, LGPL, GPL, MPL, MIT

* GPL (General Public License): Một khi sử dụng và phân phối , bắt buộc phải sử dụng giấy phép GPL, không được phép đóng mã nguồn và thay đổi giấy phép .
* LGPL (Lesser General Public License): là giấy phép tự do ít ràng buộc hơn, là giấy phép sửa đổi của GPL, được sử dụng cho một số thư viện phần mềm ( các thư viện dùng ngôn ngữ thường áp dụng giấy phép này )
* MPL (Mozilla Public License): MPL dung hoà giữa BSD và GPL.MPL cho phép dùng MPL software để tạo ra một sản phẩm khác ( thương mại hoặc không ) , tuy nhiên nếu thay đổi MPL software thì phải được đưa miễn phí lên Internet .
* Apache: được phép đóng mã nguồn , thương mại hoá và giữ bản quyền sản phẩm .
* BSD (Berkeley Software Distribution License): là giấy phép tự do ít ràng buộc hơn, các giấy phép kiểu BSD để những sản phẩm phái sinh được tái phân phối như phần mềm thương mại.
* MIT (Massachusetts Institute of Technology License)

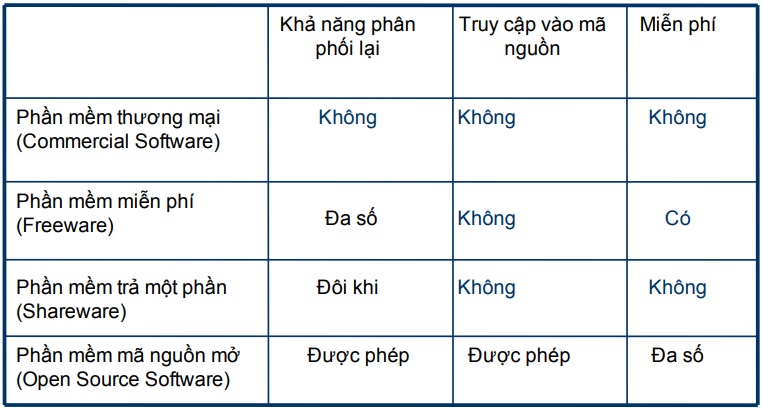
# Câu 2. Khái niệm Chủ sở hữu phần mềm

Khi một phần mềm được tạo ra nó thuộc một chủ sở hữu nào đó. Chủ sở hữu có thể là một cá nhân (lập trình viên viết ra phần mềm) hoặc là một công ty phần mềm (người bỏ tiền ra thuê mướn lập trình viên trực tiếp viết phần mềm cho công ty).Chủ sở hữu phần mềm có toàn quyền trên phần mềm mà họ là chủ sở hữu, và sẽ quyết định mức độ sử dụng và khai thác của những người khác trên phần mềm mà họ là chủ sở hữu. Khi muốn sử dụng một phần mềm đó, người sử dụng phải xin phép chủ sở hữu phần mềm thông qua một giấy phép được cập bởi chủ sở hữu phần mềm.

# Câu 3. Khái niệm Giấy phép sử dụng phần mềm (License)

Được chủ sở hữu phần mềm cấp cho người muốn sử dụng phần mềm. Nó là một bản hợp đồng gồm các điều khoản và điều kiện, mô tả những gì mà chủ sở hữu phần mềm cho phép bạn khai thác phiên bản phần mềm liên quan. Nó quy định về những khả năng mà bạn có thể có được trên phần mềm mà bạn được cấp giấy phép sử dụng.

# Câu 4. Khác biệt về giấy phép giữa 1 số phần mềm thông dụng



# Câu 5. 10 điều kiện (tiêu chí) mà các điều khoản phân phối phần mềm mã nguồn mở phải tuân theo

* Tiêu chí (1): Tự do phân phối lại ( Free Redistribution) Bản quyền sẽ không hạn chế bất cứ ai bán hoặc cho phần mềm; và không đòi hỏi tiền bản quyền hay một chi phí nào cho thương vụ này.
* Tiêu chí (2): Mã nguồn ( Source Code) Chương trình phải được phân phối cùng với mã nguồn được công bố bằng những phương tiện công cộng mà người ta có thể lấy được mã nguồn với một chi phí sao chép hợp lý nhất
* Tiêu chí (3): Sản phẩm kế thừa (Derived Works) Giấy phép phải công nhận những sửa đổi và những sản phẩm kế thừa; và phải cho phép chúng được phân phối với cùng những điều khoản như giấy phép của phần mềm ban đầu
* Tiêu chí (4) Tính toàn vẹn của mã nguồn của tác giả (Integrity of The Author's Source Code) Giấy phép có thể ngăn cản việc phân phối mã nguồn dưới dạng bị sửa đổi chỉ khi giấy phép chấp nhận sự phân phối các tập tin vá lỗi (patch file) với mã nguồn vì mục đích sửa đổi chương trình tại thời điểm xây dựng (built time) chương trình. Giấy phép phải cho phép một cách tường minh việc phân phối phần mềm tạo ra từ mã nguồn bị sửa đổi. Giấy phép có thể yêu cầu những sản phẩm kế thừa phải mang một cái tên khác hoặc số phiên bản khác so với phần mềm gốc.
* Tiêu chí (5): Không phân biệt đối xử giữa các cá nhân và các nhóm (No Discrimination Against Persons or Groups)
* Tiêu chí (6): Không phân biệt đối xử với mục đích sử dụng (No Discrimination Against Fields of Endeavor)
* Tiêu chí (7): Phân phối giấy phép (Distribution of License) Những quyền được kèm với chương trình phải được áp dụng đối với tất cả những người mà sau đó chương trình được phân phối lại mà không cần thiết phải thực thi thêm những giấy phép phụ của những thành phần này
* Tiêu chí (8): Giấy phép không được dành riêng cho một sản phẩm ( License Must Not Be Specific to a Product) Những quyền được kèm theo chương trình thì không bị phụ thuộc vào việc chương trình là thành phần của một bản phân phối phần mềm cụ thể. Nếu phần mềm được rút trích từ bản phân phối đó và được sử dụng hoặc phân phối lại với những điều khoản của giấy phép của chương trình thì tất cả các bên mà chương trình được phân phối đến cũng nên có được các quyền lợi ngang bằng như những quyền lợi được đưa ra theo bản phân phối phần mềm gốc.
* Tiêu chí (9): Giấy phép không được cản trở phần mềm khác ( License Must Not Restrict Other Software) Giấy phép không được đặt những hạn chế lên những phần mềm khác cùng được phân phối với phần mềm của giấy phép này. Ví dụ, giấy phép không được khăng khăng rằng tất cả các phần mềm khác được phân phối trên cùng một phương tiện thì phải là phần mềm mã nguồn mở
* Tiêu chí (10): Giấy phép phải trung lập về mặt công nghệ (License Must Be Technology-Neutral) Không có sự dự trù nào của giấy phép dành cho một công nghệ riêng hay một kiểu giao diện nào đó

# Câu 6. Sự khác biệt giữa mô hình phát triển phần mềm truyền thống và Phần mềm Mã Nguồn Mở (PMMNM)

| **phần mềm truyền thống** | **Phần mềm Mã Nguồn Mở** |
| --- | --- |
| Khang hiếm và tốn kém, vì thế cần quản lý chặt chẽ  Cần xây dựng môi trường để bảo vệ tài nguyên này | Lập trình viên là tình nguyện  Sử dụng hạ tầng cơ sở (ví dụ máy tính) sẵn có  Phân phối qua Internet |

# Câu 7. Khái niệm Phát triển phần mềm mã nguồn mở

Tên: Free and open-source software development (FOSSD)

Phân biệt “miễn phí” và “mở” (free/open)

– Đều là tự do sử dụng, nghiên cứu, chỉnh sửa và lại đóng góp cho mã nguồn.

– Tuy nhiên, miễn phí thì luôn luôn mở, nhưng mở chưa chắc là miễn phí.

FOSSD không phải là “CNPM”

– Vì FOSSD có anh chất nhanh hơn, tốt hơn, rẻ hơn so với CNPM trong một số tình huống.

– Các nhóm FOSSD dùng 10-50 công cụ OSSD tools và các ứng dụng giao tiếp để hỗ trợ công việc phát triển của họ.

# Câu 8. Đặc trưng của dự án FOSSD (Free and open-source software development)

* Những người phát triển FOSS cũng chính là người sử dụng, trong khi ~1% người dùng FOSS là người phát triển.
* Cần có khối lượng lớn người đóng góp và các thành phần FOSS được kết nối qua mạng tương tác kỹ thuật-xã hội.
* Các dự án FOSSD có thể xuất hiện thông qua những cái sẵn có.
* Kết cấu kiến trúc không đoán trước được: Những tích hợp thành phần đa dự án

# Câu 9. Mô hình phát triển phần mềm mã nguồn mở

Là mô hình xây dựng chợ: Không có một thiết kế ban đầu rõ ràng, không có một quy trình quản lý chính thức.

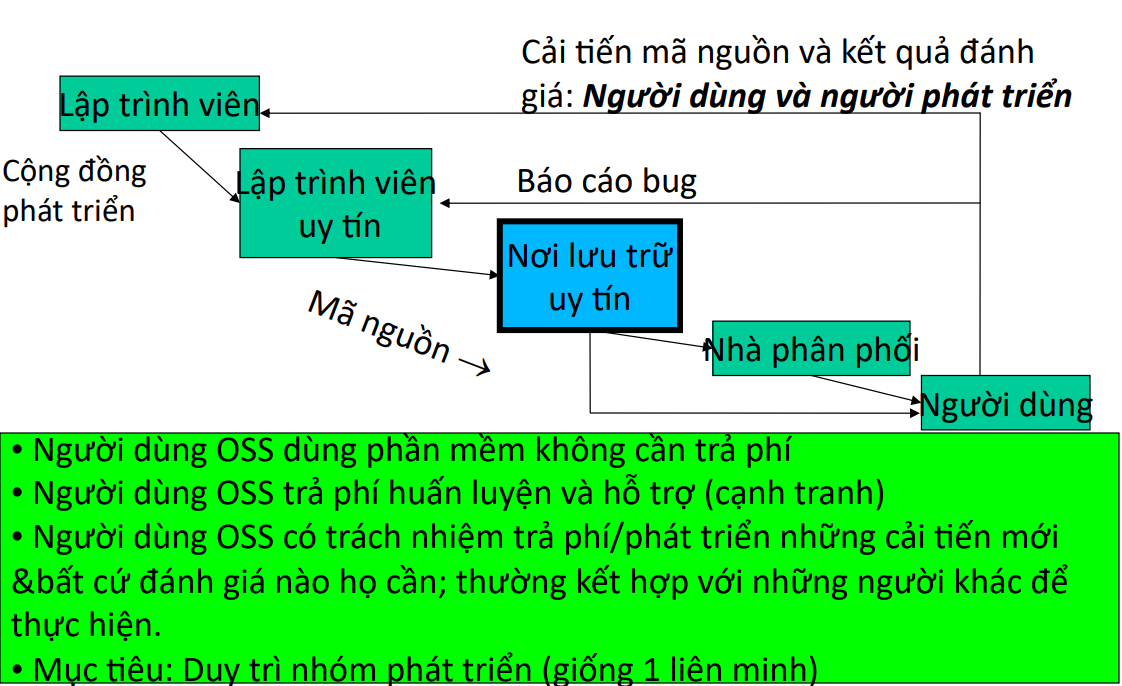
Sử dụng một chính sách lỏng lẻo trong việc:

* Phát hành mã nguồn
* Quản lý ai là người viết mã nguồn cho việc sửa lỗi và cho các chức năng mới

Nguyên tắc căn bản: «Viết mã lệnh thường xuyên, phát hành thường xuyên». Đây là mô hình tăng trưởng: Tự phát triển khi phần mềm đạt đến một số chức năng cơ bản nào đó. Mô hình phát triển gồm 2 giai đoạn:

* Giai đoạn khởi đầu:
* Phần mềm chưa đủ các chức năng để có thể hấp dẫn các lập trình viên khác
* Cần một số tài trợ về tài chánh để có thể đạt đến điểm có thể sử dụng được, sẽ chuyển sang giai đoạn tăng trưởng
* Giai đoạn tăng trưởng
* Nhận được thêm nhiều chức năng mới và các gói sửa lỗi từ cộng đồng

# Câu 10. Vẽ mô hình phát triển OSS (open-source software development) tiêu biểu

****

# Câu 11. Môi trường phát triển mã nguồn mở FOSSD

Môi trường phát triển PMMNM cần cung cấp các chức năng sau:

* Các kênh truyền thông (communication channel)
* Các cơ sở dữ liệu về lỗi (Bug database)
* Hệ thống quản lý mã nguồn (Version control)

# Câu 12. Xưởng phát triển mã nguồn mở

Được biết đến với tên Forge, là các website cung cấp môi trường phát triển PMMNM Ví dụ: Codendi, FusionForge (ex-GForge) QualiPSo, SourceForge, Google Code…

# Câu 13. Động cơ của người phát triển PMMNM

* Vì phần thưởng tiền bạc: Không nhiều
* Phần lớn PMMNM là kết quả của
  + Niềm đam mê lập trình,
  + Kết quả của một số bài tập trong các chương trình đại học,
  + Vì Lợi ích cộng đồng
* Một số công ty dùng như mô hình kinh tế để:
  + Thâm nhập thị trường đã bị thống trị bởi công ty khác
  + Phát hành sản phẩm nhanh hơn nhờ sử dụng lại PMMNM

# Câu 14. Khái niệm Mô hình MVC ?

Khái niệm: MVC là viết tắt của cụm từ “Model-View-Controller“. Đây là mô hình thiết kế được sử dụng trong kỹ thuật phần mềm.

MVC chia thành ba phần được kết nối với nhau và mỗi thành phần đều có một nhiệm vụ riêng của nó và độc lập với các thành phần khác. Tên gọi 3 thành phần:

* Model (dữ liệu): Quản lí xử lí các dữ liệu.
  + Model chứa một cấu trúc dữ liệu có chức năng lưu trữ toàn bộ các thông tin dữ liệu của một ứng dụng. Trong mô hình MVC thì Model đóng vai trò kết nối cho 2 thành phần View và Controller.
  + Được thiết lập như một cơ sở dữ liệu hoặc đơn giản hóa như một file XML thông thường
  + Ví dụ: Chẳng hạn như biểu tượng hay là một nhân vật trong game.
* View (giao diện): Nơi hiển thị dữ liệu cho người dùng.
  + Phần giao diện dành cho người sử dụng. View là phương tiện hiển thị các đối tượng trong một ứng dụng (như hiển thị một cửa sổ, nút…). Nó bao gồm bất cứ thứ gì mà người dùng có thể nhìn thấy được.
  + Thông qua dữ liệu của MVC , người dùng sẽ thực hiện các thao tác tìm kiếm, sử dụng thông tin website, ứng dụng.
  + Chức năng khác của thành phần View này chính là khả năng ghi nhận hành vi của người dùng để tương tác được với Controller.
* Controller (bộ điều khiển): Điều khiển sự tương tác của hai thành phần Model và View.
  + Là bộ phận có nhiệm vụ xử lý các yêu cầu người dùng đưa đến thông qua View, từ đó, đưa ra dữ liệu phù hợp với người dùng
  + Ngoài ra, Controller còn có chức năng kết nối với model.

Luồng tương tác giữa các thành phần trong MVC.

Ví dụ: Chức năng thêm bài viết mới trong trang quản trị website. Nơi đây có 2 trường nhập về tiêu đề và nội dung bài viết, thì trong đó:

* View sẽ hiển thị ra phần nhập form tiêu đề và nội dung.
* Controller lấy dữ liệu từ 2 trường và gửi tới Model.
* Model lúc này sẽ nhận dữ liệu từ Controller để lưu trữ trong cơ sở dữ liệu.

# Câu 15. Khi nào nên sử dụng MVC

* Khi cần 1 quy trình phát triển nhanh hơn
* Có khả năng cung cấp nhiều chế độ view
* Muốn các sửa đổi không ảnh hưởng đến toàn bộ mô hình
* Trả về dữ liệu mà không cần định dạng

# Câu 16. Cách hoạt động của MVC

* Model quản lý hành vi và dữ liệu của miền ứng dụng.
* Khi xử lý sự kiện, model thông báo cho các bộ quan sát (thường là view) khi thông tin được thay đổi để nó có phản ứng thích hợp.
* View render model thành một form phù hợp cho việc tương tác
* Trên cùng một model có thể có nhiều view để phục vụ cho các mục đích khác nhau
* Controller tiếp nhận yêu cầu người dùng và khởi động một respond bằng cách gọi đối tượng model.
* Controller nhận yêu cầu từ người dùng rồi chỉ định cho model và view thực hiện hành động dựa trên yêu cầu đó
* Một ứng dụng MVC là tập hợp của nhiều bộ ba model – controller – view, mỗi bộ chịu trách nhiệm với một UI khác nhau
* MVC thường thấy trên các ứng dụng web mà view là HTML hoặc XHTML được sinh ra bởi ứng dụng
* Controller nhận GET hoặc POST input và quyết định sẽ làm gì với nó
* Model không chỉ đơn thuần là một cơ sở dữ liệu, model trong MVC là cả dữ liệu và logic nghiệp vụ cần thiết để thao tác dữ liệu trong ứng dụng

# Câu 17. Ưu và nhược điểm của mô hình MVC

| **Ưu điểm** |
| --- |
| - **Tiết kiệm băng thông**: MVC không sử dụng viewstate nên sẽ giúp tiết kiệm băng thông cho website. Người dùng có thể sử dụng ứng dụng trên web cần tương tác để gửi và nhận dữ liệu một cách liên tục. Điều này cũng giúp giảm băng thông cho website hoạt động tốt cũng như ổn định hơn.  - **Dễ dàng kiểm tra**: Với MVCC, bạn có thể dễ dàng thực hiện các công việc như kiểm tra, rà soát lỗi phần mềm trước khi đưa tới tay người tiêu dùng. Việc này đảm bảo chất lượng và độ uy tín sản phẩm cao hơn.  **- Điều khiển đa dạng**:Sử dụng mô hình MVC sẽ giúp bạn điều khiển nền tảng của các ngôn ngữ lập trình hiện đại như HTML, CSS, Javascript,… với nhiều dạng khác nhau.  **- View và size:** View là nơi dùng để lưu trữ dữ liệu. Khi có yêu cầu được thực hiện càng lớn thì càng kích thích tệp lớn hơn. Khi đó, tốc độ đường truyền mạnh cũng bị giảm. Việc sử dụng mô hình MVC sẽ giúp bạn tiết kiệm được diện tích băng thông một cách tối ưu.  **- Chức năng SOC (Separation of Concern):** Chức năng này cho phép bạn có thể phân tách rõ ràng các thành phần như model, data, giao diện, nghiệp vụ.  **- Kết hợp:** Mô hình MVC cho phép bạn code thoải mái trên nền tảng web. giúp giảm tải dữ liệu máy chủ  **- Đơn giản:** Cấu trúc của mẫu MVC tương đối đơn giản và dễ sử dụng ngay cả khi bạn không có chuyên môn. |
| **Nhược điểm** |
| - Đối với các dự án có tính phức tạp cao thì mô hình MVC trở nên không khả dụng**.**  - Việc ứng dụng mô hình MVC trong các dự án nhỏ sẽ dễ bị cồng kềnh, tốn kém nguồn lực khi phát triển dự án. |

# Câu 18. Observer Pattern

* Định nghĩa phụ thuộc {một – nhiều} giữa các đối tượng sao cho khi một đối tượng thay đổi trạng thái, tất cả các phụ thuộc của nó sẽ tự động được thông báo và cập nhật.
* Dùng để tách chủ đề khỏi observer, do chủ đề cần ít thông tin để đánh thức observer

# Câu 19. Khái niệm Mysql

● Hệ quản trị cơ sở dữ liệu quan hệ

● Dùng cho các ứng dụng vừa và nhỏ

● Hỗ trợ chuẩn SQL

● Phần mềm mã nguồn mở, miễn phí

● Chạy trên nhiều platforms (Unix, Linux, Windows)

● Đơn giản, tốc độ nhanh

● Phổ biến (~ 8 triệu đơn vị/cá nhân cài đặt sử dụng)

● Tồn tại hơn 10 năm qua, có nguồn gốc từ mSQL (ISAM)

# Câu 20. Tính năng của mysql

* SQL cơ bản (tạo bảng, chèn/xóa/cập nhật mẫu tin, truy vấn, etc)
* Nhiều tính năng tiên tiến của SQL
* Những câu truy vấn phức tạp
* Ràng buộc khóa, ràng buộc dữ liệu, Trigger
* View (bảng ảo)
* Cursor
* Truy cập cạnh tranh

# Câu 21. Quản trị căn bản MySQL

■ Chạy/tắt MySQL server:

● Từ console gõ lệnh: /etc/init.d/mysqld [start|stop]

■ Đổi password của DBAdmin (root)

● Từ console gõ lệnh: mysqladmin –u root password xxxx

■ Làm việc từ mysql client

● Từ console gõ lệnh: mysql –u root -p

● Nhập password xxxx

● Dấu nhắc: mysql>

● exit: thoát, help: trợ giúp, etc.

■ Tạo một cơ sở dữ liệu: create database db\_name;

● Ví dụ: mysql> create database mydb;

■ Chọn một cơ sở dữ liệu để làm việc: use db\_name;

● Ví dụ: mysql> use mydb;

■ Xem thông tin về cơ sở dữ liệu và bảng

● Ví dụ: mysql> show databases;

● Ví dụ: mysql> show tables;

● Ví dụ: mysql> describe table\_name;

■ Xóa một cơ sở dữ liệu: drop database db\_name;

● Ví dụ: mysql> drop database mydb;

■ Tạo và phân quyền cho user mới:

● grant priv\_type privileges on db\_obj to username@”servername” identified by ‘password’;

● Ví dụ: mysql> grant all privileges on mydb.\* to nghi@”localhost” identified by ‘nghipasswd’;

■ Xóa phân quyền của user:

● revoke priv\_type privileges on db\_obj from username@”servername”;

● Ví dụ: mysql> revoke all privileges on mydb.\* from nghi@”localhost”;

● mysql> drop user nghi@”localhost”;

■ Backup cơ sở dữ liệu

● mysqldump --add-drop-table -u [username] -p[password] [db\_name] > [backup\_file]

● Ví dụ từ console: mysqldump --add-drop-table -u root –pxxxx mydb > mydb.bak

■ Phục hồi cơ sở dữ liệu

● mysql -u [username] -p[password] [db\_name] < [backup\_file]

● Ví dụ từ console: mysql -u root –pxxxx mydb < mydb.bak

■ Backup bảng của cơ sở dữ liệu: select into outfile from

● Ví dụ: mysql> select \* into outfile ‘/tmp/test.sql’ from test;

■ Phục hồi bảng của cơ sở dữ liệu: load data infile replace into table

● Ví dụ: mysql> load data infile ‘/tmp/test.sql’ replace into table test;

# Câu 22. Những nguyên tắc cơ bản của luật bản quyền

* Công ước Berne: bản quyền gắn kết với sự thể hiện của ý tưởng
* Bản quyền không bảo hộ ý tưởng mà chỉ bảo hộ sự thể hiện của ý tưởng đó
* Không ai ngoài tác giả có quyền tạo ra các tác phẩm kế thừa từ tác phẩm gốc có bản quyền
* Thời gian được bản quyền bảo vệ: phụ thuộc vào quốc gia
* Hoa kỳ: min(70 năm sau khi tác giả mất, 95 năm từ khi tác phẩm phát hành, 120 từ ngày tạo ra sản phẩm)
* Tác phẩm làm thuê (work for hire), bao gồm cả dịch thuật: bản quyền thuộc về người thuê hoặc tác giả của bản gốc (người ủy quyền).
* Sử dụng đẹp (fair use): Tác phẩm kế thừa có thay đổi: bản quyền thuộc về tác giả mới.
* Sau khi hết thời hạn bảo vệ các tác phẩm có bản quyền đi vào môi trường công cộng (public domain)

# Câu 23. Copyright

Bản quyền là một khái niệm pháp luật được chính phủ ban hành, giao cho người sáng tác (creator) của một tác phẩm nào đó những quyền riêng biệt trên tác phẩm của mình. “quyền sao chép” (the right of copy) là một dạng của sở hữu trí tuệ

# Câu 24. Copyleft

* Copyleft là một phương pháp tổng quát cho việc làm cho một chương trình/phần mềm trở nên tự do (free) và yêu cầu tất cả các phiên bản được chỉnh sửa hoặc mở rộng của nó cũng phải tự do.
* Copyleft phát biểu rằng bất cứ người nào phân phối lại phần mềm, có hay không thay đổi, phải để lại sự tự do sao chép và thay đổi.
* Copyleft bảo đảm cho MỌI NGƯỜI có sự tự do trên phần mềm.
* Copyleft một phần mềm: thừa nhận có bản quyền (copyrighted), thêm vào những phần liên quan đến việc phân phối: hướng dẫn hợp pháp cho quyền sử dụng, thay đổi và phân phối lại mã lệnh (code) của phần mềm hay bất cứ chương trình dẫn xuất từ nó chỉ với điều kiện là phần liên quan tới việc phân phối không thay đổi
* Các nhà phát triển phần mềm có quyền sở hữu dùng copyright để lấy đi sự tự do của người sử dụng.
* Copyleft dùng copyright để bảo đảm sự tự do của người sử dụng.
* Copyleft là một cách dùng copyright trên một chương trình (nó không có nghĩa là bỏ đi copyright).

# Câu 25. Phân loại phần mềm

Dựa trên 3 tiêu chí để phân loại:

* Khả năng phân phối lại
* Truy cập mã nguồn
* Phí sử dụng

Các loại phần mềm hiện tại

* Phần mềm tự do (free software)
* Phần mềm mã nguồn mở
* Phần mềm công cộng
* Phần mềm được copyleft
* Phần mềm tự do không được copyleft
* Phần mềm phân phối với giấy phép GPL
* Hệ thống GNU
* Phần mềm không tự do (non-free)
* Phần mềm bán tự do
* Phần mềm có chủ sở hữu (proprietary software)
* Phần mềm miễn phí (freeware)
* Phần mềm chia sẻ
* Phần mềm riêng tư (private software)
* Phần mềm thương mại

# Câu 26. Khái niệm PHP

* PHP là Hypertext Preprocessor
* Ngôn ngữ script chạy trên server
* PHP scripts chứa text, thẻ HTML, script
* Sử dụng phần mở rộng tên file : .php, .phtml
* PHP scripts sẽ trả về kết quả cho trình duyệt một plain HTML
* PHP hỗ trợ để làm việc với nhiều hệ QTCSDL khác nhau
* MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.
* Phần mềm mã nguồn mở, miễn phí
* Chạy trên nhiều platforms (Unix, Linux, Windows)

# Câu 27. Biến trong PHP

Biến người dùng định nghĩa

* Chứa dữ liệu
* Biến được bắt đầu bởi dấu $
* Tên biến bắt đầu bằng một ký tự chữ cái hoặc \_
* Phân biệt giữa ký tự thường và hoa
* Kiểu được tính ở thời điểm gán giá trị
* Gán giá trị với =
* Sử dụng & như tham chiếu

Biến có sẵn trong PHP

* $GLOBALS : tất cả các biến trong phạm vi toàn cục của script
* $\_SERVER : tập hợp biến môi trường của Web server
* $\_GET, $\_POST : biến được cung cấp các chuỗi query URL cho script
* $\_COOKIE : biến cung cấp HTTP cookies cho script
* $\_FILES : biến cung cấp HTTP POST file uploads cho script
* $\_ENV : biến cung cấp môi trường cho script
* $\_REQUEST : cung cấp các $\_GET, $\_POST, $\_COOKIE

Phạm vi biến

* Toàn cục : sử dụng từ khóa global hoặc biến $GLOBALS
* Cục bộ
* Biến tĩnh : sử dụng từ khóa static

# Câu 28. Ứng dụng quản lý nội dung (CMA)

Khái niệm: CMA (Content Management Application) là giao diện người dùng của một trang web cho phép bạn thiết kế, thay đổi và xóa nội dung trên trang web của mình

* Ứng dụng quản lý nội dung (CMA) quản lý vòng đời đầy đủ của các thành phần nội dung từ khi bắt đầu đến khi xóa bỏ chúng.
* CMA tạo các thành phần nội dung trong kho lưu trữ, bảo trì chúng trong thời gian chúng tồn tại, và loại bỏ các thành phần nội dung này khỏi kho lưu trữ khi không cần thiết

# Câu 29. Ứng dụng quản lý cấu trúc nội dung Metacontent management Application (MMA)

* MMA là ứng dụng quản lý vòng đời đầy đủ của cấu trúc nội dung.
* Ta có thể định nghĩa cấu trúc nội dung, đặc biệt là cách các thành phần nội dung được hiển thị trên các website như thế nào.
* Mục đích của MMA là sinh ra các cấu trúc nội dung thay vì các thành phần nội dung

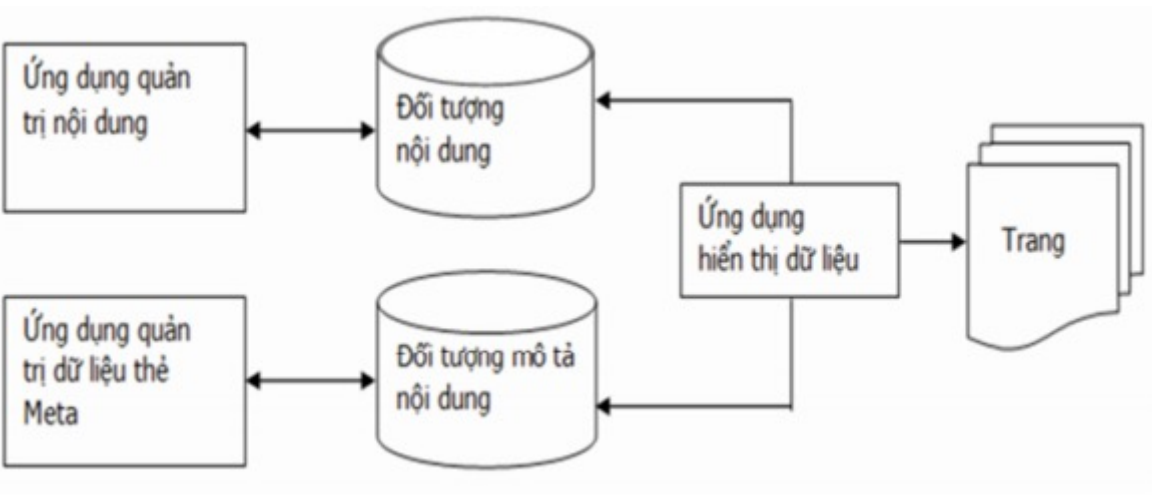
# Câu 30. Ứng dụng phân phối nội dung Content Delivery Application (CDA)

* Công việc của ứng dụng phân phối nội dung CDA là lấy nội dung từ nơi “Quản trị nội dung” và hiển thị chúng theo những định nghĩa trong MMA ra trang web.
* CDA chỉ có quyền đọc kho lưu trữ, điều này sẽ tăng khả năng bảo mật cho trang web.
* CDA cung cấp cho người sử dụng là hàm tìm kiếm trên một thực thể nội dung.

# Câu 31. Giới thiệu hệ thống quản lý nội dung Content Management System (CMS)

* Là một ứng dụng người sử dụng xây dựng và quản lý nội dung (text, video, nhạc, hình ảnh, files…) website mà không cần sự can thiệp, hỗ trợ từ người lập trình trang web..
* Với CMS, bạn có thể tạo dựng, quản lý, chỉnh sửa và xuất bản nội dung một cách dễ dàng và thân thiện với người dùng (user-friendly interface). Bạn hoàn toàn có thể tùy chỉnh thiết kế và các chức năng của website bằng cách tải về hoặc mua template và extension, không cần code gì cả.
* Các chức năng cơ bản:
  + Tạo, lưu trữ các nội dung
  + Chỉnh sửa, thêm, bớt nội dung
  + Chuyển và chia sẻ nội dung
  + Quản lý và phân quyền sử dụng cho người dùng
* Các loại CMS:
  + Open Source
  + Tự code hoặc framwork
  + Mất phí
* Các loại CMS được yêu thích: Joomla, Drupal, WordPress,...
* Các CMS trên sử dụng ngôn ngữ PHP, hệ quản trị MySQL.

**Cấu trúc**

****

**Phần 2: Code**

**//Kiểm tra chuỗi đối xứng**

function palindromeString($string)

{

for ($i = 0; $i < strlen($string); $i++) {

if ($string[$i] != $string[strlen($string) - $i - 1]) {

return 0;

}

}

return 1;

}

echo palindromeString("AbccbA");

**//Đảo ngược 1 chuỗi**

function reverseString($str)

{

for ($i = 0; $i < strlen($str) / 2; $i++) {

$tg = $str[$i];

$str[$i] = $str[strlen($str) - $i - 1];

$str[strlen($str) - $i - 1] = $tg;

}

return $str;

}

echo reverseString("Abcsdh");

**//Tìm số lượng tương ứng với mỗi ký tự và in chuỗi gồm số lượng và ký tự**

function stringToNumber($string)

{

$tempStr = '';

$count = 1;

for ($i = 0; $i < strlen($string); $i++) {

if ($i == strlen($string) - 1) {

if ($string[$i] != $string[$i - 1]) {

$tempStr .= '1' . $string[$i];

} else {

$tempStr .= $count . $string[$i];

}

break;

} else {

if ($string[$i] == $string[$i + 1]) {

$count++;

} else {

$tempStr .= $count . $string[$i];

$count = 1;

}

}

}

return $tempStr;

}

**//Xóa các ký tự giống nhau**

function deleteRepetitiveCharacters($str)

{

$tempstr = $str[0];

for ($i = 1; $i < strlen($str); $i++) {

$kt = true;

for ($j = $i - 1; $j >= 0; $j--) {

if ($str[$i] == $str[$j]) {

$kt = false;

break;

}

}

if ($kt == true) {

$tempstr .= $str[$i];

}

}

return $tempstr;

}