TRƯỜNG ĐẠI HỌC TRÀ VINH

KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN MÔN HỌC**

**THIẾT KẾ GIAO DIỆN WEBSITE REVIEW TRUYỆN ONLINE**

|  |  |
| --- | --- |
| **Giáo viên hướng dẫn:**  Nguyễn Ngọc Đan Thanh | **Sinh viên thực hiện:**  Mã số sinh viên: 110122191  Họ tên: Kim Thạch Minh Trí  Lớp: DA22TTC |

***Trà Vinh, tháng 4 năm 2024***

TRƯỜNG ĐẠI HỌC TRÀ VINH

KHOA KỸ THUẬT VÀ CÔNG NGHỆ

**BỘ MÔN CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN MÔN HỌC**

**THIẾT KẾ GIAO DIỆN WEBSITE REVIEW TRUYỆN ONLINE**

|  |  |
| --- | --- |
| **Giáo viên hướng dẫn:**  Nguyễn Ngọc Đan Thanh | **Sinh viên thực hiện:**  Mã số sinh viên: 110122191  Họ tên: Kim Thạch Minh Trí  Lớp: DA22TTC |

***Trà Vinh, tháng 4 năm 2024***

**MỤC LỤC**

[CHƯƠNG 1. Tổng quan về nội dung nghiên cứu 1](#_Toc167201778)

[1.1. Quy trình phát triển website 1](#_Toc167201779)

[1.1.1. Lên kế hoạch và thu thập yêu cầu: 1](#_Toc167201780)

[1.1.2. Thiết kế giao diện: 1](#_Toc167201781)

[1.1.3. Phát triển website: 1](#_Toc167201782)

[1.1.4. Thử nghiệm và triển khai: 1](#_Toc167201783)

[1.1.5. Bảo trì và cập nhật: 1](#_Toc167201784)

[1.1.6. Ngoài ra, quy trình phát triển website có thể bao gồm thêm các bước sau: 2](#_Toc167201785)

[1.2. Ngôn ngữ HTML 2](#_Toc167201786)

[1.2.1. Ngôn ngữ HTML là gì ? 2](#_Toc167201787)

[1.2.2. Đặc điểm của HTML: 2](#_Toc167201788)

[1.2.3. Cấu trúc cơ bản của một trang HTML: [1] [2] 2](#_Toc167201789)

[1.2.4. Danh sách các phần tử HTML: 3](#_Toc167201790)

[1.2.4.1 Phần tử cơ bản: [2] [3] [4] 3](#_Toc167201791)

[1.2.4.2 Phần tử định dạng văn bản: [2] [3] [4] 4](#_Toc167201792)

[1.2.4.3 Phần tử tạo liên kết: [2] [3] [4] 4](#_Toc167201793)

[1.2.4.4 Phần tử tạo danh sách: [2] [3] [4] 4](#_Toc167201794)

[1.2.4.5 Phần tử tạo hình ảnh: [2] [3] [4] 5](#_Toc167201795)

[1.2.4.6 Phần tử tạo bảng: [2] [3] [4] 5](#_Toc167201796)

[1.2.4.7 Phần tử tạo biểu mẫu: [2] [3] [4] 5](#_Toc167201797)

[1.2.4.8 Phần tử tạo khung: [2] [3] [4] 5](#_Toc167201798)

[1.2.4.9 Phần tử tạo âm thanh và video: [2] [3] [4] 5](#_Toc167201799)

[1.2.4.10 Phần tử tạo các yếu tố khác: [2] [3] [4] 5](#_Toc167201800)

[1.2.5. Danh sách các thuộc tính HTML: 6](#_Toc167201801)

[1.2.5.1 Thuộc tính chung: [2] [3] [4] 6](#_Toc167201802)

[1.2.5.2 Thuộc tính cho phần tử <a>: [2] [3] [4] 6](#_Toc167201803)

[1.2.5.3 Thuộc tính cho phần tử <img>: [2] [3] [4] 6](#_Toc167201804)

[1.2.5.4 Thuộc tính cho phần tử <table>: [2] [3] [4] 6](#_Toc167201805)

[1.2.5.5 Thuộc tính cho phần tử <tr>: [2] [3] [4] 7](#_Toc167201806)

[1.2.5.6 Thuộc tính cho phần tử <th> và <td>: [2] [3] [4] 7](#_Toc167201807)

[1.2.5.7 Thuộc tính cho phần tử <input>: [2] [3] [4] 7](#_Toc167201808)

[1.2.5.8 Thuộc tính cho phần tử <select>: [2] [3] [4] 7](#_Toc167201809)

[1.2.6. Tiêu đề trong HTML 7](#_Toc167201810)

[1.2.6.1 Các cấp tiêu đề: 7](#_Toc167201811)

[1.2.6.2 Cách sử dụng thẻ tiêu đề: 8](#_Toc167201812)

[1.2.6.3 Lưu ý: 8](#_Toc167201813)

[1.2.6.4 Một số mẹo sử dụng thẻ tiêu đề hiệu quả: 8](#_Toc167201814)

[1.2.7. Đoạn văn bản trong HTML 9](#_Toc167201815)

[1.2.7.1 Cách sử dụng thẻ <p>: 9](#_Toc167201816)

[1.2.7.2 Lưu ý: 9](#_Toc167201817)

[1.2.7.3 Một số mẹo sử dụng thẻ <p> hiệu quả: 9](#_Toc167201818)

[1.2.8. Định dạng văn bản trong HTML 9](#_Toc167201819)

[1.2.8.1 Sử dụng thẻ định dạng: [2] [3] [4] 10](#_Toc167201820)

[1.2.8.2 Sử dụng thuộc tính CSS: 10](#_Toc167201821)

[1.2.8.3 Sử dụng các lớp CSS: 11](#_Toc167201822)

[1.2.8.4 Một số lưu ý: 11](#_Toc167201823)

[1.2.9. Nhận xét trong HTML 12](#_Toc167201824)

[1.2.9.1 Cách viết nhận xét: 12](#_Toc167201825)

[1.2.9.2 Ví dụ: 12](#_Toc167201826)

[1.2.9.3 Lưu ý: 12](#_Toc167201827)

[1.2.9.4 Một số mẹo sử dụng nhận xét hiệu quả: 12](#_Toc167201828)

[1.2.10. Liên kết trong HTML 13](#_Toc167201829)

[1.2.10.1 Cấu trúc của thẻ <a>: 13](#_Toc167201830)

[1.2.10.2 Ví dụ: 13](#_Toc167201831)

[1.2.10.3 Một số lưu ý: 13](#_Toc167201832)

[1.2.10.4 Một số mẹo sử dụng liên kết hiệu quả: 14](#_Toc167201833)

[1.2.11. Danh sách trong HTML 14](#_Toc167201834)

[1.2.11.1 Danh sách không theo thứ tự (Unordered List): [2] [3] [4] 14](#_Toc167201835)

[1.2.11.2 Danh sách theo thứ tự (Ordered List): [2] [3] [4] 15](#_Toc167201836)

[1.2.11.3 Định dạng danh sách: [2] [3] [4] 15](#_Toc167201837)

[1.2.11.4 Danh sách lồng nhau: [2] [3] [4] 16](#_Toc167201838)

[1.2.11.5 Một số lưu ý: 16](#_Toc167201839)

[1.2.12. Hình ảnh trong HTML 16](#_Toc167201840)

[1.2.12.1 1. Cấu trúc thẻ <img>: 17](#_Toc167201841)

[1.2.12.2 Một số lưu ý: 17](#_Toc167201842)

[1.2.12.3 Một số mẹo sử dụng hình ảnh hiệu quả: 17](#_Toc167201843)

[1.2.13. Bảng trong HTML 18](#_Toc167201844)

[1.2.13.1 Cấu trúc cơ bản của bảng: 18](#_Toc167201845)

[1.2.13.2 Giải thích: 18](#_Toc167201846)

[1.2.13.3 Một số thuộc tính hữu ích: [2] [3] [4] 18](#_Toc167201847)

[1.2.13.4 Một số lưu ý: 19](#_Toc167201848)

[1.2.14. Biểu mẫu trong HTML 19](#_Toc167201849)

[1.2.14.1 Cấu trúc cơ bản của biểu mẫu: [2] [3] [4] 19](#_Toc167201850)

[1.2.14.2 . Giải thích: [2] [3] [4] 20](#_Toc167201851)

[1.2.14.3 Một số loại phần tử nhập dữ liệu phổ biến: [2] [3] [4] 20](#_Toc167201852)

[1.2.14.4 Một số lưu ý: 21](#_Toc167201853)

[1.2.15. Khung trong HTML (Iframe) 21](#_Toc167201854)

[1.2.15.1 Cấu trúc cơ bản của Iframe: [2] [3] [4] 21](#_Toc167201855)

[1.2.15.2 Một số thuộc tính hữu ích khác của Iframe: [2] [3] [4] 22](#_Toc167201856)

[1.2.15.3 Một số lưu ý: 22](#_Toc167201857)

[1.2.16. Phần tử khối và phần tử nội tuyến trong HTML 22](#_Toc167201858)

[1.2.16.1 Phân biệt phần tử khối và phần tử nội tuyến: 22](#_Toc167201859)

[1.2.16.2 Ví dụ minh họa: 23](#_Toc167201860)

[1.2.16.3 Một số lưu ý: 23](#_Toc167201861)

[1.2.17. Đường dẫn tệp trong HTML 24](#_Toc167201862)

[1.2.17.1 Loại đường dẫn tệp: 24](#_Toc167201863)

[1.2.17.2 Sử dụng đường dẫn tệp: 24](#_Toc167201864)

[1.2.17.3 Một số lưu ý: 25](#_Toc167201865)

[1.2.18. Âm thanh và video trong HTML 25](#_Toc167201866)

[1.2.18.1 Âm thanh: [2] [3] [4] 25](#_Toc167201867)

[1.2.18.2 Video: [2] [3] [4] 25](#_Toc167201868)

[1.2.18.3 Một số lưu ý: 26](#_Toc167201869)

[1.3. Bảng định kiểu CSS 26](#_Toc167201870)

[1.3.1. CSS là gì ? 26](#_Toc167201871)

[1.3.2. Một số đặc điểm của CSS: 26](#_Toc167201872)

[1.3.3. Vai trò của CSS: 27](#_Toc167201873)

[1.3.4. Cấu trúc cơ bản của CSS: [5] [6] 27](#_Toc167201874)

[1.3.5. Một số thuộc tính CSS phổ biến: [5] [6] 28](#_Toc167201875)

[1.3.6. Một số lưu ý: 28](#_Toc167201876)

[1.3.7. CSS nội tuyến (Inline CSS) 29](#_Toc167201877)

[1.3.7.1 Cấu trúc: 29](#_Toc167201878)

[1.3.7.2 .Ưu điểm: 29](#_Toc167201879)

[1.3.7.3 Nhược điểm: 29](#_Toc167201880)

[1.3.7.4 Khi nào nên sử dụng CSS nội tuyến: 30](#_Toc167201881)

[1.3.7.5 Nên hạn chế sử dụng CSS nội tuyến cho các trường hợp sau: 30](#_Toc167201882)

[1.3.8. CSS nội bộ (Internal CSS) 30](#_Toc167201883)

[1.3.8.1 Cấu trúc: 30](#_Toc167201884)

[1.3.8.2 Ưu điểm: 31](#_Toc167201885)

[1.3.8.3 Nhược điểm: 31](#_Toc167201886)

[1.3.8.4 Khi nào nên sử dụng CSS nội bộ: 31](#_Toc167201887)

[1.3.8.5 Nên hạn chế sử dụng CSS nội bộ cho các trường hợp sau: 32](#_Toc167201888)

[1.3.9. CSS bên ngoài (External CSS) 32](#_Toc167201889)

[1.3.9.1 Cấu trúc: 32](#_Toc167201890)

[1.3.9.2 Ưu điểm: 33](#_Toc167201891)

[1.3.9.3 Nhược điểm: 33](#_Toc167201892)

[1.3.9.4 Khi nào nên sử dụng CSS bên ngoài: 34](#_Toc167201893)

[1.3.9.5 Nên hạn chế sử dụng CSS bên ngoài cho các trường hợp sau: 34](#_Toc167201894)

[1.3.10. Phông chữ trong CSS 34](#_Toc167201895)

[1.3.10.1 Thuộc tính font-family: 34](#_Toc167201896)

[1.3.10.2 Thuộc tính font-size: 35](#_Toc167201897)

[1.3.10.3 Thuộc tính font-weight: 35](#_Toc167201898)

[1.3.10.4 Thuộc tính font-style: 35](#_Toc167201899)

[1.3.10.5 Thuộc tính line-height: 36](#_Toc167201900)

[1.3.10.6 Một số thuộc tính khác: 36](#_Toc167201901)

[1.3.10.7 Ví dụ: 36](#_Toc167201902)

[1.3.10.8 Lưu ý: 37](#_Toc167201903)

[1.3.11. Màu sắc cho văn bản trong CSS 37](#_Toc167201904)

[1.3.11.1 Sử dụng mã màu HEX: 37](#_Toc167201905)

[1.3.11.2 Sử dụng tên màu: 37](#_Toc167201906)

[1.3.11.3 Sử dụng giá trị RGB: 37](#_Toc167201907)

[1.3.11.4 Sử dụng giá trị HSL: 38](#_Toc167201908)

[1.3.11.5 Sử dụng chức năng rgba() và hsla(): 38](#_Toc167201909)

[1.3.11.6 Lưu ý: 38](#_Toc167201910)

[1.3.12. Căn chỉnh văn bản trong CSS 38](#_Toc167201911)

[1.3.12.1 Căn chỉnh theo chiều ngang: [5] [6] [7] 38](#_Toc167201912)

[1.3.12.2 Căn chỉnh theo chiều dọc: [5] [6] [7] 39](#_Toc167201913)

[1.3.12.3 Sử dụng các giá trị khác: [5] [6] [7] 39](#_Toc167201914)

[1.3.12.4 Lưu ý: 40](#_Toc167201915)

[1.3.13. Khoảng cách giữa các phần tử trong CSS 40](#_Toc167201916)

[1.3.13.1 Thuộc tính margin: 40](#_Toc167201917)

[Ví dụ: 40](#_Toc167201918)

[1.3.13.2 Thuộc tính padding: 41](#_Toc167201919)

[1.3.13.3 Thuộc tính border-spacing: 41](#_Toc167201920)

[1.3.13.4 Sử dụng các đơn vị: 41](#_Toc167201921)

[1.3.13.5 Sử dụng các thuộc tính flexbox và grid: 42](#_Toc167201922)

[1.3.13.6 Lưu ý: 42](#_Toc167201923)

[1.3.14. Chiều rộng và chiều cao của phần tử trong CSS 42](#_Toc167201924)

[1.3.14.1 Thuộc tính width: 42](#_Toc167201925)

[1.3.14.2 Thuộc tính height: 43](#_Toc167201926)

[1.3.14.3 Sử dụng các đơn vị: 43](#_Toc167201927)

[1.3.14.4 Giá trị tự động: 43](#_Toc167201928)

[1.3.14.5 Sử dụng giá trị max-width và max-height: 43](#_Toc167201929)

[1.3.14.6 Sử dụng giá trị min-width và min-height: 44](#_Toc167201930)

[1.3.14.7 Sử dụng các thuộc tính flexbox và grid: 44](#_Toc167201931)

[1.3.14.8 Lưu ý: 44](#_Toc167201932)

[1.3.15. Màu nền cho phần tử trong CSS 44](#_Toc167201933)

[1.3.15.1 Cú pháp: 45](#_Toc167201934)

[1.3.15.2 Ví dụ: 45](#_Toc167201935)

[1.3.15.3 Giá trị màu sắc: 45](#_Toc167201936)

[1.3.15.4 Hình ảnh nền: 45](#_Toc167201937)

[1.3.15.5 Thuộc tính liên quan: 46](#_Toc167201938)

[1.3.15.6 Lưu ý: 46](#_Toc167201939)

[1.3.16. Đường viền cho phần tử trong CSS 46](#_Toc167201940)

[1.3.16.1 Thuộc tính border: 46](#_Toc167201941)

[1.3.16.2 Cú pháp: 47](#_Toc167201942)

[1.3.16.3 Các thuộc tính riêng lẻ: [5] [6] [7] 47](#_Toc167201943)

[1.3.16.4 Giá trị đường viền: [5] [6] [7] 47](#_Toc167201944)

[1.3.16.5 Đường viền cho các cạnh riêng lẻ: [5] [6] [7] 48](#_Toc167201945)

[1.3.16.6 Lưu ý: 48](#_Toc167201946)

[1.3.17. Cách hiển thị phần tử trong CSS 48](#_Toc167201947)

[1.3.17.1 Sử dụng thuộc tính display: 49](#_Toc167201948)

[1.3.17.2 Sử dụng thuộc tính visibility: 49](#_Toc167201949)

[1.3.17.3 Sử dụng thuộc tính opacity: 50](#_Toc167201950)

[1.3.17.4 Sử dụng các kỹ thuật bố cục: 50](#_Toc167201951)

[1.3.17.5 Lưu ý: 50](#_Toc167201952)

[1.3.18. Vị trí của phần tử trong CSS 51](#_Toc167201953)

[1.3.18.1 Thuộc tính position: 51](#_Toc167201954)

[1.3.18.2 Các thuộc tính vị trí khác: 52](#_Toc167201955)

[1.3.18.3 Lưu ý: 52](#_Toc167201956)

[1.4. Ngôn ngữ Javascript 52](#_Toc167201957)

[1.4.1. Ngôn ngữ JavaScript 52](#_Toc167201958)

[1.4.2. Đặc điểm chính của JavaScript: 53](#_Toc167201959)

[1.4.3. Ứng dụng của JavaScript: 54](#_Toc167201960)

[1.4.4. Ưu điểm của JavaScript: 54](#_Toc167201961)

[1.4.5. Nhược điểm của JavaScript: 54](#_Toc167201962)

[1.4.6. Cách thêm JavaScript vào trang web 55](#_Toc167201963)

[1.4.6.1 Sử dụng thẻ <script>: 55](#_Toc167201964)

[1.4.6.2 Sử dụng thẻ <script> trong phần <body>: 55](#_Toc167201965)

[1.4.6.3 Lưu ý: 56](#_Toc167201966)

[1.4.7. Javascript cơ bản 56](#_Toc167201967)

[1.4.7.1 Các kiểu dữ liệu: 56](#_Toc167201968)

[1.4.7.2 Các toán tử: 57](#_Toc167201969)

[1.4.7.3 Các biến: 57](#_Toc167201970)

[1.4.7.4 Cấu trúc điều khiển: 57](#_Toc167201971)

[1.4.7.5 Hàm: 57](#_Toc167201972)

[1.4.7.6 Sự kiện: 58](#_Toc167201973)

[1.4.7.7 DOM (Document Object Model): 58](#_Toc167201974)

[1.4.8. Các kiểu dữ liệu trong Javascript 58](#_Toc167201975)

[1.4.8.1 Kiểu dữ liệu nguyên thủy (Primitive Data Types): 58](#_Toc167201976)

[1.4.8.2 Kiểu dữ liệu tham chiếu (Reference Data Types): 59](#_Toc167201977)

[1.4.8.3 Lưu ý: 59](#_Toc167201978)

[1.4.8.4 Ngoài ra, Javascript còn có một số kiểu dữ liệu khác: 59](#_Toc167201979)

[1.4.8.5 Để kiểm tra kiểu dữ liệu của một biến, bạn có thể sử dụng toán tử typeof: [8] [9] [10] [11] 60](#_Toc167201980)

[1.4.9. Các toán tử trong Javascript 60](#_Toc167201981)

[1.4.9.1 Toán tử số học: [8] [9] [10] [11] 60](#_Toc167201982)

[1.4.9.2 Toán tử so sánh: [8] [9] [10] [11] 61](#_Toc167201983)

[1.4.9.3 Toán tử logic: [8] [9] [10] [11] 61](#_Toc167201984)

[1.4.9.4 Toán tử gán: [8] [9] [10] [11] 62](#_Toc167201985)

[1.4.9.5 Toán tử chuỗi: [8] [9] [10] [11] 62](#_Toc167201986)

[1.4.10. Các biến trong Javascript 63](#_Toc167201987)

[1.4.10.1 Khai báo biến: 63](#_Toc167201988)

[1.4.10.2 Sử dụng biến: 64](#_Toc167201989)

[1.4.10.3 Quy tắc đặt tên biến: 64](#_Toc167201990)

[1.4.10.4 Kiểu dữ liệu của biến: 64](#_Toc167201991)

[1.4.10.5 Ví dụ về sử dụng biến: 64](#_Toc167201992)

[1.4.10.6 Lưu ý: 65](#_Toc167201993)

[1.4.11. Cấu trúc điều khiển trong Javascript 65](#_Toc167201994)

[1.4.11.1 Câu lệnh if: 65](#_Toc167201995)

[1.4.11.2 Câu lệnh switch: 66](#_Toc167201996)

[1.4.11.3 Vòng lặp for: 67](#_Toc167201997)

[1.4.11.4 Vòng lặp while: 67](#_Toc167201998)

[1.4.11.5 Cấu trúc điều khiển lồng nhau: 68](#_Toc167201999)

[1.4.11.6 Sử dụng các cấu trúc điều khiển hiệu quả: 68](#_Toc167202000)

[1.4.12. Hàm trong Javascript 68](#_Toc167202001)

[1.4.12.1 Khai báo hàm: 68](#_Toc167202002)

[1.4.12.2 Gọi hàm: 68](#_Toc167202003)

[1.4.12.3 Hàm có tham số: 69](#_Toc167202004)

[1.4.12.4 Hàm trả về giá trị: 69](#_Toc167202005)

[1.4.12.5 Phạm vi biến: 69](#_Toc167202006)

[1.4.12.6 Ví dụ về sử dụng hàm: 69](#_Toc167202007)

[1.4.12.7 Lưu ý: 70](#_Toc167202008)

[1.4.13. Sự kiện trong Javascript 70](#_Toc167202009)

[1.4.13.1 Các loại sự kiện: 70](#_Toc167202010)

[1.4.13.2 Lắng nghe sự kiện: 70](#_Toc167202011)

[1.4.13.3 Hàm xử lý sự kiện: 71](#_Toc167202012)

[1.4.13.4 Loại bỏ sự kiện: 71](#_Toc167202013)

[1.4.13.5 Sử dụng sự kiện hiệu quả: 72](#_Toc167202014)

[1.4.14. DOM (Document Object Model) trong Javascript 72](#_Toc167202015)

[1.4.14.1 Các phương thức DOM thông dụng: [8] [9] [10] [11] 72](#_Toc167202016)

[1.4.14.2 Những điều bạn có thể làm với DOM: 73](#_Toc167202017)

[1.4.14.3 Lợi ích của việc sử dụng DOM: 74](#_Toc167202018)

[1.5. Kết chương 74](#_Toc167202019)

[1.5.1. HTML (HyperText Markup Language): 74](#_Toc167202020)

[1.5.2. CSS (Cascading Style Sheets): 75](#_Toc167202021)

[1.5.3. JavaScript: 75](#_Toc167202022)

[1.5.4. Sự kết hợp của HTML, CSS và JavaScript: 75](#_Toc167202023)

[1.5.5. Bên cạnh kiến thức cơ bản về HTML, CSS và JavaScript, chương này cũng đề cập đến một số chủ đề quan trọng khác: 75](#_Toc167202024)

[CHƯƠNG 2. Phân tích thiết kế hệ thống 77](#_Toc167202025)

[2.1. Mô tả bài toán 77](#_Toc167202026)

[2.1.1. Mục tiêu chính: 77](#_Toc167202027)

[2.1.2. Mục tiêu phụ: 78](#_Toc167202028)

[2.2. Đặc tả các yêu cầu 78](#_Toc167202029)

[2.2.1. Yêu cầu chức năng 78](#_Toc167202030)

[2.2.1.1 Chức Năng Chính: 78](#_Toc167202031)

[2.2.1.2 Chức Năng Phụ: 79](#_Toc167202032)

[2.2.2. Yêu cầu phi chức năng 80](#_Toc167202033)

[2.2.2.1 Hiệu suất 80](#_Toc167202034)

[2.2.2.2 Khả năng mở rộng 80](#_Toc167202035)

[2.2.2.3 Bảo mật 81](#_Toc167202036)

[2.2.2.4 Khả năng sử dụng 81](#_Toc167202037)

[2.2.2.5 Khả năng bảo trì 81](#_Toc167202038)

[2.2.2.6 Khả năng tương thích 82](#_Toc167202039)

[2.2.2.7 Tiêu chí đánh giá yêu cầu phi chức năng 82](#_Toc167202040)

[2.3. Thiết kế dữ liệu 82](#_Toc167202041)

[2.3.1. Mô hình dữ liệu 82](#_Toc167202042)

[2.3.2. Danh sách các thực thể và mối kết hợp 84](#_Toc167202043)

[2.3.3. Chi tiết thực thể và mối kết hợp 84](#_Toc167202044)

[2.4. Thiết kế xử lý 85](#_Toc167202045)

[2.4.1. Sơ đồ DFD mức ngữ cảnh 85](#_Toc167202046)

[2.4.2. Mô tả: 85](#_Toc167202047)

[2.4.2.1 Tác nhân bên ngoài: 85](#_Toc167202048)

[2.4.2.2 Luồng dữ liệu: 85](#_Toc167202049)

[2.4.3. Sơ đồ DFD mức 1 85](#_Toc167202050)

[2.4.4. Mô tả: 85](#_Toc167202051)

[2.4.4.1 Phân hệ chức năng: 85](#_Toc167202052)

[2.4.4.2 Luồng dữ liệu: 86](#_Toc167202053)

[2.4.5. Sơ đồ DFD mức 2 (Theo từng chức năng) 86](#_Toc167202054)

[2.4.6. Mô tả: 86](#_Toc167202055)

[2.4.6.1 Quá trình: 86](#_Toc167202056)

[2.4.6.2 Luồng dữ liệu: 86](#_Toc167202057)

[2.4.7. Diễn giải từng bước: 87](#_Toc167202058)

[2.4.7.1 Phân hệ Quản lý truyện: 87](#_Toc167202059)

[2.4.7.2 Phân hệ Tìm kiếm truyện: 88](#_Toc167202060)

[2.4.7.3 Phân hệ Xem thông tin truyện: 89](#_Toc167202061)

[2.5. Thiết kế giao diện 90](#_Toc167202062)

[2.5.1. Sơ đồ hệ thống 90](#_Toc167202063)

[2.5.2. Giao diện trang chủ người dùng 91](#_Toc167202064)

[2.5.2.1 Header (Phần đầu trang) 92](#_Toc167202065)

[2.5.2.2 Main Content (Nội dung chính) 92](#_Toc167202066)

[2.5.2.3 Sidebar (Thanh bên phải) 93](#_Toc167202067)

[2.5.2.4 Footer (Chân trang) 94](#_Toc167202068)

[2.5.2.5 Đăng Ký Nhận Tin: 94](#_Toc167202069)

[2.5.2.6 Các Thành Phần Khác 94](#_Toc167202070)

[2.5.2.7 Tổng Quan và Giao Diện 95](#_Toc167202071)

[2.5.3. Giao diện trang con người dùng 96](#_Toc167202072)

[2.5.3.1 Phần đầu trang: 97](#_Toc167202073)

[2.5.3.2 Phần thông tin truyện: 97](#_Toc167202074)

[2.5.3.3 Phần nội dung chính: 97](#_Toc167202075)

[2.5.3.4 Phần nhận xét và cảm nhận về bộ truyện: 98](#_Toc167202076)

[2.5.3.5 Phần bình luận của độc giả: 98](#_Toc167202077)

[2.5.3.6 Thanh bên phải: 98](#_Toc167202078)

[2.5.3.7 Phần chân trang: 98](#_Toc167202079)

[2.5.4. Giao diện chức năng đăng ký và đăng nhập 99](#_Toc167202080)

[2.5.4.1 Tiêu đề chính: 99](#_Toc167202081)

[2.5.4.2 Thông tin đăng nhập và đăng ký: 99](#_Toc167202082)

[2.5.4.3 Tổng Quan Bố Cục 100](#_Toc167202083)

[2.5.4.4 Thành Phần Chi Tiết 101](#_Toc167202084)

[2.5.4.5 Tổng Quan Thiết Kế 102](#_Toc167202085)

[2.5.4.6 Tổng Quan Chung: 103](#_Toc167202086)

[2.5.4.7 Các Thành Phần Cụ Thể: 103](#_Toc167202087)

[2.5.4.8 Bố Cục và Sự Trực Quan: 105](#_Toc167202088)

[2.5.4.9 Trải Nghiệm Người Dùng: 105](#_Toc167202089)

[2.5.4.10 Tổng Quan Chung: 106](#_Toc167202090)

[2.5.4.11 Các Thành Phần Cụ Thể: 106](#_Toc167202091)

[2.5.4.12 Bố Cục và Sự Trực Quan: 108](#_Toc167202092)

[2.5.4.13 Trải Nghiệm Người Dùng: 108](#_Toc167202093)

[2.5.5. Giao diện chức năng tìm kiếm 108](#_Toc167202094)

[2.5.5.1 Sidebar (Thanh bên phải) 109](#_Toc167202095)

[2.5.6. Giao diện chức năng bình luận 110](#_Toc167202096)

[2.5.6.1 Phần bình luận của độc giả: 110](#_Toc167202097)

[2.5.7. Giao diện trang chủ quản trị 110](#_Toc167202098)

[2.5.7.1 Giao diện chức năng … 110](#_Toc167202099)

[2.5.7.2 Giao diện chức năng … 110](#_Toc167202100)

[2.5.7.3 Giao diện chức năng … 110](#_Toc167202101)

[2.5.8. Kết chương 110](#_Toc167202102)

[2.5.8.1 Tầm quan trọng của thiết kế giao diện: 110](#_Toc167202103)

[2.5.8.2 Các yếu tố ảnh hưởng đến thiết kế giao diện: 111](#_Toc167202104)

[2.5.8.3 Quy trình thiết kế giao diện: 111](#_Toc167202105)

[2.5.8.4 Kết luận: 112](#_Toc167202106)

[CHƯƠNG 3. Cài đặt thực nghiệm 113](#_Toc167202107)

[3.1. Bộ dữ liệu thử nghiệm 113](#_Toc167202108)

[3.2. Kết quả thực nghiệm 113](#_Toc167202109)

[3.2.1. Chức năng gợi ý tìm kiếm 113](#_Toc167202110)

[3.2.2. Chức năng tìm kiếm 114](#_Toc167202111)

[3.2.3. Chức năng đăng ký 115](#_Toc167202112)

[3.2.4. Chức năng đăng nhập 116](#_Toc167202113)

[3.2.5. Chức năng quên mật khẩu 116](#_Toc167202114)

[3.2.6. Chức năng bình luận 116](#_Toc167202115)

[3.3. Kết chương 118](#_Toc167202116)

[3.3.1. Tóm tắt kết quả 118](#_Toc167202117)

[3.3.2. Kết quả thử nghiệm 119](#_Toc167202118)

[CHƯƠNG 4. Kết luận và hướng phát triển 120](#_Toc167202119)

[4.1. Kết luận 120](#_Toc167202120)

[4.1.1. Phân tích thị trường: 120](#_Toc167202121)

[4.1.2. Xác định USP (Unique Selling Proposition): 120](#_Toc167202122)

[4.1.3. Thiết kế giao diện: 120](#_Toc167202123)

[4.1.4. Lựa chọn công nghệ phù hợp: 120](#_Toc167202124)

[4.1.5. Chú trọng trải nghiệm người dùng: 120](#_Toc167202125)

[4.1.6. Đảm bảo bảo mật: 120](#_Toc167202126)

[4.1.7. Tối ưu hóa SEO: 121](#_Toc167202127)

[4.1.8. Marketing và quảng bá: 121](#_Toc167202128)

[4.2. Hướng phát triển 121](#_Toc167202129)

[4.2.1. Hệ thống cá nhân hóa: 121](#_Toc167202130)

[4.2.2. Hệ thống cộng đồng: 121](#_Toc167202131)

[4.2.3. Hệ thống gamification: 121](#_Toc167202132)

[4.2.4. Hệ thống livestream: 121](#_Toc167202133)

[4.2.5. Hệ thống hỗ trợ đa ngôn ngữ: 121](#_Toc167202134)

[4.2.6. Hệ thống thanh toán: 122](#_Toc167202135)

[4.2.7. Cập nhật nội dung thường xuyên: 122](#_Toc167202136)

[4.2.8. Hợp tác với tác giả và reviewer: 122](#_Toc167202137)

[4.2.9. Tổ chức các sự kiện: 122](#_Toc167202138)

[4.2.10. Phân tích dữ liệu: 122](#_Toc167202139)

[4.2.11. Cập nhật xu hướng: 122](#_Toc167202140)

[4.2.12. Bên cạnh các tính năng trên, website review truyện online cũng cần chú trọng đến: 122](#_Toc167202141)

**DANH MỤC HÌNH ẢNH**

[Hình 2.1. Sơ đồ hệ thống 90](#_Toc167201759)

[Hình 2.2. Giao diện trang chủ của người dùng 91](#_Toc167201760)

[Hình 2.3. Giao diện trang con của người dùng 96](#_Toc167201761)

[Hình 2.4. Giao diện chức năng Đăng Ký và Đăng Nhập 99](#_Toc167201762)

[Hình 2.5. Giao diện chức năng Đăng Ký 100](#_Toc167201763)

[Hình 2.6. Giao diện chức năng Đăng Nhập 102](#_Toc167201764)

[Hình 2.7. Giao diện chức năng Quên mật khẩu 106](#_Toc167201765)

[Hình 2.8. Giao diện chức năng tìm kiếm 108](#_Toc167201766)

[Hình 2.9. Giao diện chức năng tìm kiếm 109](#_Toc167201767)

[Hình 2.10. Giao diện chức năng bình luận 110](#_Toc167201768)

[Hình 3.1. Chức năng gợi ý tìm kiếm 113](#_Toc167201769)

[Hình 3.2. Chức năng tìm kiếm 114](#_Toc167201770)

[Hình 3.3. Chức năng tìm kiếm 115](#_Toc167201771)

[Hình 3.4. Chức năng đăng ký 115](#_Toc167201772)

[Hình 3.5. Chức năng đăng nhập 116](#_Toc167201773)

[Hình 3.6. Chức năng quên mật khẩu 116](#_Toc167201774)

[Hình 3.7. Chức năng bình luận 117](#_Toc167201775)

[Hình 3.8. Chức năng bình luận 117](#_Toc167201776)

[Hình 3.9. Chức năng bình luận 118](#_Toc167201777)

**DANH MỤC BẢNG BIỂU**

[Bảng 2.1 Danh sách các thực thể và mối kết hợp 2](#_Toc164154346)

[Bảng 2.2 Chi tiết thực thể … 2](#_Toc164154347)

# Tổng quan về nội dung nghiên cứu

## Quy trình phát triển website

### Lên kế hoạch và thu thập yêu cầu:

* Xác định mục tiêu của website.
* Phân tích đối tượng mục tiêu.
* Lập kế hoạch cho các chức năng và nội dung của website.
* Thu thập yêu cầu từ khách hàng.

### Thiết kế giao diện:

* Tạo bản đồ trang web (sitemap) để xác định cấu trúc của website.
* Thiết kế giao diện người dùng (UI) và trải nghiệm người dùng (UX) của website.
* Tạo bản mẫu giao diện (wireframe) để phác thảo bố cục của website.

### Phát triển website:

* Lựa chọn công nghệ phù hợp để phát triển website, ví dụ như HTML, CSS, JavaScript, PHP, MySQL.
* Viết mã website theo các yêu cầu đã được xác định.
* Kiểm tra và sửa lỗi website.

### Thử nghiệm và triển khai:

* Thử nghiệm website trên các trình duyệt và thiết bị khác nhau.
* Sửa lỗi và hoàn thiện website.
* Triển khai website lên hosting.

### Bảo trì và cập nhật:

* Cập nhật nội dung website thường xuyên.
* Sửa lỗi website khi cần thiết.
* Cập nhật website để phù hợp với các công nghệ mới.

### Ngoài ra, quy trình phát triển website có thể bao gồm thêm các bước sau:

* Phân tích thị trường và đối thủ cạnh tranh.
* Tối ưu hóa công cụ tìm kiếm (SEO) cho website.
* Marketing và quảng bá website.

## Ngôn ngữ HTML

### Ngôn ngữ HTML là gì ?

**HTML** (viết tắt của **HyperText Markup Language**) là ngôn ngữ đánh dấu siêu văn bản được sử dụng để tạo ra các trang web. HTML là ngôn ngữ cơ bản nhất trong việc xây dựng website, nó được sử dụng để mô tả cấu trúc và nội dung của trang web. [1]

### Đặc điểm của HTML:

* **Dễ học:** HTML có cú pháp đơn giản và dễ học, ngay cả đối với người mới bắt đầu.
* **Phổ biến:** HTML được hỗ trợ bởi tất cả các trình duyệt web hiện đại.
* **Linh hoạt:** HTML có thể được sử dụng để tạo ra nhiều loại trang web khác nhau, từ trang web đơn giản đến trang web phức tạp.
* **Có thể mở rộng:** HTML có thể được kết hợp với các ngôn ngữ khác như CSS và JavaScript để tạo ra các trang web tương tác và đẹp mắt.

### Cấu trúc cơ bản của một trang HTML: [1] [2]

* **<!DOCTYPE html>**: Khai báo loại tài liệu là HTML5.
* **<html>**: Thẻ bắt đầu của tài liệu HTML.
* **<head>**: Chứa thông tin về trang web, như tiêu đề trang, meta tags, v.v.
* **Tiêu đề trang:** được khai báo bằng thẻ <title>.
* **Liên kết đến các tập tin CSS:** được khai báo bằng thẻ <link>.
* **Khai báo meta:** cung cấp các thông tin về trang web như từ khóa, mô tả, v.v.
* **Khai báo JavaScript:** được khai báo bằng thẻ <script>..
* **</head>**: Thẻ kết thúc của phần <head>.
* **<body>**: Chứa nội dung chính của trang web.
* **Tiêu đề:** được khai báo bằng các thẻ <h1>, <h2>, <h3>, v.v.
* **Đoạn văn:** được khai báo bằng thẻ <p>.
* **Hình ảnh:** được khai báo bằng thẻ <img>.
* **Liên kết:** được khai báo bằng thẻ <a>.
* **Danh sách:** được khai báo bằng các thẻ <ul>, <ol>, <li>, v.v.
* **Bảng:** được khai báo bằng các thẻ <table>, <tr>, <td>, v.v.
* **</body>**: Thẻ kết thúc của phần <body>.
* **</html>**: Thẻ kết thúc của tài liệu HTML.

### Danh sách các phần tử HTML:

HTML có rất nhiều phần tử khác nhau, mỗi phần tử có chức năng riêng. Dưới đây là danh sách các phần tử HTML phổ biến nhất:

#### Phần tử cơ bản: [2] [3] [4]

* **<!DOCTYPE html>:** Khai báo phiên bản HTML của trang web.
* <html>: Khai báo bắt đầu của tài liệu HTML.
* <head>: Chứa các thông tin về trang web, nhưng không hiển thị trực tiếp trên trình duyệt.
* <title>: Khai báo tiêu đề của trang web.
* <link>: Liên kết đến các tập tin CSS.
* <meta>: Khai báo meta cho trang web.
* <script>: Khai báo JavaScript cho trang web.
* <body>: Chứa nội dung chính của trang web, sẽ hiển thị trực tiếp trên trình duyệt.
* <h1> đến <h6>: Khai báo các cấp tiêu đề khác nhau.
* <p>: Khai báo đoạn văn.
* <br>: Ngắt dòng.
* <hr>: Kẻ ngang.

#### Phần tử định dạng văn bản: [2] [3] [4]

* <b>: In đậm văn bản.
* <i>: In nghiêng văn bản.
* <u>: Gạch chân văn bản.
* <strike>: Xuyên tâm văn bản.
* <pre>: Định dạng văn bản dạng mã nguồn.
* <mark>: Tô sáng văn bản.
* <del>: Xóa văn bản.
* <ins>: Chèn văn bản.

#### Phần tử tạo liên kết: [2] [3] [4]

* <a>: Tạo liên kết đến trang web khác hoặc tài nguyên khác.

#### Phần tử tạo danh sách: [2] [3] [4]

* <ul>: Tạo danh sách không có thứ tự.
* <ol>: Tạo danh sách có thứ tự.
* <li>: Khai báo mục trong danh sách.

#### Phần tử tạo hình ảnh: [2] [3] [4]

* <img>: Chèn hình ảnh vào trang web.

#### Phần tử tạo bảng: [2] [3] [4]

* <table>: Khai báo bảng.
* <tr>: Khai báo hàng trong bảng.
* <th>: Khai báo tiêu đề cột trong bảng.
* <td>: Khai báo dữ liệu trong bảng.

#### Phần tử tạo biểu mẫu: [2] [3] [4]

* <form>: Tạo biểu mẫu để thu thập dữ liệu từ người dùng.
* <input>: Tạo các trường nhập liệu khác nhau như text, checkbox, radio button, v.v.
* <select>: Tạo danh sách lựa chọn.
* <textarea>: Tạo vùng nhập liệu văn bản dài.
* <button>: Tạo nút bấm để gửi biểu mẫu.

#### Phần tử tạo khung: [2] [3] [4]

* <iframe>: Tạo khung để hiển thị nội dung từ trang web khác.

#### Phần tử tạo âm thanh và video: [2] [3] [4]

* <audio>: Tạo trình phát âm thanh.
* <video>: Tạo trình phát video.

#### Phần tử tạo các yếu tố khác: [2] [3] [4]

* <div>: Tạo một khối phần tử để nhóm các phần tử khác.
* <span>: Nhóm một đoạn văn bản để định dạng riêng.
* <comment>: Thêm chú thích vào mã HTML (không hiển thị trên trình duyệt).

### Danh sách các thuộc tính HTML:

Thuộc tính HTML được sử dụng để bổ sung thông tin cho các phần tử HTML, giúp định dạng và điều chỉnh hành vi của các phần tử đó. Mỗi phần tử HTML có thể có một số thuộc tính khác nhau. Một số thuộc tính phổ biến nhất bao gồm:

#### Thuộc tính chung: [2] [3] [4]

* **id:** Xác định ID duy nhất cho phần tử.
* **class:** Xác định một hoặc nhiều lớp cho phần tử.
* **style:** Định dạng trực tiếp phần tử bằng CSS.
* **title:** Hiển thị văn bản bật lên khi di chuột qua phần tử.
* **lang:** Xác định ngôn ngữ của phần tử.
* **dir:** Xác định hướng hiển thị văn bản (ltr hoặc rtl).

#### Thuộc tính cho phần tử <a>: [2] [3] [4]

* **href:** Xác định URL của liên kết.
* **target:** Xác định vị trí mở liên kết (ví dụ: \_blank, \_self).
* **rel:** Xác định mối quan hệ giữa trang hiện tại và trang được liên kết.
* **type:** Xác định loại nội dung của liên kết.

#### Thuộc tính cho phần tử <img>: [2] [3] [4]

* **src:** Xác định URL của hình ảnh.
* **alt:** Xác định văn bản thay thế cho hình ảnh (hiển thị khi không hiển thị được hình ảnh).
* **width:** Xác định chiều rộng của hình ảnh.
* **height:** Xác định chiều cao của hình ảnh.

#### Thuộc tính cho phần tử <table>: [2] [3] [4]

* **border:** Xác định đường viền cho bảng.
* **cellspacing:** Xác định khoảng cách giữa các ô trong bảng.
* **cellpadding:** Xác định khoảng cách giữa nội dung và viền của các ô trong bảng.

#### Thuộc tính cho phần tử <tr>: [2] [3] [4]

* **align:** Xác định cách căn chỉnh nội dung trong hàng.
* **valign:** Xác định cách căn chỉnh nội dung theo chiều dọc trong hàng.

#### Thuộc tính cho phần tử <th> và <td>: [2] [3] [4]

* **scope:** Xác định phạm vi của tiêu đề hoặc dữ liệu trong bảng.
* **abbr:** Xác định dạng rút gọn của tiêu đề hoặc dữ liệu.

#### Thuộc tính cho phần tử <input>: [2] [3] [4]

* **type:** Xác định loại trường nhập liệu (text, checkbox, radio button, v.v.).
* **value:** Xác định giá trị mặc định của trường nhập liệu.
* **checked:** Xác định trạng thái được chọn của checkbox hoặc radio button.
* **placeholder:** Xác định văn bản gợi ý trong trường nhập liệu.

#### Thuộc tính cho phần tử <select>: [2] [3] [4]

* **size:** Xác định số lượng tùy chọn hiển thị trong danh sách lựa chọn.
* **multiple:** Xác định cho phép chọn nhiều tùy chọn hay không.
* **disabled:** Xác định vô hiệu hóa danh sách lựa chọn.

### Tiêu đề trong HTML

Tiêu đề đóng vai trò quan trọng trong việc tổ chức nội dung và giúp người dùng dễ dàng nắm bắt thông tin trên trang web. HTML cung cấp các thẻ tiêu đề từ <h1> đến <h6> để định dạng các cấp tiêu đề khác nhau. [2] [3] [4]

#### ****Các cấp tiêu đề:****

* <h1>: Tiêu đề cấp 1 - Đây là tiêu đề quan trọng nhất trên trang web, thường được sử dụng cho tiêu đề trang.
* <h2>: Tiêu đề cấp 2 - Được sử dụng cho các tiêu đề chính trong một phần của trang.
* <h3>: Tiêu đề cấp 3 - Được sử dụng cho các tiêu đề phụ trong một phần của trang.
* <h4>: Tiêu đề cấp 4 - Được sử dụng cho các tiêu đề con trong một phần của trang.
* <h5>: Tiêu đề cấp 5 - Ít được sử dụng hơn, thường được sử dụng cho các tiêu đề nhỏ trong một phần của trang.
* <h6>: Tiêu đề cấp 6 - Ít được sử dụng nhất, thường được sử dụng cho các tiêu đề rất nhỏ trong một phần của trang.

#### Cách sử dụng thẻ tiêu đề:

<h1 >Nội dung tiêu đề</h1>

<h2 >Nội dung tiêu đề</h2>

<h3 >Nội dung tiêu đề</h3>

<h4 >Nội dung tiêu đề</h4>

<h5 >Nội dung tiêu đề</h5>

<h6 >Nội dung tiêu đề</h6>

#### Lưu ý:

* Nên sử dụng các cấp tiêu đề một cách hợp lý để tạo ra cấu trúc rõ ràng cho trang web.
* Không nên sử dụng quá nhiều thẻ tiêu đề cấp cao (<h1>, <h2>) trong một trang.
* Nên sử dụng thẻ tiêu đề để thể hiện nội dung chính của trang, không nên sử dụng thẻ tiêu đề để định dạng văn bản.

#### Một số mẹo sử dụng thẻ tiêu đề hiệu quả:

* Sử dụng từ khóa trong các thẻ tiêu đề để giúp người dùng dễ dàng tìm kiếm trang web của bạn trên Google.
* Viết các thẻ tiêu đề ngắn gọn và súc tích, dễ đọc và dễ hiểu.
* Sử dụng các cấp tiêu đề một cách nhất quán để tạo ra cấu trúc rõ ràng cho trang web.
* Sử dụng thẻ tiêu đề để tóm tắt nội dung chính của mỗi phần trong trang web.

### Đoạn văn bản trong HTML

Đoạn văn bản là một phần quan trọng trong nội dung của trang web. HTML sử dụng thẻ <p> để định dạng đoạn văn bản. [2] [3] [4]

#### **Cách sử dụng thẻ** <p>**:**

<p>Đây là một đoạn văn bản.</p>

<p>Đây là một đoạn văn bản khác.</p>

#### Lưu ý:

* Mỗi đoạn văn bản phải được bao bọc bởi thẻ <p> và </p>.
* Trình duyệt web sẽ tự động thêm khoảng cách (lề) trước và sau mỗi đoạn văn bản.
* Bạn có thể sử dụng các thẻ HTML khác bên trong thẻ <p> để định dạng văn bản, chẳng hạn như <b>, <i>, <u>, v.v.

#### Một số mẹo sử dụng thẻ <p> hiệu quả:

* Viết các đoạn văn bản ngắn gọn và súc tích, dễ đọc và dễ hiểu.
* Sử dụng các từ khóa trong các đoạn văn bản để giúp người dùng dễ dàng tìm kiếm thông tin trên trang web của bạn.
* Sử dụng các đoạn văn bản để chia nhỏ nội dung thành các phần dễ đọc hơn.
* Sử dụng thẻ <p> để tóm tắt nội dung chính của mỗi phần trong trang web.

### Định dạng văn bản trong HTML

HTML cung cấp nhiều thẻ và thuộc tính để định dạng văn bản trong trang web, giúp bạn tạo ra nội dung đẹp mắt và thu hút người đọc. Dưới đây là một số cách thức phổ biến để định dạng văn bản trong HTML:

#### ****Sử dụng thẻ định dạng:**** [2] [3] [4]

* **In đậm:** <b>Nội dung in đậm</b>
* **In nghiêng:** <i>Nội dung in nghiêng</i>
* **Gạch chân:** <u>Nội dung gạch chân</u>
* **Xuyên tâm:** <strike>Nội dung xuyên tâm</strike>
* **Định dạng văn bản dạng mã nguồn:** <pre>Nội dung dạng mã nguồn</pre>
* **Tô sáng văn bản:** <mark>Nội dung tô sáng</mark>
* **Xóa văn bản:** <del>Nội dung đã xóa</del>
* **Chèn văn bản:** <ins>Nội dung được chèn</ins>

#### ****Sử dụng thuộc tính CSS:****

Ngoài việc sử dụng các thẻ định dạng, bạn có thể sử dụng thuộc tính CSS để định dạng văn bản một cách linh hoạt hơn. Ví dụ:

* **Định dạng phông chữ:** [5] [6]
  + font-family: Thay đổi kiểu phông chữ
  + font-size: Thay đổi kích thước phông chữ
  + font-weight: Thay đổi độ dày phông chữ
  + font-style: Thay đổi kiểu chữ (italic, normal)
* **Định dạng màu sắc:** [5] [6]
  + color: Thay đổi màu sắc của văn bản
* **Định dạng căn lề:** [5] [6]
  + text-align: Căn lề trái, phải, giữa, hai bên
* **Định dạng khoảng cách:** [5] [6]
  + line-height: Thay đổi khoảng cách giữa các dòng văn bản
  + letter-spacing: Thay đổi khoảng cách giữa các chữ cái
  + word-spacing: Thay đổi khoảng cách giữa các từ

#### ****Sử dụng các lớp CSS:****

Bạn có thể tạo các lớp CSS để định dạng nhiều phần văn bản trong trang web một cách đồng nhất. [5] [6] Ví dụ:

CSS

.tieude {

font-family: Arial, sans-serif;

font-size: 24px;

font-weight: bold;

text-align: center;

}

.noidung {

font-family: Times New Roman, serif;

font-size: 16px;

line-height: 1.5;

} [5] [6]

HTML

<h1 class="tieude">Tiêu đề</h1>

<p class="noidung">Nội dung đoạn văn bản</p>

#### Một số lưu ý:

* Nên sử dụng các thẻ định dạng một cách hợp lý để tránh làm cho trang web rối mắt.
* Nên sử dụng thuộc tính CSS để định dạng văn bản một cách linh hoạt và hiệu quả hơn.
* Nên sử dụng các lớp CSS để định dạng nhiều phần văn bản trong trang web một cách đồng nhất.

### Nhận xét trong HTML

Nhận xét trong HTML là những đoạn văn bản được thêm vào mã HTML nhưng không hiển thị trên trình duyệt web. Nhận xét được sử dụng để giải thích hoặc ghi chú về mã, giúp người lập trình dễ dàng hiểu và sửa đổi mã sau này. [2] [3] [4]

#### ****Cách viết nhận xét:****

* Dấu <!-- --> đánh dấu cuối của nhận xét.

#### ****Ví dụ:****

<!DOCTYPE html>

<html>

<head>

<title>Trang web của tôi</title>

</head>

<body>

<h1>Đây là tiêu đề</h1>

<p>Đây là nội dung của trang web.</p>

<!-- Đây là nội dung ẩn bí mật của trang web -->

</body>

</html>

#### Lưu ý:

* Nhận xét có thể được viết ở bất kỳ vị trí nào trong mã HTML.
* Nhận xét không ảnh hưởng đến cách hiển thị trang web trên trình duyệt web.
* Nên sử dụng nhận xét để giải thích hoặc ghi chú về những phần mã phức tạp hoặc khó hiểu.
* Nên tránh sử dụng nhận xét để thêm thông tin không liên quan đến mã.

#### Một số mẹo sử dụng nhận xét hiệu quả:

* Sử dụng nhận xét để giải thích mục đích của các đoạn mã.
* Sử dụng nhận xét để ghi chú về những thay đổi đã thực hiện trong mã.
* Sử dụng nhận xét để đánh dấu các phần mã chưa hoàn thiện.
* Sử dụng nhận xét để ghi lại các thông tin quan trọng về mã.

### Liên kết trong HTML

Liên kết trong HTML, hay còn gọi là siêu liên kết (hyperlink), cho phép người dùng di chuyển giữa các trang web hoặc tài nguyên khác nhau trên internet. Liên kết được tạo bằng thẻ <a> và có thể được đặt vào bất kỳ vị trí nào trong mã HTML. [2] [3] [4]

#### ****Cấu trúc của thẻ**** <a>****:****

<a href="URL" target="\_blank">Nội dung liên kết</a>

* href: Thuộc tính xác định URL của trang web hoặc tài nguyên mà liên kết trỏ đến.
* target: Thuộc tính xác định vị trí mở liên kết. Ví dụ: \_blank mở liên kết trong tab mới, \_self mở liên kết trong cùng tab.
* Nội dung liên kết: Văn bản hiển thị cho người dùng, là phần họ sẽ nhấp chuột để truy cập liên kết. [2] [3] [4]

#### Ví dụ:

<a href="https://www.google.com/">Truy cập Google</a>

<a href="https://vi.wikipedia.org/">Truy cập Wikipedia tiếng Việt</a>

<a href="mailto:example@gmail.com">Gửi email</a>

#### Một số lưu ý:

* Nên sử dụng văn bản liên kết mô tả chính xác nội dung của trang web hoặc tài nguyên mà liên kết trỏ đến.
* Nên sử dụng URL đầy đủ và chính xác cho các liên kết.
* Nên sử dụng thuộc tính target một cách hợp lý để mang lại trải nghiệm tốt nhất cho người dùng.
* Nên thử nghiệm các liên kết để đảm bảo rằng chúng hoạt động bình thường.

#### Một số mẹo sử dụng liên kết hiệu quả:

* Sử dụng liên kết nội bộ để liên kết đến các trang khác trên cùng một trang web.
* Sử dụng liên kết bên ngoài để liên kết đến các trang web khác có liên quan đến nội dung của bạn.
* Sử dụng liên kết theo ngữ cảnh để giúp người dùng dễ dàng tìm thấy thông tin họ cần.
* Sử dụng liên kết kêu gọi hành động để khuyến khích người dùng thực hiện một hành động cụ thể, chẳng hạn như nhấp chuột để mua hàng hoặc đăng ký nhận bản tin.

### Danh sách trong HTML

Danh sách được sử dụng để liệt kê các mục theo thứ tự hoặc không theo thứ tự trong HTML. Có hai loại danh sách chính:

#### ****Danh sách không theo thứ tự (Unordered List):**** [2] [3] [4]

* Sử dụng thẻ <ul> để tạo danh sách.
* Mỗi mục trong danh sách được đánh dấu bằng thẻ <li>.
* Các mục trong danh sách được đánh dấu bằng dấu chấm tròn hoặc kiểu ký tự khác.

**Ví dụ:**

<ul>

<li>Mục 1</li>

<li>Mục 2</li>

<li>Mục 3</li>

</ul>

#### Danh sách theo thứ tự (Ordered List): [2] [3] [4]

* Sử dụng thẻ <ol> để tạo danh sách.
* Mỗi mục trong danh sách được đánh dấu bằng thẻ <li>.
* Các mục trong danh sách được đánh số theo thứ tự (1, 2, 3, ...).

**Ví dụ:**

<ol>

<li>Bước 1</li>

<li>Bước 2</li>

<li>Bước 3</li>

</ol>

#### Định dạng danh sách: [2] [3] [4]

Có thể sử dụng thuộc tính type của thẻ <ul> hoặc <ol> để thay đổi kiểu đánh dấu cho các mục trong danh sách. Ví dụ:

<ul type="square">

<li>Mục 1</li>

<li>Mục 2</li>

<li>Mục 3</li>

</ul>

Có thể sử dụng thuộc tính style của thẻ <li> để định dạng riêng cho từng mục trong danh sách. Ví dụ:

<ul>

<li style="color: red;">Mục 1</li>

<li>Mục 2</li>

<li>Mục 3</li>

</ul>

#### Danh sách lồng nhau: [2] [3] [4]

Có thể tạo danh sách lồng nhau bằng cách đặt thẻ <ul> hoặc <ol> bên trong một thẻ <li>.

Ví dụ:

<ul>

<li>Mục 1

<ul>

<li>Mục con 1.1</li>

<li>Mục con 1.2</li>

</ul>

</li>

<li>Mục 2</li>

<li>Mục 3</li>

</ul>

#### Một số lưu ý:

* Nên sử dụng danh sách để liệt kê các mục có mối quan hệ với nhau.
* Nên sử dụng loại danh sách phù hợp cho từng trường hợp.
* Nên định dạng danh sách một cách hợp lý để dễ đọc và dễ nhìn.
* Nên sử dụng danh sách lồng nhau một cách cẩn thận để tránh tạo ra cấu trúc phức tạp.

### Hình ảnh trong HTML

Hình ảnh đóng vai trò quan trọng trong việc thu hút người dùng và tăng tính thẩm mỹ cho trang web. HTML sử dụng thẻ <img> để chèn hình ảnh vào trang web. [2] [3] [4]

#### **1. **Cấu trúc thẻ**** <img>****:****

HTML

<img src="URL\_ảnh" alt="Văn bản thay thế">

* src: Thuộc tính bắt buộc, xác định URL của ảnh.
* alt: Thuộc tính bắt buộc, cung cấp văn bản thay thế cho ảnh, hiển thị khi ảnh không thể tải được hoặc khi người dùng di chuột qua ảnh.

**Ví dụ:**

HTML

<img src="https://example.com/image.jpg" alt="Hình ảnh đại diện">

#### ****Một số lưu ý:****

* Nên sử dụng URL đầy đủ và chính xác cho ảnh.
* Nên sử dụng văn bản thay thế mô tả chính xác nội dung của ảnh.
* Nên tối ưu hóa kích thước ảnh để giảm thời gian tải trang.
* Nên sử dụng thuộc tính width và height để đặt kích thước cho ảnh.
* Nên sử dụng thuộc tính border để đặt đường viền cho ảnh.
* Nên sử dụng thuộc tính margin và padding để đặt khoảng cách cho ảnh.

#### ****Một số mẹo sử dụng hình ảnh hiệu quả:****

* Sử dụng hình ảnh chất lượng cao và phù hợp với nội dung trang web.
* Sử dụng hình ảnh để minh họa cho các ý tưởng và thông tin quan trọng.
* Sử dụng hình ảnh một cách vừa phải để tránh làm cho trang web quá nặng nề.
* Sử dụng chú thích cho hình ảnh để cung cấp thêm thông tin cho người dùng.
* Sử dụng thuộc tính alt để tối ưu hóa trang web cho công cụ tìm kiếm.

### Bảng trong HTML

Bảng được sử dụng để sắp xếp dữ liệu theo hàng và cột trong HTML. Bảng được tạo ra bằng thẻ <table>, bên trong có các thẻ <tr> để tạo hàng và thẻ <td> để tạo ô. [2] [3] [4]

#### ****Cấu trúc cơ bản của bảng:****

HTML

<table>

<tr>

<th>Tiêu đề 1</th>

<th>Tiêu đề 2</th>

<th>Tiêu đề 3</th>

</tr>

<tr>

<td>Nội dung 1.1</td>

<td>Nội dung 1.2</td>

<td>Nội dung 1.3</td>

</tr>

<tr>

<td>Nội dung 2.1</td>

<td>Nội dung 2.2</td>

<td>Nội dung 2.3</td>

</tr>

</table> [2] [3] [4]

#### ***Giải thích:***

* <table>: Thẻ bắt đầu bảng.
* <tr>: Thẻ bắt đầu hàng trong bảng.
* <th>: Thẻ tạo tiêu đề cho cột trong bảng.
* <td>: Thẻ tạo ô dữ liệu trong bảng.
* Nội dung bên trong thẻ <th> và <td> sẽ hiển thị trong bảng.

#### ****Một số thuộc tính hữu ích:**** [2] [3] [4]

* border: Thiết lập đường viền cho bảng.
* cellspacing: Thiết lập khoảng cách giữa các ô trong bảng.
* cellpadding: Thiết lập khoảng trống giữa nội dung ô và đường viền ô.
* width: Thiết lập chiều rộng cho bảng.
* height: Thiết lập chiều cao cho bảng.
* align: Căn lề nội dung trong các ô.

**Ví dụ:**

HTML

<table border=1 cellspacing=2 cellpadding=5 width=300>

<tr>

<th>STT</th>

<th>Họ và tên</th>

<th>Email</th>

</tr>

<tr>

<td>1</td>

<td>Nguyễn Văn A</td>

<td>nguyenvana@gmail.com</td>

</tr>

<tr>

<td>2</td>

<td>Trần Thị B</td>

<td>tranthib@gmail.com</td>

</tr>

</table> [2] [3] [4]

#### ****Một số lưu ý:****

* Nên sử dụng bảng khi cần sắp xếp dữ liệu một cách rõ ràng và khoa học.
* Nên sử dụng các thẻ <th> và <td> một cách hợp lý để phân biệt tiêu đề và dữ liệu.
* Nên sử dụng các thuộc tính để định dạng bảng một cách phù hợp với nội dung.
* Nên sử dụng chú thích cho bảng để giải thích nội dung của bảng.

### Biểu mẫu trong HTML

Biểu mẫu đóng vai trò quan trọng trong việc tương tác với người dùng trên trang web. Biểu mẫu cho phép người dùng nhập dữ liệu, gửi thông tin hoặc thực hiện các hành động khác. Biểu mẫu được tạo ra bằng thẻ <form> trong HTML. [2] [3] [4]

#### ****Cấu trúc cơ bản của biểu mẫu:**** [2] [3] [4]

HTML

<form action="URL\_xử\_lý" method="POST">

<label for="ten">Tên:</label>

<input type="text" id="ten" name="ten">

<br>

<label for="email">Email:</label>

<input type="email" id="email" name="email">

<br>

<input type="submit" value="Gửi">

</form>

#### ***. Giải thích:*** [2] [3] [4]

* form: Thẻ bắt đầu biểu mẫu.
* action: Thuộc tính xác định URL của trang web sẽ xử lý dữ liệu được gửi từ biểu mẫu.
* method: Thuộc tính xác định phương thức gửi dữ liệu (POST hoặc GET).
* label: Thẻ tạo nhãn cho các phần tử nhập dữ liệu.
* for: Thuộc tính liên kết nhãn với phần tử nhập dữ liệu có ID tương ứng.
* input: Thẻ tạo các phần tử nhập dữ liệu khác nhau (text, email, submit, v.v.).
* type: Thuộc tính xác định loại phần tử nhập dữ liệu.
* id: Thuộc tính xác định ID cho phần tử nhập dữ liệu.
* name: Thuộc tính xác định tên cho phần tử nhập dữ liệu.
* value: Thuộc tính xác định giá trị mặc định cho phần tử nhập dữ liệu.

#### ****Một số loại phần tử nhập dữ liệu phổ biến:**** [2] [3] [4]

* text: Cho phép người dùng nhập văn bản.
* email: Cho phép người dùng nhập địa chỉ email.
* password: Cho phép người dùng nhập mật khẩu.
* checkbox: Cho phép người dùng chọn một hoặc nhiều tùy chọn.
* radio: Cho phép người dùng chọn một trong nhiều tùy chọn.
* textarea: Cho phép người dùng nhập một đoạn văn bản dài.
* select: Cho phép người dùng chọn một tùy chọn từ danh sách.
* submit: Gửi dữ liệu biểu mẫu đến máy chủ.

**Ví dụ:**

HTML

<form action="https://example.com/xuly.php" method="POST">

<label for="hoten">Họ và tên:</label>

<input type="text" id="hoten" name="hoten" required>

<br>

<label for="sodienthoai">Số điện thoại:</label>

<input type="tel" id="sodienthoai" name="sodienthoai">

<br>

<label for="noidung">Nội dung:</label>

<textarea id="noidung" name="noidung" rows="5" cols="30"></textarea>

<br>

<input type="submit" value="Gửi liên hệ">

</form>

#### ****Một số lưu ý:****

* Nên sử dụng biểu mẫu khi cần thu thập dữ liệu từ người dùng.
* Nên sử dụng các loại phần tử nhập dữ liệu phù hợp với mục đích sử dụng.
* Nên sử dụng nhãn cho các phần tử nhập dữ liệu để hướng dẫn người dùng.
* Nên sử dụng thuộc tính required để yêu cầu người dùng nhập dữ liệu cho các phần tử quan trọng.
* Nên sử dụng thuộc tính placeholder để gợi ý nội dung cho người dùng.
* Nên kiểm tra dữ liệu đầu vào trước khi gửi đến máy chủ.

### Khung trong HTML (Iframe)

**Khung nội tuyến (Iframe)**, hay còn gọi là khung nhúng, là một phần tử HTML cho phép bạn nhúng một trang web khác vào trang web hiện tại của mình. Iframe thường được sử dụng để hiển thị các video, bản đồ, nội dung từ các trang web khác hoặc để tạo ra các ứng dụng web phức tạp. [2] [3] [4]

#### ****Cấu trúc cơ bản của Iframe:**** [2] [3] [4]

HTML

<iframe src="URL\_trang\_web\_nhúng" width="600" height="400"></iframe>

* iframe: Thẻ bắt đầu khung nội tuyến.
* src: Thuộc tính xác định URL của trang web sẽ được nhúng vào khung.
* width: Thuộc tính xác định chiều rộng của khung (bằng pixel).
* height: Thuộc tính xác định chiều cao của khung (bằng pixel).

**Ví dụ:**

HTML

<iframe src="https://www.youtube.com/embed/v=J---AIyznWw" width="560" height="315"></iframe>

Ví dụ trên sẽ nhúng video YouTube có ID J---AIyznWw vào khung với chiều rộng 560 pixel và chiều cao 315 pixel.

#### ****Một số thuộc tính hữu ích khác của Iframe:**** [2] [3] [4]

* scrolling: Xác định có hiển thị thanh cuộn trong khung hay không (yes, no, auto).
* frameborder: Xác định có hiển thị đường viền xung quanh khung hay không (0, 1).
* allowfullscreen: Cho phép video được phát toàn màn hình (true, false).
* title: Cung cấp tiêu đề cho khung.

#### ****Một số lưu ý:****

* Nên sử dụng Iframe khi cần nhúng nội dung từ trang web khác vào trang web hiện tại.
* Nên sử dụng các thuộc tính để định dạng khung một cách phù hợp với nội dung.
* Nên sử dụng Iframe một cách có trách nhiệm và tuân thủ các điều khoản dịch vụ của các trang web được nhúng.

### Phần tử khối và phần tử nội tuyến trong HTML

**Phần tử khối** và **phần tử nội tuyến** là hai loại phần tử cơ bản trong HTML, đóng vai trò quan trọng trong việc định dạng và bố cục trang web. [2] [3] [4]

#### ****Phân biệt phần tử khối và phần tử nội tuyến:****

##### **Phần tử khối:** [2] [3] [4]

* Bắt đầu từ dòng mới.
* Chiếm toàn bộ chiều rộng của phần tử cha.
* Có thể chứa các phần tử con khác (phần tử khối và phần tử nội tuyến).
* Ví dụ: <div>, <h1>, <h2>, <p>, <ul>, <ol>, <table>, <form>, v.v.

##### **Phần tử nội tuyến:** [2] [3] [4]

* Không bắt đầu từ dòng mới.
* Chiếm chiều rộng cần thiết để hiển thị nội dung.
* Không thể chứa các phần tử con khác (chỉ có thể chứa văn bản).
* Ví dụ: <span>, <em>, <strong>, <a>, <img>, <br>, <input>, <select>, v.v.

#### ****Ví dụ minh họa:****

HTML

<div>

<h1>Tiêu đề</h1>

<p>Nội dung đoạn văn.</p>

<img src="image.jpg" alt="Hình ảnh">

</div>

* Trong ví dụ trên, <div> là phần tử khối, chứa các phần tử con khác là <h1>, <p> và <img>.
* <h1> là phần tử khối, bắt đầu từ dòng mới và chiếm toàn bộ chiều rộng của phần tử cha (<div>).
* <p> là phần tử khối, bắt đầu từ dòng mới và chiếm toàn bộ chiều rộng của phần tử cha (<div>).
* <img> là phần tử nội tuyến, không bắt đầu từ dòng mới và chỉ chiếm chiều rộng cần thiết để hiển thị hình ảnh.

#### **Một số lưu ý**:****

* Có thể sử dụng thuộc tính display để thay đổi loại hiển thị của phần tử (block, inline, inline-block, none).
* Việc sử dụng hợp lý các phần tử khối và phần tử nội tuyến giúp tạo bố cục trang web rõ ràng, khoa học và dễ đọc.
* Nên sử dụng các phần tử khối để tạo cấu trúc chính cho trang web, sử dụng các phần tử nội tuyến để định dạng nội dung.

### Đường dẫn tệp trong HTML

Đường dẫn tệp (file path) trong HTML được sử dụng để xác định vị trí của các tệp tin liên quan đến trang web, chẳng hạn như hình ảnh, CSS, JavaScript, video, âm thanh, v.v. Việc sử dụng đường dẫn tệp chính xác giúp trình duyệt web có thể tải và hiển thị các tệp tin này một cách chính xác. [2] [3] [4]

#### ****Loại đường dẫn tệp:****

Có hai loại đường dẫn tệp chính trong HTML:

##### Đường dẫn tuyệt đối: [2] [3] [4]

Bắt đầu từ gốc của trang web (ví dụ: /images/logo.png, https://example.com/css/style.css).

##### Đường dẫn tương đối: [2] [3] [4]

Bắt đầu từ vị trí của tệp HTML hiện tại (ví dụ: images/logo.png, ../css/style.css).

#### ****Sử dụng đường dẫn tệp:****

##### **Hình ảnh:** [2] [3] [4]

Sử dụng thẻ <img> và thuộc tính src để xác định đường dẫn đến tệp hình ảnh.

* + Ví dụ: <img src="images/logo.png" alt="Logo">

##### **CSS:** [5] [6] [7]

Sử dụng thẻ <link> và thuộc tính href để xác định đường dẫn đến tệp CSS.

* + Ví dụ: <link rel="stylesheet" href="css/style.css">

##### **JavaScript:** [8] [9] [10] [11]

Sử dụng thẻ <script> và thuộc tính src để xác định đường dẫn đến tệp JavaScript.

* + Ví dụ: <script src="js/script.js"></script>

##### **Video:** [2] [3] [4]

Sử dụng thẻ <video> và thuộc tính src để xác định đường dẫn đến tệp video.

* + Ví dụ: <video src="videos/video.mp4" controls></video>

##### **Âm thanh:** [2] [3] [4]

Sử dụng thẻ <audio> và thuộc tính src để xác định đường dẫn đến tệp âm thanh.

* + Ví dụ: <audio src="audio/audio.mp3" controls></audio>

#### ****Một số lưu ý:****

* Nên sử dụng đường dẫn tệp chính xác và phù hợp với cấu trúc thư mục của trang web.
* Nên sử dụng đường dẫn tuyệt đối khi nhúng tệp từ trang web khác.
* Nên sử dụng đường dẫn tương đối khi nhúng tệp từ thư mục con của thư mục chứa tệp HTML hiện tại.
* Nên kiểm tra kỹ đường dẫn tệp trước khi sử dụng để tránh lỗi.

### Âm thanh và video trong HTML

HTML cung cấp các thẻ để nhúng nội dung âm thanh và video trực tiếp vào trang web, giúp bạn tạo ra các trang web đa phương tiện phong phú và thu hút hơn.

#### ****Âm thanh:**** [2] [3] [4]

* **Thẻ <audio>:** Thẻ này được sử dụng để nhúng tệp âm thanh vào trang web.
  + **Thuộc tính src:** Xác định đường dẫn đến tệp âm thanh.
  + **Thuộc tính controls:** Hiển thị thanh điều khiển cho phép người dùng phát, tạm dừng, tua nhanh/tua lui và điều chỉnh âm lượng.

**Ví dụ:**

HTML

<audio src="audio/music.mp3" controls>

Trình duyệt của bạn không hỗ trợ thẻ audio.

</audio>

**Các định dạng âm thanh phổ biến:** mp3, wav, ogg

#### ****Video:**** [2] [3] [4]

* **Thẻ <video>:** Thẻ này được sử dụng để nhúng tệp video vào trang web.
  + **Thuộc tính src:** Xác định đường dẫn đến tệp video.
  + **Thuộc tính controls:** Hiển thị thanh điều khiển cho phép người dùng phát, tạm dừng, tua nhanh/tua lui, điều chỉnh âm lượng và toàn màn hình.

**Ví dụ:**

HTML

<video src="videos/video.mp4" controls>

Trình duyệt của bạn không hỗ trợ thẻ video.

</video>

**Các định dạng video phổ biến:** mp4, webm, ogg

#### ****Một số lưu ý:****

* Nên sử dụng các định dạng âm thanh và video phổ biến để đảm bảo khả năng tương thích với nhiều trình duyệt.
* Nên kiểm tra kích thước tệp âm thanh và video để tránh làm chậm tốc độ tải trang.
* Nên sử dụng các chú thích và phụ đề để nâng cao khả năng tiếp cận cho người khiếm thính.
* Nên cân nhắc sử dụng các tính năng tự động phát hoặc tắt tiếng âm thanh/video để phù hợp với trải nghiệm người dùng.

## Bảng định kiểu CSS

### CSS là gì ?

**CSS** (viết tắt của Cascading Style Sheets - Bảng định kiểu tầng) là một ngôn ngữ sheet kiểu được sử dụng để định dạng và trình bày nội dung được viết bằng HTML. Nó tách biệt nội dung của trang web (HTML) với giao diện hiển thị (CSS), giúp cho việc quản lý và bảo trì website dễ dàng hơn. [1]

### Một số đặc điểm của CSS:

* **Dễ học:** CSS có cú pháp tương đối đơn giản và dễ học, ngay cả đối với người mới bắt đầu.
* **Linh hoạt:** CSS có thể được sử dụng để tạo ra nhiều kiểu hiển thị khác nhau cho trang web.
* **Kiểm soát tốt giao diện:** Cho phép bạn kiểm soát nhiều yếu tố về giao diện trang web như màu sắc, font chữ, kích thước, khoảng cách, và hiệu ứng.
* **Tính kế thừa (Cascading):** Các kiểu định nghĩa trong CSS được thừa hưởng theo một thứ tự nhất định, giúp bạn dễ dàng tạo ra các kiểu toàn cục và ghi đè chúng cho các phần tử cụ thể.

### ****Vai trò của CSS:****

* **Tách biệt nội dung và trình bày:** CSS giúp tách biệt nội dung (HTML) và trình bày (CSS), cho phép bạn thay đổi giao diện của trang web mà không cần sửa đổi nội dung.
* **Kiểm soát bố cục:** CSS cho phép bạn kiểm soát bố cục của trang web, bao gồm cách sắp xếp các yếu tố, kích thước và vị trí của chúng.
* **Định dạng văn bản:** CSS cho phép bạn định dạng văn bản trên trang web, bao gồm phông chữ, màu sắc, kích thước, căn chỉnh và nhiều hơn nữa.
* **Tạo hiệu ứng:** CSS cho phép bạn tạo hiệu ứng thị giác cho các yếu tố trên trang web, chẳng hạn như bóng đổ, đường viền, hiệu ứng di chuột và chuyển đổi.
* **Tăng tính tương thích:** CSS giúp tăng tính tương thích của trang web với các trình duyệt khác nhau.

### **Cấu trúc cơ bản của CSS:** [5][6]

CSS

selector {

property: value;

property: value;

/\* ... \*/

}

* **Selector:** Xác định các phần tử HTML mà bạn muốn định dạng.
* **Property:** Xác định thuộc tính CSS mà bạn muốn áp dụng cho các phần tử được chọn.
* **Value:** Xác định giá trị cho thuộc tính CSS.

**Ví dụ:**

CSS

/\* Định dạng tất cả các tiêu đề h1 \*/

h1 {

font-family: Arial, sans-serif;

color: #333;

text-align: center;

margin-top: 20px;

}

/\* Định dạng các đoạn văn bản \*/

p {

font-size: 16px;

line-height: 1.5;

margin-bottom: 15px;

}

### ****Một số thuộc tính CSS phổ biến:**** [5][6]

* font-family: Xác định phông chữ cho văn bản.
* color: Xác định màu sắc cho văn bản.
* text-align: Xác định căn chỉnh văn bản (trái, phải, giữa, hai bên).
* margin: Xác định khoảng cách giữa các phần tử.
* padding: Xác định khoảng cách giữa nội dung và viền của phần tử.
* width: Xác định chiều rộng của phần tử.
* height: Xác định chiều cao của phần tử.
* background-color: Xác định màu nền của phần tử.
* border: Xác định đường viền cho phần tử.
* display: Xác định cách hiển thị phần tử (block, inline, none).
* position: Xác định vị trí của phần tử (static, relative, absolute, fixed).

### ****Một số lưu ý:****

* CSS có thể được viết trực tiếp trong thẻ HTML, trong tệp CSS riêng biệt hoặc được liên kết đến tệp HTML.
* Nên sử dụng các quy tắc CSS chung để tránh lặp lại code.
* Nên sử dụng các công cụ hỗ trợ CSS để viết và kiểm tra code.
* Nên cập nhật kiến thức về CSS thường xuyên để sử dụng các tính năng mới nhất.

### CSS nội tuyến (Inline CSS)

**CSS nội tuyến** là phương pháp định dạng các phần tử HTML bằng cách viết trực tiếp các quy tắc CSS vào thuộc tính style của thẻ HTML. [5] [6] [7]

#### ****Cấu trúc:****

HTML

<tag style="property: value; property: value;">Nội dung</tag>

* tag: Thẻ HTML cần định dạng.
* property: Thuộc tính CSS muốn áp dụng.
* value: Giá trị cho thuộc tính CSS.

**Ví dụ:**

HTML

<h1 style="color: blue; font-size: 24px;">Tiêu đề</h1>

<p style="text-align: center;">Nội dung đoạn văn.</p>

#### **.**Ưu điểm:****

* **Đơn giản:** Dễ dàng sử dụng và hiểu, phù hợp cho người mới bắt đầu.
* **Kiểm soát trực tiếp:** Cho phép định dạng các phần tử cụ thể một cách chính xác.
* **Sử dụng nhanh chóng:** Thích hợp cho những thay đổi nhỏ, đơn giản.

#### ****Nhược điểm:****

* **Lặp lại code:** Viết CSS trực tiếp trong HTML có thể dẫn đến lặp lại code, làm cho trang web trở nên khó bảo trì.
* **Khó quản lý:** Khi cần thay đổi định dạng cho nhiều phần tử, việc sử dụng CSS nội tuyến sẽ trở nên khó khăn.
* **Ảnh hưởng đến tính thẩm mỹ:** Viết CSS trực tiếp trong HTML có thể làm cho code HTML trở nên lộn xộn và khó đọc.

#### ****Khi nào nên sử dụng CSS nội tuyến:****

* **Định dạng các phần tử cụ thể một cách độc đáo:** Ví dụ, bạn có thể sử dụng CSS nội tuyến để định dạng một tiêu đề duy nhất với màu sắc và kích thước khác biệt so với các tiêu đề khác trên trang.
* **Thay đổi định dạng nhanh chóng:** Ví dụ, bạn có thể sử dụng CSS nội tuyến để thay đổi màu nền của một phần tử cho mục đích thử nghiệm hoặc demo.

#### ****Nên hạn chế sử dụng CSS nội tuyến cho các trường hợp sau:****

* **Định dạng nhiều phần tử giống nhau:** Thay vào đó, hãy sử dụng CSS bên ngoài hoặc các lớp CSS để định dạng các phần tử một cách chung.
* **Tạo giao diện phức tạp:** CSS bên ngoài và các công cụ CSS nâng cao sẽ giúp bạn tạo ra giao diện phức tạp một cách hiệu quả và dễ quản lý hơn.
* **Dự án web lớn:** Việc sử dụng CSS nội tuyến trong các dự án web lớn có thể dẫn đến code lộn xộn, khó bảo trì và dễ xảy ra lỗi.

### CSS nội bộ (Internal CSS)

**CSS nội bộ** là phương pháp định dạng các phần tử HTML bằng cách viết các quy tắc CSS trong một khối <style> nằm trong phần <head> của tệp HTML. [5] [6] [7]

#### ****Cấu trúc:****

HTML

<!DOCTYPE html>

<html>

<head>

<style>

/\* Các quy tắc CSS \*/

</style>

</head>

<body>

Nội dung trang web

</body>

</html>

**Ví dụ:**

HTML

<!DOCTYPE html>

<html>

<head>

<style>

h1 {

color: blue;

font-size: 24px;

}

p {

text-align: center;

margin-bottom: 15px;

}

</style>

</head>

<body>

<h1>Tiêu đề</h1>

<p>Nội dung đoạn văn.</p>

</body>

</html>

#### ****Ưu điểm:****

* **Giữ code HTML gọn gàng:** Viết CSS trong <head> giúp tách biệt code CSS khỏi code HTML, làm cho code HTML dễ đọc và dễ quản lý hơn.
* **Kiểm soát phạm vi:** Các quy tắc CSS trong <style> chỉ áp dụng cho các phần tử trong cùng một tệp HTML, giúp tránh xung đột với CSS từ các tệp khác.
* **Dễ dàng chỉnh sửa:** Việc chỉnh sửa CSS trong <style> đơn giản hơn so với việc thay đổi tệp CSS bên ngoài.

#### ****Nhược điểm:****

* **Phạm vi hạn chế:** CSS nội bộ chỉ áp dụng cho tệp HTML hiện tại, không thể áp dụng cho các tệp HTML khác.
* **Lặp lại code:** Nếu cần sử dụng cùng một quy tắc CSS cho nhiều tệp HTML, bạn sẽ phải lặp lại code trong mỗi tệp.
* **Khó quản lý:** Khi dự án web phát triển lớn, việc quản lý CSS nội bộ có thể trở nên khó khăn.

#### ****Khi nào nên sử dụng CSS nội bộ:****

* **Định dạng các phần tử trong một tệp HTML riêng biệt:** Ví dụ, bạn có thể sử dụng CSS nội bộ để định dạng giao diện cho một trang web con hoặc một phần cụ thể của trang web.
* **Tạo giao diện đơn giản:** Nếu giao diện của trang web tương đối đơn giản và không cần sử dụng lại CSS cho nhiều tệp HTML, CSS nội bộ có thể là lựa chọn phù hợp.
* **Dự án web nhỏ:** CSS nội bộ có thể là lựa chọn tốt cho các dự án web nhỏ với số lượng tệp HTML hạn chế.

#### ****Nên hạn chế sử dụng CSS nội bộ cho các trường hợp sau:****

* **Dự án web lớn:** Việc sử dụng CSS nội bộ trong các dự án web lớn có thể dẫn đến code lộn xộn, khó bảo trì và dễ xảy ra lỗi.
* **Cần sử dụng lại CSS:** Nếu cần sử dụng cùng một quy tắc CSS cho nhiều tệp HTML, hãy sử dụng CSS bên ngoài để tránh lặp lại code.
* **Tạo giao diện phức tạp:** CSS bên ngoài và các công cụ CSS nâng cao sẽ giúp bạn tạo ra giao diện phức tạp một cách hiệu quả và dễ quản lý hơn.

### CSS bên ngoài (External CSS)

**CSS bên ngoài** là phương pháp định dạng các phần tử HTML bằng cách viết các quy tắc CSS trong một tệp riêng biệt có đuôi .css, sau đó liên kết tệp CSS này với tệp HTML bằng thẻ <link>. [5] [6] [7]

#### ****Cấu trúc:****

##### **Tệp CSS:**

CSS

/\* Các quy tắc CSS \*/

##### **Tệp HTML:**

HTML

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="style.css">

</head>

<body>

Nội dung trang web

</body>

</html>

**Ví dụ:**

**1. Tệp style.css:**

CSS

h1 {

color: blue;

font-size: 24px;

}

p {

text-align: center;

margin-bottom: 15px;

}

**2. Tệp index.html:**

HTML

<!DOCTYPE html>

<html>

<head>

<link rel="stylesheet" href="style.css">

</head>

<body>

<h1>Tiêu đề</h1>

<p>Nội dung đoạn văn.</p>

</body>

</html>

#### ****Ưu điểm:****

* **Tái sử dụng:** Các quy tắc CSS trong tệp CSS bên ngoài có thể được áp dụng cho nhiều tệp HTML khác nhau, giúp tiết kiệm thời gian và công sức.
* **Dễ dàng quản lý:** Việc chỉnh sửa và cập nhật CSS chỉ cần thực hiện trong một tệp duy nhất, giúp code dễ quản lý hơn.
* **Giảm tải trang:** Trình duyệt chỉ cần tải tệp CSS một lần và áp dụng cho tất cả các tệp HTML liên kết, giúp cải thiện tốc độ tải trang.
* **Tăng tính tương thích:** CSS bên ngoài giúp đảm bảo tính tương thích của giao diện trên các trình duyệt khác nhau.

#### ****Nhược điểm:****

* **Phụ thuộc vào tệp CSS:** Nếu tệp CSS bị lỗi hoặc không được liên kết chính xác, giao diện trang web sẽ bị ảnh hưởng.
* **Cần thêm bước liên kết:** Việc liên kết tệp CSS với tệp HTML cần thêm một bước so với CSS nội tuyến và CSS nội bộ.
* **Có thể gây chậm tải trang ban đầu:** Việc tải tệp CSS có thể khiến trang web tải chậm hơn trong lần truy cập đầu tiên.

#### ****Khi nào nên sử dụng CSS bên ngoài:****

* **Định dạng nhiều tệp HTML:** CSS bên ngoài là lựa chọn tốt nhất khi cần định dạng nhiều tệp HTML với cùng một giao diện hoặc cần sử dụng lại CSS cho nhiều phần tử trong nhiều trang.
* **Tạo giao diện phức tạp:** CSS bên ngoài giúp bạn dễ dàng tổ chức và quản lý các quy tắc CSS phức tạp cho giao diện trang web.
* **Dự án web lớn:** CSS bên ngoài là lựa chọn thiết yếu cho các dự án web lớn với nhiều tệp HTML và nhiều phần tử cần định dạng.

#### ****Nên hạn chế sử dụng CSS bên ngoài cho các trường hợp sau:****

* **Định dạng các phần tử trong một tệp HTML riêng biệt:** Nếu chỉ cần định dạng các phần tử trong một tệp HTML cụ thể, CSS nội tuyến hoặc CSS nội bộ có thể là lựa chọn phù hợp hơn.
* **Tạo giao diện đơn giản:** CSS nội tuyến hoặc CSS nội bộ có thể là lựa chọn nhanh chóng và đơn giản cho các giao diện đơn giản.
* **Dự án web nhỏ:** CSS nội tuyến hoặc CSS nội bộ có thể là lựa chọn phù hợp cho các dự án web nhỏ với số lượng tệp HTML hạn chế.

### Phông chữ trong CSS

Để định dạng phông chữ cho văn bản trong CSS, bạn có thể sử dụng thuộc tính font-family, font-size, font-weight, font-style, line-height và các thuộc tính khác liên quan đến phông chữ. [5] [6] [7]

#### ****Thuộc tính**** font-family****:****

Thuộc tính này xác định các phông chữ mà bạn muốn sử dụng cho văn bản. Nên liệt kê các phông chữ theo thứ tự ưu tiên, từ phông chữ mong muốn nhất đến phông chữ dự phòng. Trình duyệt sẽ chọn phông chữ đầu tiên trong danh sách mà nó hỗ trợ. [5] [6] [7]

VD:

CSS

p {

font-family: Arial, sans-serif;

}

#### ****Thuộc tính**** font-size****:****

Thuộc tính này xác định kích thước của phông chữ. Kích thước có thể được định dạng bằng đơn vị pixel (px), em hoặc rem. [5] [6] [7]

VD:

CSS

h1 {

font-size: 24px; /\* Kích thước bằng pixel \*/

}

p {

font-size: 1em; /\* Kích thước tương đối so với phông chữ cha \*/

}

#### ****Thuộc tính**** font-weight****:****

Thuộc tính này xác định độ đậm của phông chữ. Các giá trị phổ biến bao gồm normal, bold, lighter, bolder. [5] [6] [7]

VD:

CSS

h1 {

font-weight: bold; /\* Phông chữ đậm \*/

}

#### ****Thuộc tính**** font-style****:****

Thuộc tính này xác định kiểu dáng của phông chữ. Các giá trị phổ biến bao gồm normal, italic, oblique. [5] [6] [7]

VD:

CSS

p {

font-style: italic; /\* Phông chữ nghiêng \*/

}

#### ****Thuộc tính**** line-height****:****

Thuộc tính này xác định khoảng cách giữa các dòng văn bản. Giá trị có thể được định dạng bằng đơn vị pixel, em hoặc rem. [5] [6] [7]

VD:

CSS

p {

line-height: 1.5em; /\* Khoảng cách dòng bằng 1.5 lần kích thước phông chữ \*/

}

#### ****Một số thuộc tính khác:****

Ngoài ra, còn có nhiều thuộc tính CSS khác liên quan đến phông chữ, chẳng hạn như: [5] [6] [7]

* text-decoration: Định dạng gạch chân, gạch ngang, phông chữ gạch xóa cho văn bản.
* text-align: Căn chỉnh văn bản (trái, phải, giữa, hai bên).
* letter-spacing: Khoảng cách giữa các ký tự trong văn bản.
* word-spacing: Khoảng cách giữa các từ trong văn bản.
* text-transform: Chuyển đổi văn bản thành hoa, thường hoặc kiểu khác.

#### ****Ví dụ:****

CSS

h1 {

font-family: "Times New Roman", Georgia, serif;

font-size: 2.5em;

font-weight: bold;

color: #333;

text-align: center;

}

p {

font-family: Arial, sans-serif;

font-size: 1em;

line-height: 1.5em;

margin-bottom: 15px;

}

#### ****Lưu ý:****

* Nên sử dụng các phông chữ web để đảm bảo hiển thị thống nhất trên các trình duyệt khác nhau.
* Nên kiểm tra khả năng tương thích của phông chữ trước khi sử dụng.
* Nên sử dụng các công cụ hỗ trợ CSS để viết và kiểm tra code.

### Màu sắc cho văn bản trong CSS

Để định dạng màu sắc cho văn bản trong CSS, bạn có thể sử dụng thuộc tính color. Thuộc tính này cho phép bạn xác định màu sắc cho văn bản bằng nhiều cách khác nhau. [5] [6] [7]

#### ****Sử dụng mã màu HEX:****

Mã màu HEX sử dụng ký tự thập lục phân để biểu thị màu sắc [5] [6] [7]. Ví dụ:

CSS

p {

color: #FF0000; /\* Màu đỏ \*/

}

#### ****Sử dụng tên màu:****

Bạn có thể sử dụng tên màu tiếng Anh để định dạng màu sắc [5] [6] [7]. Ví dụ:

CSS

h1 {

color: red; /\* Màu đỏ \*/

}

#### ****Sử dụng giá trị RGB:****

Giá trị RGB (Red, Green, Blue) biểu thị màu sắc bằng cách sử dụng ba thành phần tương ứng với cường độ của ba màu đỏ, xanh lá cây và xanh lam. Mỗi thành phần có giá trị từ 0 đến 255. [5] [6] [7] Ví dụ:

CSS

p {

color: rgb(255, 0, 0); /\* Màu đỏ \*/

}

#### ****Sử dụng giá trị HSL:****

Giá trị HSL (Hue, Saturation, Lightness) biểu thị màu sắc bằng ba thành phần: Hue (sắc thái), Saturation (độ bão hòa) và Lightness (độ sáng). [5] [6] [7] Ví dụ:

CSS

h1 {

color: hsl(0, 100%, 50%); /\* Màu đỏ \*/

}

#### ****Sử dụng chức năng**** rgba() ****và**** hsla()****:****

Các chức năng này cho phép bạn định dạng màu sắc với giá trị alpha, xác định độ mờ của màu. [5] [6] [7] Ví dụ:

CSS

p {

color: rgba(255, 0, 0, 0.5); /\* Màu đỏ mờ 50% \*/

}

h1 {

color: hsla(0, 100%, 50%, 0.7); /\* Màu đỏ mờ 30% \*/

}

#### ****Lưu ý:****

* Nên sử dụng các màu sắc phù hợp với giao diện chung của trang web và dễ đọc cho người dùng.
* Nên kiểm tra khả năng tương thích của màu sắc trên các trình duyệt khác nhau.
* Nên sử dụng các công cụ hỗ trợ CSS để chọn và kiểm tra màu sắc.

### Căn chỉnh văn bản trong CSS

Để căn chỉnh văn bản trong CSS, bạn có thể sử dụng thuộc tính text-align. Thuộc tính này cho phép bạn căn chỉnh văn bản theo chiều ngang (trái, phải, giữa, hai bên) hoặc theo chiều dọc (trên, dưới, trung tâm).

#### ****Căn chỉnh theo chiều ngang:**** [5][6][7]

* text-align: left;: Căn chỉnh văn bản sang trái.
* text-align: right;: Căn chỉnh văn bản sang phải.
* text-align: center;: Căn chỉnh văn bản vào giữa.
* text-align: justify;: Căn chỉnh văn bản hai bên, tạo ra các dòng văn bản thẳng hàng.

**Ví dụ:**

CSS

/\* Căn chỉnh tiêu đề sang trái \*/

h1 {

text-align: left;

}

/\* Căn chỉnh đoạn văn sang phải \*/

p {

text-align: right;

}

/\* Căn chỉnh nội dung bảng vào giữa \*/

table td {

text-align: center;

}

/\* Căn chỉnh văn bản hai bên \*/

.justified-text {

text-align: justify;

}

#### ****Căn chỉnh theo chiều dọc:**** [5][6][7]

* vertical-align: top;: Căn chỉnh văn bản ở vị trí trên cùng của phần tử.
* vertical-align: middle;: Căn chỉnh văn bản ở vị trí giữa của phần tử theo chiều dọc.
* vertical-align: bottom;: Căn chỉnh văn bản ở vị trí dưới cùng của phần tử.

**Ví dụ:**

CSS

/\* Căn chỉnh nội dung ô bảng theo chiều dọc \*/

table td {

vertical-align: middle;

}

/\* Căn chỉnh ảnh và văn bản theo chiều dọc \*/

.image-text {

display: flex;

align-items: center;

}

#### ****Sử dụng các giá trị khác:**** [5][6][7]

Ngoài các giá trị được đề cập ở trên, thuộc tính text-align còn có một số giá trị khác như:

* text-align: inherit;: Kế thừa giá trị căn chỉnh từ phần tử cha.
* text-align: initial;: Đặt lại giá trị căn chỉnh về mặc định ban đầu của trình duyệt.

#### ****Lưu ý:****

* Hiệu ứng căn chỉnh văn bản có thể phụ thuộc vào loại phần tử và các thuộc tính CSS khác được áp dụng cho phần tử đó.
* Nên sử dụng các giá trị căn chỉnh phù hợp với bố cục và nội dung của trang web.
* Nên kiểm tra khả năng hiển thị của văn bản đã căn chỉnh trên các trình duyệt khác nhau.

### Khoảng cách giữa các phần tử trong CSS

Để điều chỉnh khoảng cách giữa các phần tử trong CSS, bạn có thể sử dụng các thuộc tính sau:

#### ****Thuộc tính**** margin****:****

Thuộc tính margin định dạng khoảng cách trống xung quanh phần tử, bao gồm khoảng cách từ phần tử đến các phần tử lân cận và khoảng cách từ phần tử đến mép trang. [5] [6] [7]

* margin-top: Khoảng cách phía trên phần tử.
* margin-right: Khoảng cách bên phải phần tử.
* margin-bottom: Khoảng cách phía dưới phần tử.
* margin-left: Khoảng cách bên trái phần tử.
* margin: Thiết lập giá trị giống nhau cho tất cả các hướng margin.

#### ****Ví dụ:****

CSS

p {

margin: 20px; /\* Khoảng cách 20px cho tất cả các hướng \*/

}

h1 {

margin-top: 30px; /\* Khoảng cách 30px phía trên \*/

margin-bottom: 15px; /\* Khoảng cách 15px phía dưới \*/

}

#### ****Thuộc tính**** padding****:****

Thuộc tính padding định dạng khoảng cách giữa nội dung của phần tử và các đường viền của phần tử. [5] [6] [7]

* padding-top: Khoảng cách phía trên nội dung.
* padding-right: Khoảng cách bên phải nội dung.
* padding-bottom: Khoảng cách phía dưới nội dung.
* padding-left: Khoảng cách bên trái nội dung.
* padding: Thiết lập giá trị giống nhau cho tất cả các hướng padding.

**Ví dụ:**

CSS

div {

padding: 10px; /\* Khoảng cách 10px cho tất cả các hướng \*/

border: 1px solid #ccc; /\* Thêm đường viền cho phần tử \*/

}

#### ****Thuộc tính**** border-spacing****:****

Thuộc tính border-spacing điều chỉnh khoảng cách giữa các đường viền của các ô trong bảng. [5] [6] [7]

**Ví dụ:**

CSS

table {

border-collapse: collapse; /\* Xóa đường viền giữa các ô \*/

border-spacing: 5px; /\* Khoảng cách 5px giữa các đường viền \*/

}

#### ****Sử dụng các đơn vị:****

Khoảng cách trong CSS có thể được định dạng bằng nhiều đơn vị khác nhau, [5] [6] [7] bao gồm:

* px: Pixel (đơn vị điểm ảnh)
* em: Kích thước tương đối so với phông chữ cha
* rem: Kích thước tương đối so với kích thước phông chữ gốc
* %: Phần trăm so với kích thước chứa
* vw: Phần trăm so với chiều rộng màn hình
* vh: Phần trăm so với chiều cao màn hình

#### ****Sử dụng các thuộc tính flexbox và grid:****

Các hệ thống bố cục flexbox và grid cung cấp các phương pháp nâng cao hơn để điều chỉnh khoảng cách giữa các phần tử, bao gồm khả năng tạo khoảng cách đều giữa các phần tử, điều chỉnh khoảng cách tự động dựa trên kích thước màn hình, v.v. [5] [6] [7]

#### ****Lưu ý:****

* Hiệu ứng của các thuộc tính khoảng cách có thể phụ thuộc vào loại phần tử và các thuộc tính CSS khác được áp dụng cho phần tử đó.
* Nên sử dụng các giá trị khoảng cách phù hợp với bố cục và nội dung của trang web.
* Nên kiểm tra khả năng hiển thị của các phần tử với các khoảng cách đã được điều chỉnh trên các trình duyệt khác nhau.

### Chiều rộng và chiều cao của phần tử trong CSS

Để điều chỉnh kích thước (chiều rộng và chiều cao) của phần tử trong CSS, bạn có thể sử dụng các thuộc tính sau:

#### ****Thuộc tính**** width****:****

Thuộc tính width xác định chiều rộng của phần tử. [5] [6] [7]

**Ví dụ:**

CSS

img {

width: 200px; /\* Chiều rộng 200 pixel \*/

}

.container {

width: 50%; /\* Chiều rộng 50% so với phần tử cha \*/

}

#### ****Thuộc tính**** height****:****

Thuộc tính height xác định chiều cao của phần tử. [5] [6] [7]

**Ví dụ:**

CSS

img {

height: 150px; /\* Chiều cao 150 pixel \*/

}

.container {

height: 300px; /\* Chiều cao 300 pixel \*/

}

#### ****Sử dụng các đơn vị:****

Chiều rộng và chiều cao trong CSS có thể được định dạng bằng nhiều đơn vị khác nhau, [5] [6] [7] bao gồm:

* px: Pixel (đơn vị điểm ảnh)
* em: Kích thước tương đối so với phông chữ cha
* rem: Kích thước tương đối so với kích thước phông chữ gốc
* %: Phần trăm so với kích thước chứa
* vw: Phần trăm so với chiều rộng màn hình
* vh: Phần trăm so với chiều cao màn hình

#### ****Giá trị tự động:****

Nếu không khai báo thuộc tính width hoặc height cho một phần tử, trình duyệt sẽ tự động tính toán kích thước dựa trên nội dung của phần tử. Tuy nhiên, điều này có thể dẫn đến kết quả không mong muốn trong một số trường hợp.

#### ****Sử dụng giá trị**** max-width ****và**** max-height****:****

Thuộc tính max-width và max-height đặt kích thước tối đa cho chiều rộng và chiều cao của phần tử, ngăn không cho phần tử lớn hơn kích thước đã chỉ định. [5] [6] [7]

**Ví dụ:**

CSS

img {

max-width: 100%; /\* Chiều rộng tối đa 100% \*/

height: auto; /\* Chiều cao tự động \*/

}

#### ****Sử dụng giá trị**** min-width ****và**** min-height****:****

Thuộc tính min-width và min-height đặt kích thước tối thiểu cho chiều rộng và chiều cao của phần tử, đảm bảo rằng phần tử không nhỏ hơn kích thước đã chỉ định. [5] [6] [7]

**Ví dụ:**

CSS

.container {

min-width: 300px; /\* Chiều rộng tối thiểu 300 pixel \*/

height: auto; /\* Chiều cao tự động \*/

}

#### ****Sử dụng các thuộc tính flexbox và grid:****

Các hệ thống bố cục flexbox và grid cung cấp các phương pháp nâng cao hơn để điều chỉnh kích thước của phần tử, bao gồm khả năng tạo các phần tử có kích thước linh hoạt, tự động thay đổi kích thước dựa trên kích thước màn hình, v.v. [5] [6] [7]

#### ****Lưu ý:****

* Hiệu ứng của các thuộc tính kích thước có thể phụ thuộc vào loại phần tử và các thuộc tính CSS khác được áp dụng cho phần tử đó.
* Nên sử dụng các giá trị kích thước phù hợp với bố cục và nội dung của trang web.
* Nên kiểm tra khả năng hiển thị của các phần tử với kích thước đã được điều chỉnh trên các trình duyệt khác nhau.

### Màu nền cho phần tử trong CSS

Để định dạng màu nền cho phần tử trong CSS, bạn có thể sử dụng thuộc tính background-color. Thuộc tính này cho phép bạn đặt màu sắc cho nền của phần tử. [5] [6] [7]

#### ****Cú pháp:****

CSS

selector {

background-color: color-value;

}

#### ****Ví dụ:****

CSS

body {

background-color: #f0f0f0; /\* Nền trang màu xám nhạt \*/

}

.container {

background-color: #333; /\* Nền container màu đen \*/

}

p {

background-color: #fff; /\* Nền đoạn văn màu trắng \*/

}

#### ****Giá trị màu sắc:****

Bạn có thể sử dụng các định dạng màu sắc khác nhau cho thuộc tính background-color, [5] [6] [7] bao gồm:

* **Mã màu HEX:** Ví dụ: #ff0000 (màu đỏ).
* **Tên màu:** Ví dụ: red (màu đỏ).
* **Giá trị RGB:** Ví dụ: rgb(255, 0, 0) (màu đỏ).
* **Giá trị HSL:** Ví dụ: hsl(0, 100%, 50%) (màu đỏ).
* **Giá trị transparent:** Để tạo nền trong suốt.

#### ****Hình ảnh nền:****

Ngoài màu sắc, bạn cũng có thể sử dụng hình ảnh làm nền cho phần tử bằng cách sử dụng thuộc tính background-image. [5] [6] [7]

CSS

.section {

background-image: url("image.jpg"); /\* Nền hình ảnh cho phần tử \*/

background-repeat: repeat; /\* Lặp lại hình ảnh \*/

background-position: center; /\* Vị trí hình ảnh \*/

}

#### ****Thuộc tính liên quan:****

Ngoài background-color và background-image, còn có một số thuộc tính CSS khác liên quan đến màu nền của phần tử, [5] [6] [7] bao gồm:

* background-repeat: Xác định cách lặp lại hình ảnh nền (lặp lại, không lặp lại, v.v.).
* background-position: Xác định vị trí của hình ảnh nền (trên cùng bên trái, trung tâm, v.v.).
* background-size: Xác định kích thước của hình ảnh nền (bao phủ, chứa, tự động, v.v.).
* background-attachment: Xác định cách hình ảnh nền di chuyển khi cuộn trang (cố định, cuộn cùng, v.v.).

#### ****Lưu ý:****

* Hiệu ứng màu nền có thể phụ thuộc vào loại phần tử và các thuộc tính CSS khác được áp dụng cho phần tử đó.
* Nên sử dụng màu nền phù hợp với giao diện chung của trang web và dễ nhìn cho người dùng.
* Nên kiểm tra khả năng hiển thị của màu nền trên các trình duyệt khác nhau.
* Nên sử dụng các công cụ hỗ trợ CSS để chọn và kiểm tra màu sắc.

### Đường viền cho phần tử trong CSS

Để tạo đường viền cho phần tử trong CSS, bạn có thể sử dụng các thuộc tính sau:

#### ****Thuộc tính**** border****:****

Thuộc tính border là viết tắt cho ba thuộc tính: border-width, border-style và border-color. Thuộc tính này cho phép bạn thiết lập độ dày, kiểu và màu sắc cho đường viền của phần tử. [5] [6] [7]

#### ****Cú pháp:****

CSS

selector {

border: width style color;

}

**Ví dụ:**

CSS

p {

border: 2px solid #ccc; /\* Đường viền dày 2px, kiểu solid, màu xám nhạt \*/

}

.button {

border: 1px dashed #000; /\* Đường viền dày 1px, kiểu dashed, màu đen \*/

}

img {

border: 5px dotted red; /\* Đường viền dày 5px, kiểu dotted, màu đỏ \*/

}

#### ****Các thuộc tính riêng lẻ:**** [5][6][7]

* border-width: Thiết lập độ dày của đường viền.
* border-style: Thiết lập kiểu đường viền (solid, dashed, dotted, none).
* border-color: Thiết lập màu sắc cho đường viền.

**Ví dụ:**

CSS

h1 {

border-width: 10px 5px 2px 1px; /\* Độ dày đường viền cho các cạnh: trên, phải, dưới, trái \*/

}

.box {

border-style: solid double dotted; /\* Kiểu đường viền cho các cạnh: trên, phải, dưới, trái \*/

}

#### ****Giá trị đường viền:**** [5][6][7]

* thin: Đường viền mỏng.
* medium: Đường viền trung bình.
* thick: Đường viền dày.
* none: Không có đường viền.

**Ví dụ:**

CSS

table td {

border: 1px solid; /\* Đường viền mỏng cho tất cả các cạnh \*/

}

.container {

border: 2px medium #666; /\* Đường viền trung bình dày 2px màu xám \*/

}

#### ****Đường viền cho các cạnh riêng lẻ:**** [5][6][7]

Bạn có thể sử dụng các thuộc tính riêng lẻ để thiết lập đường viền cho từng cạnh của phần tử:

* border-top: Đường viền cạnh trên.
* border-right: Đường viền cạnh phải.
* border-bottom: Đường viền cạnh dưới.
* border-left: Đường viền cạnh trái.

**Ví dụ:**

CSS

img {

border-top: 3px solid red; /\* Đường viền cạnh trên dày 3px màu đỏ \*/

border-bottom: 5px dashed blue; /\* Đường viền cạnh dưới dày 5px kiểu dashed màu xanh dương \*/

}

#### ****Lưu ý:****

* Hiệu ứng đường viền có thể phụ thuộc vào loại phần tử và các thuộc tính CSS khác được áp dụng cho phần tử đó.
* Nên sử dụng kiểu và màu đường viền phù hợp với giao diện chung của trang web.
* Nên kiểm tra khả năng hiển thị của đường viền trên các trình duyệt khác nhau.

### Cách hiển thị phần tử trong CSS

Có nhiều cách để hiển thị phần tử trong CSS, tùy thuộc vào hiệu ứng mong muốn và loại phần tử bạn muốn hiển thị. Dưới đây là một số phương pháp phổ biến:

#### ****Sử dụng thuộc tính**** display****:****

Thuộc tính display là một trong những cách cơ bản nhất để kiểm soát hiển thị của phần tử. Thuộc tính này có nhiều giá trị khác nhau, [5] [6] [7] bao gồm:

* block: Hiển thị phần tử như một khối, chiếm toàn bộ chiều rộng của phần tử cha.
* inline: Hiển thị phần tử như một phần tử nội tuyến, chỉ chiếm phần không gian cần thiết cho nội dung của nó.
* none: Ẩn phần tử hoàn toàn.
* flex: Hiển thị phần tử theo cách bố cục flexbox.
* grid: Hiển thị phần tử theo cách bố cục grid.

**Ví dụ:**

CSS

/\* Hiển thị phần tử p như một khối \*/

p {

display: block;

}

/\* Hiển thị phần tử span như một phần tử nội tuyến \*/

span {

display: inline;

}

/\* Ẩn phần tử div \*/

.hidden {

display: none;

}

#### ****Sử dụng thuộc tính**** visibility****:****

Thuộc tính visibility cũng có thể được sử dụng để kiểm soát hiển thị của phần tử, [5] [6] [7] nhưng nó khác với display ở một số điểm:

* visible: Hiển thị phần tử (mặc định).
* hidden: Ẩn phần tử nhưng vẫn giữ nguyên vị trí và kích thước trong bố cục.
* collapse: Ẩn phần tử và thu gọn vị trí của nó trong bố cục.

**Ví dụ:**

CSS

/\* Ẩn phần tử p nhưng vẫn giữ nguyên vị trí trong bố cục \*/

p.invisible {

visibility: hidden;

}

/\* Ẩn phần tử div và thu gọn vị trí của nó trong bố cục \*/

.collapsed {

visibility: collapse;

}

#### ****Sử dụng thuộc tính**** opacity****:****

Thuộc tính opacity cho phép bạn điều chỉnh độ mờ của phần tử, từ 0 (hoàn toàn trong suốt) đến 1 (hoàn toàn mờ đục). [5] [6] [7]

**Ví dụ:**

CSS

/\* Làm mờ phần tử p 50% \*/

p {

opacity: 0.5;

}

/\* Làm mờ phần tử img 75% \*/

img {

opacity: 0.75;

}

#### ****Sử dụng các kỹ thuật bố cục:****

Ngoài các thuộc tính đơn lẻ như display, visibility và opacity, bạn cũng có thể sử dụng các kỹ thuật bố cục nâng cao như flexbox và grid để kiểm soát cách hiển thị các phần tử trong CSS. Các kỹ thuật này cung cấp nhiều khả năng linh hoạt hơn để định vị và sắp xếp các phần tử trên trang. [5] [6] [7]

#### ****Lưu ý:****

* Hiệu ứng hiển thị của phần tử có thể phụ thuộc vào loại phần tử và các thuộc tính CSS khác được áp dụng cho phần tử đó.
* Nên sử dụng các kỹ thuật hiển thị phù hợp với mục đích và nội dung của trang web.
* Nên kiểm tra khả năng hiển thị của các phần tử trên các trình duyệt khác nhau.

### Vị trí của phần tử trong CSS

Để xác định vị trí của phần tử trong CSS, bạn có thể sử dụng các thuộc tính sau:

#### ****Thuộc tính**** position****:****

Thuộc tính position xác định cách phần tử được định vị trong bối cảnh của tài liệu hoặc phần tử cha. [5] [6] [7]

Có nhiều giá trị cho thuộc tính position, bao gồm:

* static: Vị trí mặc định của phần tử, được xác định bởi dòng chảy bình thường của tài liệu.
* relative: Vị trí của phần tử được di chuyển so với vị trí bình thường của nó trong dòng chảy tài liệu.
* fixed: Vị trí của phần tử được cố định so với khung nhìn của trình duyệt, không di chuyển khi cuộn trang.
* absolute: Vị trí của phần tử được định vị hoàn toàn so với phần tử cha gần nhất, tách khỏi dòng chảy tài liệu.

**Ví dụ:**

CSS

/\* Định vị phần tử p tương đối so với vị trí bình thường của nó \*/

p {

position: relative;

top: 20px; /\* Di chuyển 20px từ vị trí trên \*/

left: 10px; /\* Di chuyển 10px từ vị trí trái \*/

}

/\* Định vị phần tử div cố định so với khung nhìn \*/

.fixed-element {

position: fixed;

top: 0; /\* Vị trí trên cùng \*/

right: 0; /\* Vị trí bên phải \*/

width: 300px; /\* Chiều rộng \*/

height: 200px; /\* Chiều cao \*/

}

/\* Định vị phần tử img tuyệt đối so với phần tử cha gần nhất \*/

img.absolute {

position: absolute;

top: 50%; /\* Vị trí 50% từ trên \*/

left: 50%; /\* Vị trí 50% từ trái \*/

transform: translate(-50%, -50%); /\* Di chuyển 50% kích thước theo cả hai hướng \*/

}

#### ****Các thuộc tính vị trí khác:****

Ngoài position, còn có một số thuộc tính khác liên quan đến vị trí của phần tử, [5] [6] [7] bao gồm:

* top: Di chuyển phần tử từ vị trí trên.
* right: Di chuyển phần tử từ vị trí bên phải.
* bottom: Di chuyển phần tử từ vị trí dưới.
* left: Di chuyển phần tử từ vị trí bên trái.
* z-index: Xác định thứ tự xếp chồng của các phần tử khi chúng chồng chéo lên nhau.

#### ****Lưu ý:****

* Hiệu ứng vị trí của phần tử có thể phụ thuộc vào loại phần tử và các thuộc tính CSS khác được áp dụng cho phần tử đó.
* Nên sử dụng các giá trị vị trí phù hợp với bố cục và nội dung của trang web.
* Nên kiểm tra khả năng hiển thị của các phần tử với vị trí đã được điều chỉnh trên các trình duyệt khác nhau.

## Ngôn ngữ Javascript

### Ngôn ngữ JavaScript

JavaScript là một ngôn ngữ lập trình kịch bản được sử dụng để thêm các tính năng tương tác và động vào các trang web. Nó cho phép bạn thao tác với các phần tử HTML, tạo ra các hiệu ứng động và xử lý dữ liệu phía người dùng. JavaScript là một ngôn ngữ mạnh mẽ và linh hoạt, nó được sử dụng trong nhiều ứng dụng web khác nhau, [1]bao gồm:

* **Tạo các trang web tương tác:** JavaScript có thể được sử dụng để tạo các trang web tương tác với người dùng, ví dụ như các biểu mẫu, trò chơi, và các ứng dụng web.
* **Phát triển các ứng dụng web một trang (SPA):** JavaScript là ngôn ngữ chính được sử dụng để phát triển các ứng dụng web một trang (SPA), cho phép bạn tạo ra các ứng dụng web mượt mà và nhanh chóng.
* **Tạo các hiệu ứng động:** JavaScript có thể được sử dụng để tạo ra các hiệu ứng động trên trang web, ví dụ như các menu thả xuống, các thanh trượt, và các hiệu ứng chuyển tiếp.
* **Xử lý dữ liệu phía người dùng:** JavaScript có thể được sử dụng để xử lý dữ liệu phía người dùng, ví dụ như thu thập dữ liệu từ biểu mẫu, xác thực dữ liệu đầu vào, và thao tác với dữ liệu JSON.

### Đặc điểm chính của JavaScript:

* **Ngôn ngữ dựa trên nguyên mẫu:** JavaScript sử dụng mô hình lập trình dựa trên nguyên mẫu, trong đó các đối tượng được tạo ra dựa trên các đối tượng khác, thừa kế các thuộc tính và phương thức của chúng.
* **Hỗ trợ các hàm hạng nhất:** JavaScript cho phép bạn tạo, truyền và sử dụng các hàm như bất kỳ loại dữ liệu nào khác, mang lại tính linh hoạt và mạnh mẽ cho ngôn ngữ.
* **Được giải thích hoặc biên dịch ngay lập tức:** JavaScript có thể được giải thích trực tiếp bởi trình duyệt hoặc biên dịch thành mã máy trước khi chạy, giúp nó linh hoạt và hiệu quả.
* **Có nhiều thư viện và khung:** JavaScript có hệ sinh thái phong phú gồm các thư viện và khung, giúp bạn dễ dàng phát triển các ứng dụng web phức tạp và các ứng dụng khác.

### Ứng dụng của JavaScript:

* **Phát triển web:** JavaScript là ngôn ngữ chính được sử dụng để tạo các trang web tương tác và năng động. Nó được sử dụng để tạo ra các ứng dụng web một trang (SPA), các trò chơi web, các hiệu ứng hình ảnh và nhiều hơn nữa.
* **Phát triển ứng dụng di động:** JavaScript có thể được sử dụng để phát triển ứng dụng di động lai bằng các khung như React Native và Cordova.
* **Phát triển ứng dụng máy chủ:** Node.js là môi trường thời gian chạy JavaScript cho phép bạn viết các ứng dụng máy chủ mạnh mẽ và có thể mở rộng.
* **Phát triển game:** JavaScript có thể được sử dụng để phát triển các trò chơi 2D và 3D bằng các thư viện như Phaser và Babylon.js.
* **Tự động hóa:** JavaScript có thể được sử dụng để tự động hóa các tác vụ và quy trình bằng các công cụ như Selenium và Puppeteer.

### Ưu điểm của JavaScript:

* **Dễ học và sử dụng:** JavaScript có cú pháp tương đối đơn giản và dễ học, ngay cả đối với những người mới bắt đầu.
* **Linh hoạt và mạnh mẽ:** JavaScript hỗ trợ nhiều tính năng và mô hình lập trình, giúp nó linh hoạt và có thể được sử dụng cho nhiều mục đích khác nhau.
* **Cộng đồng lớn và tích cực:** JavaScript có cộng đồng lập trình viên lớn và tích cực, luôn sẵn sàng giúp đỡ và chia sẻ kiến ​​thức.
* **Nhiều thư viện và khung:** JavaScript có hệ sinh thái phong phú gồm các thư viện và khung, giúp bạn dễ dàng phát triển các ứng dụng phức tạp.

### Nhược điểm của JavaScript:

* **Có thể khó gỡ lỗi:** JavaScript có thể khó gỡ lỗi do tính năng động và dựa trên nguyên mẫu của nó.
* **Có thể không an toàn:** JavaScript có thể bị khai thác để thực thi mã độc hại nếu không được viết cẩn thận.
* **Hiệu suất có thể khác nhau:** Hiệu suất của JavaScript có thể khác nhau tùy thuộc vào trình duyệt và môi trường thực thi.

### Cách thêm JavaScript vào trang web

Có hai phương pháp chính để thêm JavaScript vào trang web:

#### ****Sử dụng thẻ**** <script>****:****

* **Đặt thẻ <script>** trong phần <head> của tài liệu HTML. Vị trí này giúp tải JavaScript trước khi hiển thị nội dung trang, đảm bảo JavaScript được thực thi đúng cách. [8] [9] [10] [11]
* **Sử dụng thuộc tính src** để **liên kết đến tệp JavaScript** bên ngoài. Ví dụ:

HTML

<head>

<title>Trang web của tôi</title>

<script src="script.js"></script>

</head>

* **Viết mã JavaScript** **trực tiếp** trong thẻ <script>. Phương pháp này chỉ phù hợp cho các đoạn mã ngắn đơn giản. Ví dụ:

HTML

<head>

<title>Trang web của tôi</title>

<script>

function chào() {

alert("Chào mừng bạn đến với trang web của tôi!");

}

</script>

</head>

#### ****Sử dụng thẻ**** <script> ****trong phần**** <body>****:****

* **Đặt thẻ <script>** trong phần <body> của tài liệu HTML. Vị trí này **tải JavaScript sau khi tải nội dung trang**, nhưng có thể ảnh hưởng đến hiệu suất trang. [8] [9] [10] [11]
* **Sử dụng thuộc tính src** để **liên kết đến tệp JavaScript** bên ngoài. Ví dụ:

HTML

<body>

<p>Nội dung trang web...</p>

<script src="script.js"></script>

</body>

* **Viết mã JavaScript** **trực tiếp** trong thẻ <script>. Phương pháp này **không khuyến khích**, vì có thể làm chậm tốc độ tải trang và gây khó khăn cho việc bảo trì mã.

#### ****Lưu ý:****

* **Đảm bảo tệp JavaScript** được đặt đúng vị trí trong thư mục dự án của bạn.
* **Sử dụng thuộc tính type="text/javascript"** trong thẻ <script> để đảm bảo trình duyệt hiểu đúng loại tập tin.
* **Đóng thẻ <script>** bằng dấu </script>.
* **Kiểm tra cú pháp JavaScript** để đảm bảo không có lỗi.
* **Sử dụng công cụ gỡ lỗi** của trình duyệt để kiểm tra xem JavaScript có hoạt động chính xác hay không.

### Javascript cơ bản

**Javascript** là một ngôn ngữ lập trình kịch bản phổ biến được sử dụng để tạo ra các trang web tương tác và năng động. Nó cho phép bạn thêm tính năng và hành vi cho trang web của mình, giúp trang web trở nên hấp dẫn và thú vị hơn cho người dùng.

**Dưới đây là một số kiến ​​thức cơ bản về Javascript mà bạn nên biết:**

#### Các kiểu dữ liệu:

Javascript có các kiểu dữ liệu cơ bản sau: [8] [9] [10] [11]

* **Số nguyên:** Ví dụ: 10, 20, -50
* **Số thực:** Ví dụ: 3.14, 9.87, -2.5
* **Chuỗi:** Ví dụ: "Hello", "World!", "This is a string."
* **Boolean:** true hoặc false
* **Mảng:** Dữ liệu được tổ chức trong một danh sách có thứ tự. Ví dụ: ["apple", "banana", "orange"]
* **Đối tượng:** Dữ liệu được tổ chức theo cặp khóa-giá trị. Ví dụ: { name: "John", age: 30, city: "Hanoi" }

#### Các toán tử:

Javascript có các toán tử toán học cơ bản [8] [9] [10] [11] như +, -, \*, /, % (lấy dư), toán tử so sánh như ==, !=, <, >, <=, >=, và toán tử logic như && (và), || (hoặc), ! (không).

#### Các biến:

Biến được sử dụng để lưu trữ dữ liệu. Bạn khai báo biến bằng cách sử dụng từ khóa var, let hoặc const. [8] [9] [10] [11] Ví dụ:

JavaScript

var name = "Alice";

let age = 25;

const city = "Hanoi";

#### Cấu trúc điều khiển:

Cấu trúc điều khiển giúp bạn kiểm soát luồng thực thi chương trình. Javascript có các cấu trúc điều khiển cơ bản [8] [9] [10] [11] như:

* **Câu lệnh if:** Thực thi một khối mã nếu điều kiện được thỏa mãn.
* **Câu lệnh switch:** Thực thi một khối mã dựa trên giá trị của một biến.
* **Vòng lặp for:** Lặp lại một khối mã một số lần nhất định.
* **Vòng lặp while:** Lặp lại một khối mã cho đến khi điều kiện được thỏa mãn.

#### Hàm:

Hàm là một khối mã được sử dụng để thực hiện một nhiệm vụ cụ thể. Bạn có thể tạo hàm bằng cách sử dụng từ khóa function. [8] [9] [10] [11] Ví dụ:

JavaScript

function greeting(name) {

console.log("Chào mừng bạn " + name + "!");

}

greeting("Bob"); // Hiển thị "Chào mừng bạn Bob!"

#### Sự kiện:

Sự kiện là những hành động xảy ra trong trang web, như khi người dùng nhấp chuột vào một nút hoặc di chuyển chuột qua một phần tử. Javascript cho phép bạn lắng nghe các sự kiện và thực thi mã khi chúng xảy ra. [8] [9] [10] [11] Ví dụ:

JavaScript

const button = document.getElementById("myButton");

button.addEventListener("click", function() {

alert("Bạn đã nhấp vào nút!");

});

#### DOM (Document Object Model):

DOM là đại diện dạng cây của trang web trong bộ nhớ của trình duyệt. Javascript cho phép bạn truy cập và thao tác với các phần tử HTML trong DOM. [8] [9] [10] [11] Ví dụ:

JavaScript

const element = document.getElementById("myElement");

element.textContent = "Nội dung mới";

### Các kiểu dữ liệu trong Javascript

Javascript có 8 **kiểu dữ liệu** cơ bản, được chia thành hai loại:

#### ****Kiểu dữ liệu nguyên thủy (Primitive Data Types):****

##### **Số nguyên (Number):**

Biểu diễn các giá trị số nguyên, ví dụ: 10, -20, 5000. [8] [9] [10] [11]

##### **Số thực (Number):**

Biểu diễn các giá trị số thực, ví dụ: 3.14, 9.87, -2.5. [8] [9] [10] [11]

##### **Chuỗi (String):**

Biểu diễn các chuỗi ký tự, được đặt trong dấu nháy đơn (') hoặc dấu nháy kép ("), ví dụ: "Hello", "World!", "This is a string.". [8] [9] [10] [11]

##### **Boolean (Boolean):**

Biểu diễn hai giá trị logic: true và false. [8] [9] [10] [11]

##### **Null (Null):**

Biểu diễn giá trị rỗng, không có giá trị nào được gán. [8] [9] [10] [11]

##### **Undefined (Undefined):**

Biểu diễn giá trị chưa được khởi tạo hoặc giá trị của một biến đã bị xóa. [8] [9] [10] [11]

#### ****Kiểu dữ liệu tham chiếu (Reference Data Types):****

##### Mảng (Array):

Biểu diễn một tập hợp các giá trị, được lưu trữ trong ngoặc vuông []. Mỗi giá trị trong mảng được gọi là một phần tử, có thể truy cập bằng chỉ mục. Ví dụ: ["apple", "banana", "orange"], [1, 2, 3, 4, 5]. [8] [9] [10] [11]

##### Đối tượng (Object):

Biểu diễn một tập hợp các cặp khóa-giá trị, được bao bọc trong dấu ngoặc nhọn {}. Mỗi cặp khóa-giá trị được gọi là một thuộc tính, có thể truy cập bằng tên khóa. Ví dụ: { name: "John", age: 30, city: "Hanoi" }, { color: "red", width: 100, height: 50 }. [8] [9] [10] [11]

#### ****Lưu ý:****

* Kiểu dữ liệu nguyên thủy được lưu trữ trực tiếp trong bộ nhớ.
* Kiểu dữ liệu tham chiếu lưu trữ một tham chiếu đến vị trí trong bộ nhớ nơi dữ liệu thực sự được lưu trữ.
* Mỗi kiểu dữ liệu có các đặc điểm và cách sử dụng riêng biệt.

#### ****Ngoài ra, Javascript còn có một số kiểu dữ liệu khác:****

* **Symbol:** Biểu diễn các giá trị duy nhất, không thể thay đổi và được sử dụng để xác định các thuộc tính riêng tư hoặc các khóa đối tượng.
* **BigInt:** Biểu diễn các số nguyên lớn hơn giới hạn của kiểu Number.

#### ****Để kiểm tra kiểu dữ liệu của một biến, bạn có thể sử dụng toán tử**** typeof****:**** [8] [9] [10] [11]

JavaScript

const myVariable = 10;

console.log(typeof myVariable); // Hiển thị "number"

const myString = "Hello";

console.log(typeof myString); // Hiển thị "string"

const myArray = ["apple", "banana", "orange"];

console.log(typeof myArray); // Hiển thị "object"

const myObject = { name: "John", age: 30 };

console.log(typeof myObject); // Hiển thị "object"

### Các toán tử trong Javascript

Javascript có nhiều **toán tử** được sử dụng để thực hiện các phép toán và so sánh dữ liệu. Các toán tử có thể được phân loại thành các nhóm sau:

#### ****Toán tử số học:**** [8] [9] [10] [11]

* +: Phép cộng
* -: Phép trừ
* \*: Phép nhân
* /: Phép chia
* %: Phép lấy dư
* ++: Tăng giá trị biến lên 1
* --: Giảm giá trị biến đi 1

**Ví dụ:**

JavaScript

const a = 10;

const b = 5;

console.log(a + b); // Hiển thị 15

console.log(a - b); // Hiển thị 5

console.log(a \* b); // Hiển thị 50

console.log(a / b); // Hiển thị 2

console.log(a % b); // Hiển thị 0

let x = 10;

x++;

console.log(x); // Hiển thị 11

let y = 5;

y--;

console.log(y); // Hiển thị 4

#### ****Toán tử so sánh:**** [8] [9] [10] [11]

* ==: So sánh bằng nhau (kiểm tra giá trị)
* !=: So sánh không bằng nhau (kiểm tra giá trị)
* ===: So sánh bằng nhau (kiểm tra cả giá trị và kiểu dữ liệu)
* !==: So sánh không bằng nhau (kiểm tra cả giá trị và kiểu dữ liệu)
* <: So sánh nhỏ hơn
* <=: So sánh nhỏ hơn hoặc bằng
* >: So sánh lớn hơn
* >=: So sánh lớn hơn hoặc bằng

**Ví dụ:**

JavaScript

const a = 10;

const b = 5;

const c = "10";

console.log(a == b); // Hiển thị false (giá trị khác nhau)

console.log(a != b); // Hiển thị true (giá trị khác nhau)

console.log(a === c); // Hiển thị false (kiểu dữ liệu khác nhau)

console.log(a !== c); // Hiển thị true (kiểu dữ liệu khác nhau)

console.log(a < b); // Hiển thị false

console.log(a <= b); // Hiển thị false

console.log(a > b); // Hiển thị true

console.log(a >= b); // Hiển thị true

#### ****Toán tử logic:**** [8] [9] [10] [11]

* &&: Phép "và" (true chỉ khi cả hai điều kiện đều true)
* ||: Phép "hoặc" (true nếu một hoặc cả hai điều kiện đều true)
* !: Phép "không" (lật ngược giá trị logic)

**Ví dụ:**

JavaScript

const age = 20;

const isStudent = true;

console.log(age >= 18 && isStudent); // Hiển thị true (cả hai điều kiện đều true)

console.log(age < 18 || isStudent); // Hiển thị true (một trong hai điều kiện true)

let isRaining = false;

console.log(!isRaining); // Hiển thị true (lật ngược giá trị logic)

#### ****Toán tử gán:**** [8] [9] [10] [11]

* =: Gán giá trị cho biến
* +=: Gán giá trị cộng thêm cho biến
* -=: Gán giá trị trừ đi cho biến
* \*=: Gán giá trị nhân cho biến
* /=: Gán giá trị chia cho biến
* %=: Gán giá trị lấy dư cho biến

**Ví dụ:**

JavaScript

let x = 10;

x += 5;

console.log(x); // Hiển thị 15

x -= 3;

console.log(x); // Hiển thị 12

x \*= 2;

console.log(x); // Hiển thị 24

#### ****Toán tử chuỗi:**** [8] [9] [10] [11]

* +: Nối hai chuỗi (phép cộng chuỗi)
* +=: Nối chuỗi và gán kết quả cho biến

**Ví dụ:**

JavaScript

const firstName = "John";

const lastName = "Doe";

const fullName = firstName + " " + lastName;

console.log(fullName); // Hiển thị "John Doe"

let message = "Hello";

message += " World!";

console.log(message); // Hi

### Các biến trong Javascript

Biến là một phần tử quan trọng trong lập trình Javascript, giúp lưu trữ dữ liệu và sử dụng dữ liệu đó trong chương trình. Biến cho phép bạn đặt tên cho một vùng nhớ trong bộ nhớ máy tính để lưu trữ giá trị, sau đó có thể truy cập và thao tác với giá trị đó bằng tên biến.

#### ****Khai báo biến:****

Để khai báo biến trong Javascript, bạn sử dụng từ khóa var, let hoặc const, theo sau là tên biến và giá trị khởi tạo (tùy chọn):

**var:** Đây là từ khóa khai báo biến truyền thống, có phạm vi toàn cục (global scope) và phạm vi hàm (function scope). Biến khai báo bằng var có thể được truy cập và thay đổi từ bất kỳ nơi nào trong chương trình. [8] [9] [10] [11]

JavaScript

var name = "Alice"; // Khai báo biến var có tên "name" với giá trị "Alice"

**let:** Từ khóa let khai báo biến với phạm vi khối (block scope). Biến khai báo bằng let chỉ có thể truy cập và thay đổi trong khối mã nơi nó được khai báo. [8] [9] [10] [11]

JavaScript

if (true) {

let age = 30; // Khai báo biến let có tên "age" với giá trị 30 trong khối if

console.log(age); // Hiển thị 30

}

console.log(age); // Lỗi: age không được khai báo ngoài khối if

**const:** Từ khóa const khai báo biến hằng số, có nghĩa là giá trị của biến không thể thay đổi sau khi được khởi tạo. [8] [9] [10] [11]

JavaScript

const city = "Hanoi"; // Khai báo biến const có tên "city" với giá trị "Hanoi"

city = "Ho Chi Minh"; // Lỗi: city là biến hằng số và không thể thay đổi giá trị

#### ****Sử dụng biến:****

Để sử dụng biến, bạn chỉ cần gọi tên biến. [8] [9] [10] [11] Ví dụ:

JavaScript

var name = "Bob";

console.log(name); // Hiển thị "Bob"

let age = 25;

console.log("Tuổi của " + name + " là: " + age); // Hiển thị "Tuổi của Bob là: 25"

#### ****Quy tắc đặt tên biến:****

* Tên biến phải bắt đầu bằng chữ cái, số hoặc dấu gạch dưới (\_).
* Không được sử dụng khoảng trắng trong tên biến.
* Tên biến không được trùng với từ khóa Javascript.
* Nên đặt tên biến theo quy ước đặt tên để dễ hiểu và dễ quản lý.

#### ****Kiểu dữ liệu của biến:****

Kiểu dữ liệu của biến xác định loại dữ liệu mà biến có thể lưu trữ. Javascript có các kiểu dữ liệu cơ bản như số nguyên, số thực, chuỗi, boolean, mảng và đối tượng. Kiểu dữ liệu của biến được xác định tự động dựa trên giá trị khởi tạo.

#### ****Ví dụ về sử dụng biến:****

JavaScript

// Khai báo biến var, let và const

var message = "Chào mừng bạn đến với Javascript!";

let counter = 0;

const PI = 3.14159;

// Sử dụng biến trong các phép toán

let sum = 10 + 20;

console.log("Tổng của 10 và 20 là: " + sum);

// Sử dụng biến trong chuỗi

let greeting = "Xin chào " + message;

console.log(greeting);

// Thay đổi giá trị biến

counter++;

console.log("Giá trị hiện tại của counter là: " + counter);

#### ****Lưu ý:****

* Nên sử dụng let và const thay vì var để khai báo biến mới vì let và const có phạm vi rõ ràng hơn và giúp tránh lỗi.
* Nên đặt tên biến theo quy ước đặt tên để dễ hiểu và dễ quản lý.
* Nên sử dụng kiểu dữ liệu phù hợp cho biến để tránh lỗi và tối ưu hóa hiệu suất chương trình.

### Cấu trúc điều khiển trong Javascript

Cấu trúc điều khiển là những khối mã giúp bạn kiểm soát luồng thực thi chương trình Javascript, cho phép bạn thực hiện các hành động khác nhau dựa trên điều kiện nhất định. Javascript có các cấu trúc điều khiển cơ bản sau:

#### ****Câu lệnh**** if****:****

Câu lệnh if sử dụng để thực thi một khối mã nếu điều kiện được thỏa mãn. Cấu trúc cơ bản của câu lệnh if [8] [9] [10] [11] như sau:

JavaScript

if (điều kiện) {

// Khối mã được thực thi nếu điều kiện đúng

} else {

// Khối mã được thực thi nếu điều kiện sai

}

**Ví dụ:**

JavaScript

const age = 20;

if (age >= 18) {

console.log("Bạn đã đủ tuổi trưởng thành.");

} else {

console.log("Bạn chưa đủ tuổi trưởng thành.");

}

#### ****Câu lệnh**** switch****:****

Câu lệnh switch sử dụng để thực thi một khối mã dựa trên giá trị của một biến. Cấu trúc cơ bản của câu lệnh switch [8] [9] [10] [11] như sau:

JavaScript

switch (biến) {

case giá trị1:

// Khối mã được thực thi nếu biến bằng giá trị1

break;

case giá trị2:

// Khối mã được thực thi nếu biến bằng giá trị2

break;

default:

// Khối mã được thực thi nếu không có giá trị nào khớp

}

**Ví dụ:**

JavaScript

const day = 1;

switch (day) {

case 1:

console.log("Thứ Hai");

break;

case 2:

console.log("Thứ Ba");

break;

case 3:

console.log("Thứ Tư");

break;

default:

console.log("Ngày không hợp lệ");

}

#### ****Vòng lặp**** for****:****

Vòng lặp for sử dụng để lặp lại một khối mã nhiều lần, với số lần lặp được xác định trước. Cấu trúc cơ bản của vòng lặp for [8] [9] [10] [11] như sau:

JavaScript

for (let i = 0; i < n; i++) {

// Khối mã được lặp lại n lần

}

**Ví dụ:**

JavaScript

for (let i = 1; i <= 5; i++) {

console.log("Số lần lặp:", i);

}

#### ****Vòng lặp**** while****:****

Vòng lặp while sử dụng để lặp lại một khối mã cho đến khi điều kiện được thỏa mãn. Cấu trúc cơ bản của vòng lặp while [8] [9] [10] [11] như sau:

JavaScript

while (điều kiện) {

// Khối mã được lặp lại cho đến khi điều kiện sai

}

**Ví dụ:**

JavaScript

let i = 0;

while (i < 10) {

console.log("Số:", i);

i++;

}

#### ****Cấu trúc điều khiển lồng nhau:****

Bạn có thể kết hợp các cấu trúc điều khiển khác nhau để tạo ra các logic phức tạp hơn. Ví dụ, bạn có thể sử dụng câu lệnh if trong vòng lặp for hoặc vòng lặp while. [8] [9] [10] [11]

#### ****Sử dụng các cấu trúc điều khiển hiệu quả:****

* Sử dụng cấu trúc điều khiển phù hợp với mục đích và nhu cầu của bạn.
* Tránh lạm dụng các cấu trúc điều khiển làm cho mã khó hiểu và khó bảo trì.
* Sử dụng các bình luận để giải thích mục đích của các cấu trúc điều khiển.

### Hàm trong Javascript

Hàm là những khối mã được sử dụng để thực hiện một nhiệm vụ cụ thể hoặc một tập hợp các hành động trong Javascript. Hàm giúp bạn tái sử dụng mã, chia nhỏ chương trình thành các phần nhỏ hơn và dễ quản lý hơn, đồng thời làm cho mã trở nên rõ ràng và dễ hiểu hơn.

#### ****Khai báo hàm:****

Để khai báo hàm trong Javascript, bạn sử dụng từ khóa function, theo sau là tên hàm, dấu ngoặc đơn (()), và ngoặc nhọn ({}) bao gồm khối mã của hàm. [8] [9] [10] [11] Ví dụ:

JavaScript

function chào() {

console.log("Chào mừng bạn đến với Javascript!");

}

#### ****Gọi hàm:****

Để gọi hàm, bạn chỉ cần sử dụng tên hàm và dấu ngoặc đơn (()). [8] [9] [10] [11] Ví dụ:

JavaScript

chào(); // Hiển thị "Chào mừng bạn đến với Javascript!"

#### ****Hàm có tham số:****

Hàm có thể nhận tham số để cung cấp dữ liệu cho hàm thực hiện. Tham số được truyền vào hàm trong dấu ngoặc đơn (()) khi gọi hàm. [8] [9] [10] [11] Ví dụ:

JavaScript

function chàoTên(name) {

console.log("Chào mừng bạn " + name + "!");

}

chàoTên("Alice"); // Hiển thị "Chào mừng bạn Alice!"

chàoTên("Bob"); // Hiển thị "Chào mừng bạn Bob!"

#### ****Hàm trả về giá trị:****

Hàm có thể trả về giá trị bằng cách sử dụng từ khóa return. Giá trị trả về được lấy sau khi hàm thực thi xong. [8] [9] [10] [11] Ví dụ:

JavaScript

function tínhTổng(a, b) {

return a + b;

}

const sum = tínhTổng(10, 20);

console.log("Tổng của 10 và 20 là:", sum); // Hiển thị "Tổng của 10 và 20 là: 30"

#### ****Phạm vi biến:****

Biến khai báo bên trong hàm chỉ có phạm vi trong hàm đó (phạm vi hàm). Biến khai báo ngoài hàm có phạm vi toàn cục (global scope). [8] [9] [10] [11]

#### ****Ví dụ về sử dụng hàm:****

JavaScript

// Hàm tính bình phương của một số

function bìnhPhương(number) {

return number \* number;

}

// Hàm tính diện tích hình vuông

function diệnTíchHìnhVuông(cạnh) {

return bìnhPhương(cạnh);

}

const a = 5;

console.log("Bình phương của " + a + " là:", bìnhPhương(a)); // Hiển thị "Bình phương của 5 là: 25"

console.log("Diện tích hình vuông cạnh " + a + " là:", diệnTíchHìnhVuông(a)); // Hiển thị "Diện tích hình vuông cạnh 5 là: 25"

#### ****Lưu ý:****

* Nên đặt tên hàm theo quy ước đặt tên để dễ hiểu và dễ quản lý.
* Nên sử dụng hàm để tái sử dụng mã và chia nhỏ chương trình thành các phần nhỏ hơn.
* Nên sử dụng bình luận để giải thích mục đích của hàm và các tham số của hàm.

### Sự kiện trong Javascript

Sự kiện trong Javascript là những hành động xảy ra trong trang web, như khi người dùng nhấp chuột vào một nút, di chuyển chuột qua một phần tử hoặc nhập văn bản vào một ô nhập liệu. Javascript cho phép bạn lắng nghe các sự kiện này và thực thi mã khi chúng xảy ra, giúp bạn tạo ra các trang web tương tác và năng động.

#### ****Các loại sự kiện:****

Có rất nhiều loại sự kiện khác nhau trong Javascript, một số loại sự kiện phổ biến [8] [9] [10] [11] bao gồm:

* **Sự kiện chuột:** click, mousedown, mouseup, mousemove, mouseover, mouseout
* **Sự kiện bàn phím:** keydown, keypress, keyup
* **Sự kiện biểu mẫu:** submit, change, focus, blur
* **Sự kiện DOM:** load, unload, resize, scroll

#### ****Lắng nghe sự kiện:****

Để lắng nghe sự kiện, bạn sử dụng phương thức addEventListener() của đối tượng DOM. Phương thức này nhận ba tham số: [8] [9] [10] [11]

* **Loại sự kiện:** Tên của sự kiện bạn muốn lắng nghe.
* **Hàm xử lý sự kiện:** Hàm sẽ được thực thi khi sự kiện xảy ra.
* **Tùy chọn: Boolean bubble:** Xác định sự kiện có nổi bọt (bubble) hay không.

Ví dụ:

JavaScript

const button = document.getElementById("myButton");

button.addEventListener("click", function() {

alert("Bạn đã nhấp vào nút!");

});

#### ****Hàm xử lý sự kiện:****

Hàm xử lý sự kiện là hàm được thực thi khi sự kiện xảy ra. Hàm này nhận một đối tượng Event làm tham số, chứa thông tin chi tiết về sự kiện. [8] [9] [10] [11]

Ví dụ:

JavaScript

const button = document.getElementById("myButton");

button.addEventListener("click", function(event) {

console.log("X: " + event.clientX);

console.log("Y: " + event.clientY);

});

#### ****Loại bỏ sự kiện:****

Để loại bỏ sự kiện, bạn sử dụng phương thức removeEventListener() của đối tượng DOM. Phương thức này nhận hai tham số: [8] [9] [10] [11]

* **Loại sự kiện:** Tên của sự kiện bạn muốn loại bỏ.
* **Hàm xử lý sự kiện:** Hàm đã được thêm trước đó.

Ví dụ:

JavaScript

const button = document.getElementById("myButton");

button.addEventListener("click", myEventHandler);

function myEventHandler(event) {

alert("Bạn đã nhấp vào nút!");

}

// Loại bỏ sự kiện sau 5 giây

setTimeout(function() {

button.removeEventListener("click", myEventHandler);

}, 5000);

#### ****Sử dụng sự kiện hiệu quả:****

* Nên sử dụng sự kiện một cách hợp lý để tránh tạo ra các sự kiện không cần thiết và ảnh hưởng đến hiệu suất trang web.
* Nên sử dụng các phương thức addEventListener() và removeEventListener() một cách cẩn thận để tránh rò rỉ bộ nhớ.
* Nên sử dụng bình luận để giải thích mục đích của các sự kiện và các hàm xử lý sự kiện.

### DOM (Document Object Model) trong Javascript

Trong lập trình Javascript, DOM (Document Object Model) đóng vai trò quan trọng như một **đại diện dạng cây** của trang web, được lưu trữ trong bộ nhớ của trình duyệt. DOM cung cấp cho Javascript một **giao diện lập trình** để truy cập và thao tác với các thành phần HTML, CSS và nội dung của trang web.

#### Các phương thức DOM thông dụng: [8] [9] [10] [11]

* getElementById: Lấy một thẻ theo ID duy nhất của nó.
* getElementsByTagName: Lấy một tập hợp các thẻ theo tên thẻ của chúng.
* createElement: Tạo một thẻ HTML mới.
* appendChild: Thêm một nút con vào một nút khác.
* removeChild: Xóa một nút con khỏi nút cha của nó.
* innerHTML: Lấy hoặc đặt nội dung HTML của một thẻ.
* style: Lấy hoặc đặt các thuộc tính style của một thẻ.

#### ****Những điều bạn có thể làm với DOM:****

* **Truy cập và thay đổi nội dung của các phần tử HTML:**
  + Lấy nội dung văn bản hiện tại của một phần tử.
  + Thay đổi nội dung văn bản đó.
  + Thêm hoặc xóa nội dung hoàn toàn.
* **Thay đổi kiểu (style) của các phần tử:**
  + Thay đổi màu sắc, font chữ, kích thước, ... của phần tử.
* **Thêm, xóa và sắp xếp lại các phần tử:**
  + Tạo các phần tử HTML mới bằng Javascript và thêm chúng vào DOM.
  + Xóa các phần tử hiện có.
  + Di chuyển các phần tử để thay đổi bố cục của trang.
* **Xử lý sự kiện:**
  + Lắng nghe các sự kiện của người dùng trên trang web (click chuột, di chuột, cuộn trang, ...).
  + Thực thi mã Javascript để phản hồi với các sự kiện này.

**Ví dụ:**

JavaScript

// Lấy element với id "myElement"

const element = document.getElementById("myElement");

// Thay đổi nội dung

element.textContent = "Nội dung mới";

// Thêm class "highlight"

element.classList.add("highlight");

// Xóa element

element.parentNode.removeChild(element);

// Lắng nghe sự kiện click trên nút

const button = document.getElementById("myButton");

button.addEventListener("click", function() {

alert("Bạn đã nhấp vào nút!");

});

#### ****Lợi ích của việc sử dụng DOM:****

* **Tạo các trang web tương tác và năng động:** DOM giúp bạn tạo các trang web phản hồi với hành động của người dùng, cập nhật nội dung theo yêu cầu và cung cấp trải nghiệm người dùng phong phú hơn.
* **Xây dựng các giao diện người dùng phức tạp:** DOM là nền tảng để xây dựng các giao diện người dùng với các thành phần có thể tương tác.
* **Cho phép Javascript tương tác với HTML và CSS:** DOM đóng vai trò trung gian, cho phép Javascript đọc và thao tác với nội dung và style của trang web được xác định trong HTML và CSS.

## Kết chương

Quy trình phát triển website là một quá trình quan trọng để đảm bảo website được xây dựng một cách hiệu quả, chất lượng và đáp ứng các yêu cầu của khách hàng. Quy trình này bao gồm nhiều bước, từ việc lên kế hoạch và thiết kế đến việc xây dựng và triển khai website.

**Bên cạnh quy trình phát triển website, bài viết này cũng đã cung cấp thông tin về các ngôn ngữ lập trình web phổ biến như HTML, CSS và JavaScript.** Đây là những ngôn ngữ cơ bản mà bất kỳ ai muốn phát triển website cũng cần phải học.Mỗi thành phần đóng vai trò riêng biệt nhưng lại phối hợp nhịp nhàng để tạo ra những trang web hiện đại và tương tác:

### ****HTML (HyperText Markup Language)****:

* Là ngôn ngữ đánh dấu được sử dụng để xây dựng cấu trúc cơ bản của trang web, bao gồm các phần tử như heading, paragraphs, images, links, v.v.
* Cung cấp khung sườn cho nội dung và xác định các thành phần khác nhau của trang.
* Không có khả năng tạo giao diện trực quan phức tạp.

### ****CSS (Cascading Style Sheets)****:

* Là ngôn ngữ định dạng được sử dụng để kiểm soát giao diện và bố cục của trang web.
* Định dạng màu sắc, font chữ, kích thước, vị trí và các thuộc tính hiển thị khác của các phần tử HTML.
* Tạo ra giao diện trực quan đẹp mắt và thu hút cho người dùng.

### ****JavaScript****:

* Là ngôn ngữ lập trình được sử dụng để thêm chức năng tương tác và động cho trang web.
* Cho phép người dùng tương tác với các phần tử HTML, phản hồi hành động của người dùng, cập nhật nội dung trang web theo thời gian thực, tạo các hiệu ứng động và trò chơi đơn giản.
* Mang lại trải nghiệm người dùng phong phú và hấp dẫn.

### ****Sự kết hợp của HTML, CSS và JavaScript:****

* Giúp tạo ra các trang web đầy đủ chức năng, đẹp mắt và tương tác.
* HTML cung cấp cấu trúc, CSS định dạng giao diện, và JavaScript mang lại sự linh hoạt và tương tác.
* Ba ngôn ngữ này là nền tảng thiết yếu cho bất kỳ ai muốn phát triển các trang web hiện đại.

### ****Bên cạnh kiến thức cơ bản về HTML, CSS và JavaScript, chương này cũng đề cập đến một số chủ đề quan trọng khác:****

* Vai trò của trình duyệt web trong việc diễn giải và hiển thị các trang web.
* Công cụ dành cho nhà phát triển web để tạo và chỉnh sửa trang web.
* Một số khái niệm cơ bản về bảo mật web.

Hiểu rõ về HTML, CSS và JavaScript là bước đầu tiên quan trọng để trở thành một nhà phát triển web thành công. Chúc bạn tiếp tục học hỏi và khám phá những kiến thức chuyên sâu hơn về lĩnh vực lập trình web đầy thú vị này.

# Phân tích thiết kế hệ thống

## Mô tả bài toán

Hệ thống review truyện là một ứng dụng web hoặc di động cho phép người dùng chia sẻ đánh giá và nhận xét về các tác phẩm truyện tranh, tiểu thuyết, hoặc sách nói. Hệ thống này có thể bao gồm các chức năng sau:

* + **Tạo tài khoản:** Người dùng có thể tạo tài khoản để truy cập đầy đủ các tính năng của hệ thống.
  + **Tìm kiếm truyện:** Người dùng có thể tìm kiếm truyện theo tiêu đề, tác giả, thể loại, hoặc các từ khóa khác.
  + **Xem thông tin truyện:** Hệ thống cung cấp thông tin chi tiết về mỗi truyện, bao gồm tóm tắt nội dung, hình ảnh bìa, danh sách tác giả, thể loại, v.v.
  + **Đọc review:** Người dùng có thể đọc các review do người khác viết về truyện.
  + **Viết review:** Người dùng có thể viết review của riêng họ về truyện, bao gồm đánh giá, nhận xét, và tóm tắt nội dung.
  + **Đánh giá truyện:** Người dùng có thể đánh giá truyện bằng cách cho điểm từ 1 đến 5 sao.
  + **Theo dõi truyện:** Người dùng có thể theo dõi các truyện mà họ quan tâm để nhận thông báo về các cập nhật mới, như review mới hoặc chương mới.
  + **Gợi ý truyện:** Hệ thống có thể gợi ý cho người dùng những truyện mà họ có thể thích dựa trên lịch sử đọc truyện và sở thích của họ.
  + **Tham gia cộng đồng:** Người dùng có thể tham gia vào các diễn đàn thảo luận về truyện, chia sẻ ý kiến ​​và kinh nghiệm đọc của họ với những người dùng khác.

### Mục tiêu chính:

* Cung cấp nền tảng trực tuyến cho người dùng chia sẻ và đánh giá các tác phẩm truyện tranh, tiểu thuyết, ngôn tình, v.v.
* Tạo cộng đồng yêu thích truyện online để kết nối, trao đổi và khám phá những tác phẩm mới.

### Mục tiêu phụ:

* Thu hút lượng truy cập cao và tạo dựng thương hiệu cho website.
* Kiếm thêm thu nhập thông qua quảng cáo hoặc các dịch vụ liên quan.

## Đặc tả các yêu cầu

### Yêu cầu chức năng

#### Chức Năng Chính:

##### Quản lý tài khoản người dùng:

* Đăng ký tài khoản.
* Đăng nhập tài khoản.
* Cập nhật thông tin cá nhân.
* Quản lý danh sách truyện yêu thích.
* Theo dõi các tác giả yêu thích.

##### Tìm kiếm và lọc truyện:

* Tìm kiếm truyện theo tên, tác giả, thể loại, trạng thái, v.v.
* Lọc truyện theo tiêu chí: mới nhất, phổ biến nhất, được đánh giá cao nhất, v.v.
* Tìm kiếm nâng cao với nhiều tiêu chí kết hợp.

##### Xem thông tin chi tiết về truyện:

* Tóm tắt nội dung truyện.
* Mô tả chi tiết về truyện.
* Hình ảnh bìa truyện.
* Danh sách các chương truyện.
* Thông tin về tác giả.
* Đánh giá và bình luận của người dùng.
* Danh sách truyện tương tự.

##### Đánh giá và bình luận về truyện:

* Viết đánh giá và bình luận cho truyện.
* Xếp hạng truyện từ 1 đến 5 sao.
* So sánh các tác phẩm truyện.
* Thích và phản hồi đánh giá của người khác.

##### Quản lý danh sách truyện yêu thích:

* Thêm truyện vào danh sách yêu thích.
* Xóa truyện khỏi danh sách yêu thích.
* Sắp xếp danh sách yêu thích theo tiêu chí.
* Nhận thông báo về các chương truyện mới của các tác phẩm trong danh sách yêu thích.

##### Tham gia cộng đồng:

* Giao lưu, kết bạn với những người cùng sở thích.
* Tham gia các nhóm thảo luận về truyện.
* Tạo nhóm thảo luận mới.
* Tham gia các sự kiện của cộng đồng.

#### Chức Năng Phụ:

##### Hệ thống đề xuất truyện:

* Gợi ý những tác phẩm phù hợp với sở thích của người dùng dựa trên lịch sử đọc truyện, đánh giá và bình luận.
* Gợi ý những tác phẩm mới đang được quan tâm.
* Gợi ý những tác phẩm được đánh giá cao bởi cộng đồng.

##### Chức năng thông báo:

* Cập nhật thông tin về các truyện mới, chương mới, sự kiện của cộng đồng.
* Thông báo về các hoạt động của người dùng mà bạn quan tâm.
* Thông báo về tin nhắn và lời mời kết bạn.

##### Hệ thống quản lý quảng cáo:

* Hiển thị quảng cáo phù hợp với đối tượng người dùng.
* Quản lý vị trí hiển thị quảng cáo.
* Theo dõi hiệu quả của các chiến dịch quảng cáo.

##### Hệ thống thống kê truy cập:

* Theo dõi số lượng người dùng, lượt truy cập, hành vi người dùng.
* Phân tích dữ liệu để hiểu rõ hơn về nhu cầu của người dùng.
* Cải thiện website dựa trên dữ liệu thống kê.

### Yêu cầu phi chức năng

#### Hiệu suất

* **Tốc độ truy cập:** Website cần đảm bảo tốc độ truy cập nhanh chóng, mượt mà cho người dùng, ngay cả khi có nhiều người truy cập cùng lúc.
* **Thời gian tải trang:** Các trang web, đặc biệt là trang chủ và trang thông tin chi tiết về truyện, cần được tải nhanh chóng để tránh gây khó chịu cho người dùng.
* **Khả năng xử lý dữ liệu:** Website cần có khả năng xử lý dữ liệu hiệu quả để đảm bảo hoạt động ổn định khi có lượng truy cập lớn hoặc khi có nhiều người dùng tương tác cùng lúc.

#### Khả năng mở rộng

* **Hỗ trợ lượng truy cập cao:** Website cần có khả năng mở rộng để có thể đáp ứng lượng truy cập ngày càng tăng trong tương lai.
* **Hỗ trợ thêm nhiều chức năng:** Website cần có cấu trúc linh hoạt để có thể dễ dàng thêm các chức năng mới trong tương lai mà không ảnh hưởng đến hiệu suất của website.
* **Hỗ trợ lưu trữ nhiều dữ liệu:** Website cần có khả năng lưu trữ lượng dữ liệu lớn, bao gồm thông tin người dùng, thông tin về truyện, hình ảnh, video, v.v.

#### Bảo mật

* **Bảo mật thông tin người dùng:** Website cần đảm bảo bảo mật thông tin người dùng, bao gồm thông tin cá nhân, mật khẩu, thông tin thanh toán, v.v.
* **Chống tấn công mạng:** Website cần được bảo vệ khỏi các cuộc tấn công mạng như tấn công SQL Injection, Cross-Site Scripting (XSS), v.v.
* **Bảo vệ dữ liệu:** Website cần có các biện pháp để bảo vệ dữ liệu khỏi bị đánh cắp, sao chép trái phép hoặc bị phá hủy.

#### Khả năng sử dụng

* **Giao diện dễ sử dụng:** Giao diện website cần dễ sử dụng, dễ hiểu và dễ điều hướng cho người dùng, bất kể trình độ kỹ thuật của họ.
* **Khả năng truy cập:** Website cần đảm bảo khả năng truy cập cho tất cả người dùng, bao gồm cả người khuyết tật.
* **Hỗ trợ đa ngôn ngữ:** Website nên hỗ trợ nhiều ngôn ngữ để đáp ứng nhu cầu của người dùng ở các quốc gia khác nhau.

#### Khả năng bảo trì

* **Cấu trúc mã rõ ràng:** Cấu trúc mã của website cần rõ ràng, dễ hiểu và dễ bảo trì.
* **Công cụ quản trị dễ sử dụng:** Website cần có công cụ quản trị dễ sử dụng để cho phép quản trị viên website cập nhật nội dung, quản lý người dùng và theo dõi hiệu suất website.
* **Tài liệu hướng dẫn đầy đủ:** Website cần có tài liệu hướng dẫn đầy đủ để hướng dẫn người dùng cách sử dụng website và hướng dẫn quản trị viên cách quản lý website.

#### Khả năng tương thích

* **Hỗ trợ đa trình duyệt:** Website cần hỗ trợ đa trình duyệt web phổ biến như Chrome, Firefox, Safari, Edge, v.v.
* **Hỗ trợ đa thiết bị:** Website cần tương thích với đa thiết bị truy cập, từ máy tính bàn, laptop đến điện thoại thông minh, máy tính bảng.
* **Thiết kế responsive:** Website cần có thiết kế responsive để tự động điều chỉnh giao diện phù hợp với kích thước màn hình của thiết bị truy cập.

#### Tiêu chí đánh giá yêu cầu phi chức năng

* **Mức độ đáp ứng các yêu cầu:** Website cần đáp ứng đầy đủ các yêu cầu phi chức năng được đề ra ở trên.
* **Mức độ ảnh hưởng đến trải nghiệm người dùng:** Các yêu cầu phi chức năng cần được đáp ứng ở mức độ cao để đảm bảo trải nghiệm người dùng tốt nhất.
* **Chi phí thực hiện:** Chi phí thực hiện các yêu cầu phi chức năng cần được cân nhắc hợp lý với khả năng của dự án.

## Thiết kế dữ liệu

### Mô hình dữ liệu

**Mô hình ERD** (Entity-Relationship Diagram) là một biểu đồ mô tả các thực thể (entity) trong hệ thống và mối quan hệ (relationship) giữa chúng. Mô hình ERD giúp trực quan hóa cấu trúc dữ liệu của hệ thống và dễ dàng xác định các yêu cầu dữ liệu cần thiết.

Entities: Người dùng, Truyện, Chương, Đánh giá, Danh sách truyện yêu thích

Attributes:

- Người dùng: user\_id (PK), username, email, password, fullname, avatar, description, created\_at, updated\_at

- Truyện: truyện\_id (PK), ten\_truyen, tacgia, theloai, tomtat, mota\_chitiet, anh\_bia, trang\_thai, luot\_xem, luot\_thich, luot\_danhgia, created\_at, updated\_at

- Chương: chuong\_id (PK), ten\_chuong, noi\_dung, truyện\_id (FK), thu\_tu, created\_at, updated\_at

- Đánh giá: danhgia\_id (PK), diem\_danhgia, binh\_luan, nguoidung\_id (FK), truyen\_id (FK), created\_at, updated\_at

- Danh sách truyện yêu thích: truyenyeuthich\_id (PK), nguoidung\_id (FK), truyen\_id (FK), created\_at, updated\_at

Relationships:

- Người dùng tạo ra Truyện: (Người dùng, Truyện)

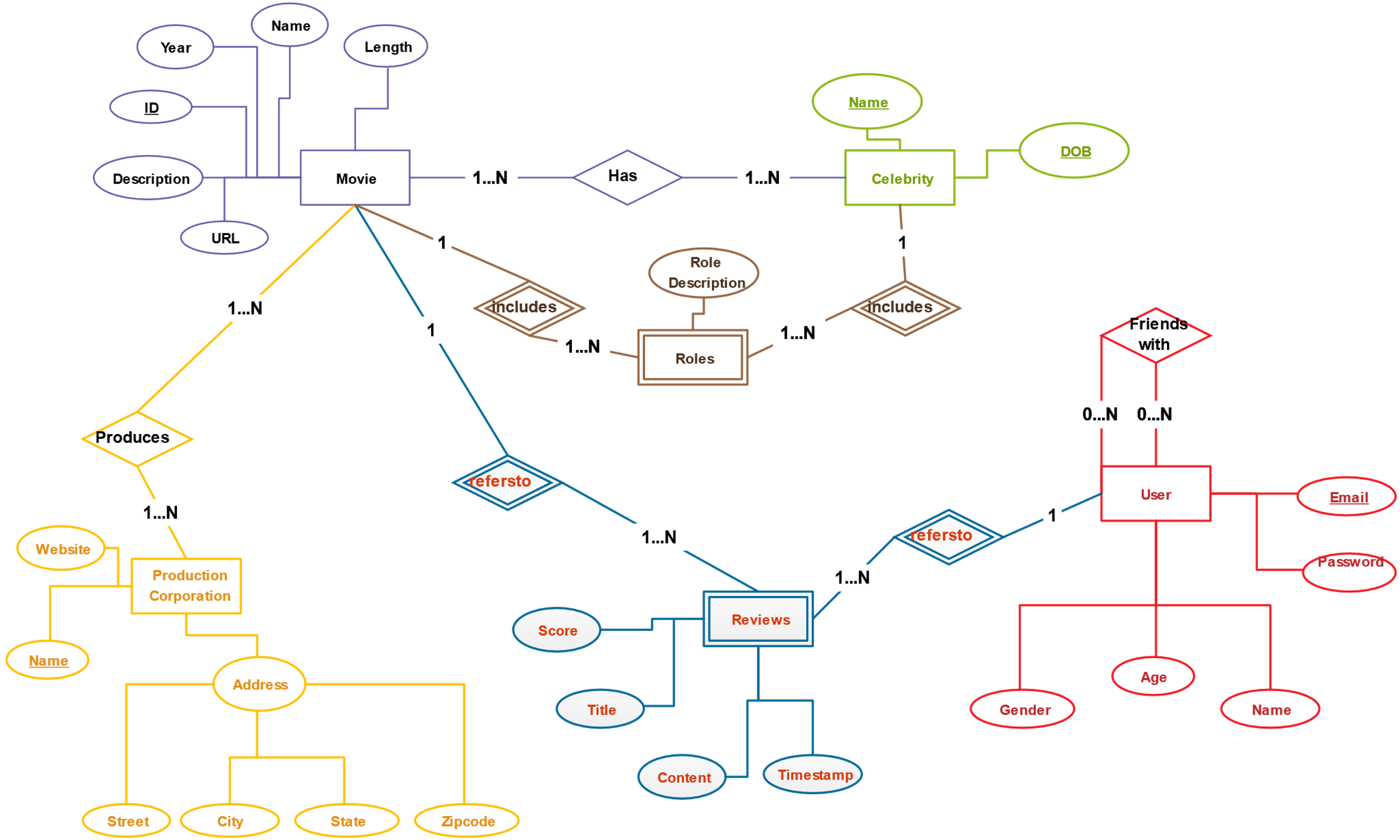
- Truyện bao gồm nhiều Chương: (Truyện, Chương)

- Người dùng đánh giá Truyện: (Người dùng, Đánh giá, Truyện)

- Người dùng có Danh sách truyện yêu thích: (Người dùng, Danh sách truyện yêu thích)

- Danh sách truyện yêu thích bao gồm nhiều Truyện: (Danh sách truyện yêu thích, Truyện)

Trình bày mô hình ERD



Hình minh họa (tạm)

### Danh sách các thực thể và mối kết hợp

Bảng 2.1 Danh sách các thực thể và mối kết hợp

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên thực thể/Mối kết hợp | Diễn giải | Ghi chú |
|  |  |  |  |
|  |  |  |  |

### Chi tiết thực thể và mối kết hợp

Thực thể

Mô tả

Chi tiết thực thể

Bảng 2.2 Chi tiết thực thể …

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc toàn vẹn |
|  |  |  |  |  |
|  |  |  |  |  |

## Thiết kế xử lý

### Sơ đồ DFD mức ngữ cảnh

**Sơ đồ DFD mức ngữ cảnh** mô tả hệ thống review truyện ở mức độ tổng quan, chỉ ra các tác nhân bên ngoài và các luồng dữ liệu chính tương tác với hệ thống.

### Mô tả:

#### Tác nhân bên ngoài:

* Người dùng: Tương tác với hệ thống thông qua giao diện web hoặc di động.
* Admin: Quản lý hệ thống, bao gồm quản lý người dùng, review, truyện, v.v.

#### Luồng dữ liệu:

* Người dùng gửi yêu cầu (tìm kiếm truyện, xem thông tin truyện, viết review, v.v.) đến hệ thống.
* Hệ thống xử lý yêu cầu và gửi dữ liệu tương ứng cho người dùng.
* Admin gửi yêu cầu quản lý hệ thống (quản lý người dùng, review, truyện, v.v.) đến hệ thống.
* Hệ thống xử lý yêu cầu quản lý và gửi thông tin phản hồi cho Admin.

### Sơ đồ DFD mức 1

**Sơ đồ DFD mức 1** mô tả hệ thống review truyện ở mức độ chi tiết hơn, chia nhỏ hệ thống thành các phân hệ chức năng chính.

### Mô tả:

#### Phân hệ chức năng:

* Quản lý người dùng: Xử lý các tác vụ liên quan đến người dùng, như tạo tài khoản, đăng nhập, cập nhật thông tin, v.v.
* Quản lý truyện: Xử lý các tác vụ liên quan đến truyện, như thêm truyện mới, chỉnh sửa thông tin truyện, xóa truyện, v.v.
* Quản lý review: Xử lý các tác vụ liên quan đến review, như thêm review mới, chỉnh sửa review, xóa review, v.v.
* Tìm kiếm truyện: Cho phép người dùng tìm kiếm truyện theo tiêu đề, tác giả, thể loại, v.v.
* Xem thông tin truyện: Hiển thị thông tin chi tiết về truyện, bao gồm tóm tắt nội dung, hình ảnh bìa, danh sách tác giả, thể loại, v.v.
* Đọc review: Hiển thị các review đã được viết về truyện.
* Viết review: Cho phép người dùng viết review về truyện.
* Đánh giá truyện: Cho phép người dùng đánh giá truyện bằng cách cho điểm từ 1 đến 5 sao.

#### Luồng dữ liệu:

* Người dùng gửi yêu cầu đến phân hệ chức năng tương ứng.
* Phân hệ chức năng xử lý yêu cầu và gửi dữ liệu tương ứng cho người dùng.

### Sơ đồ DFD mức 2 (Theo từng chức năng)

**Sơ đồ DFD mức 2** mô tả chi tiết từng phân hệ chức năng trong hệ thống review truyện.

### Mô tả:

#### Quá trình:

* Tạo tài khoản
* Đăng nhập
* Cập nhật thông tin
* Xóa tài khoản

#### Luồng dữ liệu:

* Người dùng gửi yêu cầu đến quá trình tương ứng.
* Quá trình xử lý yêu cầu và truy cập dữ liệu từ cơ sở dữ liệu.
* Cơ sở dữ liệu cung cấp dữ liệu cho quá trình.
* Quá trình xử lý dữ liệu và gửi kết quả cho người dùng.

### Diễn giải từng bước:

Để diễn giải từng bước chi tiết cho các phân hệ chức năng khác, bạn cần cung cấp thêm thông tin về chức năng cụ thể của từng phân hệ. Ví dụ:

#### Phân hệ Quản lý truyện:

* Chức năng: Thêm truyện mới, chỉnh sửa thông tin truyện, xóa truyện, v.v.
* Diễn giải từng bước:

##### Thêm review mới:

* Người dùng truy cập trang viết review.
* Người dùng nhập nội dung review, đánh giá, và chọn truyện muốn review.
* Người dùng nhấp vào nút "Gửi".
* Hệ thống kiểm tra tính hợp lệ của thông tin review.
* Nếu thông tin review hợp lệ:
  + Hệ thống lưu trữ thông tin review mới vào cơ sở dữ liệu.
  + Hệ thống hiển thị thông báo thành công cho người dùng.
* Nếu thông tin review không hợp lệ:
  + Hệ thống hiển thị thông báo lỗi cho người dùng và yêu cầu người dùng nhập lại thông tin.

##### Chỉnh sửa review:

* Người dùng truy cập trang danh sách review.
* Người dùng nhấp vào nút "Chỉnh sửa" bên cạnh review muốn chỉnh sửa.
* Hệ thống hiển thị trang chỉnh sửa review.
* Người dùng chỉnh sửa nội dung review, đánh giá, hoặc chọn truyện khác.
* Người dùng nhấp vào nút "Lưu".
* Hệ thống cập nhật thông tin review đã chỉnh sửa vào cơ sở dữ liệu.
* Hệ thống hiển thị thông báo thành công cho người dùng.

##### Xóa review:

* Người dùng truy cập trang danh sách review.
* Người dùng nhấp vào nút "Xóa" bên cạnh review muốn xóa.
* Hệ thống hiển thị thông báo xác nhận xóa review.
* Người dùng nhấp vào nút "Xóa" để xác nhận.
* Hệ thống xóa thông tin review khỏi cơ sở dữ liệu.
* Hệ thống hiển thị thông báo thành công cho người dùng.

#### Phân hệ Tìm kiếm truyện:

**Chức năng:** Cho phép người dùng tìm kiếm truyện theo tiêu đề, tác giả, thể loại, v.v.

**Diễn giải từng bước:**

##### Người dùng nhập thông tin tìm kiếm:

* Người dùng truy cập trang tìm kiếm truyện.
* Người dùng nhập thông tin tìm kiếm vào ô tìm kiếm, ví dụ như tiêu đề truyện, tác giả, thể loại, v.v.
* Người dùng nhấp vào nút "Tìm kiếm".

##### Hệ thống xử lý thông tin tìm kiếm:

* Hệ thống lấy thông tin tìm kiếm từ người dùng.
* Hệ thống truy vấn cơ sở dữ liệu để tìm kiếm các truyện phù hợp với thông tin tìm kiếm.

##### Hệ thống hiển thị kết quả tìm kiếm:

* Hệ thống hiển thị danh sách các truyện tìm được cho người dùng.
* Mỗi truyện trong danh sách hiển thị thông tin cơ bản như tiêu đề, tác giả, ảnh bìa, v.v.
* Người dùng có thể nhấp vào một truyện để xem thông tin chi tiết về truyện đó.

#### Phân hệ Xem thông tin truyện:

**Chức năng:** Hiển thị thông tin chi tiết về truyện, bao gồm tóm tắt nội dung, hình ảnh bìa, danh sách tác giả, thể loại, v.v.

**Diễn giải từng bước:**

##### Người dùng truy cập trang thông tin truyện:

* Người dùng truy cập trang danh sách truyện.
* Người dùng nhấp vào một truyện để xem thông tin chi tiết về truyện đó.

##### Hệ thống hiển thị thông tin truyện:

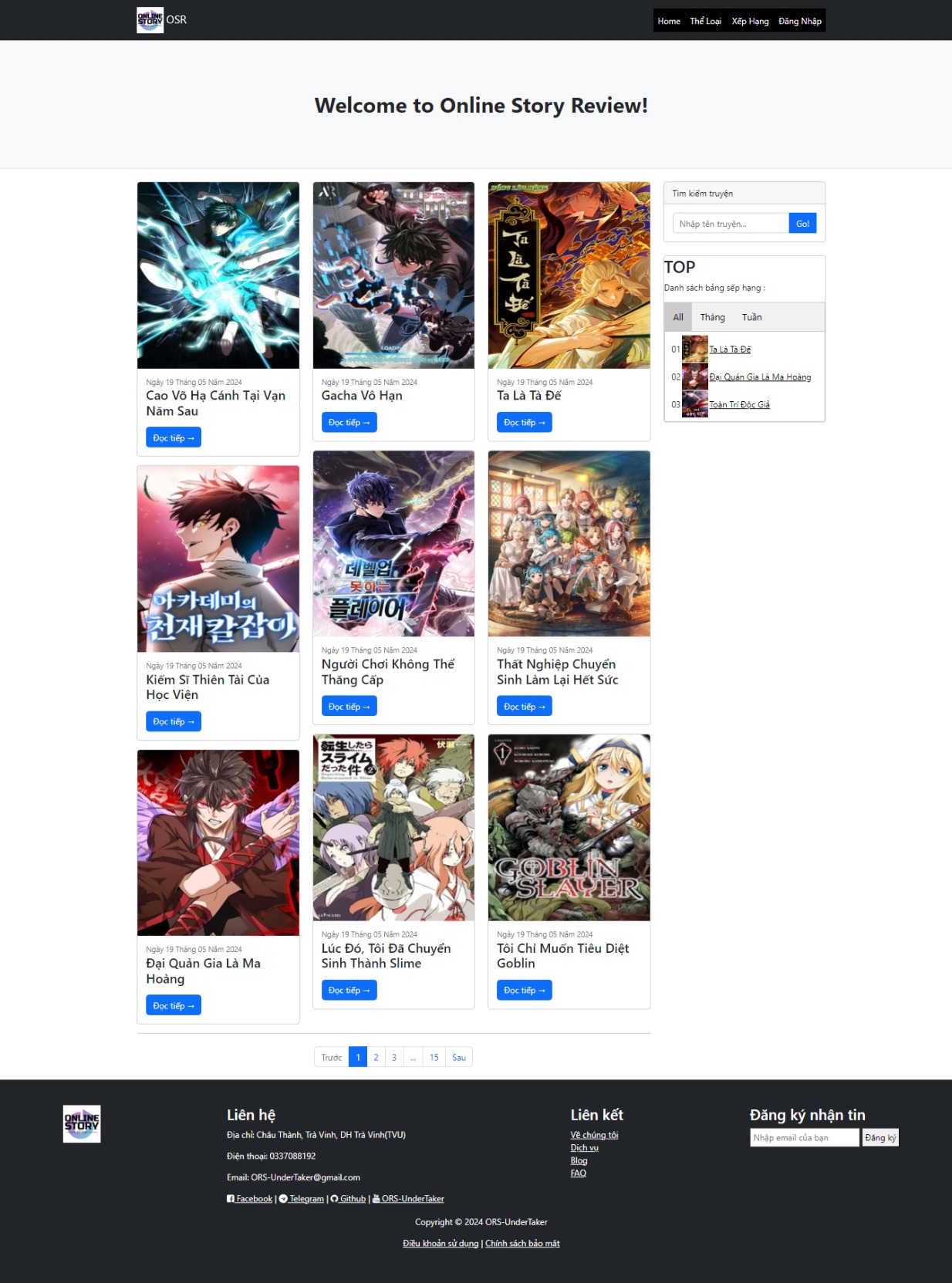
* + Hệ thống hiển thị thông tin chi tiết về truyện, bao gồm:
* Tiêu đề truyện
* Tác giả truyện
* Thể loại truyện
* Tóm tắt nội dung truyện
* Ảnh bìa truyện
* Danh sách review của truyện
* Đánh giá trung bình của truyện
  + Người dùng có thể đọc review của người khác về truyện.
  + Người dùng có thể viết review của riêng họ về truyện.
  + Người dùng có thể đánh giá truyện bằng cách cho điểm từ 1 đến 5 sao.

## Thiết kế giao diện

### Sơ đồ hệ thống

Hình 2.1. Sơ đồ hệ thống

### Giao diện trang chủ người dùng



Hình 2.2. Giao diện trang chủ của người dùng

Mô tả chi tiết về màn hình bao gồm các thành phần và cách bố cục

Trang web "Online Story Review" (OSR) được thiết kế với sự tinh tế và chú trọng đến trải nghiệm người dùng, nhằm mang đến một nền tảng đánh giá truyện trực tuyến hiệu quả và hấp dẫn. Dưới đây là mô tả chi tiết và toàn diện về các thành phần và bố cục của trang web:

#### Header (Phần đầu trang)

##### **Logo và Tên Trang Web**:

* **Logo OSR**: Nằm ở góc trên bên trái, logo được thiết kế đơn giản nhưng nổi bật, với chữ "OSR" là viết tắt của "Online Story Review". Logo có thể bao gồm biểu tượng hoặc hình ảnh liên quan đến sách hoặc truyện, tạo ấn tượng thị giác ngay lập tức.
* **Tên Trang Web**: Ngay bên cạnh logo, tên đầy đủ "Online Story Review!" được hiển thị bằng phông chữ lớn và đậm, dễ đọc.

##### **Thanh Menu Điều Hướng**:

* **Home**: Liên kết về trang chủ của trang web, giúp người dùng quay lại trang chính bất kỳ lúc nào.
* **Thể Loại**: Một trang liệt kê các thể loại truyện có sẵn, chẳng hạn như hành động, phiêu lưu, giả tưởng, lãng mạn, kinh dị, và nhiều thể loại khác. Người dùng có thể chọn thể loại mình yêu thích để tìm truyện.
* **Xếp Hạng**: Trang này hiển thị các bảng xếp hạng truyện theo nhiều tiêu chí khác nhau như lượt đọc, đánh giá của người dùng, hoặc độ phổ biến.
* **Đăng Nhập**: Liên kết đến trang hoặc cửa sổ đăng nhập, nơi người dùng có thể đăng nhập vào tài khoản của mình hoặc đăng ký tài khoản mới nếu chưa có.

#### Main Content (Nội dung chính)

##### **Tiêu Đề Chào Mừng**:

* **Tiêu Đề**: "Welcome to Online Story Review!" được đặt ngay trung tâm, dưới thanh menu. Tiêu đề này được viết bằng phông chữ lớn và dễ đọc, tạo cảm giác chào đón người dùng ngay khi truy cập vào trang web.

##### Danh Sách Truyện:

* **Bố Cục Dạng Lưới**: Các truyện được sắp xếp theo dạng lưới gồm 3 cột và nhiều hàng, mỗi hàng có 3 thẻ truyện, tổng cộng 9 truyện hiển thị trên một trang.
* **Thẻ Truyện**: Mỗi thẻ truyện bao gồm các thành phần sau:
* **Ảnh Bìa**: Ảnh minh họa lớn và rõ nét nằm ở phần trên của thẻ, giúp người dùng dễ nhận diện truyện. Ảnh bìa thường là một hình ảnh nghệ thuật hoặc cảnh đặc trưng của truyện.
* **Tiêu Đề Truyện**: Tiêu đề được đặt ngay dưới ảnh bìa, in đậm và dễ đọc, giúp người dùng nhanh chóng biết tên truyện.
* **Ngày Cập Nhật**: Thông tin về ngày cập nhật mới nhất của truyện, nằm ngay dưới tiêu đề. Thông tin này giúp người dùng biết truyện có được cập nhật thường xuyên hay không.
* **Nút "Đọc tiếp"**: Nút bấm màu xanh dương với chữ "Đọc tiếp", nằm ở phía dưới cùng của thẻ truyện. Khi người dùng nhấn vào nút này, họ sẽ được chuyển đến trang chi tiết của truyện, nơi có thêm thông tin và chương truyện.

#### Sidebar (Thanh bên phải)

##### **Tìm Kiếm Truyện**:

* **Hộp Tìm Kiếm**: Gồm một ô nhập văn bản để người dùng nhập tên truyện hoặc từ khóa cần tìm kiếm.
* **Nút "Go!"**: Nút màu xanh dương nằm ngay cạnh hộp tìm kiếm, khi bấm vào, trang web sẽ thực hiện tìm kiếm và hiển thị kết quả liên quan đến từ khóa nhập vào.

##### **Bảng Xếp Hạng TOP**:

* **Khung Xếp Hạng**: Một khung nhỏ hiển thị bảng xếp hạng các truyện phổ biến nhất. Bảng xếp hạng có thể thay đổi theo thời gian (ngày, tuần, tháng) hoặc theo các tiêu chí khác.
* **Danh Sách Truyện**: Mỗi truyện trong bảng xếp hạng bao gồm:
* **Thứ Tự**: Số thứ tự của truyện trong bảng xếp hạng.
* **Tên Truyện**: Tên truyện được hiển thị rõ ràng. Người dùng có thể nhấn vào tên truyện để chuyển đến trang chi tiết của truyện đó.

#### Footer (Chân trang)

##### Thông Tin Liên Hệ:

* + **Địa Chỉ**: Địa chỉ văn phòng hoặc trụ sở của trang web: Châu Thành, Tà Vinh, ĐH Trà Vinh (TVU).
  + **Điện Thoại**: Số điện thoại liên hệ: 0338708912.
  + **Email**: Địa chỉ email hỗ trợ: OSR-UnderTaker@gmail.com.

##### **Liên Kết Hữu Ích**:

* + **Về Chúng Tôi**: Trang giới thiệu về đội ngũ và sứ mệnh của trang web.
  + **Dịch Vụ**: Trang mô tả các dịch vụ mà OSR cung cấp cho người dùng.
  + **Blog**: Liên kết đến blog của trang web, nơi chứa các bài viết và tin tức liên quan đến truyện.
  + **FAQ**: Trang câu hỏi thường gặp giúp người dùng giải đáp các thắc mắc thông thường.

#### ****Đăng Ký Nhận Tin****:

* + **Ô Nhập Email**: Một ô để người dùng nhập địa chỉ email của họ.
  + **Nút Đăng Ký**: Nút để người dùng nhấn vào và đăng ký nhận bản tin từ trang web, cập nhật về các truyện mới và thông tin liên quan.

#### Các Thành Phần Khác

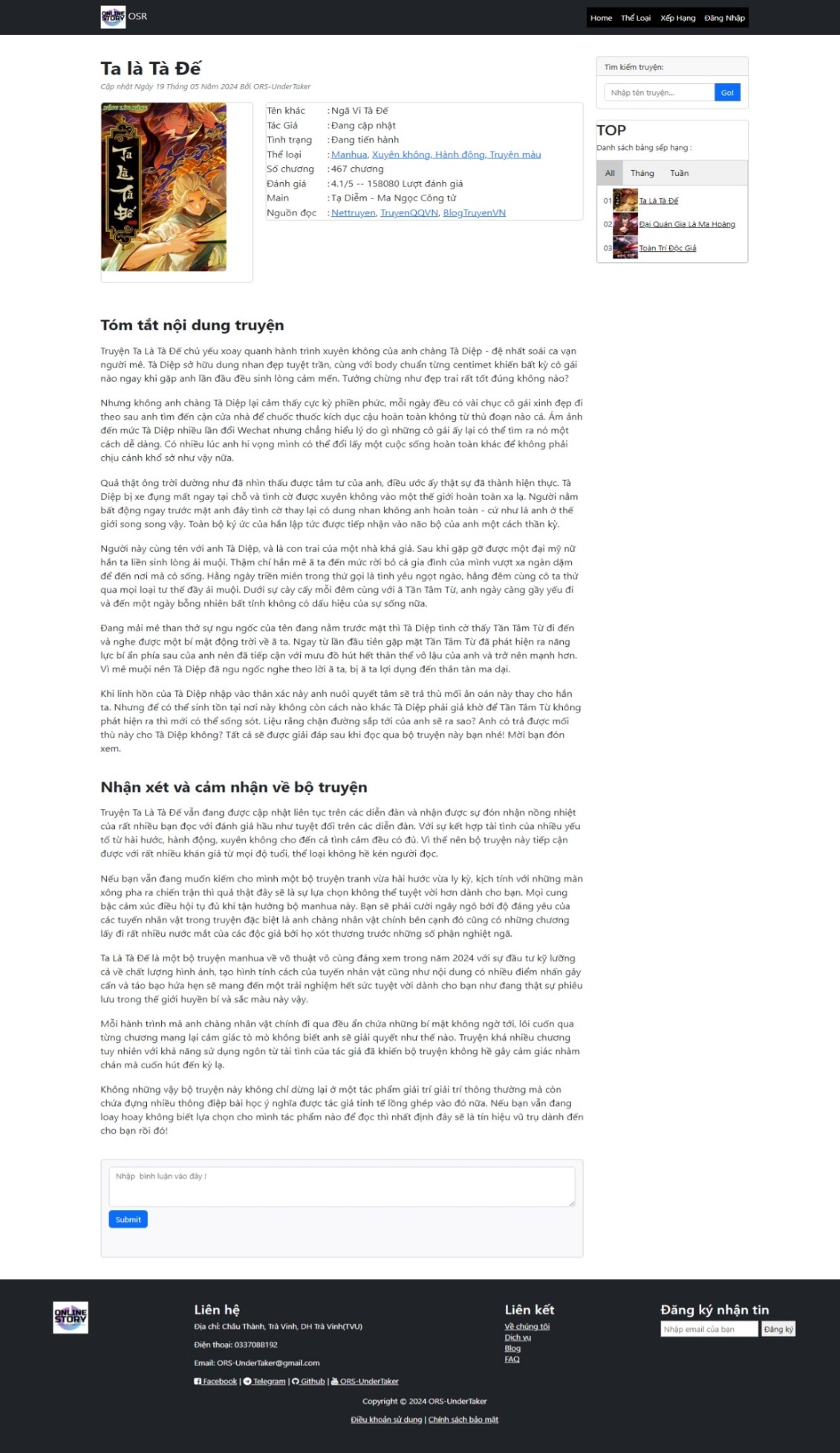
**Nút Điều Hướng Trang**: Dưới danh sách truyện, có các nút để người dùng chuyển đổi giữa các trang khác nhau:

* + **Trang Trước**: Nút để quay lại trang trước đó.
  + **Số Trang**: Các nút số trang cho phép người dùng chọn trang cụ thể để xem.
  + **Trang Sau**: Nút để chuyển đến trang tiếp theo.

#### Tổng Quan và Giao Diện

* + **Màu Sắc và Phông Chữ**: Trang web sử dụng màu sắc hài hòa, chủ yếu là màu trắng nền với các chi tiết màu đen và xanh dương. Các phông chữ được chọn lựa kỹ lưỡng để đảm bảo dễ đọc và tạo cảm giác thân thiện.
  + **Tính Thân Thiện Với Người Dùng**: Các thành phần trên trang được sắp xếp một cách logic và dễ tiếp cận, giúp người dùng dễ dàng tìm kiếm và đọc truyện. Thiết kế trang web cũng đảm bảo tính phản hồi tốt trên các thiết bị di động và máy tính để bàn.

### Giao diện trang con người dùng



Hình 2.3. Giao diện trang con của người dùng

#### ****Phần đầu trang:****

##### **Tiêu đề trang web**:

Ở phần đầu của trang web, phía trên cùng là logo của trang web với tên "OSR" và một thanh điều hướng đơn giản bao gồm các mục "Home", "Thể Loại", "Xếp Hạng", và "Đăng Nhập".

#### ****Phần thông tin truyện:****

##### **Tiêu đề truyện**:

Ngay dưới thanh điều hướng là tiêu đề "Ta Là Tà Đế" được in đậm, to và nổi bật, dễ dàng thu hút sự chú ý của người đọc.

##### **Thông tin cơ bản về truyện**:

Ngay bên dưới tiêu đề là một bảng thông tin chi tiết về truyện, bao gồm:

* + **Tên khác**: "Ngã Vì Tà Đế".
  + **Tác giả**: Đang cập nhật.
  + **Tình trạng**: Đang tiến hành.
  + **Thể loại**: Manhua, Xuyên không, Hành động, Truyện màu.
  + **Số chương**: 467 chương.
  + **Đánh giá**: 4.1/5 - 150080 lượt đánh giá.
  + **Nguồn**: Nettruyen, TruyenQQVN, BlogTruyenVN.

##### **Hình ảnh bìa truyện**:

Phía bên trái của bảng thông tin là hình ảnh bìa truyện. Hình ảnh này giúp người đọc dễ dàng nhận diện truyện, với hình ảnh màu sắc hấp dẫn và bắt mắt.

#### ****Phần nội dung chính:****

##### **Tóm tắt nội dung truyện**:

* Tiêu đề "Tóm tắt nội dung truyện" được in đậm và đặt ngay phía trên phần văn bản tóm tắt.
* Nội dung tóm tắt được viết chi tiết, kể về câu chuyện của nhân vật chính Tà Diệp, người sở hữu khả năng xuyên không. Tóm tắt nêu rõ các khó khăn và thách thức mà anh phải đối mặt trong hành trình của mình, cũng như sự phát triển của các nhân vật và tình tiết trong truyện.

#### ****Phần nhận xét và cảm nhận về bộ truyện:****

##### **Nhận xét và cảm nhận**:

* Tiêu đề "Nhận xét và cảm nhận về bộ truyện" cũng được in đậm để phân biệt với các phần khác.
* Phần này bao gồm các nhận xét chi tiết về truyện từ góc nhìn của người viết bài. Các nhận xét này phân tích các yếu tố như cốt truyện, nhân vật, hình ảnh và nghệ thuật của truyện. Người viết cũng đánh giá về cách xây dựng tình tiết, mạch truyện và cảm xúc mà truyện mang lại cho người đọc.

#### ****Phần bình luận của độc giả:****

**Bình luận**:

* + Ở phần dưới của trang, có một khung bình luận cho phép người đọc nhập nội dung bình luận của mình và gửi (Submit) để chia sẻ ý kiến cá nhân về truyện. Khung bình luận này giúp tạo sự tương tác giữa người đọc với nhau và với nội dung của trang web.

#### ****Thanh bên phải:****

##### **Thanh tìm kiếm**:

* Bên phải trang là thanh tìm kiếm, cho phép người đọc tìm kiếm các truyện khác một cách dễ dàng.

##### **Danh sách xếp hạng**:

* Dưới thanh tìm kiếm là danh sách các truyện được xếp hạng cao nhất, giúp người đọc dễ dàng tìm kiếm những truyện đang phổ biến và được yêu thích nhất hiện nay.

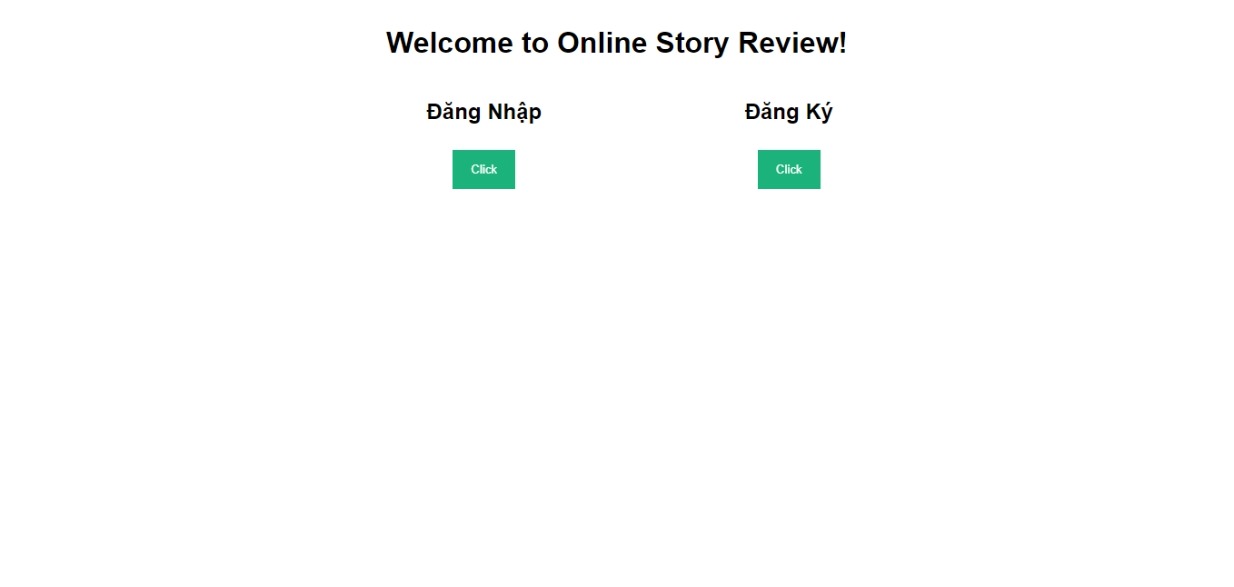
#### ****Phần chân trang:****

##### **Liên hệ và thông tin thêm**:

* Chân trang web cung cấp các thông tin liên hệ như địa chỉ, số điện thoại, email, và liên kết đến các trang mạng xã hội của trang web (Facebook, Telegram, GitHub).
* Có các liên kết đến các trang như "Về chúng tôi", "Blog", "FAQ" để người đọc có thể tìm hiểu thêm về trang web.
* Các điều khoản sử dụng và chính sách bảo mật cũng được liệt kê ở đây, đảm bảo sự minh bạch và đáng tin cậy của trang web.

Trang web có thiết kế đơn giản nhưng hiệu quả, giúp người đọc dễ dàng điều hướng và tìm kiếm thông tin. Mỗi phần đều được trình bày một cách rõ ràng và mạch lạc, với các tiêu đề in đậm giúp phân biệt giữa các mục nội dung khác nhau. Các yếu tố hình ảnh và văn bản được kết hợp một cách hài hòa, tạo ra một trải nghiệm đọc thân thiện và dễ chịu cho người sử dụng.

### Giao diện chức năng đăng ký và đăng nhập



Hình 2.4. Giao diện chức năng Đăng Ký và Đăng Nhập

#### Tiêu đề chính:

Dòng chữ "Welcome to Online Story Review!" được đặt ở trung tâm phía trên cùng của trang, với kích thước chữ lớn hơn và được in đậm để nổi bật.

#### Thông tin đăng nhập và đăng ký:

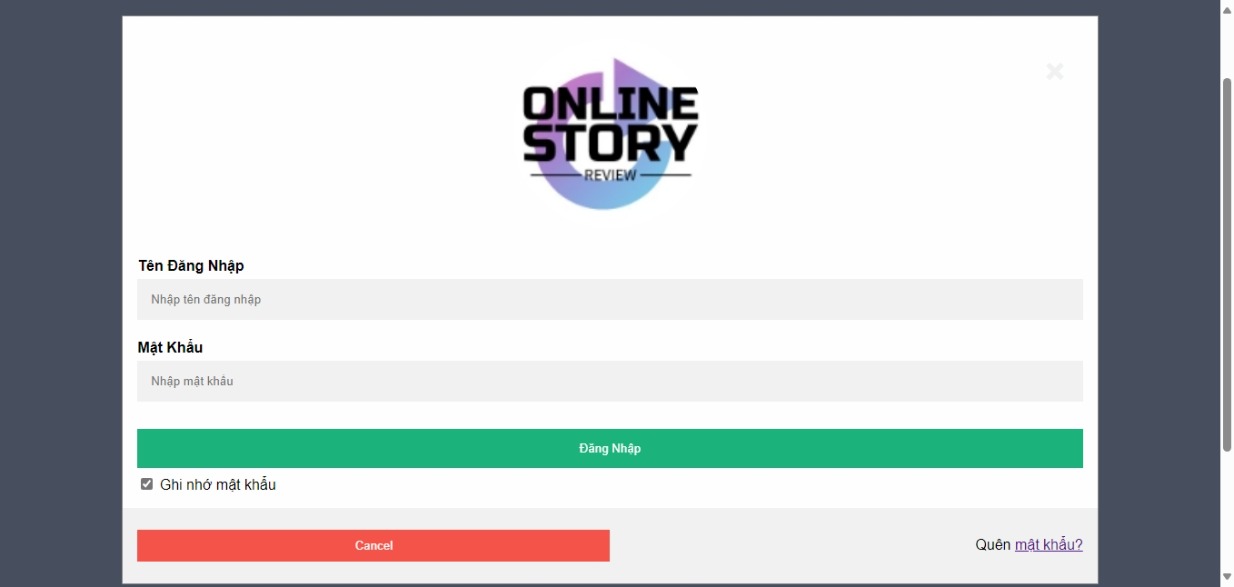
Dòng chữ "Đăng Nhập và Đăng Ký" nằm ngay dưới tiêu đề chính, cũng được đặt ở trung tâm nhưng với kích thước chữ nhỏ hơn.

##### Các nút chức năng:

Hai nút chức năng "Click" được đặt cạnh nhau, nằm ở giữa trang dưới dòng chữ "Đăng Nhập và Đăng Ký".

Cả hai nút đều có nền màu xanh lá cây và chữ trắng. Nút "Click" nằm bên trái nhấn vào sẽ hiện thị form “Đăng Nhập”và nút "Click" nằm bên phải nhấn vào sẽ hiện thị form “Đăng Ký”.

Bố cục của trang rất đơn giản và trực quan, với các thành phần chính được căn giữa để người dùng dễ dàng nhìn thấy và tương tác. Màu sắc của các nút giúp chúng nổi bật trên nền trắng của trang.



Hình 2.5. Giao diện chức năng Đăng Ký

Trang đăng nhập của trang web "Online Story Review" (OSR) được thiết kế đơn giản, trực quan và dễ sử dụng. Dưới đây là mô tả chi tiết về các thành phần và cách bố cục của màn hình đăng nhập:

#### Tổng Quan Bố Cục

* + **Nền Tối**: Phần nền của trang đăng nhập có màu tối (màu xám đậm hoặc xanh đậm) giúp làm nổi bật khung đăng nhập trắng ở trung tâm.
  + **Khung Đăng Nhập**: Khung chính của trang đăng nhập có màu trắng, được đặt ở trung tâm của màn hình, giúp người dùng tập trung vào các trường nhập liệu.

#### Thành Phần Chi Tiết

##### **Logo và Tên Trang Web**:

* + **Logo OSR**: Ở phần trên cùng của khung đăng nhập, logo "Online Story Review" được hiển thị rõ ràng. Logo này có hình dạng hình tròn với các màu sắc gradient từ tím đến xanh, tạo nên sự nhận diện thương hiệu mạnh mẽ.
  + **Tên Trang Web**: Ngay dưới logo, tên "Online Story Review!" được viết bằng phông chữ lớn và đậm, dễ đọc.

##### Trường Nhập Liệu**:**

1. **Tên Đăng Nhập**:
   * **Nhãn**: "Tên Đăng Nhập" được viết bằng chữ đậm, đặt phía trên ô nhập liệu.
   * **Ô Nhập Liệu**: Một ô nhập liệu màu trắng với chữ nhạt "Nhập tên đăng nhập" bên trong, hướng dẫn người dùng nhập tên đăng nhập của họ.
2. **Mật Khẩu**:
   * **Nhãn**: "Mật Khẩu" được viết bằng chữ đậm, đặt phía trên ô nhập liệu.
   * **Ô Nhập Liệu**: Một ô nhập liệu màu trắng với chữ nhạt "Nhập mật khẩu" bên trong, hướng dẫn người dùng nhập mật khẩu của họ.

##### **Tùy Chọn Ghi Nhớ**:

* + **Checkbox**: Một hộp kiểm (checkbox) nhỏ nằm ngay dưới trường mật khẩu.
  + **Nhãn**: "Ghi nhớ mật khẩu" nằm bên phải checkbox, cho phép người dùng chọn lưu thông tin đăng nhập trên trình duyệt của họ.

##### Nút Đăng Nhập:

* + **Nút Đăng Nhập**: Một nút lớn màu xanh lá cây với chữ "Đăng Nhập" màu trắng, đặt ngay dưới các trường nhập liệu. Nút này giúp người dùng gửi thông tin đăng nhập để truy cập vào tài khoản của họ.

##### **Nút Hủy (Cancel)**:

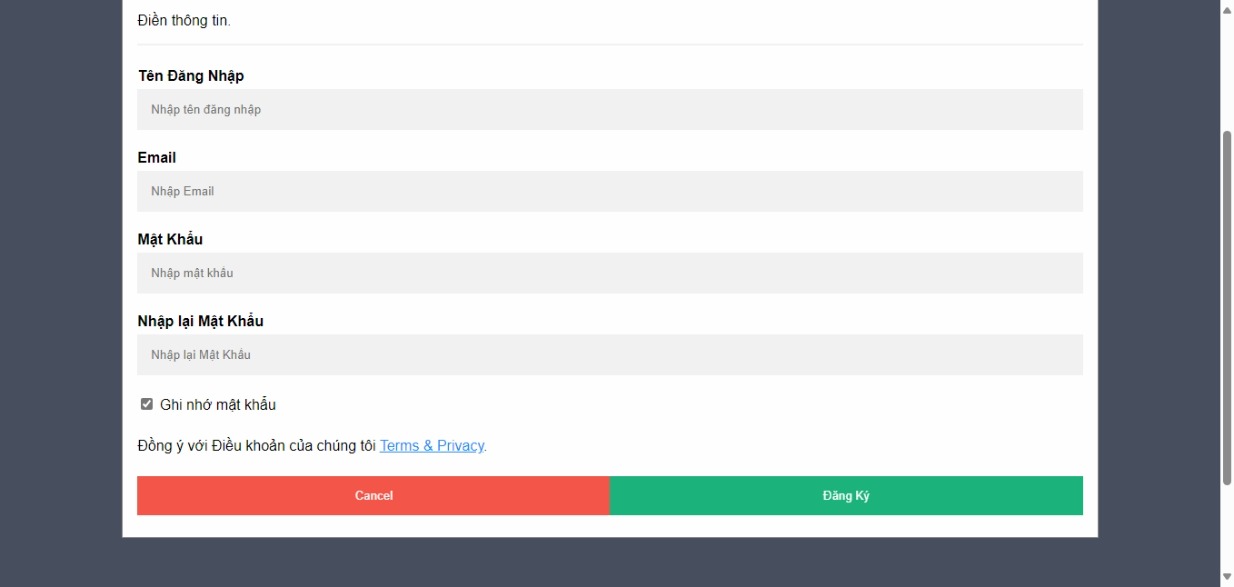
* + **Nút Hủy**: Một nút lớn màu đỏ với chữ "Cancel" màu trắng, nằm ngay dưới nút đăng nhập. Nút này cho phép người dùng hủy bỏ quá trình đăng nhập và quay lại trang trước đó.

##### **Liên Kết Quên Mật Khẩu**:

* + **Liên Kết**: Ở góc dưới bên phải của khung đăng nhập, có một liên kết nhỏ với dòng chữ "Quên mật khẩu?" màu xanh dương. Khi nhấn vào, người dùng sẽ được chuyển đến trang hoặc cửa sổ để khôi phục mật khẩu.

#### Tổng Quan Thiết Kế

* + **Màu Sắc**: Trang sử dụng màu sắc chủ đạo là màu trắng cho khung đăng nhập, màu xanh lá cây cho nút đăng nhập, và màu đỏ cho nút hủy, giúp tạo sự phân biệt rõ ràng giữa các thành phần. Liên kết "Quên mật khẩu?" có màu xanh dương, nổi bật trên nền trắng.
  + **Phông Chữ**: Sử dụng phông chữ đơn giản, dễ đọc, với các phần nhãn và nút bấm được viết bằng chữ đậm để dễ nhìn hơn.
  + **Giao Diện Thân Thiện**: Thiết kế đơn giản và trực quan, giúp người dùng dễ dàng thực hiện các thao tác đăng nhập mà không gặp khó khăn.



Hình 2.6. Giao diện chức năng Đăng Nhập

#### Tổng Quan Chung:

Màn hình này là giao diện đăng ký tài khoản trên một trang web. Mục đích của giao diện này là để thu thập thông tin từ người dùng cần thiết cho việc tạo tài khoản mới. Thiết kế hướng đến sự đơn giản, trực quan và dễ sử dụng để người dùng có thể hoàn thành quy trình đăng ký một cách thuận tiện.

#### Các Thành Phần Cụ Thể:

##### **Nền và Hộp Nội Dung**:

* + **Nền**: Toàn bộ màn hình có nền màu xám đậm, tạo ra một vùng nền tương phản cao so với hộp nội dung, giúp hộp nội dung nổi bật hơn.
  + **Hộp Nội Dung**: Nằm ở trung tâm màn hình, hình chữ nhật màu trắng, bao quanh bởi đường viền mỏng. Hộp này chứa tất cả các thành phần cần thiết cho việc đăng ký tài khoản.

##### **Tiêu Đề** :

* + **Tiêu Đề**: Nằm ở phần trên cùng của hộp nội dung, với văn bản "Điền thông tin" viết bằng chữ đậm màu đen. Cỡ chữ của tiêu đề lớn hơn các văn bản khác để thu hút sự chú ý của người dùng.

##### **Ô Nhập Liệu**:

Có bốn ô nhập liệu chính, được sắp xếp theo chiều dọc từ trên xuống dưới, mỗi ô bao gồm một nhãn và một ô văn bản để người dùng nhập thông tin.

1. **Tên Đăng Nhập**:
   * **Nhãn**: "Tên Đăng Nhập" nằm phía trên ô nhập liệu, màu đen.
   * **Ô Nhập Liệu**: Hộp văn bản màu trắng với dòng chữ mờ "Nhập tên đăng nhập". Người dùng sẽ nhập tên đăng nhập mà họ mong muốn vào đây.
2. **Email**:
   * **Nhãn**: "Email" nằm phía trên ô nhập liệu, màu đen.
   * **Ô Nhập Liệu**: Hộp văn bản màu trắng với dòng chữ mờ "Nhập Email". Người dùng sẽ nhập địa chỉ email của họ vào đây.
3. **Mật Khẩu**:
   * **Nhãn**: "Mật Khẩu" nằm phía trên ô nhập liệu, màu đen.
   * **Ô Nhập Liệu**: Hộp văn bản màu trắng với dòng chữ mờ "Nhập mật khẩu". Người dùng sẽ nhập mật khẩu mà họ mong muốn vào đây.
4. **Nhập lại Mật Khẩu**:
   * **Nhãn**: "Nhập lại Mật Khẩu" nằm phía trên ô nhập liệu, màu đen.
   * **Ô Nhập Liệu**: Hộp văn bản màu trắng với dòng chữ mờ "Nhập lại Mật Khẩu". Người dùng sẽ nhập lại mật khẩu họ đã nhập ở ô trên để xác nhận.

##### **Tùy Chọn Ghi Nhớ Mật Khẩu**:

* + **Checkbox**: Nằm dưới các ô nhập liệu. Người dùng có thể chọn hộp kiểm này để trình duyệt ghi nhớ mật khẩu của họ. Văn bản nhãn là "Ghi nhớ mật khẩu" nằm bên phải hộp kiểm.

##### **Điều Khoản và Điều Kiện**:

* + **Văn Bản**: "Đồng ý với Điều khoản của chúng tôi Terms & Privacy" nằm dưới hộp kiểm. "Terms & Privacy" là một liên kết màu xanh dẫn đến trang điều khoản và chính sách bảo mật của trang web, cho phép người dùng đọc chi tiết các điều kiện sử dụng và chính sách bảo mật trước khi đồng ý.

##### Nút Bấm:

* + **Vị Trí**: Hai nút bấm nằm dưới các ô nhập liệu và điều khoản, được đặt cạnh nhau theo chiều ngang.

1. **Nút Cancel**:
   * **Màu Sắc**: Màu đỏ tươi.
   * **Văn Bản**: "Cancel" viết bằng chữ trắng, căn giữa trên nền nút. Khi nhấn vào nút này, thao tác đăng ký sẽ bị hủy bỏ và người dùng có thể được đưa trở lại trang trước đó hoặc màn hình chính.
2. **Nút Đăng Ký**:

* **Màu Sắc**: Màu xanh lá cây.
* **Văn Bản**: "Đăng Ký" viết bằng chữ trắng, căn giữa trên nền nút. Khi người dùng nhấn vào nút này, thông tin sẽ được gửi đi để xử lý việc tạo tài khoản mới. Nếu thành công, người dùng có thể được chuyển đến một trang chào mừng hoặc trang đăng nhập.

#### Bố Cục và Sự Trực Quan:

* **Sự Đơn Giản**: Bố cục của màn hình được thiết kế đơn giản và trực quan. Mỗi thành phần đều có mục đích rõ ràng và được sắp xếp một cách hợp lý để người dùng dễ dàng hiểu và sử dụng.
* **Tương Phản Màu Sắc**: Nền xám đậm giúp hộp nội dung màu trắng nổi bật. Các màu đỏ và xanh lá cây của các nút bấm không chỉ tạo sự tương phản mạnh mẽ mà còn giúp người dùng dễ nhận biết và phân biệt các hành động.
* **Căn Giữa**: Tất cả các thành phần được căn giữa theo chiều dọc trong hộp nội dung, tạo sự cân đối và dễ nhìn.

#### Trải Nghiệm Người Dùng:

* **Dễ Hiểu**: Các nhãn và văn bản hướng dẫn đều ngắn gọn, rõ ràng và trực tiếp, giúp người dùng nhanh chóng hiểu được các thông tin cần nhập.
* **Dễ Sử Dụng**: Các ô nhập liệu lớn và rõ ràng, cùng với các nút bấm lớn và dễ nhấn, giúp người dùng thực hiện các thao tác một cách dễ dàng và nhanh chóng. Bố cục đơn giản và trực quan cũng giúp giảm thiểu nhầm lẫn và lỗi khi nhập thông tin.



Hình 2.7. Giao diện chức năng Quên mật khẩu

#### Tổng Quan Chung:

Màn hình này là một giao diện người dùng của một trang web cho phép người dùng đặt lại mật khẩu của họ trong trường hợp quên mật khẩu. Giao diện này được thiết kế tối giản, dễ sử dụng và trực quan.

#### Các Thành Phần Cụ Thể:

##### **Nền và Hộp Nội Dung**:

* **Nền**: Màn hình có nền màu xám đậm, giúp làm nổi bật hộp nội dung ở trung tâm.
* **Hộp Nội Dung**: Ở giữa màn hình là một hộp hình chữ nhật màu trắng, nơi chứa tất cả các thành phần cần thiết cho việc khôi phục mật khẩu.

##### **Tiêu Đề**:

* **Vị Trí**: Nằm ở phần trên cùng của hộp nội dung.
* **Văn Bản**: "Quên Mật Khẩu" được viết bằng chữ in đậm và cỡ chữ lớn để thu hút sự chú ý của người dùng. Màu chữ là màu đen, tạo sự tương phản rõ ràng với nền trắng của hộp nội dung.

##### **Hướng Dẫn Người Dùng**:

* **Vị Trí**: Nằm ngay dưới tiêu đề.
* **Văn Bản**: "Vui lòng nhập địa chỉ email của bạn để nhận hướng dẫn đặt lại mật khẩu." Văn bản này được viết bằng chữ thường, màu đen, và cỡ chữ nhỏ hơn tiêu đề. Hướng dẫn này ngắn gọn và trực tiếp, giúp người dùng hiểu ngay họ cần làm gì.

##### **Ô Nhập Liệu**:

* **Vị Trí**: Nằm ngay dưới đoạn hướng dẫn.
* **Nhãn**: Bên trái ô nhập liệu có nhãn "Email" để chỉ rõ loại thông tin cần nhập.
* **Ô Nhập Liệu**: Là một hộp văn bản màu trắng với viền mỏng. Bên trong ô có dòng chữ mờ "Nhập Email" để gợi ý người dùng nhập địa chỉ email của họ vào đây.

##### **Nút Bấm**:

**Vị Trí**: Nằm dưới ô nhập liệu và được đặt cạnh nhau theo chiều ngang.

1. **Nút Cancel**:
   * **Màu Sắc**: Màu đỏ.
   * **Văn Bản**: "Cancel" viết bằng chữ trắng.
   * **Chức Năng**: Hủy bỏ thao tác và có thể quay lại trang trước hoặc đóng hộp nội dung.
2. **Nút Gửi**:

* **Màu Sắc**: Màu xanh lá cây.
* **Văn Bản**: "Gửi" viết bằng chữ trắng.
* **Chức Năng**: Gửi email đã nhập để nhận hướng dẫn đặt lại mật khẩu. Khi người dùng nhấn vào nút này, một yêu cầu sẽ được gửi đến máy chủ để xử lý việc khôi phục mật khẩu.

1. **Nút Đóng**:

* **Vị Trí**: Nằm ở góc trên cùng bên phải của hộp nội dung.
* **Hình Dạng**: Là một dấu "X" màu trắng, cho phép người dùng đóng hộp nội dung và quay lại màn hình trước đó.

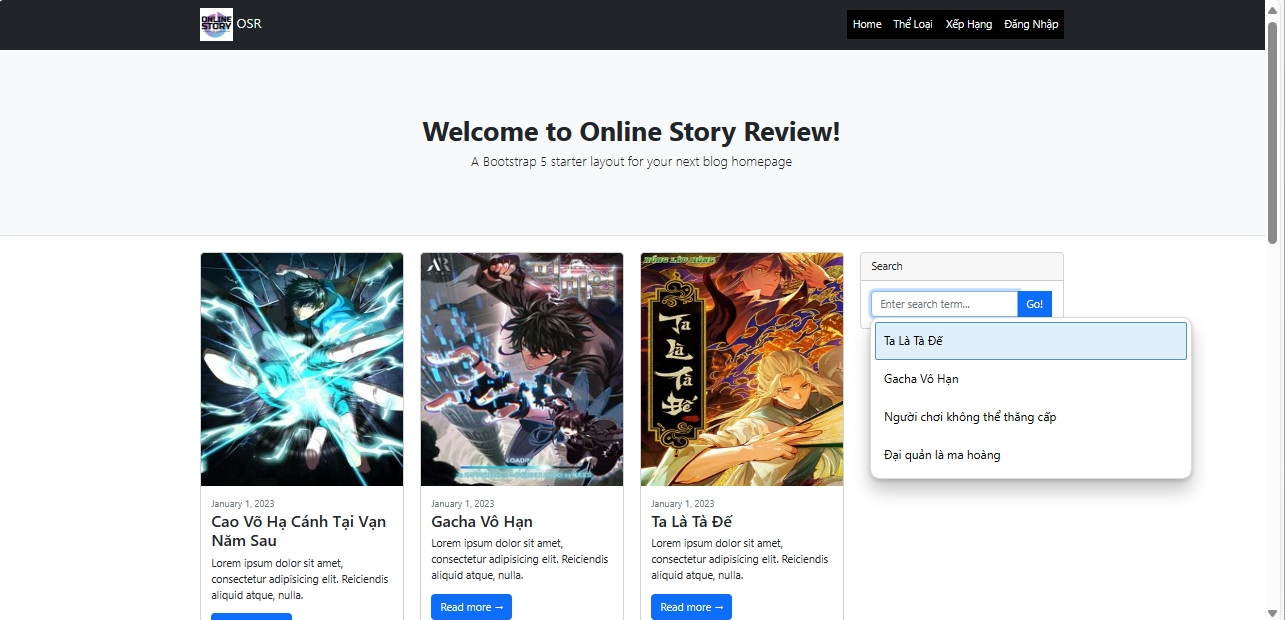
#### Bố Cục và Sự Trực Quan:

* **Sự Đơn Giản**: Bố cục của màn hình rất đơn giản và rõ ràng, giúp người dùng dễ dàng thực hiện các bước cần thiết mà không bị rối mắt.
* **Tương Phản Màu Sắc**: Sự tương phản giữa nền xám đậm và hộp nội dung màu trắng giúp tập trung sự chú ý vào các thông tin quan trọng. Màu sắc của các nút (đỏ và xanh lá cây) không chỉ giúp phân biệt chức năng mà còn tăng cường tính trực quan cho người dùng.
* **Căn Giữa**: Tất cả các thành phần được căn giữa theo chiều dọc, giúp giao diện trông cân đối và dễ nhìn.

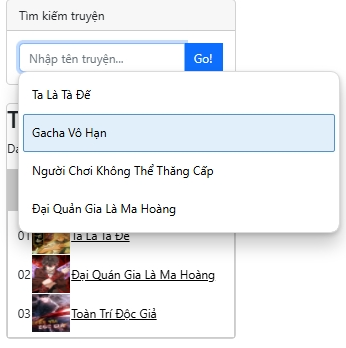
#### Trải Nghiệm Người Dùng:

* **Dễ Hiểu**: Các thành phần và văn bản hướng dẫn được trình bày một cách rõ ràng và trực tiếp, giúp người dùng nhanh chóng hiểu được các bước cần thực hiện.
* **Dễ Sử Dụng**: Bố cục hợp lý và các nút bấm lớn, dễ nhấn, giúp người dùng thực hiện thao tác một cách dễ dàng và nhanh chóng.

### Giao diện chức năng tìm kiếm



Hình 2.8. Giao diện chức năng tìm kiếm



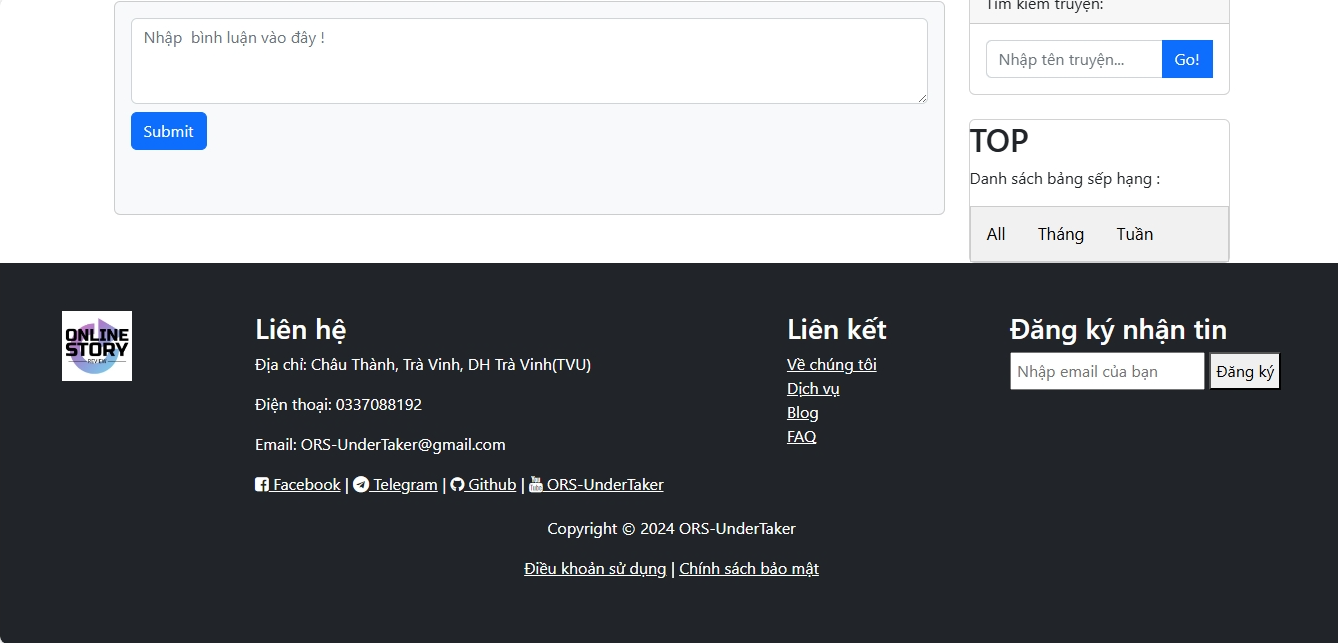
Hình 2.9. Giao diện chức năng tìm kiếm

#### Sidebar (Thanh bên phải)

##### **Tìm Kiếm Truyện**:

* **Hộp Tìm Kiếm**: Gồm một ô nhập văn bản để người dùng nhập tên truyện hoặc từ khóa cần tìm kiếm.
* **Nút "Go!"**: Nút màu xanh dương nằm ngay cạnh hộp tìm kiếm, khi bấm vào, trang web sẽ thực hiện tìm kiếm và hiển thị kết quả liên quan đến từ khóa nhập vào.
* Thanh tìm kiếm: Ở phía trên, có một thanh tìm kiếm với chữ “Nhập tiên truyện…” và nút “Go!” bên phải.
* Các tab: Dưới thanh tìm kiếm, có hai tab: “Tả Là Tả Đế” và “Gacha Vô Hạn”. Tab thứ hai (“Gacha Vô Hạn”) có vẻ được chọn.

### Giao diện chức năng bình luận



Hình 2.10. Giao diện chức năng bình luận

#### ****Phần bình luận của độc giả:****

**Bình luận**:

* + Ở phần dưới của trang, có một khung bình luận cho phép người đọc nhập nội dung bình luận của mình và gửi (Submit) để chia sẻ ý kiến cá nhân về truyện. Khung bình luận này giúp tạo sự tương tác giữa người đọc với nhau và với nội dung của trang web.

### Giao diện trang chủ quản trị

#### Giao diện chức năng …

#### Giao diện chức năng …

#### Giao diện chức năng …

### Kết chương

#### Tầm quan trọng của thiết kế giao diện:

* Giao diện đóng vai trò trung gian kết nối người dùng với hệ thống.
* Giao diện tốt giúp người dùng dễ dàng sử dụng, hiểu thông tin và hoàn thành mục tiêu.
* Giao diện kém có thể gây khó khăn cho người dùng, dẫn đến thất vọng và bỏ dùng hệ thống.

#### Các yếu tố ảnh hưởng đến thiết kế giao diện:

* **Mục tiêu của hệ thống:** Xác định mục đích sử dụng của hệ thống để thiết kế giao diện phù hợp.
* **Đối tượng người dùng:** Hiểu rõ nhu cầu, hành vi và đặc điểm của người dùng để đáp ứng trải nghiệm phù hợp.
* **Nội dung:** Sắp xếp và tổ chức nội dung hợp lý để người dùng dễ dàng truy cập và sử dụng.
* **Chức năng:** Thiết kế các chức năng cần thiết cho hệ thống một cách trực quan và dễ hiểu.
* **Giao diện trực quan:** Sử dụng các yếu tố như màu sắc, bố cục, hình ảnh, phông chữ để tạo giao diện đẹp mắt và thu hút.
* **Khả năng tương tác:** Thiết kế giao diện cho phép người dùng tương tác với hệ thống một cách dễ dàng và hiệu quả.
* **Khả năng tiếp cận:** Đảm bảo giao diện có thể truy cập được bởi mọi người dùng, bao gồm cả người khuyết tật.

#### Quy trình thiết kế giao diện:

* **Phân tích yêu cầu:** Xác định mục tiêu, đối tượng người dùng, nội dung, chức năng cần thiết cho hệ thống.
* **Lập kế hoạch:** Phác thảo các ý tưởng thiết kế, lựa chọn công cụ thiết kế phù hợp.
* **Thiết kế:** Tạo ra các mẫu giao diện, bao gồm bản vẽ giao diện, nguyên mẫu tương tác.
* **Đánh giá:** Kiểm tra và thu thập phản hồi từ người dùng để cải thiện thiết kế.
* **Thực thi:** Chuyển đổi thiết kế thành giao diện thực tế của hệ thống.
* **Bảo trì:** Theo dõi và cập nhật giao diện khi cần thiết.

#### Kết luận:

Thiết kế giao diện là một quá trình quan trọng trong việc phát triển hệ thống phần mềm. Bằng cách áp dụng các nguyên tắc và quy trình thiết kế hiệu quả, có thể tạo ra giao diện người dùng thân thiện, dễ sử dụng và đáp ứng nhu cầu của người dùng.

Ngoài ra, chương này cũng cung cấp một số tài nguyên hữu ích cho việc học thêm về thiết kế giao diện, bao gồm sách, bài báo, trang web và công cụ thiết kế.

Thiết kế giao diện là một lĩnh vực quan trọng trong việc phát triển sản phẩm phần mềm và web thành công. Việc tạo ra giao diện người dùng hiệu quả có thể giúp nâng cao trải nghiệm người dùng, tăng tỷ lệ chuyển đổi và thúc đẩy doanh nghiệp. Bằng cách áp dụng các nguyên tắc và thực hành thiết kế UI được trình bày trong chương này, các nhà thiết kế có thể tạo ra giao diện hấp dẫn và dễ sử dụng, thu hút người dùng và đạt được mục tiêu kinh doanh.

# Cài đặt thực nghiệm

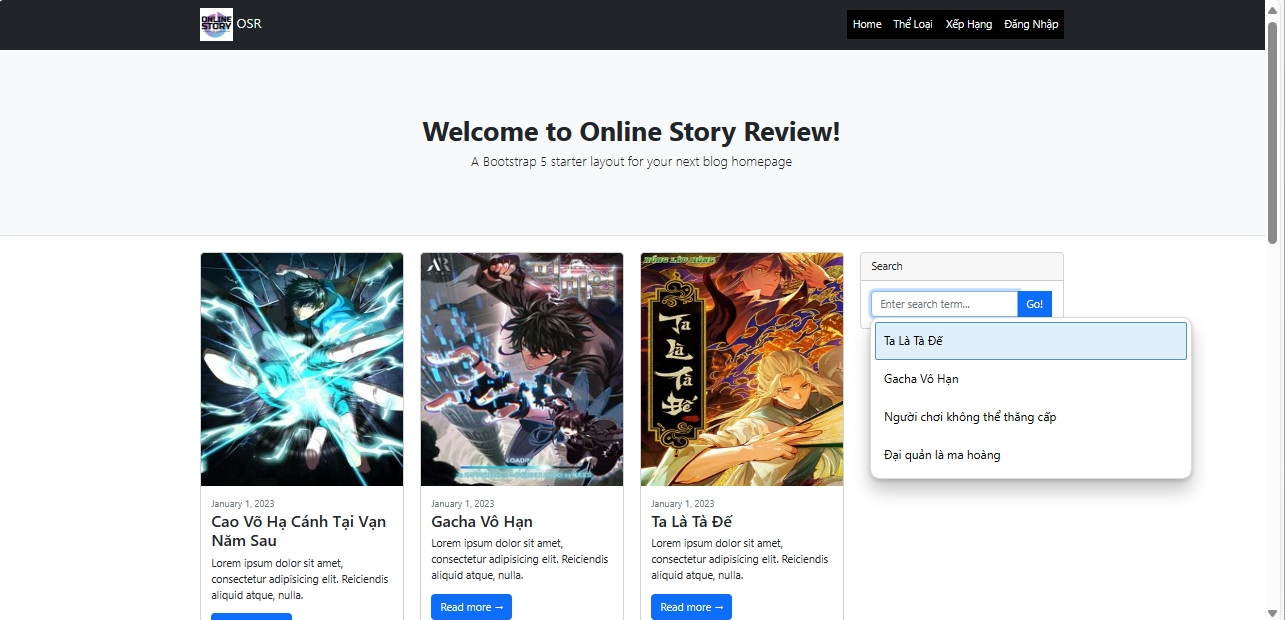
## Bộ dữ liệu thử nghiệm

Thử nghiệm chức năng tìm kiếm với tên của một truyện như: “Ta Là Tà Đế’, “Gacha Vô Hạn”.

Thử nghiệm chức năng bình luận bằng cách nhập bình luân: “review rất hay”, “review truyện đủ ý”

## Kết quả thực nghiệm

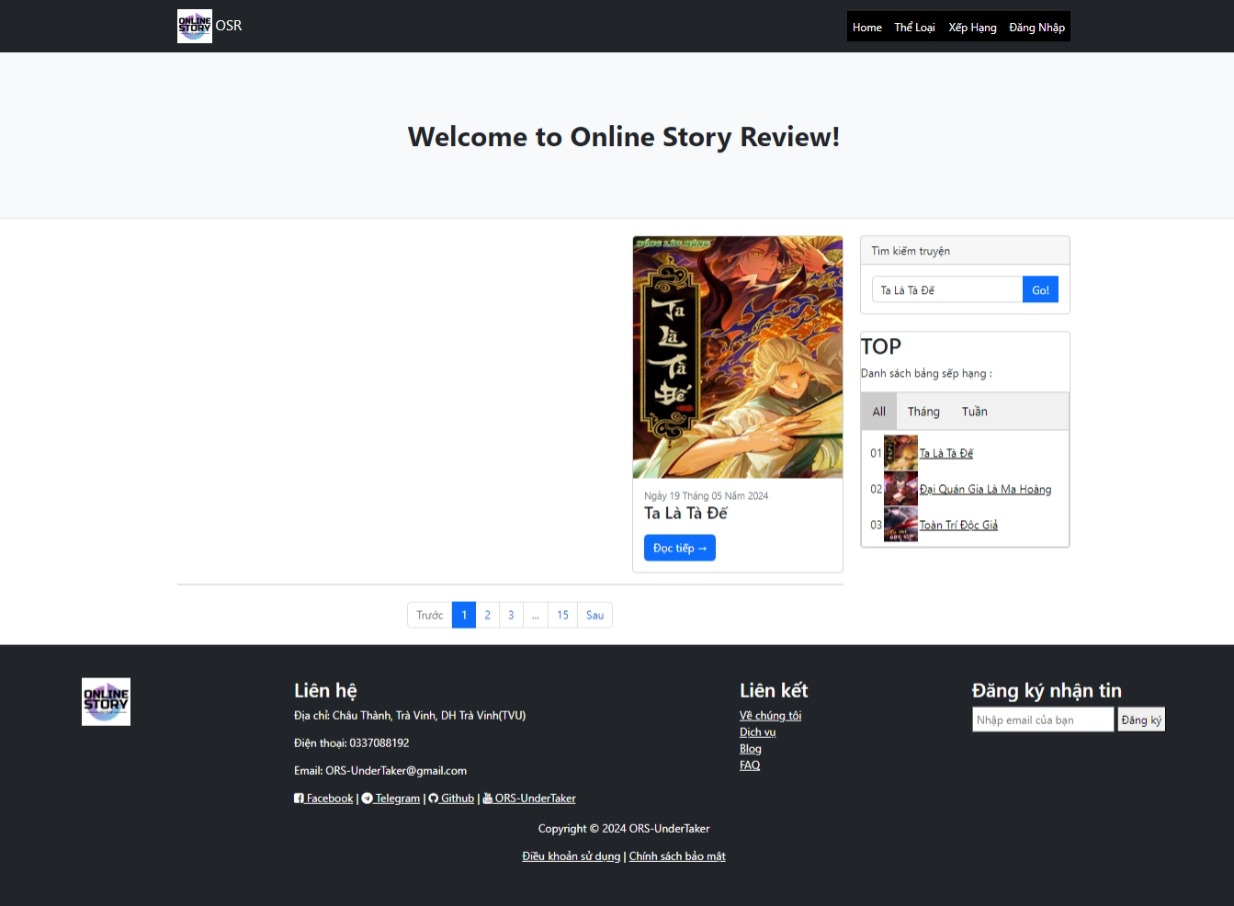
### Chức năng gợi ý tìm kiếm



Hình 3.1. Chức năng gợi ý tìm kiếm

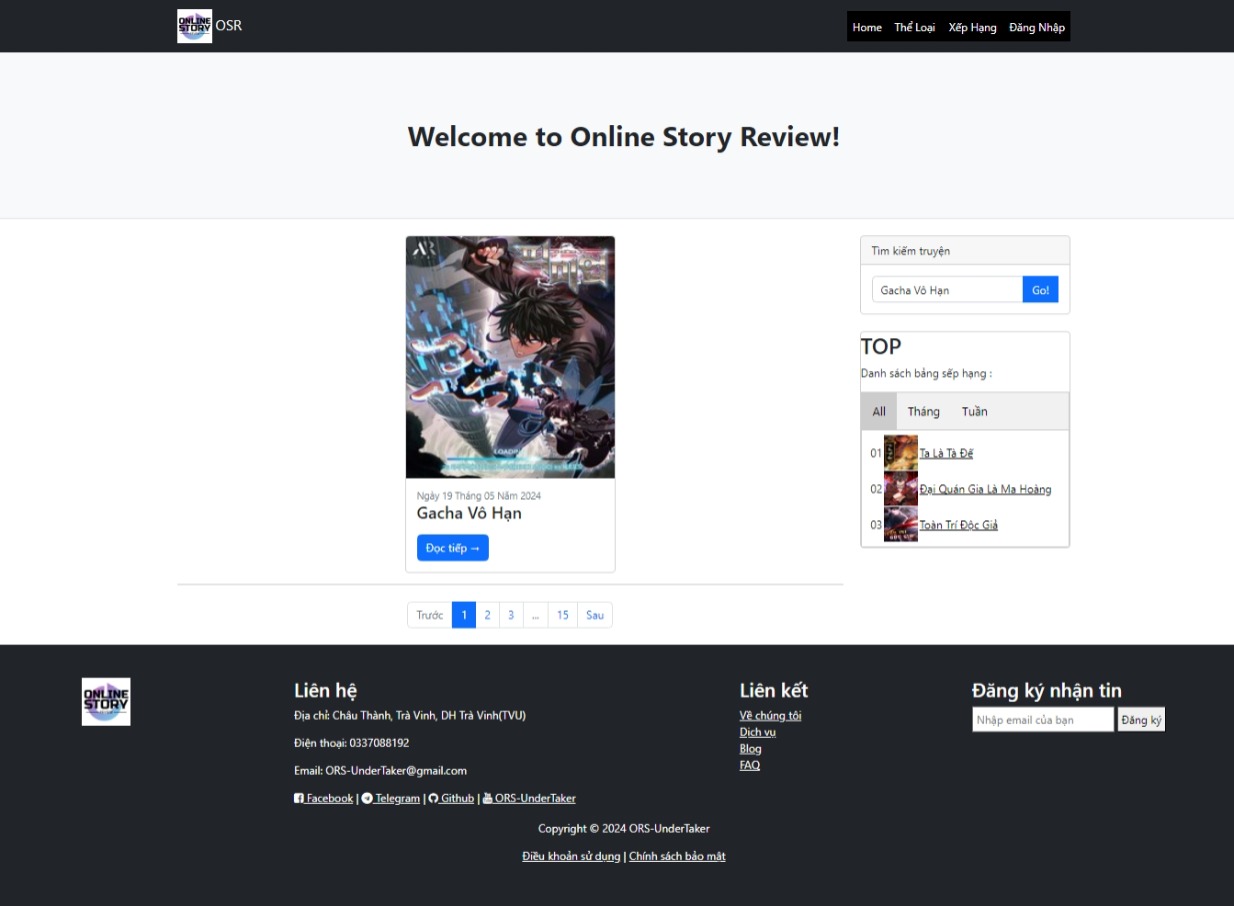
Khi nhấn con trỏ chuột vào thanh tìm kiếm chúng ta sẽ thấy được các gợi ý tên của tên của các bộ truyện dưới thanh tìm kiếm

### Chức năng tìm kiếm



Hình 3.2. Chức năng tìm kiếm

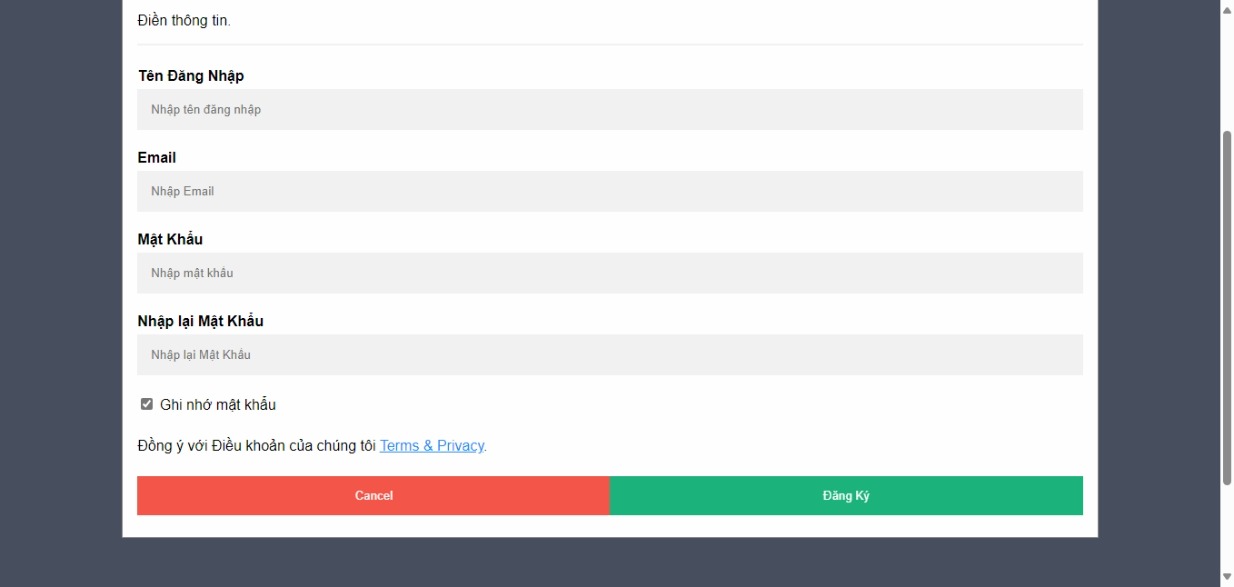
Khi nhập tên truyện là: “Ta Là Tà Đế” hệ thống sẽ xuất hiện tên truyện như đã nhập ở trên.



Hình 3.3. Chức năng tìm kiếm

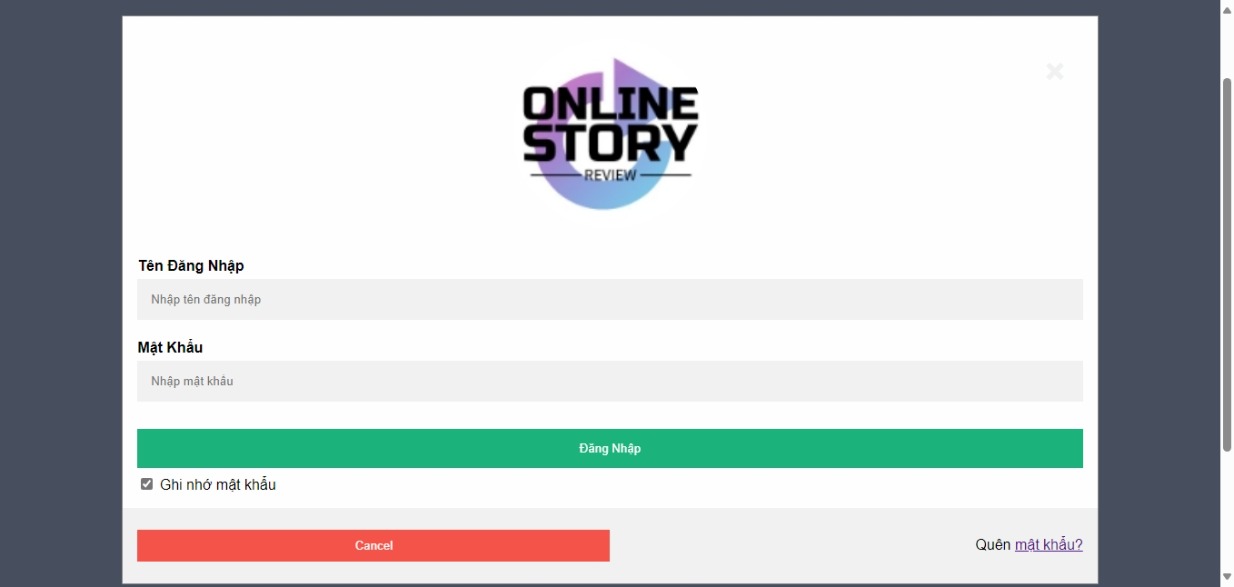
Tương tự như khi nhập tên truyện là: “Gacha Vô Hạn” thì hệ thống sẽ hiện bộ nhập ở trên ra.

### Chức năng đăng ký



Hình 3.4. Chức năng đăng ký

### Chức năng đăng nhập



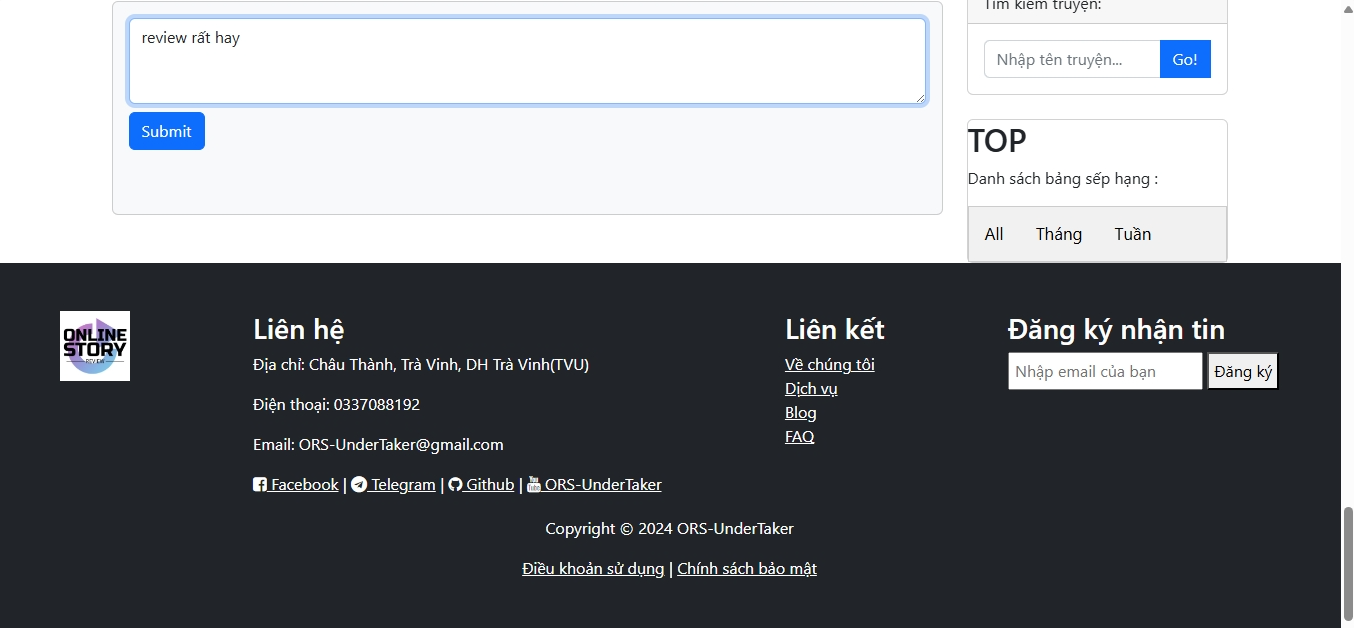
Hình 3.5. Chức năng đăng nhập

### Chức năng quên mật khẩu



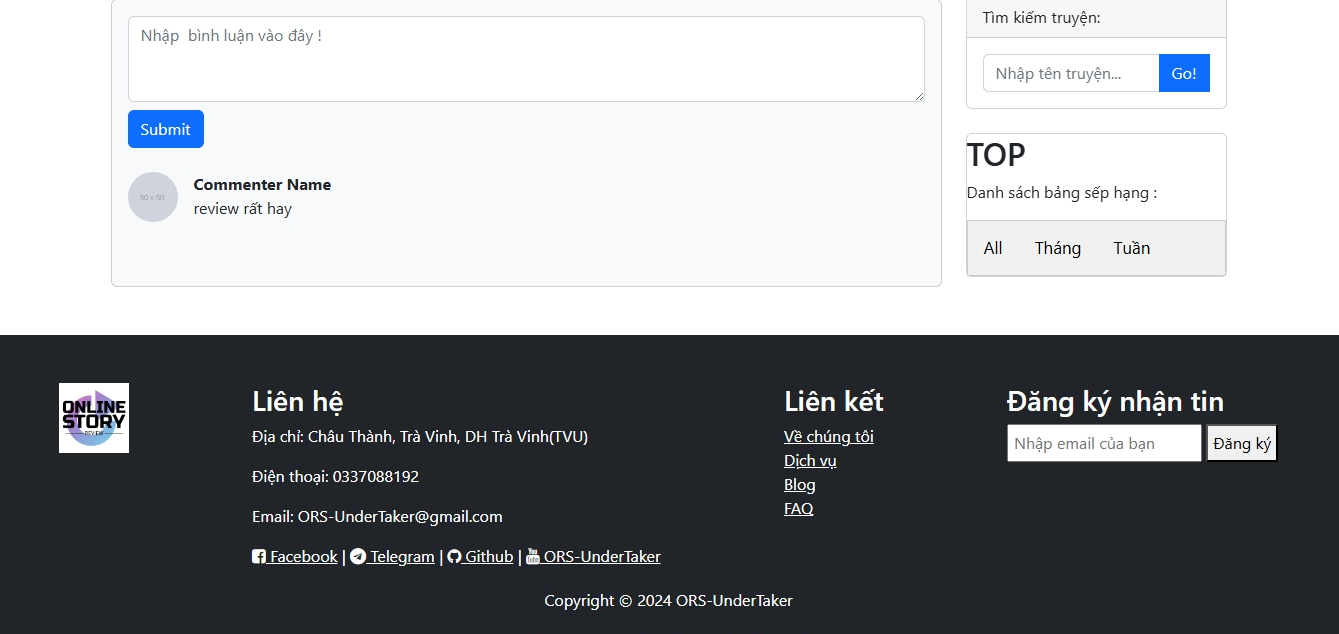
Hình 3.6. Chức năng quên mật khẩu

### Chức năng bình luận

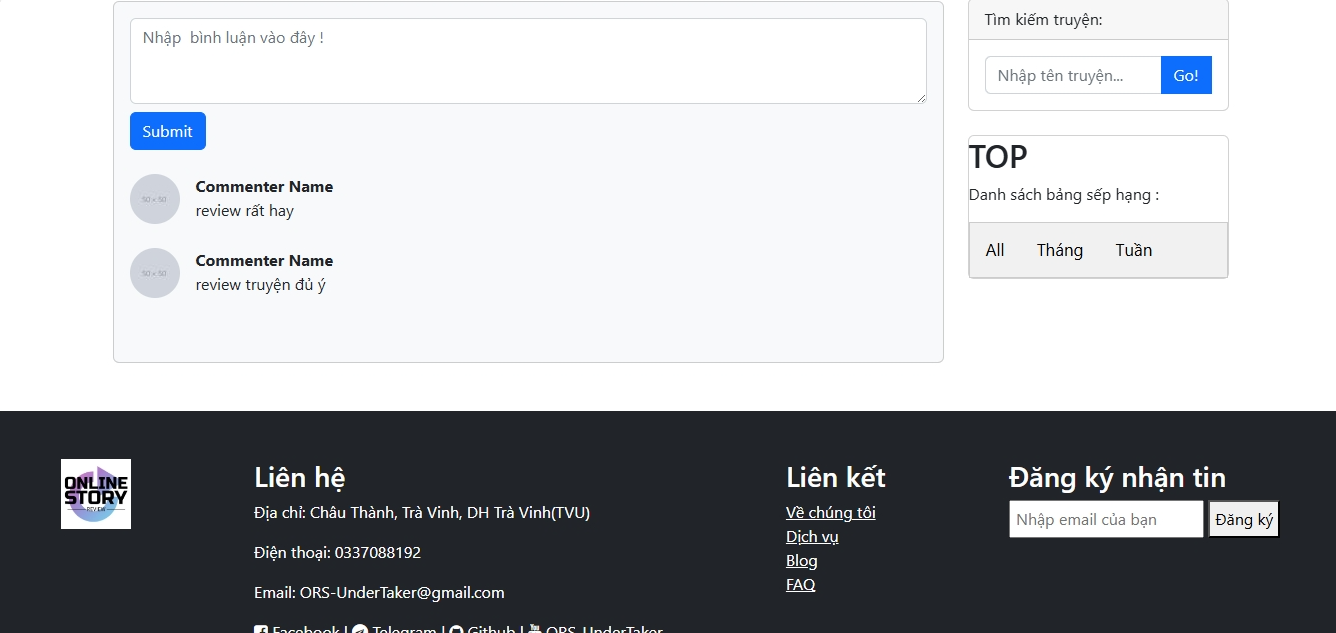


Hình 3.7. Chức năng bình luận

Khi nhập bình luận “review rất hay” thì hệ thốn sẽ hiện ở dưới thanh bình luận vừa nhập vào như hình bên dưới



Hình 3.8. Chức năng bình luận



Hình 3.9. Chức năng bình luận

Khi nhập thêm một bình luận “review truyện đủ ý” thì bình luận đó sẽ xuất hiện ở bên dưới bình luận trước đó như trong hình trên

## Kết chương

Chương trình cài đặt thực nghiệm Website Review Truyện Online đã được thực hiện thành công với nhiều tính năng nổi bật nhằm đáp ứng nhu cầu của người dùng yêu thích truyện. Hệ thống không chỉ cung cấp giao diện thân thiện và dễ sử dụng mà còn tích hợp các tính năng quan trọng như đăng ký, đăng nhập, tìm kiếm truyện, đọc truyện trực tuyến, và đánh giá truyện.

### Tóm tắt kết quả

* **Giao diện người dùng (UI)**: Thiết kế trực quan, dễ sử dụng, tương thích với nhiều thiết bị từ máy tính đến điện thoại di động.
* **Chức năng người dùng**:
  + Đăng ký và đăng nhập tài khoản cá nhân.
  + Tìm kiếm truyện theo tên, tác giả, thể loại, và đánh giá.
  + Đọc truyện trực tuyến với trải nghiệm mượt mà.
  + Đánh giá và bình luận truyện, giúp người dùng khác có cái nhìn tổng quan về truyện trước khi đọc.
* **Quản lý truyện**: Hệ thống quản trị viên dễ dàng quản lý dữ liệu truyện, bao gồm thêm mới, chỉnh sửa, xóa và cập nhật thông tin truyện.

### Kết quả thử nghiệm

Quá trình thử nghiệm cho thấy hệ thống hoạt động ổn định, đáp ứng tốt các yêu cầu đặt ra. Các tính năng chính được kiểm thử kỹ lưỡng và đảm bảo hoạt động chính xác. Tốc độ tải trang nhanh, trải nghiệm người dùng mượt mà và không gặp lỗi nghiêm trọng nào.

Website Review Truyện Online đã và đang trở thành một nền tảng hữu ích cho cộng đồng yêu thích truyện. Việc cài đặt và thử nghiệm thành công hệ thống này là bước đệm quan trọng để tiếp tục phát triển và nâng cao chất lượng dịch vụ. Những định hướng phát triển trong tương lai sẽ giúp website ngày càng hoàn thiện và thu hút nhiều người dùng hơn

# Kết luận và hướng phát triển

## Kết luận

Việc xây dựng một website review truyện online hiệu quả đòi hỏi sự kết hợp nhuần nhuyễn giữa các yếu tố kỹ thuật, thẩm mỹ và chiến lược kinh doanh. Bài viết này đã cung cấp một lộ trình chi tiết để thiết kế và phát triển website, bao gồm:

### ****Phân tích thị trường:****

Hiểu rõ thị hiếu của người đọc truyện, xu hướng thị trường và các đối thủ cạnh tranh để định vị website phù hợp.

### ****Xác định USP (Unique Selling Proposition):****

Điểm độc đáo và khác biệt của website so với các website review truyện khác để thu hút người dùng.

### ****Thiết kế giao diện:****

Giao diện cần đẹp mắt, thân thiện, dễ sử dụng và phù hợp với đối tượng mục tiêu. Nên tối ưu hóa website cho các thiết bị di động để tiếp cận người dùng mọi lúc mọi nơi.

### ****Lựa chọn công nghệ phù hợp:****

Sử dụng các ngôn ngữ lập trình web, framework và công nghệ cơ sở dữ liệu phù hợp để đảm bảo website hoạt động ổn định, hiệu quả và có thể mở rộng trong tương lai.

### ****Chú trọng trải nghiệm người dùng:****

Website cần cung cấp trải nghiệm mượt mà, dễ dàng tìm kiếm và truy cập thông tin, tương tác với website và thực hiện các hành động mong muốn.

### ****Đảm bảo bảo mật:****

Website cần được bảo mật tốt để bảo vệ thông tin người dùng và dữ liệu website.

### ****Tối ưu hóa SEO:****

Áp dụng các kỹ thuật SEO để website dễ dàng được tìm kiếm trên các công cụ tìm kiếm như Google, thu hút lượng truy cập tự nhiên.

### ****Marketing và quảng bá:****

Thực hiện các chiến dịch marketing và quảng bá hiệu quả để thu hút người dùng truy cập website.

## Hướng phát triển

Ngoài những chức năng cơ bản, website review truyện online có thể được phát triển thêm nhiều tính năng nâng cao để tăng khả năng cạnh tranh và đáp ứng nhu cầu ngày càng cao của người dùng:

### ****Hệ thống cá nhân hóa:****

Ghi nhớ sở thích đọc truyện của người dùng và đề xuất những tác phẩm phù hợp dựa trên lịch sử đọc, đánh giá và tương tác của họ.

### ****Hệ thống cộng đồng:****

Tạo lập diễn đàn, nhóm thảo luận để người dùng chia sẻ cảm nhận, đánh giá và bình luận về truyện, giao lưu, kết bạn và tổ chức các hoạt động cộng đồng.

### ****Hệ thống gamification:****

Áp dụng các yếu tố trò chơi như điểm thưởng, huy hiệu, bảng xếp hạng để khuyến khích người dùng tham gia tương tác với website, đọc truyện, viết đánh giá và bình luận.

### ****Hệ thống livestream:****

Cho phép tác giả và reviewer livestream giới thiệu, chia sẻ về truyện hoặc tổ chức các buổi giao lưu trực tuyến với người đọc.

### ****Hệ thống hỗ trợ đa ngôn ngữ:****

Hỗ trợ nhiều ngôn ngữ để tiếp cận người dùng ở nhiều quốc gia khác nhau.

### ****Hệ thống thanh toán:****

Triển khai hệ thống thanh toán trực tuyến để người dùng có thể mua truyện điện tử, ủng hộ tác giả hoặc thanh toán cho các dịch vụ khác trên website.

### ****Cập nhật nội dung thường xuyên:****

Cung cấp cho người dùng những nội dung mới mẻ, hấp dẫn như truyện mới, chương mới, đánh giá mới, tin tức về ngành truyện tranh, v.v.

### ****Hợp tác với tác giả và reviewer:****

Hợp tác với các tác giả và reviewer uy tín để thu hút độc giả và tạo nội dung chất lượng cao cho website.

### ****Tổ chức các sự kiện:****

Tổ chức các sự kiện online hoặc offline như hội chợ truyện tranh, minigame, giveaway để thu hút người dùng và quảng bá website.

### ****Phân tích dữ liệu:****

Thu thập và phân tích dữ liệu về hành vi người dùng để hiểu rõ hơn về nhu cầu và sở thích của họ, từ đó điều chỉnh nội dung và chức năng website phù hợp.

### ****Cập nhật xu hướng:****

Theo dõi và cập nhật những xu hướng mới nhất trong ngành truyện tranh để website luôn bắt kịp thị hiếu của người dùng.

### Bên cạnh các tính năng trên, website review truyện online cũng cần chú trọng đến:

* Tạo dựng cộng đồng người dùng gắn kết và tích cực.
* Cung cấp dịch vụ khách hàng chuyên nghiệp và tận tình.
* Đảm bảo website hoạt động ổn định và mượt mà.
* Cập nhật và cải tiến website thường xuyên để đáp ứng nhu cầu ngày càng cao của người dùng.

Bằng cách đầu tư vào phát triển website một cách bài bản và không ngừng sáng tạo, website review truyện online có thể trở thành một kênh giải trí và kết nối cộng đồng hiệu quả, đồng thời mang lại lợi nhuận cho nhà đầu tư.

**DANH MỤC TÀI LIỆU THAM KHẢO**

Trình bày theo quy định của chuẩn IEEE

|  |  |
| --- | --- |
| [1] | J. N. Robbins, Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics, O'Reilly Media, 1993. |
| [2] | W3Schools, “W3Schools HTML Tutorial,” 2024. [Trực tuyến]. Available: https://www.w3schools.com/html/. [Đã truy cập 1 Tháng hai 2024]. |
| [3] | M. D. N. (MDN), “Mozilla Developer Network (MDN) HTML Guide,” 2024. [Trực tuyến]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML. [Đã truy cập 10 Tháng tư 2024]. |
| [4] | HTML5Doctor, “HTML5 Doctor,” [Trực tuyến]. Available: http://html5doctor.com/. [Đã truy cập 10 Tháng tư 2024]. |
| [5] | W3Schools, “W3Schools CSS Tutoria,” 2024. [Trực tuyến]. Available: https://www.w3schools.com/css/default.asp. [Đã truy cập 10 Tháng ba 2024]. |
| [6] | M. D. N. (MDN), “Mozilla Developer Network (MDN) CSS Guide,” 2024. [Trực tuyến]. Available: https://developer.mozilla.org/en-US/docs/Web/CSS. [Đã truy cập 10 Tháng tư 2024]. |
| [7] | CSSTricks, “CSS Tricks,” 2024. [Trực tuyến]. Available: https://css-tricks.com/. [Đã truy cập 10 Tháng tư 2024]. |
| [8] | W3Schools, “W3Schools JavaScript Tutorial,” 2024. [Trực tuyến]. Available: https://www.w3schools.com/js/default.asp. [Đã truy cập 10 Tháng tư 2024]. |
| [9] | M. D. N. (MDN), “Mozilla Developer Network (MDN) JavaScript Guide,” 2024. [Trực tuyến]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript. [Đã truy cập 10 Tháng tư 2024]. |
| [10] | EloquentJavaScript, “Eloquent JavaScript,” 2024. [Trực tuyến]. Available: https://eloquentjavascript.net/. [Đã truy cập 10 Tháng tư 2024]. |
| [11] | JavaScriptInfo, “JavaScript Info,” 2024. [Trực tuyến]. Available: https://javascript.info/. [Đã truy cập 10 Tháng tư 2024]. |
| [12] | W3Schools, “W3Schools Bootstrap 5 Tutorial,” 2024. [Trực tuyến]. Available: https://www.w3schools.com/bootstrap5/index.php. [Đã truy cập 10 Tháng tư 2024]. |
| [13] | Bootstrap, “Bootstrap,” 2024. [Trực tuyến]. Available: https://getbootstrap.com/. [Đã truy cập 10 Tháng tư 2024]. |
| [14] | FreeCodeCamp, “FreeCodeCamp,” 2024. [Trực tuyến]. Available: https://www.freecodecamp.org/. [Đã truy cập 10 Tháng tư 2024]. |

|  |  |
| --- | --- |
| [15] | Codecademy, “Codecademy,” 2024. [Trực tuyến]. Available: https://www.codecademy.com/. [Đã truy cập 10 Tháng tư 2024]. |