

BÀI TẬP THỰC HÀNH SỐ 3

Học phần: CSE485 - Công nghệ Web

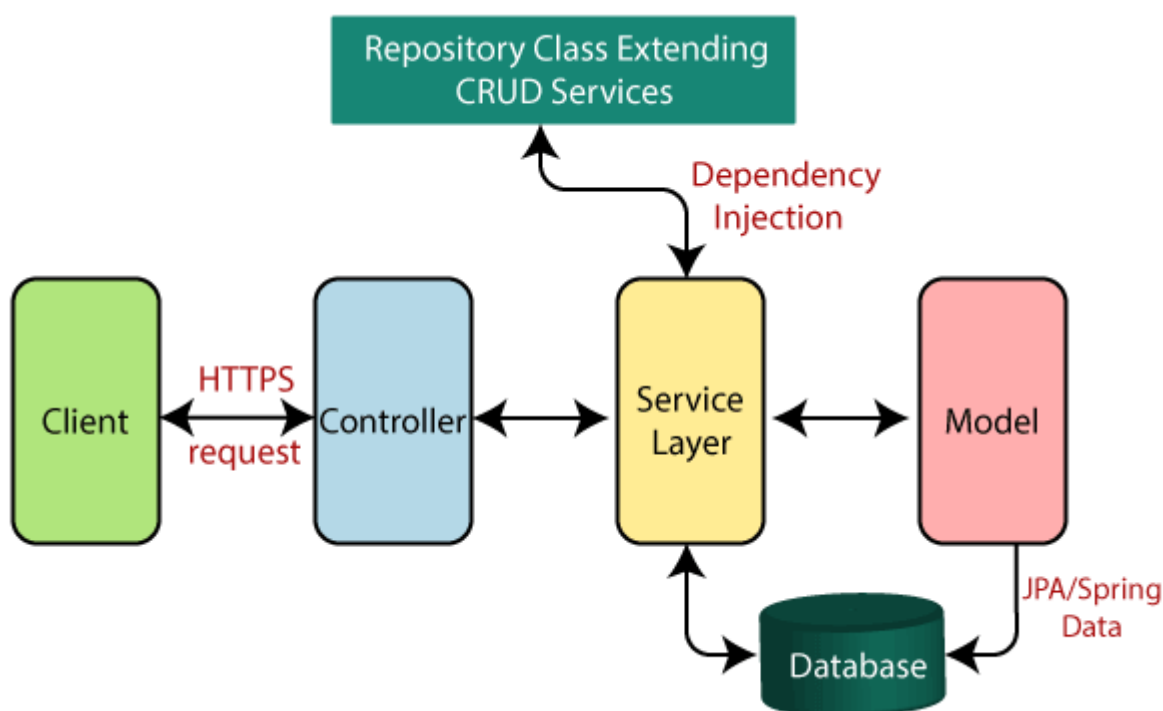
Bài tập 1: Sử dụng CSDL phpbook-1 đã có làm ví dụ minh họa.

Bài tập 2: Sử dụng code **MVC simple minh họa** cho trước (tải về từ VietCodeDi)

Đọc code và hiểu được luồng hoạt động của ứng dụng theo MVC

Người dùng Request ➡ Routes (index.php) đóng vai trò phân tích Request để xác định Controller và action (hàm cần gọi trong controller) cần xử lý ➡ Controller được xác định tiếp nhận xử lý và gọi ra hàm được chỉ định: Làm việc với Service và View tương ứng.

Mô hình hoạt động gần tương tự kiến trúc dưới đây (Tham khảo kiến trúc Spring Boot)

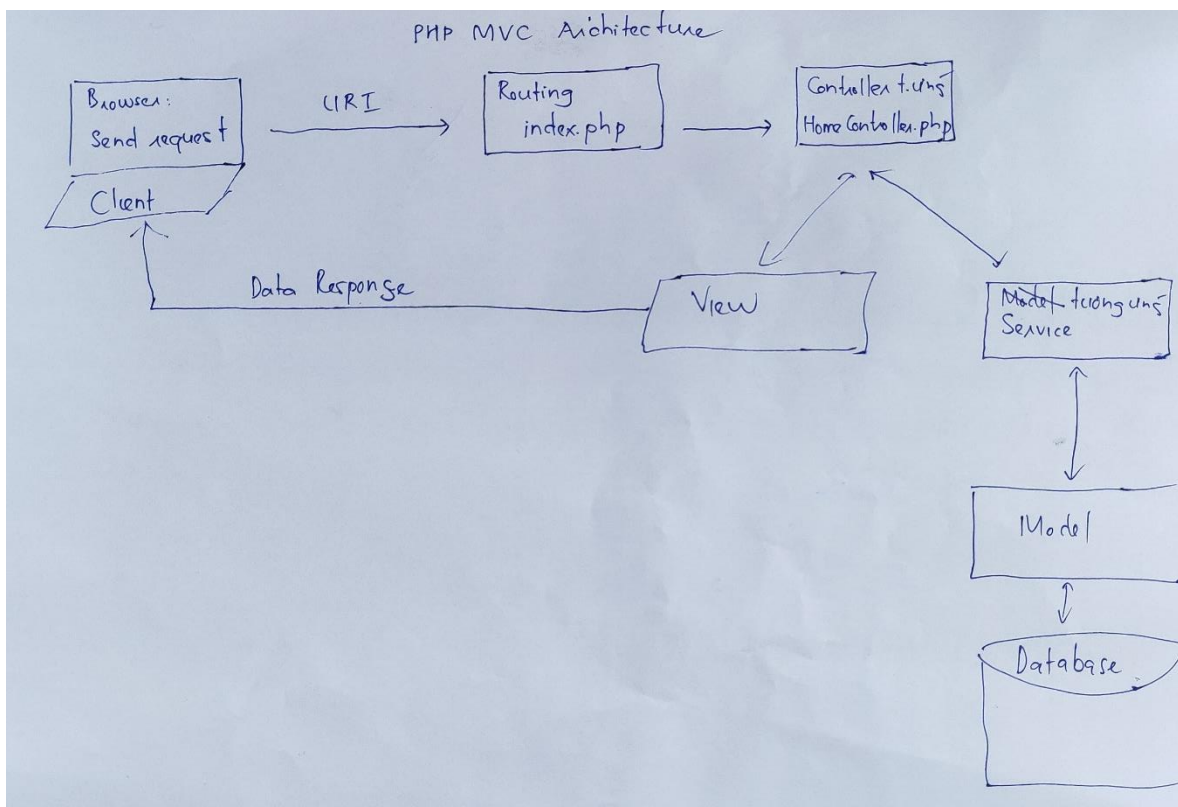


Hoặc xem chi tiết ở trang (2)

Bài tập 3: Viết chức năng theo yêu cầu sau:

- Định nghĩa Article Model
- Định nghĩa ArticleService: thực hiện các hoạt động thêm/sửa/xóa/truy vấn dữ liệu từ Article
- Định nghĩa ArticleController: điều khiển logic các hoạt động thao tác dữ liệu qua ArticleService và view tương ứng liên quan đến Article.
- Định nghĩa HomeController: điều khiển logic các hoạt động thao tác dữ liệu lấy dữ liệu từ Article và hiển thị lên trang chủ.


MÔ HÌNH KIẾN TRÚC MVC MẪU



Tham khảo cấu trúc dự án nên tạo:

CSS Copy code

```
article-website/  
├─ app/  
│   ├─ controllers/  
│   │   └─ ArticleController.php  
│   ├─ services/  
│   │   └─ ArticleService.php  
│   ├─ models/  
│   │   └─ Article.php  
│   └─ views/  
│       └─ article/  
│           ├─ index.php  
│           ├─ create.php  
│           ├─ edit.php  
│           └─ show.php  
├─ public/  
│   ├─ css/  
│   │   └─ style.css  
│   ├─ js/  
│   │   └─ script.js  
│   └─ index.php  
├─ database/  
│   └─ article_website.sql  
└─ README.md
```

 Regenerate response

HƯỚNG DẪN GIẢI CHI TIẾT

1. Tạo cấu trúc thư mục dự án ở ảnh trên.

2. Thiết lập CSDL:

- Tạo CSDL mới và thực thi các lệnh từ tệp phpbook-1.sql để tạo Bảng và dữ liệu minh họa.
- Hoặc tự tạo lại CSDL và dữ liệu minh họa

3. Định nghĩa routes:

- Tạo, mở tệp **routes.php** trong thư mục **app**.
- Định nghĩa các route tương ứng với các hành động trong ArticleController (ví dụ., index, create, edit, show).
- Ánh xạ mỗi route tới phương thức điều khiển tương ứng (ví dụ: GET **/articles** ánh xạ tới **ArticleController@index**).

4. Tạo Article model:

- Bên trong thư mục **models**, tạo một lớp PHP và đặt tên là **Article**.
- Xác định các thuộc tính lớp đại diện cho các thuộc tính của bài viết (ví dụ, id, title, content, created_at).
- Triển khai các phương thức để tương tác với cơ sở dữ liệu, chẳng hạn như lấy dữ liệu bài viết, tạo bài viết mới, cập nhật bài viết và xóa bài viết..

5. Tạo ArticleService:

- Bên trong thư mục **services**, tạo một lớp PHP và đặt tên là **ArticleService**.
- Triển khai các phương thức gói gọn logic nghiệp vụ liên quan đến bài viết, chẳng hạn như xác thực, thao tác dữ liệu hoặc truy vấn phức tạp.
- Lớp ArticleService có thể sử dụng mô hình Article để thực hiện các thao tác cơ sở dữ liệu.

6. Tạo ArticleController:

- Bên trong thư mục **controllers**, tạo một lớp PHP và đặt tên là **ArticleController**.
- Triển khai các phương thức tương ứng với các route đã xác định trong tệp **routes.php**.
- Sử dụng ArticleService để xử lý logic cho từng route.
- Truy xuất dữ liệu từ service, chuẩn bị và chuyển dữ liệu đến các view.

7. Tạo các view:

- Bên trong thư mục **views**, tạo một thư mục con, đặt tên là **article**.
- Bên trong thư mục con **article**, tạo các tệp view cho mỗi route (ví dụ, **index.php**, **create.php**, **edit.php**, **show.php**).
- Sử dụng HTML, CSS và PHP để tạo giao diện người dùng và hiển thị dữ liệu nhận được từ controller.
- Bao gồm gửi biểu mẫu, xác thực và tương tác với ArticleService khi cần thiết.

8. Triển khai route và view tương ứng:

- Bên trong ArticleController, tạo phương thức **index()** để lấy dữ liệu tất cả các bài viết từ ArticleService và truyền tới **article/index.php**.
- Trong **article/index.php**, duyệt và hiển thị bài viết trên bảng.

9. Triển khai route tạo và thêm với view tương ứng:

- Trong ArticleController, tạo một phương thức **create()** mà kết xuất ra **article/create.php**, chứa form để thêm bài viết với.
- Trong **article/create.php**, hiển thị form để thêm bài viết
- Gửi dữ liệu từ form tới phương thức **store()** của ArticleController, mà sử dụng ArticleService để lưu bài viết mới vào CSDL.

10. Triển khai route sửa và cập nhật với view tương ứng:

- Trong ArticleController, tạo một phương thức **edit(\$id)** mà lấy bản ghi của article theo id qua ArticleService và truyền dữ liệu đó tới **article/edit.php**.
- Trong **article/edit.php**, hiển thị một biểu mẫu được điền sẵn dữ liệu hiện tại của bài viết.
- Gửi biểu mẫu tới phương thức **update(\$id)** của ArticleController, phương thức này sử dụng ArticleService để cập nhật bài viết trong cơ sở dữ liệu.

11. Triển khai route show với view tương ứng:

- Trong ArticleController, hãy tạo một phương thức show(\$id) để tìm nạp bài viết với ID đã cho bằng ArticleService và chuyển nó tới **article/show.php**.
- Trong **article/show.php**, hiển thị tiêu đề, nội dung của bài viết và bất kỳ thông tin liên quan nào khác.

12. Cấu hình autoloading (tự nạp) và dependencies (phụ thuộc):

- Thực hiện cơ chế tự động tải để tự động tải các lớp cần thiết khi cần.
- Định cấu hình mọi phụ thuộc cần thiết, chẳng hạn như kết nối cơ sở dữ liệu hoặc thư viện, trong tệp cấu hình trung tâm.

13. Thiết lập front controller:

- Trong thư mục **public**, tạo một tệp **index.php** đóng vai trò là điểm vào cho tất cả các yêu cầu.
- Sử dụng tệp **routes.php** để xác định các tuyến khả dụng và xử lý các yêu cầu đến bằng cách gửi chúng đến các phương thức điều khiển tương ứng.

14. Tạo các tệp CSS và JavaScript:

- Bên trong thư mục **public**, hãy tạo các thư mục con **css** và **js** để lưu trữ tệp định kiểu và tệp JavaScript tương ứng.
- Đưa các tệp này vào các **view** phù hợp để nâng cao khả năng trình bày và tính tương tác của trang web..

15. Kiểm thử và tinh chỉnh:

- Chạy ứng dụng và kiểm tra chức năng của từng tuyến đường và chế độ xem..
- Tinh chỉnh việc triển khai nếu cần, sửa bất kỳ lỗi nào hoặc thêm các tính năng bổ sung dựa trên yêu cầu.

Bằng cách làm theo các bước này, bạn có thể xây dựng một trang web quản lý bài viết bằng cách sử dụng mẫu MVC trong PHP. Cấu trúc MVC giúp phân tách các mối quan tâm và làm cho mã có tổ chức hơn, dễ bảo trì và có thể mở rộng hơn. Mỗi thành phần (mô hình, dạng xem, bộ điều khiển) có một vai trò cụ thể, thúc đẩy khả năng sử dụng lại mã và cải thiện cấu trúc tổng thể của ứng dụng của bạn.

HẾT