**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**

****

**ISO 9001:2015**

**TRẦN VĂN TRƯỜNG HUY**

**THIẾT KẾ WEBSITE QUẢN LÝ SỰ KIỆN**

**ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**VĨNH LONG, THÁNG 08 NĂM 2025**

**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ**

****

**ISO 9001:2015**

**THIẾT KẾ WEBSITE QUẢN LÝ SỰ KIỆN**

**ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

Giảng viên hướng dẫn:**ThS. DƯƠNG NGỌC VÂN KHANH**

Sinh viên thực hiện:**TRẦN VĂN TRƯỜNG HUY**

Mã số sinh viên: **110121028**

Lớp: **DA21TTB**

Khóa: **2021**

**VĨNH LONG, THÁNG 08 NĂM 2025**

**LỜI MỞ ĐẦU**

Ngày trước, để tham gia một sự kiện, người ta thường phải đăng ký thủ công và nhận vé giấy. Giờ đây, với sự phát triển của công nghệ, mọi thứ đã tiện lợi hơn nhiều nhờ các nền tảng bán vé trực tuyến. Người tham dự chỉ cần vài thao tác là có thể tìm hiểu thông tin, đặt vé và lưu trữ vé điện tử ngay trên điện thoại. Về phía ban tổ chức, hệ thống này giúp họ dễ dàng quản lý, theo dõi và mở rộng quy mô sự kiện một cách hiệu quả hơn.

Một nền tảng bán vé trực tuyến được coi là tốt khi vừa nhanh chóng, dễ dùng, hoạt động mượt trên nhiều thiết bị, lại vừa bảo mật và có khả năng mở rộng khi nhu cầu tăng. Không chỉ đơn thuần là nơi phân phối vé, các hệ thống này còn dần trở thành công cụ hỗ trợ quản lý và tương tác toàn diện với người tham gia.

Để đáp ứng những yêu cầu đó, nhiều nhà phát triển hiện nay lựa chọn kết hợp các công nghệ web hiện đại như Next.js, TailwindCSS, NestJS, MongoDB và Firebase Storage. Bộ công nghệ này vừa đảm bảo hiệu năng, vừa tạo nên một kiến trúc client–server rõ ràng, ổn định và bảo mật. Nhờ vậy, trải nghiệm của người dùng trở nên thuận tiện hơn, đồng thời ban tổ chức cũng tối ưu được việc vận hành, dễ dàng mở rộng thêm nhiều tính năng mới trong tương lai.

**LỜI CẢM ƠN**

Xin gửi lời cảm ơn đến toàn thể giảng viên Trường Đại học Trà Vinh, Trường Kỹ thuật và Công nghê đã tạo điều kiện thuận lợi trong suốt quá trình thực hiện khoá luận. Đặc biệt xin gửi lời cảm ơn quý thầy cô trong Khoa Công nghệ thông tin đã tận tình giảng dạy, truyền đạt kiến thức, giúp hoàn thành khoá luận tốt nghiệp.

Đặc biệt, xin bày tỏ lòng biết ơn sâu sắc đến giảng viên hướng dẫn Dương Ngọc Vân Khanh người đã tận tâm đồng hành, đưa ra những định hướng chuyên môn quý báu và góp ý thiết thực trong suốt quá trình thực hiện đề tài.

Mặc dù đã nỗ lực hết sức để hoàn thành khóa luận này, song chắc chắn vẫn còn tồn tại những hạn chế. Kính mong nhận được những ý kiến đóng góp từ quý thầy cô để nội dung được hoàn thiện hơn trong tương lai.

Trân trọng cảm ơn!

Sinh viên thực hiện

Trần Văn Trường Huy

**HẬN XÉT**

**NHẬN XÉT**

***(Của giảng viên hướng dẫn trong đồ án, khoá luận của sinh viên)***

**Giảng viên hướng dẫn**

(ký và ghi rõ họ tên)

***(Của giảng viên chấm trong đồ án, khoá luận của sinh viên)***

**Giảng viên chấm**

(ký và ghi rõ họ tên)

UBND TỈNH TRÀ VINH **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**TRƯỜNG ĐẠI HỌC TRÀ VINH Độc lập – Tự do – Hạnh Phúc**

**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**

*(Của giảng viên hướng dẫn)*

Họ và tên sinh viên: Trần Văn Trường Huy MSSV: 110121028

Ngành: Công nghệ Thông tin Khóa: 2021

Tên đề tài: Thiết kế website quản lý sự kiện

Họ và tên Giáo viên hướng dẫn: Dương Ngọc Vân Khanh

Chức danh: Giảng viên. Học vị: Thạc sĩ

**NHẬN XÉT**

1. Nội dung đề tài:

1. Ưu điểm:

1. Khuyết điểm:

1. Điểm mới đề tài:

1. Giá trị thực trên đề tài:

7. Đề nghị sửa chữa bổ sung:

8. Đánh giá:

Vĩnh Long*, ngày tháng năm 20…*

Giảng viên hướng dẫn

*(Ký & ghi rõ họ tên)*

**NHẬN XÉT**

***(Của cán bộ chấm đồ án, khoá luận của sinh viên)***

**Cán bộ chấm khoá luận**

(ký và ghi rõ họ tên)

UBND TỈNH TRÀ VINH **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**TRƯỜNG ĐẠI HỌC TRÀ VINH Độc lập - Tự do - Hạnh phúc**

**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**

*(Của cán bộ chấm đồ án, khóa luận)*

Họ và tên người nhận xét:

Chức danh: Học vị:

Chuyên ngành:

Cơ quan công tác:

Tên sinh viên:

Tên đề tài đồ án, khóa luận tốt nghiệp:

**I. Ý KIẾN NHẬN XÉT**

1. Nội dung:

2. Điểm mới các kết quả của đồ án, khóa luận:

3. Ứng dụng thực tế:

**II. CÁC VẤN ĐỀ CẦN LÀM RÕ**

(Các câu hỏi của giáo viên phản biện)

**III. KẾT LUẬN**

(Ghi rõ đồng ý hay không đồng ý cho bảo vệ đồ án khóa luận tốt nghiệp)

*……………, ngày …… tháng …… năm 20…*

Người nhận xét

*(Ký & ghi rõ họ tên)*

**MỤC LỤC**

[CHƯƠNG 1. TỔNG QUAN 1](#_Toc207501143)

[1.1 Giới thiệu đề tài 1](#_Toc207501144)

[1.2 Lý do chọn đề tài 1](#_Toc207501145)

[1.3 Mục tiêu của đề tài 2](#_Toc207501146)

[1.4 Phạm vi và đối tượng nghiên cứu 3](#_Toc207501147)

[1.5 Phương pháp nghiên cứu 4](#_Toc207501148)

[CHƯƠNG 2. NGHIÊN CỨU LÝ THUYẾT 6](#_Toc207501149)

[2.1 Tổng quan về hệ thống quản lý sự kiện 6](#_Toc207501150)

[2.2 Giới thiệu về NextJS 7](#_Toc207501151)

[2.2.1 Tính năng Rendering 7](#_Toc207501152)

[2.2.2 Client-Side Rendering (CSR) 10](#_Toc207501153)

[2.3 Tính năng Routing 10](#_Toc207501154)

[2.3.1 File-based Routing 10](#_Toc207501155)

[2.3.2 Dynamic Routing 11](#_Toc207501156)

[2.3.3 Nested Routing 11](#_Toc207501157)

[2.3.4 API Routing 12](#_Toc207501158)

[2.4 Styling trong NextJS 13](#_Toc207501159)

[2.4.1 CSS Modules 13](#_Toc207501160)

[2.4.2 Styled JSX 13](#_Toc207501161)

[2.4.3 CSS-in-JS 14](#_Toc207501162)

[2.4.4 CSS Frameworks 14](#_Toc207501163)

[2.5 Giới thiệu về TailwindCSS 15](#_Toc207501164)

[2.6 Giới thiệu về NestJS 16](#_Toc207501165)

[2.7 Giới thiệu về MongoDB 18](#_Toc207501166)

[2.8 Giới thiệu về Firebase Storage 19](#_Toc207501167)

[2.9 Giới thiệu về kiến trúc Client-Server 20](#_Toc207501168)

[CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU 22](#_Toc207501169)

[3.1 Mô tả bài toán 22](#_Toc207501170)

[3.2 Phân tích thiết kế hệ thống 23](#_Toc207501171)

[3.2.1 Đặc tả yêu cầu hệ thống 23](#_Toc207501172)

[3.2.2 Kiến trúc hệ thống 27](#_Toc207501173)

[3.2.3 Sơ đồ use case 28](#_Toc207501174)

[3.2.4 Sơ đồ lớp 33](#_Toc207501175)

[3.2.5 Sơ đồ hoạt động 34](#_Toc207501176)

[3.2.6 Sơ đồ tuần tự 37](#_Toc207501177)

[3.3 Thiết kế dữ liệu 40](#_Toc207501178)

[3.3.1 Danh sách các thực thể 40](#_Toc207501179)

[3.3.2 Chi tiết thực thể 40](#_Toc207501180)

[3.4 Thiết kế giao diện 46](#_Toc207501181)

[3.4.1 Sơ đồ website người dùng cuối 46](#_Toc207501182)

[3.4.2 Giao diện website cho người dùng 46](#_Toc207501183)

[CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU 54](#_Toc207501184)

[4.1 Dữ liệu thử nghiệm 54](#_Toc207501185)

[4.2 Kết quả thử nghiệm 58](#_Toc207501186)

[4.2.1 Hiển thị thông tin sự kiện 58](#_Toc207501187)

[4.2.2 Hiển thị các trang chức năng đặt vé 63](#_Toc207501188)

[4.2.3 Hiển thị các trang tác vụ ban tổ chức 66](#_Toc207501189)

[4.2.4 Hiển thị các tác vụ quản trị viên 71](#_Toc207501190)

[4.2.5 Hiển thị tin tức 78](#_Toc207501191)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 81](#_Toc207501192)

[5.1 Kết luận 81](#_Toc207501193)

[5.2 Hướng phát triển 82](#_Toc207501194)

**DANH MỤC HÌNH ẢNH - BẢNG BIỂU**

**Hình ảnh**

[Hình 1 Kiến trúc hệ thống Client-Server 27](#_Toc207501195)

[Hình 2 Sơ đồ use case Người dùng 28](#_Toc207501196)

[Hình 3 Sơ đồ use case Ban tổ chức 30](#_Toc207501197)

[Hình 4.Sơ đồ use case Quản trị 32](#_Toc207501198)

[Hình 5 Sơ đồ lớp 33](#_Toc207501199)

[Hình 6 Sơ đồ hoạt động đặt vé 34](#_Toc207501200)

[Hình 7 Sơ đồ hoạt động quản lý sự kiện 35](#_Toc207501201)

[Hình 8 Sơ đồ tuần tự đặt vé 37](#_Toc207501202)

[Hình 9 Sơ đồ tuần tự tạo sự kiện 38](#_Toc207501203)

[Hình 10 Sơ đồ tuần tự quét vé 39](#_Toc207501204)

[Hình 11 Sơ đồ Website người dùng cuối 46](#_Toc207501205)

[Hình 12 Lo-fi prototype Trang chủ 47](#_Toc207501206)

[Hình 13 Lo-fi prototype Trang danh sách ve 48](#_Toc207501207)

[Hình 14 Lo-fi prototype Trang chi tiết sự kiện 49](#_Toc207501208)

[Hình 15 Lo-fi prototype Trang chọn vé 50](#_Toc207501209)

[Hình 16 Lo-fi prototype Trang danh sách vé đã mua 51](#_Toc207501210)

[Hình 17 Lo-fi prototype Trang chi tiết ve đã mua 51](#_Toc207501211)

[Hình 18 Lo-fi prototype Trang danh sách bản tin 52](#_Toc207501212)

[Hình 19 Lo-fi prototype Trang chi tiết bản tin 53](#_Toc207501213)

[Hình 20 Trang chủ 59](#_Toc207501214)

[Hình 21 Trang sự kiện theo loại 60](#_Toc207501215)

[Hình 22 Chức năng lọc sự kiện 60](#_Toc207501216)

[Hình 23 Trang chi tiết sự kiện 62](#_Toc207501217)

[Hình 24 Trang chọn vé 63](#_Toc207501218)

[Hình 25 Trang thông tin thanh toán 63](#_Toc207501219)

[Hình 26 Email nhận vé 65](#_Toc207501220)

[Hình 27 Trang danh sách vé đã mua 65](#_Toc207501221)

[Hình 28 Trang chi tiết đơn vé 66](#_Toc207501222)

[Hình 29 Trang tạo sự kiện 67](#_Toc207501223)

[Hình 30 Trang tạo sự kiện tiếp theo 67](#_Toc207501224)

[Hình 31 Trang danh sách sự kiện ban tổ chức 68](#_Toc207501225)

[Hình 32 Trang thống kê sự kiện 69](#_Toc207501226)

[Hình 33 Trang quét vé 69](#_Toc207501227)

[Hình 34 Các thao tác chỉnh sửa sự kiện. 70](#_Toc207501228)

[Hình 35 Form sửa sự kiện 70](#_Toc207501229)

[Hình 36 Form sửa xuất 70](#_Toc207501230)

[Hình 37 Form sửa vé 71](#_Toc207501231)

[Hình 38 Form thêm mới xuất 71](#_Toc207501232)

[Hình 39 Trang chủ quản trị 72](#_Toc207501233)

[Hình 40 Quản trị tab sự kiện 73](#_Toc207501234)

[Hình 41 Quản trị tab sự kiện thống kê 74](#_Toc207501235)

[Hình 42 Quản trị tab người dùng 75](#_Toc207501236)

[Hình 43 Quản trị tab ban tổ chức 75](#_Toc207501237)

[Hình 44 Quản trị tab giao dịch 76](#_Toc207501238)

[Hình 45 Quản trị tab thanh toán 76](#_Toc207501239)

[Hình 46 Quản trị tab thêm bản tin 77](#_Toc207501240)

[Hình 47 Quản trị tab sửa sự kiện 77](#_Toc207501241)

[Hình 48 Danh sách sự kiện hoàn vé 78](#_Toc207501242)

[Hình 49 Email gửi cho người mua vé được hoàn tiền 78](#_Toc207501243)

[Hình 50 Trang danh sách sự kiện 79](#_Toc207501244)

[Hình 51 Trang chi tiết sự kiện 80](#_Toc207501245)

**Bảng biểu**

[Bảng 1 Bảng danh sách thực thể 40](#_Toc207501246)

[Bảng 2 Bảng thực thể Tài khoản 40](#_Toc207501247)

[Bảng 3 Thực thể Người dùng 41](#_Toc207501248)

[Bảng 4 Thực thể Ban tổ chức 42](#_Toc207501249)

[Bảng 5 Thực thể Vé 43](#_Toc207501250)

[Bảng 6 Thực thể Xuất sự kiện 44](#_Toc207501251)

[Bảng 7 Thực thể Vé sự kiện 44](#_Toc207501252)

[Bảng 8 Thực thể Đơn vé 45](#_Toc207501253)

[Bảng 9 Thực thể Vé đã đơn hàng 45](#_Toc207501254)

[Bảng 10 Thực thể Bản tin 46](#_Toc207501255)

[Bảng 11 Dữ liệu tài khoản 54](#_Toc207501256)

[Bảng 12 Dữ liệu bảng ban tổ chức 54](#_Toc207501257)

[Bảng 13 Dữ liệu người dùng 55](#_Toc207501258)

[Bảng 14 Dữ liệu sự kiện 55](#_Toc207501259)

[Bảng 15 Dữ liệu xuất sự kiện 56](#_Toc207501260)

[Bảng 16 Dữ liệu vé sự kiện 56](#_Toc207501261)

[Bảng 17 Dữ liệu đơn vé 56](#_Toc207501262)

[Bảng 18 Dữ liệu vé sử dụng 56](#_Toc207501263)

[Bảng 19 Dữ liệu tin tức 57](#_Toc207501264)

# TỔNG QUAN

## Giới thiệu đề tài

Đề tài “Thiết kế Website quản lý sự kiện” được thực hiện nhằm xây dựng một hệ thống website có chức năng chính là hỗ trợ tổ chức các loại sự kiện bán vé trực tuyến để đáp ứng cho nhu cầu tham gia các sự kiện giải trí, hội thảo, biểu diễn nghệ thuật ngày càng gia tăng hiện nay.

Hệ thống cho phép người tổ chức dễ dàng tạo sự kiện, quản lý thông tin vé, theo dõi người mua vé đồng thời giúp người dùng tìm kiếm sự kiện, thanh toán và nhận vé điện tử tích hợp mã QR. Website được thiết kế với giao diện thân thiện, hiệu suất cao và đảm bảo an toàn dữ liệu người dùng, hướng tới việc đơn giản hóa toàn bộ quy trình bán vé và tham gia sự kiện trên nền tảng số.

Về mặt kỹ thuật, hệ thống ứng dụng các công nghệ hiện đại như NextJS kết hợp TailwinCSS cho frontend nhầm tạo nên một giao diện hiện đại, thân thiện. NestJS cho phần backend, MongoDB cho lưu trữ dữ liệu phi quan hệ kết hợp Firebase Storage để quản lý tài nguyên hình ảnh. Việc sử dụng những công nghệ này nhằm đảm bảo tính ổn định, bảo mật và khả năng mở rộng của hệ thống trong quá trình triển khai thực tế.

Đề tài hướng tới việc cung cấp một giải pháp số hóa cho hoạt động tổ chức và tham gia sự kiện, giúp tiết kiệm thời gian, nâng cao hiệu quả quản lý và mang lại trải nghiệm tốt hơn cho người dùng. Một ứng dụng có tiềm năng triển khai thực tế cao trong bối cảnh chuyển đổi số hiện nay.

## Lý do chọn đề tài

Hiện nay, nhu cầu tổ chức và tham gia các sự kiện ngày càng tăng, đặc biệt là tại các thành phố lớn. Tuy nhiên, việc quản lý vé, kiểm soát số lượng người tham dự và đảm bảo quy trình mua vé diễn ra thuận tiện vẫn còn gặp nhiều hạn chế nếu chỉ thực hiện theo phương thức thủ công hoặc qua các kênh phân phối truyền thống.

Các nền tảng bán vé sự kiện trực tuyến hiện nay đã phần nào giải quyết được bài toán này, tuy nhiên chi phí triển khai hoặc thiếu tính tùy biến theo nhu cầu riêng khiến nhiều đơn vị tổ chức nhỏ và vừa không dễ tiếp cận. Từ thực tế đó, việc phát triển một hệ thống website bán vé sự kiện riêng, dễ triển khai và quản lý, sẽ giúp các đơn vị tổ chức chủ động hơn trong công tác vận hành, đồng thời nâng cao trải nghiệm người dùng thông qua các chức năng tự động như đăng ký, thanh toán, nhận vé điện tử và quét mã QR.

Bên cạnh đó, việc xây dựng đề tài này cũng là cơ hội để vận dụng và củng cố các kiến thức đã học về thiết kế giao diện, lập trình backend, quản lý cơ sở dữ liệu và triển khai hệ thống website thực tế. Đề tài không chỉ mang tính ứng dụng cao mà còn phù hợp với xu hướng chuyển đổi số hiện nay trong lĩnh vực sự kiện, giải trí và thương mại dịch vụ.

## Mục tiêu của đề tài

Mục tiêu chính của đề tài là xây dựng một website bán vé sự kiện trực tuyến, cho phép người tổ chức dễ dàng tạo sự kiện, quản lý vé và kiểm soát người tham gia, đồng thời cung cấp cho người dùng một nền tảng thuận tiện để tìm kiếm, đăng ký và mua vé tham dự các sự kiện một cách nhanh chóng và an toàn.

Các mục tiêu cụ thể bao gồm:

* **Thiết kế giao diện người dùng đơn giản, dễ sử dụng:** Giao diện hiện đại, thân thiện với trải nghiệm người dùng, hỗ trợ thao tác dễ dàng trên cả máy tính và thiết bị di động.
* **Xây dựng hệ thống tạo và quản lý sự kiện:** Cho phép người tổ chức tạo sự kiện, chỉnh sửa nội dung. Đồng thời theo dõi tình trạng bán vé.
* **Chức năng đăng ký và mua vé trực tuyến:** Hỗ trợ người dùng tìm kiếm sự kiện, lựa chọn loại vé, thực hiện thanh toán với VNPay và nhận vé điện tử dưới dạng mã QR.
* **Phân quyền người dùng:** Phân biệt rõ các vai trò như quản trị viên, người tổ chức sự kiện và người tham gia đảm bảo mỗi nhóm người dùng có quyền truy cập và thao tác phù hợp.
* **Tối ưu hiệu suất và bảo mật:** Đảm bảo hệ thống hoạt động ổn định, dữ liệu người dùng được bảo vệ an toàn, đồng thời tăng khả năng xử lý truy cập đồng thời.
* **Tích hợp lưu trữ đám mây:** Sử dụng Firebase Storage để lưu trữ ảnh sự kiện một cách hiệu quả và linh hoạt.
* **Triển khai hệ thống backend vững chắc:** Sử dụng NestJS để xây dựng các API, MongoDB làm cơ sở dữ liệu nhằm đảm bảo khả năng mở rộng và duy trì dữ liệu hiệu quả.

Thông qua việc đạt được các mục tiêu trên, hệ thống hướng tới việc trở thành một nền tảng bán vé sự kiện hiệu quả, đáng tin cậy và có khả năng ứng dụng thực tiễn cao.

## Phạm vi và đối tượng nghiên cứu

Phạm vi nghiên cứu: Đề tài tập trung vào việc xây dựng một hệ thống website bán vé sự kiện trực tuyến phục vụ cho nhu cầu tổ chức và tham gia các sự kiện có thu phí. Phạm vi cụ thể của hệ thống bao gồm:

* Khảo sát các mô hình bán vé sự kiện trực tuyến hiện nay, phân tích ưu và nhược điểm để rút ra giải pháp phù hợp.
* Nghiên cứu trải nghiệm người dùng trong quy trình mua vé trực tuyến, bao gồm giao diện, các thông tin hiển thị.
* Tìm hiểu và áp dụng công nghệ vé điện tử với mã QR nhằm phục vụ việc xác thực khi tham dự sự kiện.
* Đánh giá khả năng ứng dụng các công nghệ web hiện đại như Next.js, TailwindCSS, NestJS, MongoDB và Firebase Storage trong việc xây dựng hệ thống ổn định, bảo mật và linh hoạt.
* Thử nghiệm tích hợp thanh toán trực tuyến qua cổng VNPAY để mô phỏng quy trình mua vé thực tế và hoàn vé tự động.
* Giới hạn nghiên cứu tập trung vào nhóm sự kiện có thu phí cơ bản, không mở rộng sang các mảng như chiến lược marketing, quảng bá sự kiện hay tích hợp đa dạng cổng thanh toán ngoài phạm vi thử nghiệm.

Đối tượng nghiên cứu của đề tài bao gồm:

* **Ban tổ chức sự kiện:** Có nhu cầu tạo và quản lý các sự kiện trực tuyến, theo dõi lượt đăng ký, kiểm soát vé và quét vé vào cổng.
* **Người tham gia sự kiện:** Có nhu cầu tìm kiếm, đăng ký, thanh toán và nhận vé tham dự sự kiện.
* **Quản trị viên hệ thống:** Có quyền quản lý toàn bộ người dùng, duyệt ban tổ chức mới, kiểm soát nội dung và bảo trì hệ thống.

Việc tập trung nghiên cứu vào các nhóm đối tượng trên sẽ giúp đảm bảo rằng hệ thống đáp ứng đúng nhu cầu thực tế, đồng thời dễ dàng mở rộng và tích hợp thêm tính năng trong tương lai nếu cần thiết.

## Phương pháp nghiên cứu

**Phương pháp nghiên cứu lý thuyết:** Tìm hiểu các tài liệu chuyên ngành liên quan đến quy trình tổ chức sự kiện, nghiệp vụ quản lý bán vé trực tuyến và hệ thống phân quyền người dùng. Đồng thời nghiên cứu các công nghệ sử dụng trong đề tài như: ReactJS cho giao diện người dùng, NestJS cho backend, MongoDB cho cơ sở dữ liệu NoSQL, Firebase Storage cho lưu trữ ảnh và mã QR dùng trong xác thực vé điện tử. Việc đọc hiểu và tổng hợp kiến thức từ tài liệu chính thống, diễn đàn kỹ thuật, và các hệ thống thực tế (như Ticketbox, Eventbrite) là cơ sở để định hướng thiết kế và triển khai hệ thống.

**Phương pháp nghiên cứu thực nghiệm:** Tiến hành thiết kế giao diện người dùng (UI) và xây dựng chức năng hệ thống dành cho ba nhóm người dùng: người tham gia, người tổ chức sự kiện và quản trị viên. Thực hiện lập trình các chức năng chính như tạo sự kiện, đăng ký mua vé, thanh toán giả lập, sinh mã QR và phân quyền truy cập. Hệ thống được kiểm thử với dữ liệu mẫu thu thập từ Internet (hình ảnh, nội dung sự kiện) để đánh giá khả năng vận hành, tính ổn định và độ chính xác trong xử lý nghiệp vụ thực tế.

Công nghệ sử dụng:

* **Next.js:** Đảm nhiệm vai trò xây dựng giao diện người dùng (frontend), kết hợp giữa phát triển phía máy khách và máy chủ (hybrid rendering). Next.js hỗ trợ tối ưu hiệu suất tải trang, định tuyến động và khả năng SEO tốt hơn. Giao diện cho phép người dùng duyệt danh sách sự kiện, mua vé, quản lý tài khoản và tương tác mượt mà trong toàn bộ quá trình đặt vé.
* **TailwindCSS:** Thiết kế giao diện hiện đại, trực quan và phản hồi tốt trên nhiều thiết bị. Tailwind giúp tùy chỉnh nhanh chóng giao diện bằng cách sử dụng hệ thống utility-first, hỗ trợ việc phát triển giao diện đồng bộ và dễ mở rộng.
* **NestJS:** Đảm nhận vai trò backend, xây dựng hệ thống API phục vụ quản lý sự kiện, người dùng, vé, thanh toán và phân quyền. NestJS giúp tổ chức mã nguồn rõ ràng, dễ bảo trì, đồng thời hỗ trợ khả năng bảo mật và mở rộng hệ thống.
* **MongoDB:** Lưu trữ dữ liệu người dùng, sự kiện, đơn đặt vé và lịch sử giao dịch dưới dạng tài liệu (document), phù hợp với đặc thù dữ liệu linh hoạt và không ràng buộc chặt chẽ về lược đồ.
* **Firebase Storage:** Dùng để lưu trữ và truy xuất ảnh sự kiện, hình ảnh người dùng hoặc các tệp đa phương tiện khác một cách nhanh chóng và hiệu quả, giúp giảm tải cho máy chủ backend.
* **Mã QR:** Tự động tạo mã QR cho từng vé điện tử sau khi thanh toán thành công. Mã này được sử dụng để quét và xác minh tại điểm check-in sự kiện, đảm bảo tính xác thực và kiểm soát lượng người tham gia.
* **JWT (JSON Web Token):** Xác thực và phân quyền người dùng khi truy cập hệ thống. Giúp bảo vệ các API và đảm bảo an toàn cho các phiên đăng nhập.

# NGHIÊN CỨU LÝ THUYẾT

## Tổng quan về hệ thống quản lý sự kiện

Hệ thống quản lý sự kiện là một nền tảng công nghệ thông tin được thiết kế nhằm hỗ trợ các cá nhân, tổ chức trong việc tổ chức, giám sát và phân phối vé cho các sự kiện như hội thảo, buổi biểu diễn nghệ thuật, hội chợ thương mại, buổi biễu diễn ca nhạc. Để đáp ứng được các nhu cầu trên thi một hệ thống quản lý sự kiện hiện đại thường bao gồm các **thành phần và chức năng cơ bản sau**:

**Tạo và quản lý sự kiện**

* Cho phép người tổ chức tạo sự kiện với thông tin chi tiết như: tiêu đề, thời gian, địa điểm, mô tả, hình ảnh và các loại vé.
* Hỗ trợ chỉnh sửa, cập nhật, xóa hoặc gỡ sự kiện.

**Quản lý vé và đơn đặt hàng**

* Tạo các loại vé, giới hạn số lượng, thiết lập mức giá, thiết lặp số lượng mua vé tối thiểu và tối đa.
* Theo dõi số lượng vé đã bán, số vé còn lại và trạng thái đơn hàng, ngừng bán vé, thay đổi số lượng vé.

**Đăng ký và thanh toán**

* Người dùng có thể tìm kiếm, chọn sự kiện, chọn vé và tiến hành thanh toán qua các cổng VNPay.
* Hệ thống trả về vé điện tử tích hợp mã QR ngay sau khi thanh toán thành công qua email và kho vé trên hệ thống.

**Quét mã và kiểm soát ra vào**

* Sử dụng thiết bị di động hoặc máy quét để xác minh tính hợp lệ của vé thông qua mã QR.
* Hỗ trợ thống kê lượng người tham dự theo thời gian thực.

**Quản lý người dùng và phân quyền**

* Phân loại người dùng thành các vai trò: người tham gia, người tổ chức và quản trị viên (admin).
* Mỗi vai trò được cấp quyền truy cập và chức năng tương ứng trong hệ thống.

**Hệ thống báo cáo và thống kê**

* Cung cấp các báo cáo về doanh thu, lượt đăng ký, số lượng vé bán ra, mức độ tương tác người dùng.
* Giúp người tổ chức có dữ liệu để đánh giá hiệu quả sự kiện và đưa ra quyết định cho các lần tổ chức tiếp theo.

Hệ thống quản lý sự kiện không chỉ giúp đơn giản hóa quy trình tổ chức mà còn góp phần chuyên nghiệp hóa dịch vụ, tăng mức độ hài lòng của người tham gia và mở rộng quy mô tiếp cận khách hàng tiềm năng. Chính vì vậy, việc nghiên cứu và triển khai một hệ thống như vậy là cần thiết và phù hợp với xu hướng phát triển công nghệ hiện nay.

## Giới thiệu về NextJS

Next.js là một framework phổ biến trong việc phát triển ứng dụng web dựa trên React và được phát triển bởi Vercel. Được giới thiệu lần đầu tiên vào năm 2016, Next.js cung cấp một nền tảng mạnh mẽ và linh hoạt để phát triển các ứng dụng React tối ưu hóa hiệu suất và thân thiện với công cụ tìm kiếm (SEO). Nhờ vào khả năng kết hợp tính năng server-side rendering (SSR) và static site generation (SSG), Next.js đã trở thành một lựa chọn hàng đầu của các lập trình viên khi xây dựng các ứng dụng web hiện đại.

Next.js có sẵn những tính năng và tự động cấu hình các công cụ cần thiết cho React như bundling, compiling, ... giúp lập trình viên tận dụng và tập trung vào việc phát triển ứng dụng một cách nhanh chóng và tối ưu thay vì mất thời gian vào việc cấu hình.

### Tính năng Rendering

Next.js hỗ trợ nhiều phương thức render, giúp tăng cường hiệu suất, SEO và trải nghiệm người dùng. Các phương thức render trong Next.js bao gồm **Static Generation (SSG)**, **Server-Side Rendering (SSR)**, **Incremental Static Regeneration (ISR)** và **Client-Side Rendering (CSR)**. Mỗi phương thức đều có ưu, nhược điểm riêng và phù hợp với những trường hợp sử dụng khác nhau.

#### Static Site Generation (SSG)

**Static Generation** là phương thức render phổ biến trong Next.js, trong đó HTML của trang được tạo ra tại thời điểm build, sau đó sẽ được lưu và phục vụ dưới dạng tĩnh cho các yêu cầu người dùng. Đây là phương thức render hiệu quả nhất về hiệu suất vì trang đã được chuẩn bị sẵn trước khi người dùng truy cập. Với Next.js, bạn có thể dễ dàng áp dụng SSG bằng cách sử dụng hàm getStaticProps, hàm này sẽ lấy dữ liệu tại thời điểm build và chuyển vào component dưới dạng props.

**Ưu điểm**:

* Hiệu năng cao: Do HTML được chuẩn bị sẵn, tốc độ tải trang rất nhanh.
* SEO tốt: HTML có sẵn ngay từ đầu, dễ dàng được các công cụ tìm kiếm thu thập thông tin.
* Chi phí thấp: Giảm tải cho máy chủ vì không cần xử lý thêm khi có người dùng truy cập.

**Nhược điểm**:

* Hạn chế thay đổi nội dung: Phương thức này chỉ phù hợp với trường hợp nội dung không thay đổi thường xuyên, còn với nội dung thay đổi liên tục sẽ cần phương thức khác.
* **SSG** phù hợp với các trang có nội dung cố định(tĩnh) như trang giới thiệu sản phẩm (landing page), trang blog, ... hoặc các trang không yêu cầu cập nhật thông tin thường xuyên.

#### Server-side Rendering (SSR)

**Server-Side Rendering**là phương thức render mà trang được tạo tại thời điểm có yêu cầu truy cập. Khi người dùng truy cập trang, máy chủ sẽ tạo HTML với dữ liệu mới nhất và gửi về cho trình duyệt. Điều này giúp cập nhật nội dung theo dữ liệu mới nhất, nhưng cũng tiêu tốn tài nguyên máy chủ. Trong Next.js, SSR có thể được thực hiện thông qua hàm getServerSideProps, hàm này lấy dữ liệu mỗi lần có yêu cầu và chuyển vào component, tạo ra trang được render ngay trên máy chủ.

**Ưu điểm:**

* Cập nhật nội dung động: Nội dung luôn cập nhật mới nhất theo mỗi yêu cầu truy cập.
* SEO tốt: HTML được render từ máy chủ nên dễ thu thập thông tin.

**Nhược điểm:**

* Hiệu suất kém hơn SSG: Do trang phải render mỗi khi có yêu cầu, nên mất thời gian xử lý.
* Chi phí cao hơn: Tăng tải cho máy chủ nếu có lượng truy cập lớn.

**SSR** thích hợp với các trang có dữ liệu thay đổi liên tục như trang thương mại điện tử, mạng xã hội, tin tức, bảng xếp hạng, ... hoặc các trang đòi hỏi tính cá nhân hoá cao.

#### Incremental Static Regeneration (ISR)

**Incremental Static Regeneration** là một trong những tính năng tiên tiến của Next.js, nó phương pháp tối ưu hóa giữa SSG và SSR. ISR cho phép tạo lại các trang tĩnh sau một khoảng thời gian nhất định (thông qua thuộc tính revalidate) mà không cần build lại toàn bộ ứng dụng. Điều này giúp trang vừa có hiệu suất cao như SSG, vừa có khả năng cập nhật nội dung thường xuyên. Với Next.js bằng cách sử dụng thuộc tính revalidate trong hàm getStaticProps, các trang có thể được tạo lại trên máy chủ mà không làm ảnh hưởng đến trải nghiệm người dùng.

**Ưu điểm**:

* Hiệu suất cao: Trang có thể lưu trữ tĩnh nhưng vẫn tự động cập nhật.
* Cập nhật định kỳ: Dễ dàng điều chỉnh thời gian tái tạo trang phù hợp.

**Nhược điểm**: Độ trễ cập nhật dữ liệu: nếu nội dung trên trang cần cập nhật ngay lập tức khi có sự thay đổi, ISR có thể gây ra độ trễ, dẫn đến việc hiển thị dữ liệu lỗi thời cho người dùng trong một thời gian nhất định.

**ISR** đặc biệt lý tưởng với các trang có dữ liệu thay đổi vừa phải, chẳng hạn trang tin tức hoặc trang blog yêu cầu cập nhật nội dung một cách đều đặn nhưng không quá thường xuyên.

### Client-Side Rendering (CSR)

**Client-Side Rendering** là phương thức render truyền thống của React, trong đó nội dung trang sẽ được render từ phía người dùng sau khi tải lên. Next.js cũng hỗ trợ CSR cho các trang không yêu cầu SEO mạnh.

**Ưu điểm**:

* Tương tác linh hoạt: Dễ dàng cập nhật dữ liệu, đặc biệt phù hợp với các ứng dụng có nhiều tương tác người dùng.
* Giảm tải cho máy chủ: Máy chủ chỉ gửi dữ liệu cần thiết, client sẽ xử lý render.

**Nhược điểm**:

* Hiệu suất tải trang ban đầu thấp: Trang ban đầu sẽ chỉ có HTML trống, nội dung chỉ hiển thị sau khi JavaScript được thực thi.
* SEO: Vì nội dung hiển thị trễ nên không thân thiện với các công cụ tìm kiếm.

**CSR** phù hợp với các ứng dụng không đòi hỏi SEO như các ứng dụng đơn trang phức tạp (SPA), ứng dụng nội bộ hoặc trang quản trị.

## Tính năng Routing

Next.js cung cấp hệ thống routing (định tuyến) linh hoạt và dễ sử dụng. Next.js áp dụng cơ chế routing theo **file-based routing** (định tuyến theo tệp tin), giúp đơn giản hóa việc xây dựng và quản lý các tuyến đường (route) trong ứng dụng. Bên cạnh đó, Next.js cũng hỗ trợ các phương pháp định tuyến **dynamic routing, nested routing, API routes** để đáp ứng các nhu cầu phức tạp hơn. Ta cùng tìm hiểu, phân tích về các loại routing trong Next.js và cách chúng hoạt động phía bên dưới.

### File-based Routing

**File-based Routing** là routing cơ bản trong Next.js. Nó sử dụng cấu trúc thư mục để tự động tạo các tuyến đường cho ứng dụng. Tất cả các file nằm trong thư mục /pages sẽ tự động trở thành một route tương ứng, không cần cấu hình thêm.

Ví dụ, một file pages/viblo.js sẽ được render khi người dùng truy cập đường dẫn /viblo. Tương tự, một file pages/viblo/post.js sẽ tạo đường dẫn viblo/post. Điều này giúp quản lý các route trở nên dễ dàng, hiệu quả và loại bỏ sự phức tạp khi phải cấu hình định tuyến như trong các ứng dụng React truyền thống, nơi thường phải sử dụng thư viện react-router.

**Ưu điểm**:

* Dễ sử dụng: Tạo và quản lý route trực tiếp bằng cách tạo các file và thư mục.
* Đồng nhất và trực quan: Cấu trúc thư mục phản ánh cấu trúc định tuyến, dễ theo dõi và kiểm soát.

Nhược điểm: Giới hạn với ứng dụng phức tạp: Khi có nhiều route phức tạp, file-based routing có thể thiếu linh hoạt, đòi hỏi nhiều thư mục lồng nhau.

### Dynamic Routing

Next.js cho phép định tuyến động (**Dynamic Routing**) thông qua các thư mục hoặc file có tên động bằng cách sử dụng cú pháp dynamic segments ([param]). Điều này rất hữu ích khi tạo ra các đường dẫn thay đổi linh hoạt, ví dụ như các trang sản phẩm, bài viết hoặc hồ sơ người dùng.

Ví dụ một file pages/post/[id].js sẽ tạo đường dẫn /post/123, với **123** là ID sản phẩm được lấy từ URL. Hay file pages/post/[slug].js sẽ tạo đường dẫn /post/dynamic-routing, với **dynamic-routing** đại diện cho giá trị **slug**. Điều này mang lại sự linh hoạt cao, đặc biệt cho các ứng dụng co lượng lớn nội dung không xác định trước.

**Ưu điểm**:

* Dễ dàng tùy chỉnh các route động: Đáp ứng nhanh chóng các yêu cầu về route động như profile người dùng, sản phẩm, bài viết, ...
* Tích hợp tốt với API: Next.js hỗ trợ getStaticPaths và getStaticProps, cho phép tạo các trang động mà vẫn tối ưu hóa hiệu suất thông qua phương thức **Static Generation**.

**Nhược điểm**: Độ phức tạp tăng theo cấu trúc: Khi có nhiều cấp độ route động, ứng dụng có thể trở nên khó quản lý.

### Nested Routing

Next.js không hỗ trợ **nested routing** một cách trực tiếp như một số framework khác. Thay vào đó**, nested routing** có thể được thiết lập bằng cách tạo ra các thư mục lồng nhau trong /pages, mỗi một thư mục con sẽ đại diện cho một cấp route mới hoặc sử dụng React components để xây dựng giao diện lồng nhau theo mong muốn.

Ví dụ file pages/post/[slug]/comments.js sẽ dẫn đến đường dẫn /post/post-title/comments. Hoặc sử dụng các component như Header hoặc Sidebar trong file pages/post/[slug]/index.js để hiển thị cấu trúc giao diện lồng nhau.

**Ưu điểm**:

* Tạo cấu trúc trang rõ ràng: Dễ dàng phân chia các thành phần giao diện thành từng phần riêng biệt.
* Tái sử dụng và tùy chỉnh giao diện: Component lồng nhau giúp xây dựng giao diện đa cấp dễ quản lý.

**Nhược điểm**:

* Thiếu hỗ trợ native cho nested routing: do các route được ánh xạ trực tiếp từ cấu trúc thư mục nên sẽ bị hạn chế việc tổ chức cấu trúc route phức tạp, nhất là khi ứng dụng lớn dần và cần nhiều cấp route.
* Khó khăn quản lý layout cho các route lồng nhau: Do không có cấu trúc layout lồng nhau mặc định nên đòi hỏi sử dụng các biện pháp thủ công như tạo nhiều component layou hoặc sử dụng App, ... để quản lý layout làm cho mã nguồn trở nên phức tạp và khó bảo trì.

### API Routing

Next.js còn cung cấp khả năng tạo các **API route**, hoạt động như một API server tích hợp trong ứng dụng mà không cần đến server back-end riêng biệt. Các API route này được tạo ra bằng cách thêm các file vào thư mục /pages/api và sẽ tạo thành các endpoint RESTful tương ứng, có thể xử lý các yêu cầu HTTP như GET, POST, PUT và DELETE. Đây là một tính năng riêng biệt tiện dụng để xây dựng các API nhỏ.

Ví dụ filepages/api/user.js sẽ tạo API route /api/user để xử lý yêu cầu từ phía client. API route này hữu ích khi muốn tạo các endpoint phục vụ các thao tác như gửi email, xử lý dữ liệu từ form hoặc tương tác với các API bên ngoài mà không cần một server backend riêng biệt.

**Ưu điểm**:

* Tích hợp dễ dàng: Xây dựng các API route mà không cần backend server riêng.
* Bảo mật và hiệu suất cao: API route có thể bảo vệ thông tin nhạy cảm vì chỉ được thực thi phía máy chủ.

**Nhược điểm**: Giới hạn về độ phức tạp: API route chỉ phù hợp với các yêu cầu đơn giản, cho các ứng dụng phức tạp sẽ cần giải pháp API khác như sử dụng server riêng.

## Styling trong NextJS

Next.js hỗ trợ nhiều phương pháp styling linh hoạt, giúp các nhà phát triển dễ dàng thiết kế giao diện và tối ưu hóa ứng dụng. **Styling**trong Next.js bao gồm nhiều option từ **CSS Modules, Styled JSX, CSS-in-JS, CSS Frameworks**đến**Tailwind CSS**và**Sass/SCSS.** Mỗi phương pháp đều có ưu, nhược điểm và đáp ứng nhiều nhu cầu khác nhau trong quá trình phát triển. Ta sẽ cùng tìm hiểu một vài kiểu styling trong Next.js phía bên dưới.

### CSS Modules

**CSS Modules** là một tính năng có sẵn trong Next.js, cho phép tạo các kiểu CSS phạm vi cục bộ (local scoped) cho từng component hoặc trang. Mỗi CSS class khi sử dụng CSS Modules đều được tự động tạo ra một tên duy nhất, tránh xung đột và không cần phải sử dụng các quy ước đặt tên phức tạp.

**Ưu điểm**:

* Phạm vi cục bộ: Mỗi class được scope riêng cho từng module, tránh xung đột tên.
* Dễ đọc và bảo trì: Tên class ngắn gọn và không bị trùng lặp.

**Nhược điểm**: Không hỗ trợ tính năng JavaScript trong CSS: CSS Modules thiếu khả năng tùy chỉnh động như CSS-in-JS.

**CSS Modules** phù hợp cho các dự án nhỏ hoặc các trang mà phong cách không cần quá phức tạp.

### Styled JSX

**Styled JSX** là một giải pháp CSS-in-JS có sẵn trong Next.js, giúp nhúng trực tiếp CSS vào trong code của component. **Styled JSX** tạo ra CSS riêng cho từng component và đảm bảo CSS này không bị ảnh hưởng bởi các component khác.

**Ưu điểm**:

* Cục bộ hoá CSS: CSS chỉ áp dụng cho component đó, hạn chế xung đột.
* Dễ dàng tái sử dụng và quản lý: Tất cả kiểu được viết cùng với logic của component, dễ quản lý.
* Tích hợp sẵn trong Next.js: Không cần cài đặt thêm thư viện bên ngoài.

**Nhược điểm**: Phức tạp khi cần mở rộng: Viết CSS cho các component phức tạp có thể gây khó quản lý và thiếu tính linh hoạt.

**Styled JSX** phù hợp cho các ứng dụng Next.js nhỏ hoặc những component có phong cách độc lập, không tái sử dụng nhiều.

### CSS-in-JS

Ngoài Styled JSX, Next.js cũng hỗ trợ các thư viện **CSS-in-JS** phổ biến như styled-components, emotion, ant-design, chakra-ui, ...\*\* CSS-in-JS\*\* là phương pháp viết CSS trực tiếp trong JavaScript, kết hợp sức mạnh của CSS với tính năng lập trình linh hoạt của JavaScript, tạo ra khả năng styling động và dễ quản lý.

**Ưu điểm**:

* Styling động: CSS-in-JS cho phép áp dụng logic JavaScript vào CSS, dễ dàng tạo kiểu phụ thuộc vào state, props hoặc context.
* Quản lý kiểu linh hoạt: Có thể tái sử dụng các styled-component trong toàn bộ ứng dụng.
* Phạm vi cục bộ tự động: CSS tự động scope cho từng component, tránh xung đột.

**Nhược điểm**:

* Hiệu suất: CSS-in-JS có thể ảnh hưởng đến hiệu suất khi chạy ở client-side, do CSS được xử lý động khi render.
* Kích thước bundle lớn hơn: Thêm thư viện CSS-in-JS có thể tăng kích thước bundle.

**CSS-in-JS** phù hợp với các ứng dụng có phong cách phức tạp, yêu cầu nhiều kiểu động hoặc tái sử dụng các component nhiều lần.

### CSS Frameworks

Next.js cũng hỗ trợ các **CSS Frameworks** phổ biến như Bootstrap, Bulma, Tailwind CSS, ... Trong đó, Tailwind CSS là một framework CSS utility-first được nhiều nhà phát triển Next.js yêu thích, với hệ thống các lớp tiện ích giúp xây dựng giao diện nhanh chóng mà không cần viết nhiều CSS tùy chỉnh.

**Ưu điểm**:

* Tốc độ phát triển nhanh: Các class, component có sẵn giúp thiết kế giao diện nhanh chóng.
* Giao diện nhất quán: Sử dụng sẵn class, component nên các giao diện thường có phong cách nhất quán.
* Đáp ứng tốt trên thiết bị di động: Framework CSS cung cấp sẵn các lớp responsive, giúp tối ưu hóa giao diện trên các thiết bị khác nhau.

**Nhược điểm**:

* Hạn chế tuỳ chỉnh: Các lớp tiện ích có thể giới hạn khi cần tạo giao diện độc đáo.
* Kích thước bundle: Một số framework như Bootstrap có thể thêm nhiều code không sử dụng, ảnh hưởng đến hiệu suất.

**CSS Frameworks** phù hợp cho các ứng dụng cần giao diện cơ bản và triển khai nhanh hoặc yêu cầu sự nhất quán cao trong thiết kế.

## Giới thiệu về TailwindCSS

**Tailwind CSS** là một framework CSS theo hướng **utility-first** giúp phát triển giao diện người dùng (UI) nhanh chóng và hiệu quả. Khi nhắc đến các CSS framework phổ biến, ta thường nghĩ đến **Bootstrap**, **Material Design** hoặc **Bulma** – những framework cung cấp sẵn các thành phần UI và chủ đề mặc định. Tuy nhiên, Tailwind CSS đi theo một triết lý khác.

Không giống như các framework truyền thống, Tailwind **không cung cấp các component UI định sẵn** như nút bấm, biểu mẫu hay thanh điều hướng. Thay vào đó, Tailwind cung cấp **một tập hợp các lớp tiện ích (utility classes)** cho phép nhà phát triển xây dựng giao diện từ đầu một cách linh hoạt, kiểm soát tốt và không bị ràng buộc bởi cấu trúc định sẵn.

Tailwind cho phép **tăng tốc quá trình phát triển UI** mà vẫn đảm bảo khả năng bảo trì mã nguồn (maintainability). Khi sử dụng đúng cách, Tailwind giúp giảm thiểu việc viết CSS lặp lại, hạn chế dư thừa và giữ cho kích thước tệp CSS ở mức tối ưu thông qua cơ chế **purge**.

Có ba cách tiếp cận chính khi thiết kế CSS và UI được đề cập. Mỗi cách đều có ưu và nhược điểm được hiển thị dưới đây:

**Component-based CSS (CSS theo thành phần):** Đây là cách tiếp cận mà các framework như Bootstrap hay Material UI sử dụng. Giao diện được xây dựng dựa trên các thành phần được định nghĩa sẵn (component), như: button, navbar, card, form,…

* **Ưu điểm**:Nhanh chóng tạo được các giao diện chuẩn, đồng nhất. Dễ sử dụng cho người mới.
* **Nhược điểm**:Khó tùy biến sâu. Giao diện dễ trở nên rập khuôn, khó tạo phong cách riêng.

**Custom CSS (Viết CSS thủ công):** Nhà phát triển tự định nghĩa toàn bộ CSS cho từng phần tử của giao diện, thường đi kèm với tổ chức theo kiểu BEM (Block Element Modifier).

* **Ưu điểm**:Kiểm soát toàn bộ giao diện, linh hoạt cao. Tạo được giao diện độc đáo, không bị rập khuôn.
* **Nhược điểm**:Tốn thời gian, dễ tạo CSS trùng lặp. Khó bảo trì khi dự án mở rộng.

**Utility-first CSS (CSS theo tiện ích):** Đây là triết lý của Tailwind CSS. Giao diện được xây dựng thông qua việc kết hợp nhiều lớp tiện ích nhỏ, ví dụ: p-4, text-center, bg-blue-500,..

* **Ưu điểm**:Tăng tốc độ phát triển. Dễ tái sử dụng, tránh CSS dư thừa. Dễ tích hợp vào component logic (đặc biệt trong React/Next.js).
* **Nhược điểm**: Giao diện có thể rối nếu không tổ chức tốt. Khó đọc khi class name dài.

## Giới thiệu về NestJS

NestJSlà một NodeJS framework dùng để phát triển server-side applications hiệu quả và có thể mở rộng. NestJS là sự kết hợp bởi OOP(Object Oriented Programming), FP(Functional Programming), FRP(Functional Reactive Programming). NestJS sử dụng TypeScript để phát triển nhưng nó cũng hỗ trợ cả Javascript.

Nest buộc các nhà phát triển sử dụng một kiến trúc cụ thể bằng cách giới thiệu các module, provider và controller, đảm bảo ứng dụng highly scalable, testable và dễ dàng maintaince. Nest rất khắt khe và chặt chẽ trong việc xây dựng cấu trúc project. Vì vậy hãy tuân thủ theo nó

**Các đặc điểm nổi bật của NestJS:**

* **Hỗ trợ TypeScript mạnh mẽ:** Toàn bộ framework được viết bằng TypeScript, cung cấp khả năng kiểm tra kiểu tĩnh, gợi ý thông minh và nâng cao độ an toàn khi phát triển.
* **Kiến trúc mô-đun rõ ràng:** Mỗi tính năng hoặc thành phần của ứng dụng được đóng gói trong một module riêng biệt, dễ bảo trì và tái sử dụng.
* **Tích hợp dễ dàng với cơ sở dữ liệu:** NestJS có thể kết nối dễ dàng với các hệ quản trị cơ sở dữ liệu như PostgreSQL, MySQL, MongoDB thông qua các thư viện ORM như TypeORM, Prisma hoặc Mongoose.
* **Hệ thống Dependency Injection mạnh mẽ:** Tăng khả năng kiểm thử và tái sử dụng mã nhờ cơ chế tiêm phụ thuộc.
* **Tích hợp WebSockets, GraphQL, Microservices:** NestJS hỗ trợ nhiều giao thức truyền thông, thuận tiện xây dựng các ứng dụng thời gian thực hoặc phân tán.

Các thành phần quan trọng của NestJS

Để xây dựng một ứng dụng NestJS hoàn chỉnh, framework này cung cấp một tập hợp các thành phần cốt lõi được tổ chức theo kiến trúc module, giúp mã nguồn rõ ràng, dễ mở rộng và dễ kiểm thử. Dưới đây là mô tả các thành phần quan trọng:

**Module** là đơn vị tổ chức chính trong NestJS. Mỗi module quản lý một nhóm chức năng liên quan như: quản lý sự kiện, người dùng, thanh toán,… Một module có thể bao gồm các controller, service và provider. Module chính (gốc) là AppModule, các module khác được import vào trong đó để cấu trúc toàn bộ ứng dụng.

**Controller** xử lý các yêu cầu HTTP (GET, POST, PUT, DELETE) từ phía client. Mỗi controller định nghĩa các endpoint (route) và định tuyến yêu cầu tới các service tương ứng để xử lý logic nghiệp vụ.

**Service** chứa các logic nghiệp vụ (business logic) của ứng dụng. Controller chỉ gọi service để thực hiện thao tác, tách biệt rõ ràng phần xử lý và phần giao tiếp HTTP.

**Provider** là bất kỳ lớp nào có thể được inject (tiêm phụ thuộc) vào các thành phần khác bằng hệ thống **Dependency Injection (DI)** của NestJS. Các service, repository, middleware, guard,... đều là các provider. NestJS tự động quản lý vòng đời và tạo instance khi cần thiết.

**Dependency Injection (DI) Container:** NestJS sử dụng hệ thống **DI container** mạnh mẽ để tự động cung cấp các phụ thuộc (service, repository, config...) cho các class cần thiết mà không phải khởi tạo thủ công. Giúp mã nguồn linh hoạt, dễ mở rộng và dễ kiểm thử đơn vị.

Middleware, Interceptor, Guard và Pipe:

* **Middleware:** xử lý request trước khi đến controller (dùng cho logging, xác thực...).
* **Interceptor:** can thiệp vào luồng request-response (có thể xử lý logic trước/sau controller).
* **Guard:** kiểm soát quyền truy cập (authorization), ví dụ phân quyền người dùng.
* **Pipe:** xử lý và kiểm tra dữ liệu đầu vào (input validation, transformation).

**Exception Filter:** Xử lý các lỗi toàn cục hoặc cục bộ, trả về thông báo lỗi một cách nhất quán cho client.

**ORM & Database Layer:** NestJS hỗ trợ tích hợp với nhiều thư viện ORM như **TypeORM**, **Prisma**, **Mongoose** để kết nối và thao tác với cơ sở dữ liệu. Lớp repository có thể inject vào service để thực hiện các truy vấn.

## Giới thiệu về MongoDB

MongoDB là một hệ thống quản lý cơ sở dữ liệu (DBMS) phi quan hệ, mã nguồn mở, được phát triển dựa trên cơ sở dữ liệu NoSQL. Công nghệ tiếp cận vấn đề lưu trữ và truy xuất dữ liệu một cách linh hoạt hơn so với cơ sở dữ liệu quan hệ truyền thống. Khác với các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) như MySQL hay PostgreSQL, MongoDB không sử dụng bảng (tables) và hàng (rows) mà thay vào đó dùng **bộ sưu tập (collections)** và **tài liệu (documents)**.

**Đặc điểm nổi bật của MongoDB:**

* **Linh hoạt và không ràng buộc schema (schema-less):** Dữ liệu trong MongoDB không cần tuân theo một cấu trúc cố định, giúp dễ dàng mở rộng và thay đổi cấu trúc dữ liệu khi cần thiết.
* **Dễ mở rộng và phân tán:** MongoDB hỗ trợ **sharding** (phân mảnh dữ liệu), cho phép xử lý và lưu trữ dữ liệu trên nhiều máy chủ.
* **Hiệu năng cao:** MongoDB tối ưu cho các thao tác đọc/ghi nhanh, thích hợp với các ứng dụng thời gian thực (real-time applications).
* **Dữ liệu dạng JSON/BSON:** Dữ liệu được lưu trữ dưới dạng document BSON, dễ dàng thao tác với JavaScript và các công nghệ web hiện đại.

**Cấu trúc dữ liệu trong MongoDB**

* **Database (Cơ sở dữ liệu):** Là nơi lưu trữ các collections.
* **Collection (Bộ sưu tập):** Tương tự như bảng trong RDBMS, lưu trữ nhiều documents.
* **Document (Tài liệu):** Là đơn vị nhỏ nhất, chứa dữ liệu dưới dạng cặp key-value, tương tự như một object JSON.
* **Ví dụ document:**

{

"\_id": "64e12adf98a4bc01",

"eventName": "Music Festival 2025",

"location": "Hà Nội",

"date": "2025-12-01",

"tickets": 250

}

MongoDB là một giải pháp cơ sở dữ liệu NoSQL mạnh mẽ, dễ mở rộng, rất phù hợp với các hệ thống có cấu trúc dữ liệu động như nền tảng bán vé sự kiện. Việc kết hợp MongoDB và NestJS thông qua Mongoose giúp ứng dụng vừa linh hoạt, vừa có khả năng mở rộng tốt và dễ dàng bảo trì trong tương lai.

## Giới thiệu về Firebase Storage

**Firebase Storage** là dịch vụ lưu trữ tệp (file storage) thuộc nền tảng Firebase của Google, cho phép người dùng lưu trữ và truy xuất các tập tin như hình ảnh, video, âm thanh, tài liệu,... một cách an toàn và hiệu quả. Firebase Storage được xây dựng trên nền tảng **Google Cloud Storage**, đảm bảo khả năng mở rộng, độ tin cậy cao và tích hợp chặt chẽ với các dịch vụ Firebase khác.

Các tính năng nổi bật của Firebase Storage:

* **Lưu trữ linh hoạt và mở rộng:** Firebase Storage có thể xử lý từ vài byte đến hàng gigabyte dữ liệu, đáp ứng tốt cho cả ứng dụng nhỏ lẫn hệ thống lớn.
* **Bảo mật và phân quyền:** Firebase sử dụng **Firebase Security Rules**, cho phép thiết lập các quy tắc truy cập chi tiết theo từng người dùng, theo vai trò hoặc điều kiện cụ thể.
* **Tích hợp chặt chẽ với Firebase Authentication:** Có thể kiểm soát quyền truy cập vào file dựa trên trạng thái đăng nhập hoặc thông tin người dùng.
* **Hỗ trợ tải lên/tải xuống không đồng bộ:** Cho phép người dùng tải file lên hoặc tải về mà không ảnh hưởng đến trải nghiệm của ứng dụng.
* **Tối ưu cho mạng yếu:** Firebase Storage hỗ trợ **resume upload** (tải lại khi bị gián đoạn) và tự động điều chỉnh tốc độ truyền phù hợp với tình trạng mạng.

Ưu điểm

* Tốc độ cao: Máy chủ phân phối toàn cầu giúp truy cập file nhanh chóng từ bất kỳ vị trí nào
* Tích hợp Firebase dễ dàng: Tương thích hoàn toàn với Authentication, Firestore, Realtime Database,…
* Quản lý chi tiết: Giao diện Firebase Console cho phép theo dõi dung lượng, số lượt truy cập, và nhật ký tải lên/tải xuống.

**Nhược điểm**

* **Hạn chế cấu hình backend chuyên sâu:** Firebase Storage phù hợp cho những hệ thống không yêu cầu xử lý file phức tạp phía server.
* **Chi phí:** Mặc dù Firebase có gói miễn phí, nhưng với lượng truy cập và lưu trữ lớn, cần tính toán chi phí sử dụng phù hợp.

Firebase Storage là giải pháp lý tưởng cho việc lưu trữ tệp trong các hệ thống web hiện đại. Trong ứng dụng quản lý sự kiện, nó đóng vai trò quan trọng trong việc quản lý nội dung đa phương tiện, đảm bảo hiệu năng, bảo mật và tính linh hoạt cao. Việc tích hợp Firebase Storage với frontend như **Next.js** giúp xử lý ảnh sự kiện, file vé, banner,… một cách nhanh chóng và trực quan.

## Giới thiệu về kiến trúc Client-Server

**Kiến trúc Client–Server** (máy khách – máy chủ) là một mô hình phổ biến trong phát triển phần mềm, đặc biệt trong các ứng dụng web và hệ thống phân tán. Trong mô hình này, **client** (máy khách) gửi yêu cầu dịch vụ, và **server** (máy chủ) tiếp nhận, xử lý yêu cầu và phản hồi kết quả tương ứng.

Mô hình này tạo ra sự phân tách rõ ràng giữa giao diện người dùng (phía client) và logic xử lý (phía server), giúp tăng khả năng mở rộng, bảo trì và tái sử dụng.

Mô hình Client–Server bao gồm hai thành phần chính:

**Client (Máy khách):**

* Là ứng dụng phía người dùng (ví dụ: trình duyệt, ứng dụng di động).
* Gửi các yêu cầu (request) đến server qua mạng.
* Hiển thị dữ liệu và tương tác với người dùng.

**Server (Máy chủ):**

* Là hệ thống hoặc ứng dụng xử lý các yêu cầu từ client.
* Truy cập cơ sở dữ liệu, xử lý nghiệp vụ, và phản hồi dữ liệu lại cho client.

**Quy trình hoạt động**

* **Client gửi yêu cầu** (HTTP Request) đến server (ví dụ: yêu cầu lấy danh sách sự kiện).
* **Server xử lý yêu cầu** (đọc dữ liệu, kiểm tra xác thực, gọi hàm xử lý,…).
* **Server phản hồi** dữ liệu hoặc trạng thái (HTTP Response) về cho client.
* **Client hiển thị dữ liệu** cho người dùng.

**Ưu điểm**

* **Tách biệt rõ ràng:** Giao diện (frontend) và logic xử lý (backend) hoạt động độc lập.
* **Mở rộng dễ dàng:** Có thể mở rộng backend hoặc frontend riêng biệt mà không ảnh hưởng toàn hệ thống.
* **Tăng tính bảo mật:** Dữ liệu nhạy cảm được xử lý và lưu trữ ở phía server.
* **Dễ bảo trì:** Các thành phần được phân tách nên dễ kiểm thử, sửa lỗi, nâng cấp

Nhược điểm

* **Phụ thuộc vào kết nối mạng:** Nếu mất kết nối, client không thể gửi yêu cầu đến server.
* **Tăng độ trễ:** Dữ liệu phải truyền qua lại giữa client và server qua mạng.
* **Chi phí vận hành:** Cần xây dựng hạ tầng server, xử lý bảo mật, giám sát,…

Mô hình **Client–Server** là một nền tảng kiến trúc quan trọng, đảm bảo hiệu quả trong việc xây dựng các hệ thống web hiện đại. Với khả năng phân tách rõ ràng giữa trình bày và xử lý, mô hình này tạo điều kiện thuận lợi để phát triển, mở rộng và bảo trì hệ thống quản lý sự kiện một cách chuyên nghiệp.

# HIỆN THỰC HÓA NGHIÊN CỨU

## Mô tả bài toán

Thiết kế **website quản lý và bán vé sự kiện** nhằm đáp ứng nhu cầu tìm kiếm, đăng ký, mua vé tham dự sự kiện của người dùng một cách tiện lợi và an toàn. Hệ thống cho phép tổ chức sự kiện đăng thông tin, bán vé online, quản lý người tham gia và tối ưu hóa quy trình vận hành sự kiện. Người dùng cuối có thể dễ dàng tra cứu, chọn lựa và mua vé các sự kiện phù hợp với nhu cầu cá nhân.

Xây dựng một trang web cung cấp giao diện hiện đại, dễ sử dụng và thân thiện với người dùng nhờ được xây dựng bằng **Next.js** cho frontend, kết hợp với **NestJS** cho backend, cùng với **MongoDB** làm cơ sở dữ liệu và **Firebase Storage** để lưu trữ hình ảnh sự kiện.

Hệ thống phân loại đối tượng người dùng theo chức năng gồm có:

* **Ban tổ chức sự kiện (Tổ chức viên)**
* **Khách tham dự (Người dùng)**
* **Quản trị viên hệ thống**

**Quản trị viên hệ thống:** Là người có quyền cao nhất trong hệ thống, có trách nhiệm giám sát và quản lý toàn bộ nền tảng. Các chức năng bao gồm:

* Quản lý người dùng (tổ chức viên, khách tham dự)
* Quản lý danh mục sự kiện (phân loại sự kiện theo chủ đề)
* Quản lý sự kiện (duyệt/từ chối sự kiện)
* Quản lý giao dịch, thanh toán (kiểm tra, thống kê doanh thu, xử lý khiếu nại)
* Quản lý bản tin, đăng tin tức.
* Xem thống kê hệ thống: lượt truy cập, số lượng sự kiện đang hoạt động, doanh thu tổng,…
* Hoàn tiền vé cho sự kiện đã huỷ.

**Ban tổ chức sự kiện (Tổ chức viên):** Là đối tượng tạo và quản lý sự kiện.

* Tạo, xoá sự kiện
* Quản lý số lượng vé, thời gian bán vé
* Xem danh sách người tham dự
* Cập nhật thông tin ban tổ chức
* Xem thống kê doanh thu từ sự kiện, tỷ lệ tham gia
* Quét vé tham gia, xem danh sách quét vé

**Khách tham dự (Người dùng)**

* Xem danh sách, chi tiết sự kiện công khai, bản tin.
* Lọc sự kiện
* Đăng ký tài khoản để mua vé
* Cập nhật thông tin cá nhân
* Chọn vé và thanh toán (VNPay)
* Xem lịch sử đặt vé
* Nhận mã QR qua email, trên hệ thống dùng để check-in tại sự kiện
* Xem tin tức

## Phân tích thiết kế hệ thống

### Đặc tả yêu cầu hệ thống

#### Yêu cầu chức năng

Hệ thống website quản lý và bán vé sự kiện được xây dựng nhằm cung cấp các chức năng đầy đủ, thuận tiện cho người dùng lẫn quản trị viên trong việc tổ chức, mua và quản lý vé. Các yêu cầu chức năng được xác định như sau:

**Chức năng quản lý sự kiện**

* Cho phép tổ chức viên thêm mới sự kiện với đầy đủ thông tin: tên sự kiện, mô tả, thời gian diễn ra, địa điểm tổ chức, hình ảnh minh họa.
* Hỗ trợ phân loại sự kiện theo loại hình (âm nhạc, thể thao...), trạng thái (đang mở bán, sắp diễn ra, đã kết thúc...).
* Cập nhật, xóa sự kiện.

Chức năng tìm kiếm và lọc sự kiện

* Người dùng có thể tìm kiếm sự kiện theo từ khóa (tên sự kiện, địa điểm, tên tổ chức...).
* Hỗ trợ lọc sự kiện theo các tiêu chí: loại hình, thời gian, địa điểm, mức giá.
* Thay đổi trạng thái vé (ngưng bán), thay đổi số lượng vé còn lại
* Xem trình trang bán vé, biểu đồ

Chức năng đặt vé

* Cho phép người dùng chọn loại vé, số lượng vé.
* Hệ thống tự động tính toán tổng tiền và hiển thị chi tiết đơn hàng.
* Người dùng lựa chọn phương thức thanh toán (VNPay).
* Sau khi đặt vé, hệ thống gửi email đơn hàng.

Chức năng người dùng

* Đăng ký tài khoản bằng email, nhận code kiểm tra email tồn tại.
* Đăng nhập/đăng xuất an toàn và nhanh chóng.
* Cập nhật thông tin cá nhân.
* Xem lịch sử mua vé, in hoặc tải lại vé đã mua.

Chức năng quản trị hệ thống

* Quản lý người dùng: phân quyền, khóa tài khoản, khôi phục tài khoản.
* Quản lý sự kiện, danh mục sự kiện, tổ chức viên.
* Thống kê doanh thu theo ngày/tháng/năm.
* Thanh toán doanh thu sự kiện cho ban tổ chức,…
* Quản lý, đăng tải bản tin.
* Hoàn vé cho sự kiện đã huỷ

#### Yêu cầu phi chức năng

Để hệ thống quản lý và bán vé sự kiện vận hành hiệu quả, ổn định, thân thiện với người dùng và có khả năng mở rộng trong tương lai, hệ thống cần đáp ứng đầy đủ các yêu cầu phi chức năng sau:

Hiệu năng (Performance)

* Hệ thống cần có khả năng phản hồi nhanh, đảm bảo thời gian tải trang tối thiểu và trải nghiệm người dùng mượt mà, kể cả trong các trường hợp có nhiều người dùng đồng thời.
* Kiến trúc hệ thống phải được tách biệt rõ ràng giữa frontend và backend nhằm tối ưu hiệu suất xử lý và tăng khả năng mở rộng.
* Các API được triển khai cần đảm bảo tốc độ xử lý nhanh, ổn định và có thể xử lý hiệu quả nhiều truy vấn đồng thời.

**Bảo mật (Security)**

* Hệ thống phải cung cấp cơ chế xác thực và phân quyền rõ ràng, đảm bảo mỗi loại người dùng (người dùng thông thường, tổ chức, quản trị viên) chỉ có thể truy cập vào các chức năng phù hợp với vai trò được cấp.
* Tất cả các API quan trọng cần được bảo vệ bằng cơ chế xác thực an toàn như JWT.
* Dữ liệu nhạy cảm (bao gồm mật khẩu, token) phải được mã hóa và lưu trữ một cách an toàn, tránh rò rỉ thông tin.
* Hệ thống phải áp dụng cơ chế CORS để giới hạn các nguồn truy cập không hợp lệ tới backend.

**Khả năng mở rộng (Scalability)**

* Hệ thống phải được thiết kế theo kiến trúc phân lớp (presentation – business – data access) nhằm dễ dàng mở rộng, tích hợp thêm tính năng mới hoặc tăng quy mô hệ thống khi có nhu cầu.
* Việc triển khai cần đảm bảo khả năng module hóa ở backend và component hóa ở frontend để hỗ trợ mở rộng từng phần độc lập mà không ảnh hưởng đến toàn bộ hệ thống.
* Hệ thống cần sẵn sàng để tích hợp với nhiều dịch vụ bên ngoài hoặc phục vụ lượng lớn người dùng trong tương lai.

**Dễ bảo trì (Maintainability)**

* Mã nguồn cần được tổ chức rõ ràng, tuân thủ các nguyên tắc lập trình chuẩn (như SOLID, DRY) để thuận tiện trong việc bảo trì, nâng cấp và kiểm thử sau này.
* Các phần logic nghiệp vụ, giao diện người dùng và xử lý dữ liệu cần được tách biệt rõ ràng để đảm bảo tính độc lập giữa các thành phần.
* Cấu trúc hệ thống phải hỗ trợ dễ dàng thay đổi một phần mà không ảnh hưởng đến toàn bộ hệ thống.

Khả năng tái sử dụng (Reusability)

* Các thành phần frontend (component), backend (service, module) phải được thiết kế sao cho có thể tái sử dụng trong nhiều màn hình hoặc tính năng khác nhau.
* Mã nguồn nên có tính module cao, dễ dàng chia sẻ hoặc tích hợp vào các dự án tương tự khác trong tương lai.

**Khả năng tích hợp (Integrability)**

* Hệ thống cần cung cấp các API theo chuẩn RESTful để dễ dàng tích hợp với các dịch vụ bên ngoài như hệ thống thanh toán (VNPay), email, chatbot AI, hoặc ứng dụng di động.
* Backend phải hoạt động độc lập, cho phép dễ dàng kết nối với các giao diện frontend khác nhau (web, mobile) mà không cần thay đổi logic nghiệp vụ cốt lõi.

**Thân thiện với người dùng (Usability)**

* Giao diện người dùng cần đơn giản, dễ sử dụng, hỗ trợ tìm kiếm, lọc sự kiện, và thao tác đặt vé nhanh chóng.
* Hệ thống phải cung cấp các thông báo trực quan (notification/toast) nhằm xác nhận các thao tác quan trọng của người dùng, ví dụ: đặt vé, thanh toán, hủy bỏ.
* Giao diện cần hiển thị tốt trên nhiều loại thiết bị (PC, tablet, mobile).

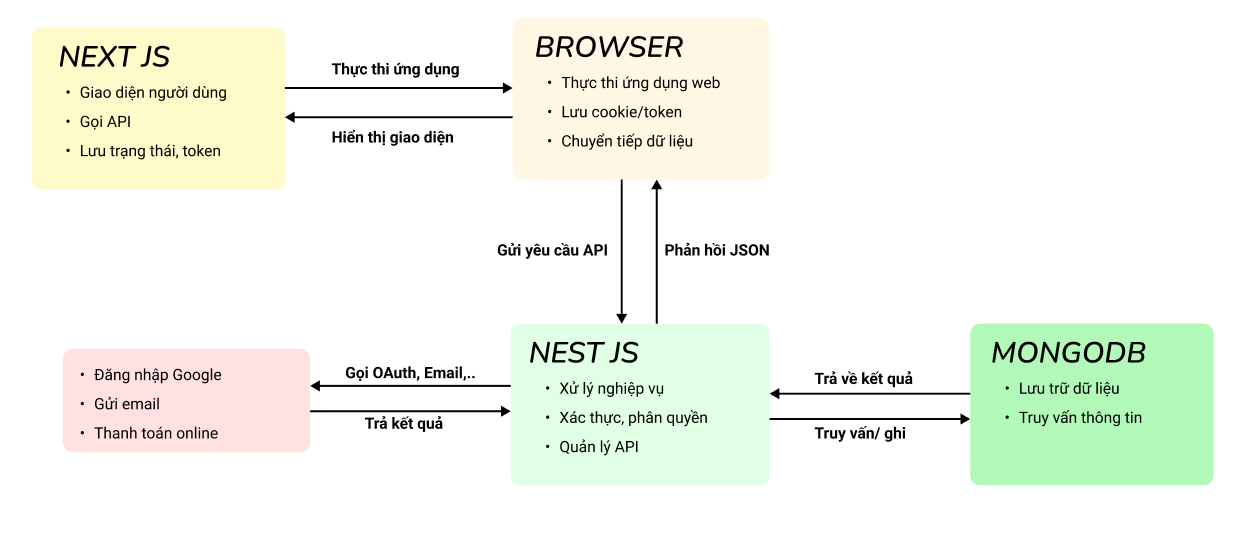
**Khả năng kiểm thử (Testability)**

* Hệ thống cần được thiết kế theo hướng module hóa để đảm bảo khả năng kiểm thử độc lập từng thành phần.
* Cần có khả năng viết và thực thi các test case tự động (unit test, integration test) cho cả backend và frontend, nhằm phát hiện lỗi sớm và đảm bảo độ ổn định của hệ thống.

Khả năng phục hồi lỗi (Reliability)

* Hệ thống phải xử lý lỗi một cách rõ ràng, cung cấp thông báo dễ hiểu và phù hợp đến người dùng cuối trong trường hợp phát sinh sự cố.
* Các lỗi hệ thống cần được ghi log chi tiết để phục vụ quá trình giám sát và xử lý sau này.
* Hệ thống cần đảm bảo hoạt động ổn định liên tục, hạn chế tối đa việc gián đoạn dịch vụ.

### Kiến trúc hệ thống



Hình Kiến trúc hệ thống Client-Server

Hệ thống được chia làm 2 phần chính:

Client Side (Frontend)

NextJS - Giao diện người dùng

* Là giao diện chính tương tác với người dùng.
* Hiển thị thông tin động (server-side rendering hoặc client-side rendering).
* Gửi yêu cầu API đến backend.
* Lưu trữ trạng thái cục bộ và token ( bằng useState, useContext).

Browser

* Thực thi ứng dụng web.
* Lưu trữ các token/cookie phục vụ cho xác thực.
* Đóng vai trò trung gian chuyển tiếp dữ liệu giữa NextJS và NestJS backend.
* Lưu trữ JWT token bằng Cookies

Server Side (Backend)

NestJS

* Framework chính xử lý logic phía server.
* Chịu trách nhiệm nhận yêu cầu từ frontend (qua browser), xác thực, xử lý nghiệp vụ và trả phản hồi.
* **Xử lý nghiệp vụ:** Thêm, sửa, xoá dữ liệu người dùng, sự kiện, đơn hàng,...
* **Xác thực & phân quyền:** Qua JWT token, middleware và decorators.
* **Tích hợp OAuth, email, thanh toán online:** Kết nối đến Google login, dịch vụ SMTP, các cổng thanh toán.

**Database Layer**

MongoDB

* Cơ sở dữ liệu NoSQL chính của hệ thống.
* Lưu trữ các thông tin như người dùng, sự kiện, vé, giao dịch,...
* Tương tác qua các module ORM/ODM của NestJS (ví dụ: Mongoose).
* Hỗ trợ truy vấn linh hoạt và mở rộng tốt.

Luồn xử lý chính:

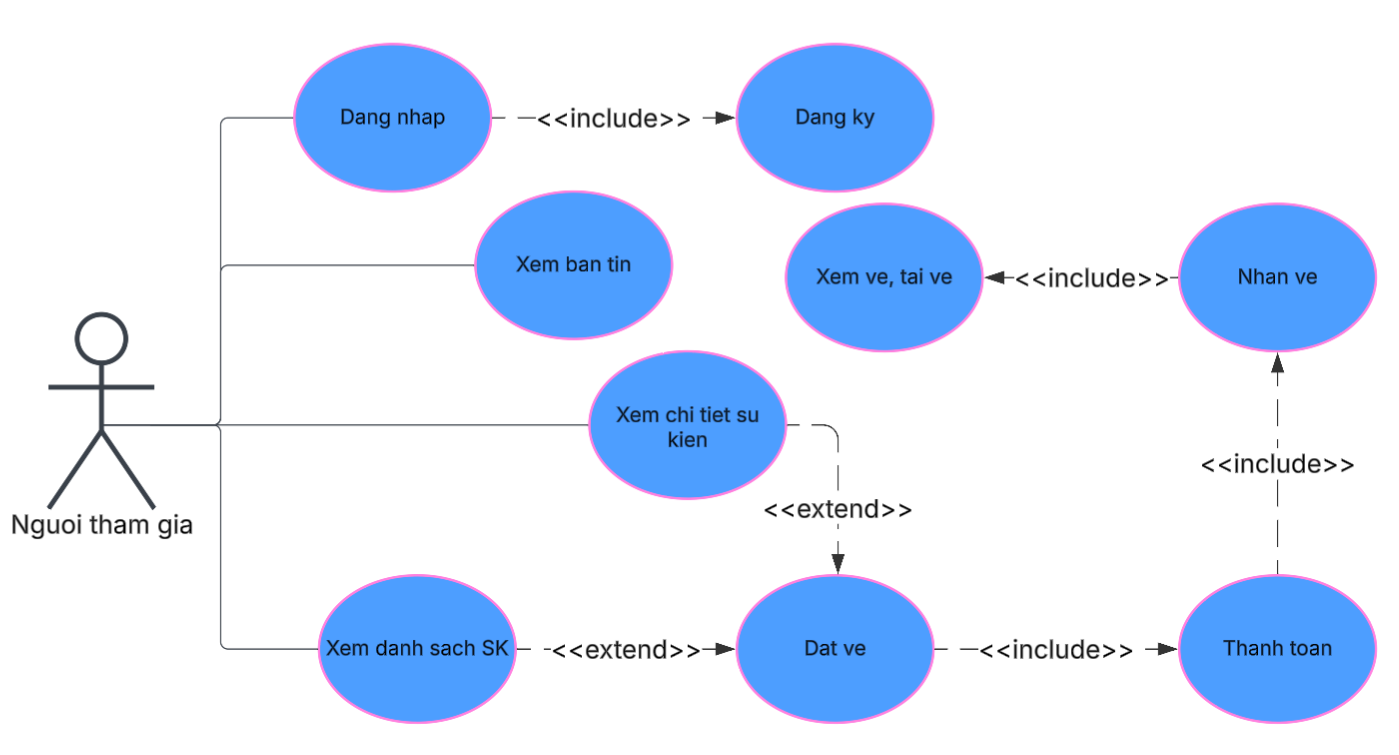
1. **Người dùng** tương tác với giao diện NextJS thông qua **browser**.
2. **NextJS** gửi yêu cầu API (dạng JSON) đến **NestJS backend**.
3. **NestJS** xác thực người dùng qua token (JWT).
4. Sau xác thực, NestJS thực hiện logic cần thiết, có thể:

* Gọi đến cơ sở dữ liệu MongoDB để lấy hoặc ghi thông tin.
* Gọi đến Firebase để lưu hoặc lấy ảnh.
* Gửi email xác thực hoặc thanh toán.

1. Kết quả từ NestJS được trả về browser → giao diện NextJS hiển thị lại thông tin.
2. Token xác thực được lưu trữ trong browser phục vụ cho các request tiếp theo.

### Sơ đồ use case

#### Sơ đồ use case Người dùng (Người tham gia sự kiện)



Hình Sơ đồ use case Người dùng

Mô tả sơ đồ use case Người dùng: Hệ thống cho phép người dùng đăng ký, đăng nhập, tìm kiếm sự kiện, mua vé và quản lý vé đã mua.

Tác nhân (Actor): Người tham gia tác nhân chính tương tác với hệ thống.

Mô tả Use Case:

Đăng nhập (Dang nhap)

* Mô tả: Cho phép người dùng đã có tài khoản truy cập hệ thống.
* Quan hệ:<<include>> Đăng ký: Hỗ trợ người dùng tạo tài khoản mới.

Đăng ký (Dang ky)

* Mô tả: Cho phép người dùng mới tạo tài khoản.
* Quan hệ: Được gọi từ "Đăng nhập".

Xem bản tin (Xem ban tin)

* Mô tả: Hiển thị các tin tức và thông tin cập nhật về sự kiện.

Xem danh sách sự kiện (Xem danh sach SK)

* Mô tả: Hiển thị danh sách các sự kiện đang hoặc sắp diễn ra.
* Quan hệ: <<extend>> Đặt vé: Cho phép người dùng đặt vé trực tiếp từ danh sách.

Xem chi tiết sự kiện (Xem chi tiet su kien)

* Mô tả: Cung cấp thông tin chi tiết về một sự kiện cụ thể.
* Quan hệ: <<extend>> Đặt vé: Cho phép người dùng đặt vé sau khi xem chi tiết.

Đặt vé (Dat ve)

* Mô tả: Cho phép người dùng chọn và mua vé cho một sự kiện.
* Quan hệ: <<include>> Thanh toán: Yêu cầu người dùng thanh toán để hoàn tất giao dịch.

Thanh toán (Thanh toan)

* Mô tả: Xử lý quá trình thanh toán cho đơn hàng đặt vé.
* Quan hệ: <<include>> Nhận vé: Sau khi thanh toán thành công, người dùng sẽ được nhận vé.

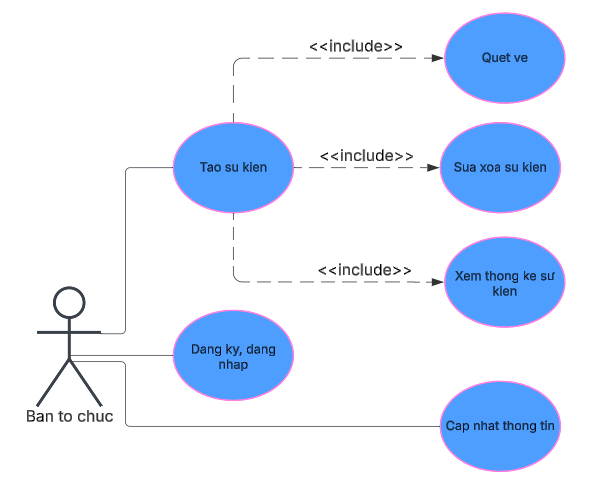
Nhận vé (Nhan ve)

* Mô tả: Tự động gửi vé điện tử đến người dùng sau khi thanh toán.
* Quan hệ: <<include>> Xem vé, tải vé: Cho phép người dùng xem hoặc tải vé đã nhận.

Xem vé, tải vé (Xem ve, tai ve)

* Mô tả: Cho phép người dùng xem lại hoặc tải về các vé đã mua.
* Quan hệ: <<include>> từ "Nhận vé".

#### Sơ đồ use case Ban tổ chức



Hình Sơ đồ use case Ban tổ chức

Mô tả use case Ban tổ chức: Hệ thống cung cấp các chức năng quản lý sự kiện cho ban tổ chức, bao gồm tạo, cập nhật, xóa sự kiện, quản lý vé và xem thống kê.

Ban tổ chức (Ban to chuc): Đây là tác nhân chính chịu trách nhiệm quản lý nội dung và các hoạt động của sự kiện.

Đăng ký, đăng nhập (Dang ky, dang nhap)

* Mô tả: Cho phép Ban tổ chức tạo tài khoản mới và truy cập vào hệ thống quản lý.
* Quan hệ:
* <<include>> Tạo sự kiện: Sau khi đăng nhập thành công, Ban tổ chức có thể thực hiện chức năng tạo sự kiện.
* <<include>> Cập nhật thông tin: Sau khi đăng nhập, Ban tổ chức có thể xem và cập nhật thông tin cá nhân hoặc thông tin tổ chức.

Cập nhật thông tin (Cap nhat thong tin)

* Mô tả: Cho phép Ban tổ chức chỉnh sửa các thông tin liên quan đến tài khoản cá nhân hoặc tổ chức của họ.
* Quan hệ: Được gọi từ use case "Đăng ký, đăng nhập".

Tạo sự kiện (Tao su kien)

* Mô tả: Cho phép Ban tổ chức tạo một sự kiện mới trên hệ thống.
* Quan hệ: <<include>> Đăng, xóa, sửa sự kiện: Sau khi tạo, sự kiện đó sẽ được quản lý thông qua chức năng "Đăng, xóa, sửa sự kiện".

Đăng, xóa, sửa sự kiện (Dang, xoa, sua su kien)

* Mô tả: Cho phép Ban tổ chức quản lý trạng thái và nội dung của các sự kiện đã tạo.
* Quan hệ:
* <<include>> Xem thống kê sự kiện: Cho phép Ban tổ chức xem các số liệu thống kê liên quan đến sự kiện này.
* <<include>> Quét vé: Sau khi sự kiện được công bố, chức năng quét vé sẽ được sử dụng để kiểm soát người tham gia.

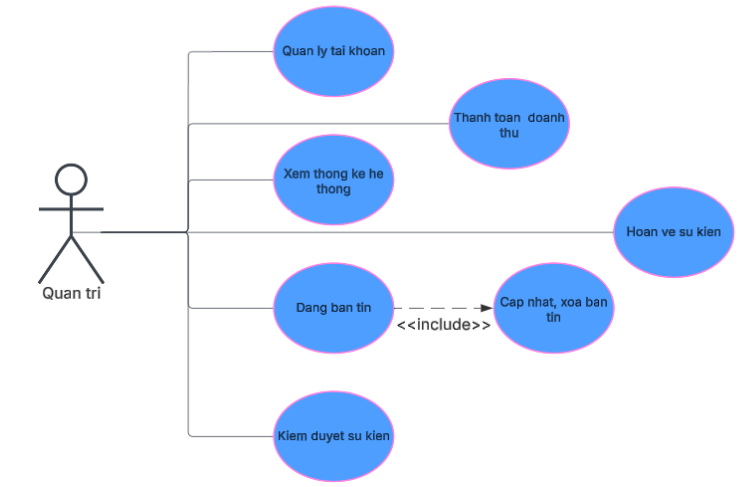
Quét vé (Quet ve)

* Mô tả: Cho phép Ban tổ chức kiểm tra và xác thực vé của người tham dự tại địa điểm sự kiện.
* Quan hệ: <<include>> từ "Đăng, xóa, sửa sự kiện".

Xem thống kê sự kiện (Xem thong ke su kien)

* Mô tả: Cung cấp các số liệu thống kê về sự kiện cho Ban tổ chức.
* Quan hệ: <<include>> từ "Đăng, xóa, sửa sự kiện".

#### Sơ đồ use case Quản trị



Hình .Sơ đồ use case Quản trị

Mô tả use case Quản trị viên: Hệ thống cung cấp các chức năng quản lý tổng quát cho người quản trị, bao gồm quản lý tài khoản người dùng, quản lý doanh thu, quản lý nội dung và kiểm duyệt sự kiện.

Quản trị: Tác nhân có quyền cao nhất trong hệ thống, chịu trách nhiệm quản lý và vận hành toàn bộ nền tảng.

Quản lý tài khoản (Quan ly tai khoan)

* Mô tả: Cho phép người quản trị xem, tìm kiếm, khóa hoặc mở khóa tài khoản của người dùng (bao gồm cả "Người tham gia" và "Ban tổ chức").

Xem thống kê hệ thống (Xem thong ke he thong)

* Mô tả: Cung cấp cái nhìn tổng quan về tình trạng hoạt động của hệ thống thông qua các số liệu thống kê.
* Quan hệ: <<include>> Thanh toán doanh thu: Dựa trên số liệu thống kê, quản trị có thể thực hiện thanh toán cho Ban tổ chức.

Thanh toán doanh thu (Thanh toan doanh thu)

* Mô tả: Cho phép người quản trị xử lý việc thanh toán doanh thu từ việc bán vé cho các Ban tổ chức.
* Quan hệ: <<include>> từ use case "Xem thống kê hệ thống".

Đăng bản tin (Dang ban tin)

* Mô tả: Cho phép người quản trị tạo và đăng tải các bản tin, thông báo lên hệ thống.
* Quan hệ: <<include>> Cập nhật, xóa bản tin: Sau khi bản tin được đăng, quản trị có thể chỉnh sửa hoặc xóa nó.

Cập nhật, xóa bản tin (Cap nhat, xoa ban tin)

* Mô tả: Cho phép người quản trị chỉnh sửa nội dung hoặc gỡ bỏ các bản tin đã được đăng.
* Quan hệ: <<include>> từ use case "Đăng bản tin".

Kiểm duyệt sự kiện (Kiem duyêt su kien)

* Mô tả: Cho phép người quản trị xem xét và phê duyệt các sự kiện do Ban tổ chức đăng tải trước khi chúng được công khai cho người dùng.

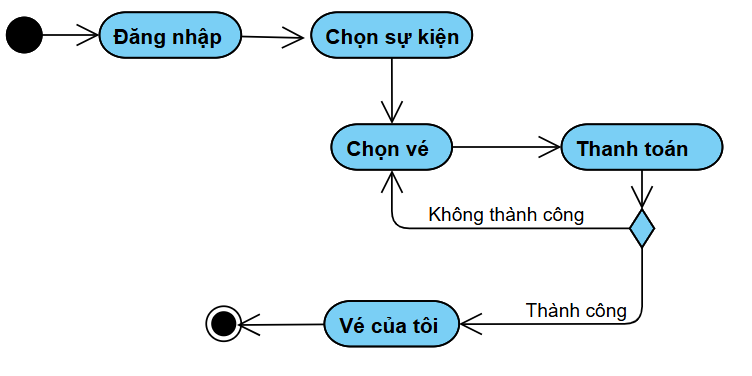
### Sơ đồ lớp



Hình Sơ đồ lớp

### Sơ đồ hoạt động

#### Sơ đồ hoạt động đặt vé



Hình Sơ đồ hoạt động đặt vé

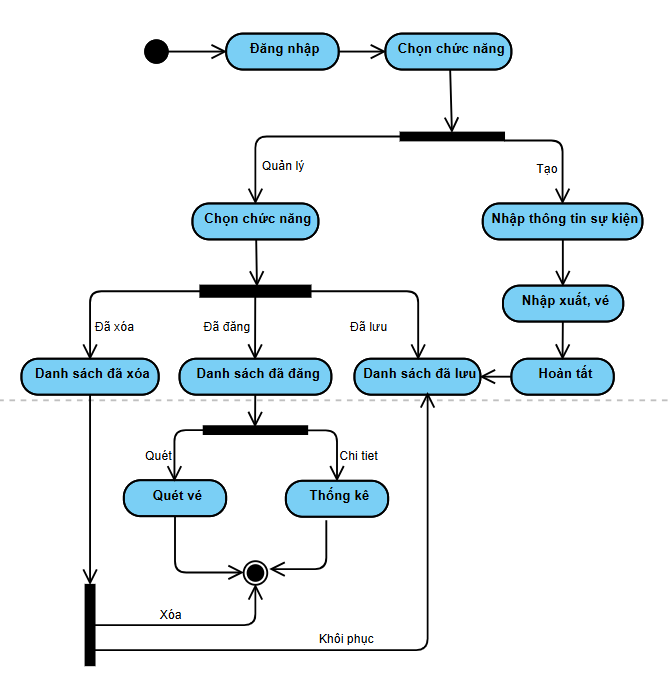
Quy trình mua vé sự kiện bắt đầu từ việc đăng nhập của người dùng. Sau đó, người dùng sẽ lần lượt chọn sự kiện và chọn vé với số lượng mong muốn. Một bước kiểm tra tự động được thực hiện để xác minh trạng thái đăng nhập của người dùng:

* Nếu chưa đăng nhập, người dùng sẽ được yêu cầu quay lại bước đầu tiên.
* Nếu đã đăng nhập, họ sẽ được chuyển đến bước thanh toán.

Quá trình thanh toán có thể dẫn đến hai kết quả:

* Thành công: Người dùng nhận được thông báo xác nhận và vé sẽ được hiển thị trong mục "Vé của tôi", hoàn tất quá trình.
* Không thành công: Hệ thống sẽ báo lỗi và cho phép người dùng quay lại bước thanh toán để thử lại.

#### Sơ đồ hoạt động quản lý sự kiện

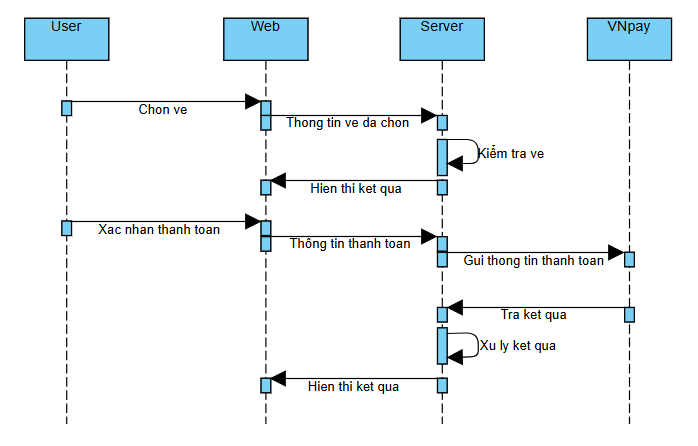


Hình Sơ đồ hoạt động quản lý sự kiện

* **Khởi động Đăng nhập:**
* Ban Tổ chức bắt đầu bằng việc **đăng nhập** vào hệ thống.
* Đây là bước xác thực thông tin nhằm đảm bảo chỉ người dùng hợp lệ mới được phép sử dụng chức năng quản lý sự kiện.
* **Chọn chức năng chính**
* Sau khi đăng nhập, hệ thống yêu cầu tác nhân **chọn chức năng**.
* Có hai nhánh xử lý chính:
* **Tạo sự kiện mới** phục vụ nhu cầu lập kế hoạch và công bố sự kiện.
* **Quản lý sự kiện** xử lý các sự kiện đã tồn tại trong hệ thống.
* **Nhánh Tạo sự kiện**
* **Nhập thông tin sự kiện**: Ban Tổ chức cung cấp dữ liệu cơ bản như tên, địa điểm, thời gian, mô tả.
* **Nhập xuất vé**: hệ thống cho phép thiết lập số lượng vé, phân loại vé theo tên, giá vé và các quy định liên quan.
* **Hoàn tất**: sau khi nhập đầy đủ dữ liệu, sự kiện được lưu vào **Danh sách đã lưu**. Đây là trạng thái nháp, sự kiện chưa được công khai để đảm bảo tính linh hoạt trong chỉnh sửa.
* **Nhánh Quản lý sự kiện**
* Trong nhánh quản lý, hệ thống chia sự kiện thành ba trạng thái riêng biệt:
* **Danh sách đã lưu**: chứa các sự kiện nháp, cho phép chỉnh sửa hoặc công khai.
* **Danh sách đã đăng**: sự kiện đã công bố, đang được bán vé và theo dõi.
* **Danh sách đã xóa**: sự kiện không còn hiệu lực nhưng có thể khôi phục nếu cần.
* Tại **Danh sách đã đăng**, Ban Tổ chức có thể thực hiện hai hoạt động song song:
* **Quét vé**: sử dụng mã QR để xác thực vé và kiểm tra người tham dự tại cổng sự kiện.
* **Thống kê**: tổng hợp và phân tích dữ liệu bán vé, số lượng người tham gia, nhằm hỗ trợ công tác quản lý và đánh giá hiệu quả sự kiện.
* **Chuyển đổi trạng thái sự kiện**
* Hệ thống cho phép sự kiện thay đổi trạng thái trong vòng đời quản lý:
* **Xóa**: chuyển sự kiện từ “đã lưu” hoặc “đã đăng” sang “đã xóa”.
* **Khôi phục**: đưa sự kiện từ “đã xóa” trở về trạng thái ban đầu.
* Cơ chế này giúp Ban Tổ chức kiểm soát linh hoạt hơn, tránh mất dữ liệu và có thể phục hồi khi cần.

### Sơ đồ tuần tự

#### Sơ đồ tuần tự đặt vé



Hình Sơ đồ tuần tự đặt vé

Sơ đồ tuần tự mô tả trình tự tương tác giữa các đối tượng **User**, **Web**, **Server**, **Database** và **VNPay** trong quá trình người dùng thực hiện đặt vé sự kiện. Quy trình diễn ra như sau:

**Chọn sự kiện và vé**

* Người dùng chọn sự kiện và vé muốn mua trên giao diện web.
* Web gửi yêu cầu lấy thông tin vé đến Server.
* Server kiểm tra tình trạng vé trong Database.
* Database trả kết quả (còn vé hoặc hết vé) về cho Server.
* Server phản hồi kết quả cho Web, hiển thị thông tin vé cho người dùng.

**Thanh toán vé**

* Khi người dùng tiến hành thanh toán, Web gửi thông tin thanh toán đến Server.
* Server tạo yêu cầu thanh toán và gửi đến hệ thống VNPay.
* VNPay xử lý giao dịch và trả kết quả về Server. Server lưu thông tin giao dịch và vé vào Database, sau đó gửi thông báo thanh toán thành công cho Web.
* Sau khi thanh toán thành công, Web chuyển người dùng đến trang “Vé của tôi” để xem thông tin vé đã mua.

#### Sơ đồ tuần tự tạo sự kiện



Hình Sơ đồ tuần tự tạo sự kiện

Sơ đồ tuần tự minh họa các bước tương tác giữa **Ban tổ chức**, **giao diện Web**, **Server** và **Cơ sở dữ liệu** trong quá trình tạo một sự kiện mới. Quy trình diễn ra như sau:

**Tạo sự kiện**:

* Ban tổ chức chọn chức năng “Tạo sự kiện” và nhập các thông tin cần thiết (tên sự kiện, thời gian, địa điểm, mô tả, số lượng vé…).
* Thông tin này được Web tạm lưu ở form tạo sự kiện cho đến khi người dùng xác nhận.

**Xác nhận tạo sự kiện**:

* Khi Ban tổ chức xác nhận, Web gửi toàn bộ dữ liệu sự kiện đến Server.
* Server tiến hành lưu dữ liệu sự kiện vào Cơ sở dữ liệu.
* Cơ sở dữ liệu phản hồi kết quả lưu trữ cho Server.
* Server gửi thông báo kết quả (thành công hoặc thất bại) về Web để hiển thị cho Ban tổ chức.

#### Sơ đồ tuần tự quét vé



Hình Sơ đồ tuần tự quét vé

Sơ đồ tuần tự mô tả quá trình **Ban tổ chức** đăng nhập, lựa chọn sự kiện, và thực hiện quét mã QR để kiểm tra vé của khách tham dự. Quy trình như sau:

**Chọn sự kiện cần quét vé**

* Ban tổ chức yêu cầu danh sách sự kiện đã đăng → Web gửi yêu cầu đến Server.
* Server lọc dữ liệu từ Database và trả kết quả về Web.
* Ban tổ chức tiếp tục yêu cầu danh sách sự kiện đang diễn ra để chọn đúng sự kiện cần quét vé.
* Server truy vấn và trả lại danh sách để hiển thị.

**Mở trang quét vé**

* Sau khi chọn sự kiện, Web hiển thị giao diện quét vé (camera hoặc nhập mã).

**Quét và kiểm tra vé**

* Ban tổ chức quét mã QR → Web gửi mã đến Server.
* Server giải mã QR, kiểm tra tính hợp lệ của vé trong Database.
* Nếu hợp lệ, Server cập nhật trạng thái vé, thời gian quét và trả kết quả “Thành công” về Web.
* Web hiển thị kết quả kiểm tra vé cho Ban tổ chức.

## Thiết kế dữ liệu

### [Danh sách các thực thể](https://docs.google.com/document/d/1I7ijQi7qoBhPV5U7vI7hOsZrXh5ROiXH/edit#heading=h.2xcytpi)

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên thực thể | Diễn giải | Ghi chú |
| 1 | account | Tài khoản | Lưu trữ thông tin về tài khoản |
| 2 | user | Người tham gia | Lưu trữ thông tin về người dùng mua vé sự kiện |
| 3 | organizer | Ban tổ chức | Lưu trữ thông tin về ban tổ chức sự kiện |
| 4 | event | Sự kiện | Lưu trữ thông tin về sự kiện |
| 5 | session | Xuất tổ chức | Lưu trữ thông tin về xuất tổ chức sự kiện |
| 6 | ticket | Vé sự kiện | Lưu trữ thông tin về vế sự kiện |
| 7 | order | Đơn mua vé | Lưu trữ thông tin về đơn mua vé |
| 8 | new | Tin tức | Lưu trữ thông tin về tin tức |
| 9 | ticket\_order | Vé sự kiện được mua | Lưu trữ thông tin về vé đã mua |

Bảng Bảng danh sách thực thể

### Chi tiết thực thể

Thực thể **accouunt**: Lưu thông tin về tài khoản để đăng nhập

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã tài khoản | ObjectId | Khóa chinh |
| 2 | email | Email tài khoản | varchar(255) | Khóa phụ |
| 3 | password | Mật khẩu tài khoản | varchar(255) |  |
| 4 | role | Quyền tài khoản | enum |  |
| 5 | status | Trạng thái của tài khoản | enum |  |

Bảng Bảng thực thể Tài khoản

Thực thể **user**: Lưu thông tin về người dùng (người tham gia)

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã người dùng | ObjectId | Khóa chinh |
| 2 | account\_id | Mã tài khoản liên kết | ObjectId | Khóa ngoại |
| 3 | name | Họ tên người dùng | varchar(255) |  |
| 4 | phone | Số điện thoại người dùng | varchar(20) |  |
| 5 | email | Email người dùng | varchar(255) | Khóa phụ |
| 6 | avatar | Ảnh người dùng | varchar(255) |  |
| 7 | province | Tỉnh của người dùng ở | varchar(100) |  |
| 8 | district | Huyện của người dùng ở | varchar(100) |  |
| 9 | ward | Xã của người dùng ở | varchar(100) |  |
| 10 | address | Ấp của người dùng ở | varchar(255) |  |

Bảng Thực thể Người dùng

Thực thể **organizer**: Lưu thông tin về Ban tổ chức

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã ban tổ chức | ObjectId | Khóa chinh |
| 2 | account\_id | Mã tài khoản liên kết | ObjectId | Khóa ngoại |
| 3 | name | Tên ban tổ chức | varchar(255) |  |
| 4 | email | Email ban tổ chức | varchar(255) | Khóa phụ |
| 5 | description | Giới thiệu ban tổ chức | text |  |
| 6 | weblink | Đường dẫn trang web | varchar(255) |  |
| 7 | phone | Số điện thoại | varchar(20) |  |
| 8 | address | Địa chỉ | varchar(255) |  |
| 9 | social\_link | Đường dẫn đến trang xã hội | varchar(255) |  |
| 10 | logo | Logo ban tổ chức | varchar(255) |  |
| 11 | bank\_name | Tên ngân hàng | varchar(100) |  |
| 12 | bank\_number | Số tài khoản ngân hàng | varchar(50) |  |
| 13 | bank\_holder | Tên chủ tài khoản | varchar(255) |  |

Bảng Thực thể Ban tổ chức

Thực thể **envent**: Lưu thông tin về Sư kiện

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã sự kiện | ObjectId | Khóa chinh |
| 2 | organizer\_id | Mã ban tổ chức đã tạo sự kiện | ObjectId | Khóa ngoại |
| 3 | title | Tiêu đề sự kiện | varchar(255) |  |
| 4 | description | Giới thiệu sự kiện | text |  |
| 5 | location\_houseNumber | Địa chỉ nhà tổ chức | varchar(255) |  |
| 6 | location\_ward | Địa chỉ xã nơi tổ chức | varchar(100) |  |
| 7 | location\_district | Địa chỉ huyện nơi tổ chức | varchar(100) |  |
| 8 | location\_pronvice | Địa chỉ tỉnh nơi tổ chức | varchar(100) |  |
| 9 | image | Hình ảnh sự kiện | varchar(255) |  |
| 10 | status | Trạng thái sự kiện | enum |  |
| 11 | event\_type | Loại sự kiện | enum |  |
| 12 | paytime | Thời gian thanh toán doanh thu cho sự kiện | date |  |

Bảng Thực thể Vé

Thực thể **session** : Lưu thông tin về Xuất hoạt động của sự kiện

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã xuất | ObjectId | Khóa chinh |
| 2 | event\_id | Mã sự kiện | ObjectId | Khóa ngoại |
| 3 | start\_time | Thời gian bắt đầu | datetime |  |
| 4 | end\_time | Thời gian kết thúc | datetime |  |
| 5 | status | Trạng thái | enum |  |

Bảng Thực thể Xuất sự kiện

Thực thể **ticket**: Lưu thông tin về Vé của sự kiện

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã vé | ObjectId | Khóa chinh |
| 2 | session\_id | Mã xuất của vé | ObjectId | Khóa ngoại |
| 3 | ticket\_name | Tên vé | varchar(255) |  |
| 4 | ticket\_price | Giá vé | decimal |  |
| 5 | ticket\_quantity | Số lượng vé | int |  |
| 6 | min\_per\_order | Số lượng tối thiểu một lần mua | int |  |
| 7 | max\_per\_order | Số lượng tôi đa một lần mua | int |  |
| 8 | description\_ticket | Mô tả vé | text |  |
| 9 | sold\_quantity | Số lượng vé đã bán | int |  |

Bảng Thực thể Vé sự kiện

Thực thể **order**: Lưu thông tin về Đơn mua vé

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã đơn hàng | ObjectId | Khóa chinh |
| 2 | user\_id | Mã người mua vé | ObjectId | Khóa ngoại |
| 3 | total\_amount | Giá trị đơn hàng | decimal |  |
| 4 | status | Trạng thái đơn | enum |  |
| 5 | vnp\_TxnRef | Mã định danh thanh toán | varchar(50) |  |
| 6 | vpn\_Reponse | Kết quả trả về từ thanh toán | json |  |
| 7 | Payment\_method | Loại thanh toán | enum |  |

Bảng Thực thể Đơn vé

Thực thể **order\_ticket**: Lưu thông tin về Mỗi vé trong đơn mua

Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã vé trong đơn mua | ObjectId | Khóa chinh |
| 2 | order\_id | Mã đơn hàng | ObjectId | Khóa ngoại |
| 3 | ticket\_id | Mã vé | ObjectId | Khóa ngoại |
| 4 | session\_id | Mã xuất | ObjectId | Khóa ngoại |
| 5 | price | Giá vé | decimal |  |
| 6 | qr\_code | Mã Qr code để quét vào cổng | varchar(255) |  |
| 7 | status | Trạng thái | enum |  |
| 8 | check\_in\_time | Thời gian quét vé | datetime |  |

Bảng Thực thể Vé đã đơn hàng

Thực thể **new**: Lưu thông tin về Bản tin

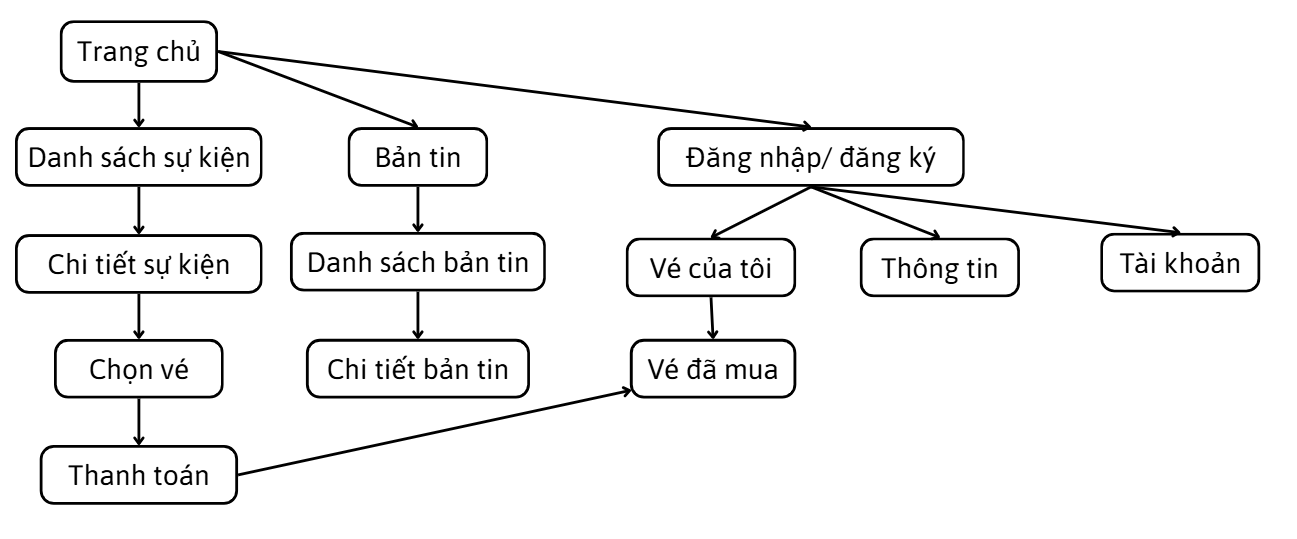
Chi tiết thực thể:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Thuộc tính | Diễn giải | Kiểu dữ liệu | Ràng buộc |
| 1 | id | Mã tin tức | ObjectId | Khóa chinh |
| 2 | title | Tiêu đề tin tức | varchar(255) |  |
| 3 | image | Ảnh tin tức | varchar(255) |  |
| 4 | content | Nội dung tin tức | text |  |

Bảng Thực thể Bản tin

## Thiết kế giao diện

### Sơ đồ website người dùng cuối



Hình Sơ đồ Website người dùng cuối

### Giao diện website cho người dùng

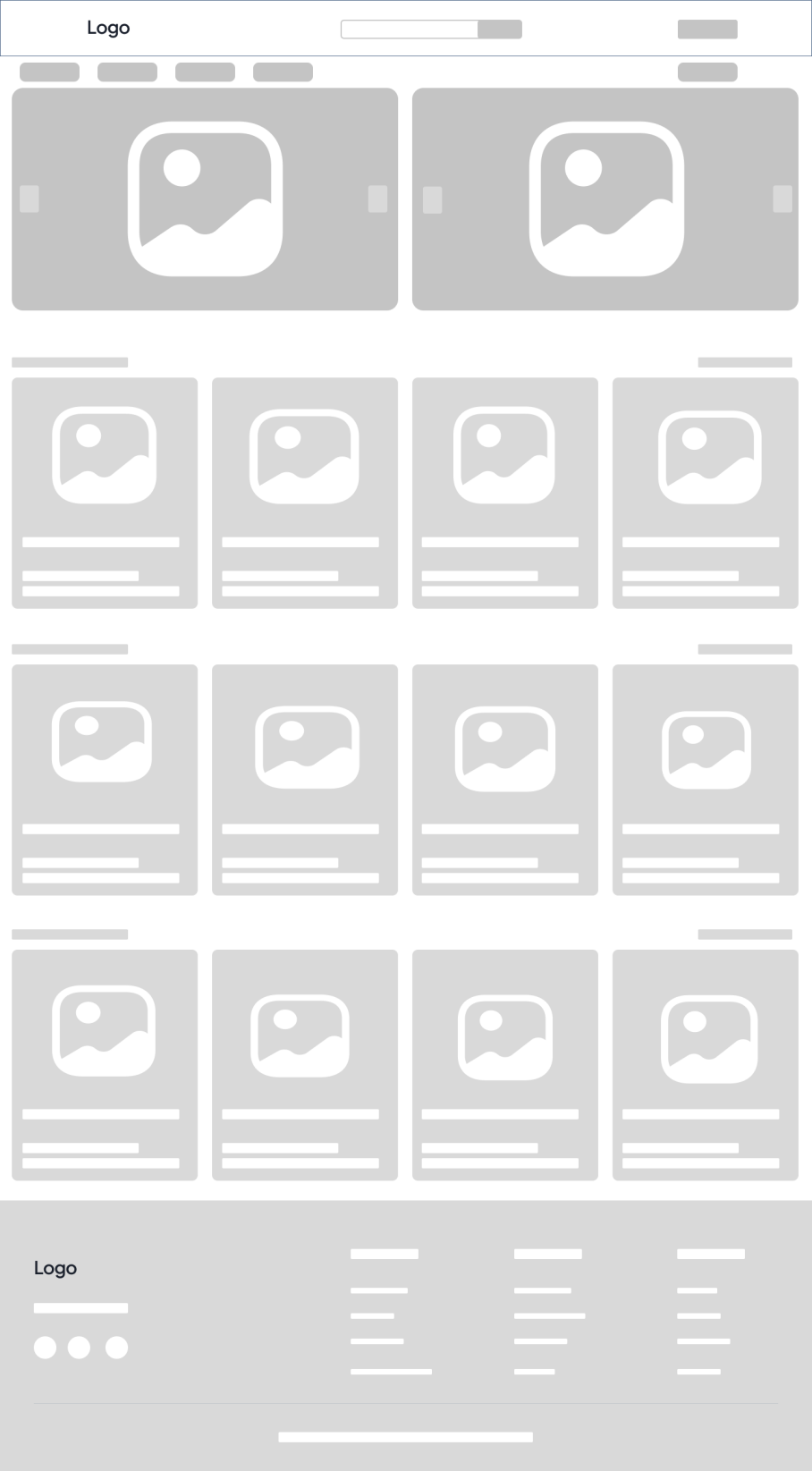
#### Trang chủ

Trang chủ của hệ thống được thiết kế với bố cục trực quan và thân thiện với người dùng, nhằm mang lại trải nghiệm thuận tiện ngay từ lần truy cập đầu tiên.

Phần đầu trang (header) hiển thị logo, thanh tìm kiếm cùng các nút điều hướng chính như đăng nhập, đăng ký và giỏ hàng, giúp người dùng dễ dàng thao tác. Ngay bên dưới là khu vực banner dạng trượt (slider) với hình ảnh lớn, dùng để giới thiệu các sự kiện nổi bật hoặc chương trình khuyến mãi, tạo điểm nhấn thu hút sự chú ý.

Nội dung chính của trang được chia thành nhiều khối sự kiện theo dạng lưới, mỗi khối bao gồm hình ảnh minh họa, tiêu đề và thông tin cơ bản, cho phép người dùng nhanh chóng nắm bắt và lựa chọn sự kiện quan tâm. Các sự kiện có thể được phân loại theo nhóm như sự kiện sắp diễn ra, sự kiện phổ biến hay gợi ý phù hợp.

Phần cuối trang (footer) bao gồm logo, liên kết mạng xã hội, các trang hỗ trợ như liên hệ, giới thiệu, điều khoản dịch vụ và chính sách bảo mật, đảm bảo cung cấp đầy đủ thông tin và tăng tính chuyên nghiệp cho website.



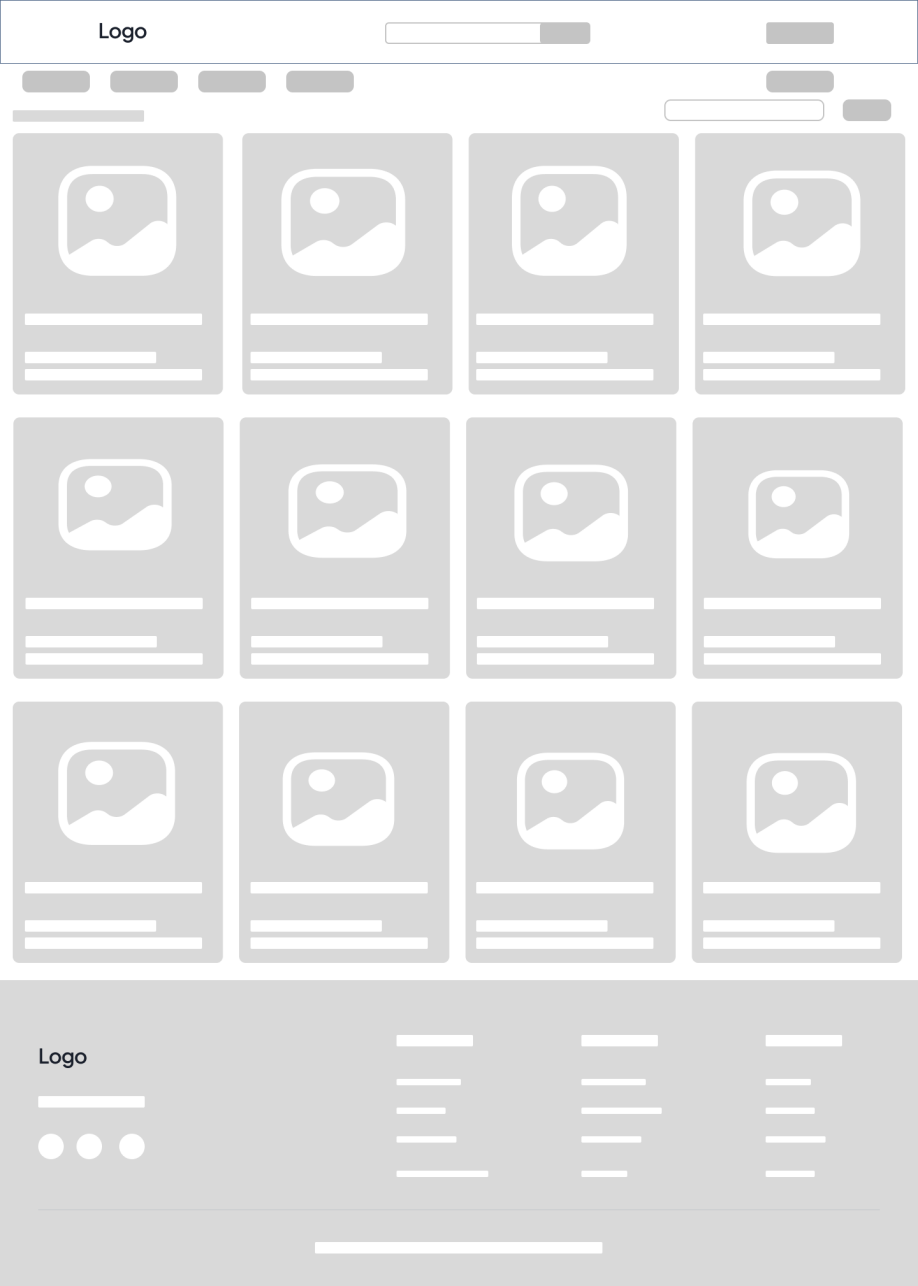
Hình Lo-fi prototype Trang chủ

#### Trang danh sách sự kiện

Trang danh sách sự kiện được xây dựng với mục tiêu giúp người dùng dễ dàng tìm kiếm và theo dõi các sự kiện đang diễn ra hoặc sắp tổ chức.

Phần đầu trang (header) vẫn giữ cấu trúc thống nhất với logo, thanh tìm kiếm và các nút điều hướng cơ bản nhằm đảm bảo trải nghiệm liền mạch giữa các trang. Ngay bên dưới là khu vực lọc và phân loại, cho phép người dùng sắp xếp sự kiện theo nhu cầu như theo thời gian, loại hình hoặc mức độ phổ biến.

Nội dung chính của trang được trình bày dưới dạng lưới nhiều cột, trong đó mỗi thẻ sự kiện hiển thị hình ảnh đại diện, tiêu đề và một số thông tin cơ bản, giúp người dùng có cái nhìn tổng quan và nhanh chóng đưa ra lựa chọn. Phần cuối trang (footer) tiếp tục cung cấp các liên kết hỗ trợ và thông tin liên hệ, đảm bảo tính đồng bộ trong toàn bộ hệ thống.



Hình Lo-fi prototype Trang danh sách ve

#### Trang chi tiết sự kiện

Trang chi tiết sự kiện được thiết kế nhằm cung cấp đầy đủ thông tin cho người dùng trước khi đưa ra quyết định tham gia hoặc mua vé.

Phần đầu trang (header) vẫn giữ cấu trúc thống nhất với logo, thanh tìm kiếm và các nút chức năng cơ bản, tạo sự đồng bộ với các trang khác trong hệ thống. Khu vực thông tin chính bao gồm ảnh đại diện sự kiện, tiêu đề, thời gian, địa điểm và nút hành động chọn mua vé.

Phần mô tả chi tiết được trình bày ngay bên dưới, cho phép ban tổ chức cung cấp nội dung xuất và vé sự kiện.



Hình Lo-fi prototype Trang chi tiết sự kiện

#### Trang chọn vé

Trang chọn giúp người dùng dễ dàng lựa chọn loại vé phù hợp. Phần đầu trang (header) vẫn giữ cấu trúc thống nhất với logo, thanh tìm kiếm và các nút điều hướng cơ bản nhằm tạo sự đồng bộ trong toàn bộ hệ thống. Nội dung chính của trang được chia thành hai khu vực:

**Khu vực chọn vé (bên trái):** hiển thị danh sách các loại vé kèm thông tin chi tiết như tên vé, giá vé. Người dùng có thể tăng/giảm số lượng vé mong muốn thông qua các nút điều chỉnh trực quan.

**Khu vực tóm tắt đặt vé (bên phải):** tổng hợp các lựa chọn của người dùng bao gồm số lượng vé, tổng giá trị đơn hàng và các thông tin bổ sung cần thiết. Phần này cũng tích hợp nút hành động (tiếp tục thanh toán) để chuyển sang bước thanh toán.



Hình Lo-fi prototype Trang chọn vé

#### Trang đơn vé đã mua

Sau khi người dùng hoàn tất thanh toán, hệ thống sẽ lưu trữ thông tin đơn hàng và hiển thị trong mục **Vé đã mua**. Tại đây, mỗi đơn hàng sẽ bao gồm các thông tin chi tiết như: mã đơn hàng, ngày tạo đơn, phương thức thanh toán đã sử dụng, cùng với thông tin người mua (họ tên và địa chỉ email).

Bên trong đơn hàng, người dùng có thể xem danh sách các vé đã mua với đầy đủ thông tin về loại vé, số lượng, đơn giá và số tiền tương ứng. Ngoài ra, hệ thống sẽ cung cấp **mã QR** cho từng vé, giúp người dùng thuận tiện sử dụng để **check-in trực tiếp tại sự kiện.**

Toàn bộ chi phí của đơn hàng cũng được hệ thống tính toán và hiển thị rõ ràng, đảm bảo người dùng dễ dàng kiểm tra và đối chiếu.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\Wireframes (5).png |

Hình Lo-fi prototype Trang danh sách vé đã mua

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\Wireframes (4).png |

Hình Lo-fi prototype Trang chi tiết ve đã mua

#### Trang danh sách bản tin, chi tiết bản tin

Trang danh sách bản tin được thiết kế theo bố cục dạng lưới (grid layout), trong đó các tin tức mới nhất được hiển thị ở khu vực nổi bật trên cùng với hình ảnh và tiêu đề lớn nhằm thu hút sự chú ý của người dùng. Các tin tức còn lại được sắp xếp thành các khối nhỏ hơn, kèm theo hình ảnh minh họa, tiêu đề và thời gian đăng tải.

Bên cạnh đó, hệ thống cung cấp thanh tìm kiếm và điều hướng phân trang, giúp người dùng dễ dàng tra cứu và tiếp cận nhiều bản tin hơn. Phần chân trang (footer) được thiết kế đồng bộ, chứa các thông tin liên hệ, liên kết nhanh và chính sách hỗ trợ.

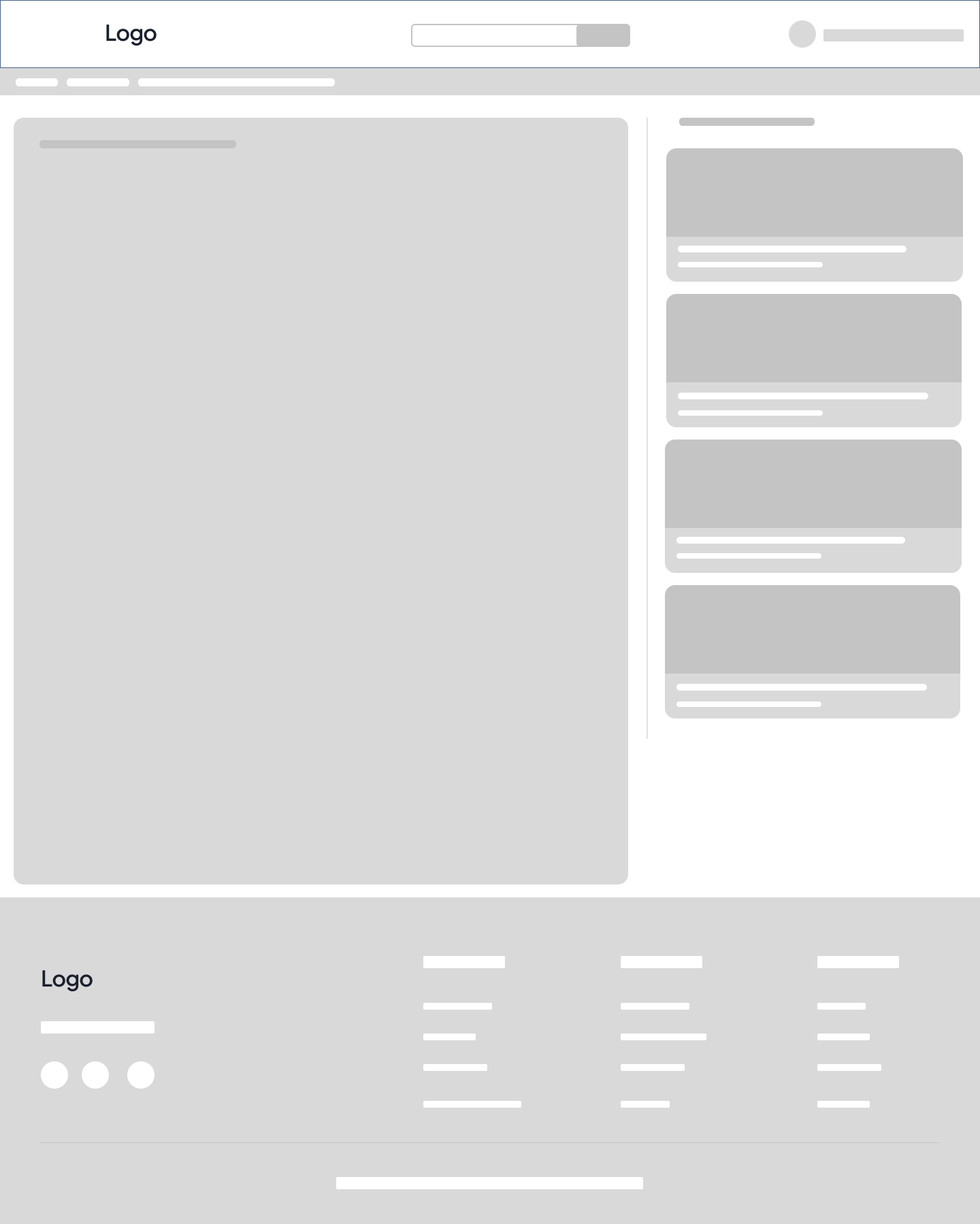


Hình Lo-fi prototype Trang danh sách bản tin

Trang chi tiết bản tin được thiết kế với bố cục hai cột:

* Cột chính hiển thị nội dung bản tin đầy đủ gồm hình ảnh, tiêu đề, thời gian đăng và phần nội dung chi tiết.
* Cột phụ bên phải hiển thị danh sách các tin liên quan, giúp tăng khả năng điều hướng và giữ chân người đọc lâu hơn trên hệ thống.

Ngoài ra, giao diện được tối ưu để đảm bảo khả năng đọc nội dung dễ dàng, sử dụng phông chữ rõ ràng, kích thước chữ phù hợp và hình ảnh minh họa sinh động. Footer vẫn giữ sự đồng bộ với trang danh sách nhằm tạo tính thống nhất cho toàn bộ hệ thống.



Hình Lo-fi prototype Trang chi tiết bản tin

# KẾT QUẢ NGHIÊN CỨU

## Dữ liệu thử nghiệm

Dữ sự kiện được thu tập từ các trang web bán vé trực tuyến. Dữ liệu được tham khảo tại trang <https://ticketbox.vn>. Dự liệu bên dưới chỉ là vài dòng sơ lượt

Dữ liệu bảng **tài khoản**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Email | Mật khẩu | Quyền | Trạng thái |
| 1 | vanhuy220033@gmail.com | google\_password | admin | active |
| 2 | tranvantruonghuyy@gmail.com | google\_password | organizer | active |
| 3 | ttruonghuy2003@gmail.com | google\_password | organizer | active |
| 4 | usertruonghuy@gmail.com | google\_password | user | active |

Bảng Dữ liệu tài khoản

Dữ liệu bảng **ban tổ chức**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Tên ban tổ chức | Email | Giới thiệu | Link trang web | Số điện thoại |
| 1 | VietSound Entertainment | vanhuy220033@gmail.com | VietSound Entertainment là đơn vi,….. | www.vietsound.vn | 0362875750 |
| 2 | Vietnam Sports Event Management | ttruonghuy2003@gmail.com | Chuyên tổ chức các sự kiện thể thao chuyên nghiệp,… | https://vietnamsports.vn | 0372875750 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Link mạng xã hội | Logo | Tên ngân hàng | Số tài khoản ngân hàng | Tên chủ tài khoản |
| 1 | facebook.com/vietsoundentertainment | https://storage.googleapis.com/… | MBbank | 123 456 789 012 | TRAN TRUONG HUY |
| 2 | https://facebook.com/vietnamsports | https://storage.googleapis.com/… | Vietcombank | 0123456789 | SPORTS EVENT MANAGEMENT |

Bảng Dữ liệu bảng ban tổ chức

Dữ liệu bảng **người dùng**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Tên | Số điện thoại | Email | Ảnh đại diện | Địa chỉ |
| 1 | Huy Trần | 0362875750 | usertruonghuy@gmail.com | https://storage.googleapis.com/… | An Bình, Tân Bình, Càng Long, Trà Vinh |
| 2 | Huy Văn | 0362875751 | taikhoancuatruonghuy@gmail.com | https://storage.googleapis.com/… | An Binh, Tân Bình, Càng Long, Trà Vinh |

Bảng Dữ liệu người dùng

Dữ liệu bảng **sự kiện**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| STT | Tiêu đề | Giới thiệu | Địa chỉ | Ảnh | Trạng thái | Loại |
| 1 | Saigon International Guitar,… | Saigon Internationa,… | Khán phòng Idecaf, 31 Thái Văn Lung,…. | https://storage.googleapis.com/book-8d19b,.. | approved | âm nhạc |
| 2 | Lion Championship 25 - 2025 | Giới hiệu,… | Quảng trường Cam Ranh,Phường Cam Linh … | https://storage.googleapis.com/book-8d19b,.. | approved | thể thao |
| 3 | THE ALMA SHOW,… | NGHỆ THUẬT ĐẲNG CẤ… | Alma Amphitheater, Alma Resort Cam Ranh,… | https://storage.googleapis.com/book-8d19b,.. | approved | văn hóa nghệ thuật |
| 4 | GRAND OPENING BENARAS PEARL | KHAI TRƯƠNG TƯNG BỪNG,… | BENARAS PEARL PHU QUOC, 179, đường 30/4,… | https://storage.googleapis.com/book-8d19b,.. | approved | khác |

Bảng Dữ liệu sự kiện

Dữ liệu bảng **xuất sự kiện**

|  |  |  |
| --- | --- | --- |
| STT | Ngày bắt đầu | Ngày kết thúc |
| 1 | 18:00 - 15/09/2025 | 22:00 - 15/09/2025 |
| 2 | 19:00 - 20/09/2025 | 22:00 - 20/09/2025 |
| 3 | 18:00 - 22/09/2025 | 20:00 - 22/09/2025 |

Bảng Dữ liệu xuất sự kiện

Dữ liệu bảng **vé sự kiện**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| STT | Tên vé | Giá vé | Số lượng | Vé mua tối thiểu | Vé mua tối đa | Mô tả | Số lượng đã bán |
| 1 | FESTIVAL PACKAGE- All 3 festival concerts | 1.200.000 | 100 | 1 | 10 |  | 0 |
| 2 | FESTIVAL CONCERT I "LATIN ROSETTE" | Fri 15 | 500.000 | 100 | 1 | 10 |  | 0 |
| 3 | FESTIVAL CONCERT II "HYMNE À LA VIE" | Sat 16 | 500.000 | 200 | 1 | 10 |  | 0 |

Bảng Dữ liệu vé sự kiện

Dữ liệu bảng **đơn vé**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Email | Tổng tiền | Trạng thái | Phương thức |
| 1 | usertruonghuy@gmail.com | 395.000 | paid | VNPay |

Bảng Dữ liệu đơn vé

Dữ liệu bảng **chi tiết vé trong đơn**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Giá | QR Code | Trạng thái | Thời gian quét |
| 1 | 395.000 | data:image/png;base64,iV,….. | valib |  |

Bảng Dữ liệu vé sử dụng

Dữ liệu bảng **tin tức**

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tiêu đề | Nội dung | Trạng thái |
| 1 | Sự kiện văn hóa nghệ thuật tại Hà Nội dịp 2/9 diễn ra tuần này | Chuỗi hoạt động biểu diễn nghệ thuật, âm nhạc, múa dân gian và triển lãm được tổ chức tại Quảng trường Cách mạng Tháng Tám nhằm kỷ niệm Quốc khánh 2/9. | Đã xuất bản |
| 2 | Hà Nội ra mắt 80 trải nghiệm du lịch vàng dịp 2/9 | Sở Du lịch Hà Nội giới thiệu 80 sản phẩm, dịch vụ du lịch mới với nhiều ưu đãi, tập trung vào ẩm thực, văn hóa và tham quan các di tích lịch sử,… | Đã xuất bản |
| 3 | Bóng hồng thép giành ba huy chương một ngày ở giải VĐQG | Vận động viên nữ của đội tuyển điền kinh đã xuất sắc giành 3 huy chương vàng trong một ngày thi đấu, khẳng định sức mạnh của thể thao nữ Việt Nam,… | Đã xuất bản |
| 4 | Các sự kiện mỹ thuật, nhiếp ảnh gây chú ý trong năm | Hàng loạt triển lãm tranh, ảnh nghệ thuật được tổ chức tại Hà Nội và TP.HCM, thu hút đông đảo giới trẻ và khách quốc tế,… | Đã xuất bản |
| 5 | 7 sự kiện thể thao nổi bật 2025 | Từ ASIAD, AFF Cup đến các giải quốc nội, năm 2025 được coi là cột mốc thành công lớn của thể thao Việt Nam,… | Đã xuất bản |
| 6 | Muôn vàng dấu ấn Masterise Homes tại diễn đàn bất động sản hạng hiệu châu Á | Masterise Homes giới thiệu loạt dự án bất động sản cao cấp, ghi dấu ấn với thiết kế hiện đại và không gian sống chuẩn quốc tế,.. | Đã xuất bản |

Bảng Dữ liệu tin tức

## Kết quả thử nghiệm

### Hiển thị thông tin sự kiện

**Trang chủ**

* Hệ thống hiển thị các sự kiện nổi bật dưới dạng banner trượt, giúp người dùng dễ dàng theo dõi các sự kiện sắp diễn ra.
* Các sự kiện được phân loại theo từng nhóm: Âm nhạc, Văn hóa nghệ thuật, Thể thao, Khác.
* Mỗi sự kiện thể hiện đầy đủ thông tin cơ bản: tên sự kiện, ngày diễn ra, giá vé, hình ảnh minh họa.
* Người dùng có thể chọn xem chi tiết hoặc nhấn vào mục “Xem tất cả” để hiển thị thêm nhiều sự kiện hơn trong cùng chuyên mục.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\trangchu.png |

Hình Trang chủ

**Trang danh sách sự kiện theo thể loại**

* Khi người dùng chọn một chuyên mục cụ thể (ví dụ: Âm nhạc), hệ thống hiển thị toàn bộ danh sách sự kiện thuộc chuyên mục đó.
* Chức năng lọc sự kiện theo vị trí và thời gian được tích hợp, giúp người dùng dễ dàng tìm sự kiện phù hợp.
* Giao diện hiển thị dạng lưới với thông tin ngắn gọn (tên, giá vé, thời gian), đảm bảo trực quan và dễ thao tác.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\tat_ca_sk_theo_danh_muc.png |

Hình Trang sự kiện theo loại

|  |
| --- |
|  |

Hình Chức năng lọc sự kiện

**Trang chi tiết sự kiện**

* Khi chọn một sự kiện, hệ thống chuyển đến trang chi tiết, hiển thị đầy đủ.
* Tên sự kiện, đơn vị tổ chức, thời gian, địa điểm.
* Giá vé và tùy chọn số lượng vé.
* Phần giới thiệu sự kiện chi tiết kèm hình ảnh minh họa.
* Thông tin về nhà tổ chức (giới thiệu, địa chỉ, liên hệ).
* Trang cũng gợi ý thêm các sự kiện liên quan, giúp tăng trải nghiệm và khuyến khích người dùng khám phá thêm.
* Nút Mua vé hoạt động bình thường, dẫn đến bước đặt vé.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\chi_tiet_sk.png |

Hình Trang chi tiết sự kiện

### Hiển thị các trang chức năng đặt vé

**Trang chọn vé**

* Người dùng có thể lựa chọn số lượng vé mong muốn.
* Thông tin sự kiện, thời gian, địa điểm và tổng tiền được hiển thị rõ ràng ở khung bên phải.
* Nút Thanh toán cho phép chuyển sang bước tiếp theo.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\chon_ve.png |

Hình Trang chọn vé

**Trang thanh toán**

* Hệ thống hỗ trợ phương thức thanh toán trực tuyến qua **VNPay.**
* Thông tin đặt vé được tóm tắt gồm loại vé, số lượng và tổng số tiền cần thanh toán.
* Người dùng xác nhận bằng cách nhấn Thanh toán, đảm bảo tính chính xác và thuận tiện trước khi giao dịch.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\thong_tin_tt.png |

Hình Trang thông tin thanh toán

Nội dung email được gửi đến người mua vé sau khi thanh toán thành công:

|  |
| --- |
| **Sự kiện: ART WORKSHOP OIL PASTEL PAINTING "MELTED THOUGHTS"**  Thời gian: 17:30:00 1/9/2025 - 07:23:00 1/9/2025  Địa điểm: Garden Art Lầu 1, 386/17C Lê Văn Sỹ, Phường 14, Quận 3, Thành phố Hồ Chí Minh   * Vé: **Tiêu Chuẩn** - Giá: **390.000 đ**  C:\Users\Admin\Downloads\qr_code_ve_Tiêu_Chuẩn_1.png * Vé: **Tiêu Chuẩn** - Giá: **390.000 đ** C:\Users\Admin\Downloads\qr_code_ve_Tiêu_Chuẩn_2.png   Điều khoản sử dụng:   * Vé chỉ có giá trị sử dụng một (01) lần cho một (01) người tham dự sự kiện vào đúng thời gian, địa điểm trên vé. * Khách hàng vui lòng xuất trình vé hợp lệ tại cổng check-in để được xác nhận tham dự. * Người mua có trách nhiệm giữ gìn và bảo mật mã vé. Ve++ không chịu trách nhiệm đối với các trường hợp mất mát, thất lạc hoặc bị sử dụng bởi bên thứ ba. * Bằng việc mua vé và tham gia sự kiện, quý khách đồng ý và cam kết tuân thủ các điều khoản và điều kiện được công bố chính thức từ Ve++ và Ban tổ chức. Điều khoản và điều kiện có thể được cập nhật mà không cần thông báo trước.   Cảm ơn bạn đã đặt vé! |

|  |
| --- |
| C:\Users\Admin\Downloads\mail.google.com_mail_u_5_.png |

Hình Email nhận vé

**Trang xác nhận và vé điện tử**

* Sau khi thanh toán thành công, vé điện tử được sinh ra và lưu trong mục **“Vé của tôi”**
* Hệ thống cung cấp **mã QR Code** để người dùng có thể sử dụng khi tham dự sự kiện.
* Vé cũng được gửi về email đăng ký, giúp người dùng lưu trữ và kiểm tra dễ dàng.

|  |
| --- |
| C:\Users\Admin\Downloads\localhost_3000_users.png |

Hình Trang danh sách vé đã mua

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\chi_tiet_ve.png |

Hình Trang chi tiết đơn vé

### Hiển thị các trang tác vụ ban tổ chức

**Tạo sự kiện**

* Nhập thông tin cơ bản: tiêu đề, ảnh, mô tả, loại sự kiện.
* Xác định địa điểm: tỉnh/thành phố, quận/huyện, phường/xã, số nhà và tên đường.

**Tạo vé**

* Thiết lập thời gian diễn ra sự kiện (bắt đầu – kết thúc).
* Có thể tạo nhiều xuất diễn khác nhau cho cùng một sự kiện.
* Gán loại vé tương ứng cho từng xuất.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\tao_sk.png |

Hình Trang tạo sự kiện

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\tao_sk_2.png |

Hình Trang tạo sự kiện tiếp theo

**Quản lý sự kiện**

* Xem danh sách các sự kiện đã tạo (đã lưu, đã đăng, đã xóa).
* Theo dõi số vé đã bán, số vé còn lại, tổng doanh thu.
* Truy cập chi tiết sự kiện để xem thống kê theo từng xuất.

**Thống kê và Báo cáo**

* Biểu đồ số lượng vé bán theo ngày.
* Biểu đồ doanh thu theo ngày.
* Lịch sử check-in khách tham dự.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\ds_sk_btc.png |

Hình Trang danh sách sự kiện ban tổ chức

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\chi_tiet_ban_ve.png |

Hình Trang thống kê sự kiện

**Quét vé & check-in**

* Sử dụng camera để quét mã QR trên vé.
* Xác nhận tình trạng vé (hợp lệ/không hợp lệ).
* Ghi nhận lượt check-in của khách tham dự.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\quet_ve.png |

Hình Trang quét vé

Các thao tác cập nhật sự kiện:

* Chỉnh sửa sự kiện.
* Sửa xuất, vé
* Thêm mới xuất, vé

|  |
| --- |
|  |

Hình Các thao tác chỉnh sửa sự kiện.

|  |
| --- |
|  |

Hình Form sửa sự kiện

|  |
| --- |
|  |

Hình Form sửa xuất

|  |
| --- |
|  |

Hình Form sửa vé

|  |
| --- |
|  |

Hình Form thêm mới xuất

### Hiển thị các tác vụ quản trị viên

**Dashboard tổng quan**

* Hiển thị số lượng sự kiện, số lượng vé đã bán, doanh thu, số người dùng tham gia hệ thống.
* Có biểu đồ trực quan giúp theo dõi xu hướng vé và doanh thu theo thời gian.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\dash.png |

Hình Trang chủ quản trị

**Quản lý sự kiện & vé**

* Theo dõi danh sách sự kiện đã được tạo bởi ban tổ chức.
* Quản lý số lượng vé phát hành, vé đã bán và tình trạng vé.
* Xem báo cáo chi tiết theo từng sự kiện.
* hoạt động đăng ký và tham gia sự kiện.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\dash_sk.png |

Hình Quản trị tab sự kiện

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\dash_sk2.png |

Hình Quản trị tab sự kiện thống kê

**Quản lý người dùng**

* Xem danh sách tài khoản người dùng.
* Chỉnh sửa hoặc khóa/mở khóa tài khoản khi cần thiết.
* Theo dõi hoạt động đăng ký và tham gia sự kiện.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\dash_user.png |

Hình Quản trị tab người dùng

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\dash_btc.png |

Hình Quản trị tab ban tổ chức

**Giao dịch**

* Hiển thị danh sách sự kiện đã qua hơn 7 ngày, lấy thông tin tài khoản ngân hàng và tải lên ảnh chuyển khoản
* Sau khi thanh toán gửi mail cho ban tổ chức

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\dash_gd.png |

Hình Quản trị tab giao dịch

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\dash_thanht.png |

Hình Quản trị tab thanh toán

**Quản lý nội dung**

* Cho phép quản trị viên **tạo mới tin tức**.
* Chức năng **sửa bài viết** đã đăng.
* Đảm bảo kênh thông tin cho người dùng và ban tổ chức.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\tao_bt.png |

Hình Quản trị tab thêm bản tin

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\sua_bt.png |

Hình Quản trị tab sửa sự kiện

|  |
| --- |
|  |

Hình Danh sách sự kiện hoàn vé

|  |
| --- |
| C:\Users\Admin\Downloads\mail.google.com_mail_u_5_ (2).png |

Hình Email gửi cho người mua vé được hoàn tiền

### Hiển thị tin tức

**Trang danh sách tin tức**

* Tin tức mới nhất được hiển thị nổi bật ở đầu trang kèm ảnh minh họa.
* Các tin bài khác được sắp xếp dạng lưới, hiển thị tiêu đề, ảnh thu nhỏ, thời gian đăng và trích đoạn nội dung.
* Cho phép phân trang để người dùng xem thêm tin cũ hơn.

|  |
| --- |
| C:\Users\Admin\Downloads\imageee\imageee\ban_tin.png |

Hình Trang danh sách sự kiện

**Trang chi tiết tin tức**

* Khi chọn một bài viết, người dùng được chuyển đến trang chi tiết.
* Nội dung tin tức hiển thị đầy đủ, bao gồm: tiêu đề, hình ảnh, nội dung văn bản, thời gian đăng.
* Bên cạnh có mục **“Bản tin liên quan”** hiển thị các bài viết khác để người dùng dễ dàng theo dõi.
* Phần chân trang (footer) đồng nhất với toàn hệ thống, cung cấp thông tin liên hệ, hỗ trợ, điều khoản.

|  |
| --- |
|  |

Hình Trang chi tiết sự kiện

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

Trong quá trình thực hiện khóa luận, sau đây là nhưng kết quả đã đạt được:

1. Thu thập và phân tích dữ liệu liên quan đến tổ chức sự kiện và bán vé trực tuyến.
2. Thiết kế thành công cơ sở dữ liệu MongoDB phục vụ cho việc quản lý sự kiện, vé, đơn hàng và người dùng.
3. Hoàn thành việc tìm hiểu và áp dụng **NestJS** cho backend, **NextJS** cho frontend và **Tailwind CSS** cho thiết kế giao diện.
4. Xây dựng và cài đặt các chức năng cơ bản: đăng ký, đăng nhập, chỉnh sửa thông tin người dùng.
5. Hoàn thành chức năng quản lý sự kiện (thêm, sửa, xóa, duyệt sự kiện).
6. Xây dựng chức năng đặt vé trực tuyến, quản lý giỏ vé và xử lý đơn hàng.
7. Hoàn thành việc tích hợp thanh toán trực tuyến thông qua **VNPay** và hỗ trợ thanh toán COD.
8. Xây dựng chức năng xuất vé điện tử kèm **mã QR** để phục vụ check-in sự kiện.
9. Hoàn thành chức năng thống kê doanh thu, vé bán ra và quản lý báo cáo cho ban tổ chức.
10. Rút ra nhiều kinh nghiệm trong việc phân tích yêu cầu, thiết kế hệ thống, áp dụng công nghệ hiện đại và xử lý các vấn đề phát sinh trong quá trình phát triển.

## Hướng phát triển

Trong tương lai, hệ thống có thể được mở rộng và phát triển theo các hướng sau:

1. **Hoàn thiện chức năng đặt vé**: Bổ sung nhiều hình thức thanh toán trực tuyến khác như Momo, ShopeePay, hoặc thẻ quốc tế Visa/MasterCard để tăng sự tiện lợi cho người dùng.
2. **Ứng dụng di động**: Phát triển ứng dụng trên nền tảng iOS và Android, giúp người dùng dễ dàng tra cứu sự kiện, đặt vé và sử dụng vé điện tử mọi lúc, mọi nơi.
3. **Cá nhân hóa trải nghiệm**: Xây dựng hệ thống gợi ý sự kiện dựa trên hành vi và sở thích của người dùng, áp dụng trí tuệ nhân tạo (AI) để tối ưu hoá trải nghiệm.
4. **Quản lý đa kênh**: Mở rộng khả năng tích hợp với các nền tảng mạng xã hội (Facebook, Zalo, Instagram) để quảng bá sự kiện và hỗ trợ bán vé trực tiếp.
5. **Nâng cấp hệ thống check-in**: Tích hợp công nghệ quét QR nâng cao, hỗ trợ quét nhanh qua thiết bị di động hoặc kiosk tự động, giảm thời gian chờ đợi của khách tham dự.
6. **Hỗ trợ ban tổ chức**: Bổ sung chức năng, tạo sự kiện tuỳ chọn cao, phân tích dữ liệu nâng cao, dự báo xu hướng và quản lý chiến dịch marketing dựa trên dữ liệu bán vé.

DANH MỤC TÀI LIỆU THAM KHẢO

**Tiếng Việt**

1. Bộ Thông tin và Truyền thông (2021), Chuyển đổi số quốc gia đến năm 2025, định hướng đến năm 2030, Hà Nội.
2. Nguyễn Văn Cường (2018), Giáo trình Cơ sở dữ liệu, Nxb Đại học Quốc gia TP. Hồ Chí Minh.
3. Trần Thị Thu Hằng (2020), “Ứng dụng công nghệ thông tin trong quản lý và tổ chức sự kiện”, Tạp chí Khoa học và Công nghệ, 18(3), tr. 45–53.
4. Trường Đại học Công nghệ Thông tin (2022), Bài giảng Phát triển ứng dụng web, Nxb Đại học Quốc gia TP. Hồ Chí Minh.
5. Ticketbox (2025), Nền tảng bán vé sự kiện trực tuyến, [Online]. Available: <https://ticketbox.vn/>
6. Ticketgo (2025), Mua vé sự kiện, vé hòa nhạc, hội thảo trực tuyến, [Online]. Available: <https://ticketgo.vn/>

**Tiếng anh**

1. Fowler, M. (2003), UML Distilled: A Brief Guide to the Standard Object Modeling Language, 3rd ed., Addison-Wesley, Boston.
2. MongoDB, Inc. (2025), MongoDB Manual, [Online]. Available: <https://www.mongodb.com/docs/manual/>
3. NestJS (2025), Official Documentation, [Online]. Available: <https://docs.nestjs.com/>
4. Next.js (2025), The React Framework for Production, [Online]. Available: <https://nextjs.org/docs>
5. Sommerville, I. (2015), Software Engineering, 10th ed., Pearson, London.