



**BÁO CÁO ĐỀ TÀI
PHÁT TRIỂN GIAO DIỆN ỨNG DỤNG**

ĐỀ TÀI:**Xây dựng giao diện ứng dụng website đặt tour du lịch.**

NGÀY ĐĂNG KÝ:**ngày 28 tháng 12 năm 2024.....**

NGÀY HOÀN THÀNH:**ngày 20 tháng 03 năm 2025.....**

NGÀY NỘP:**ngày 22 tháng 03 năm 2025.....**

GIÁO VIÊN HƯỚNG DẪN:**ThS Từ Thị Xuân Hiền.....**

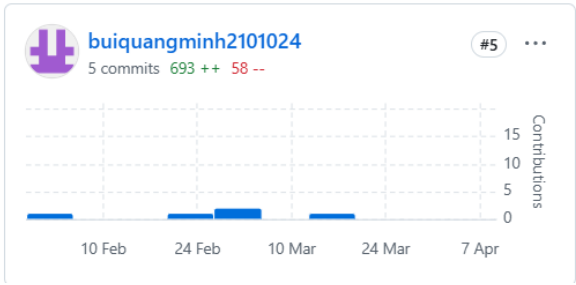
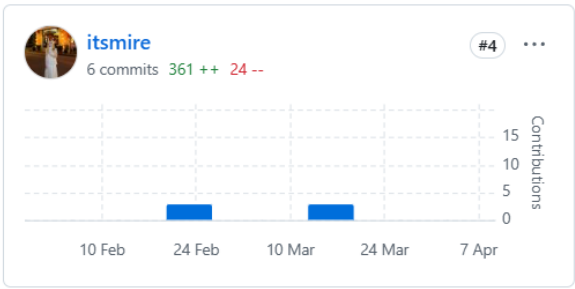
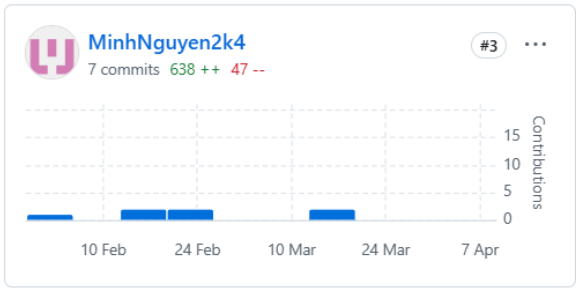
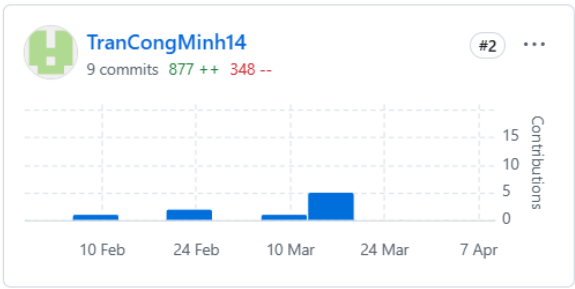
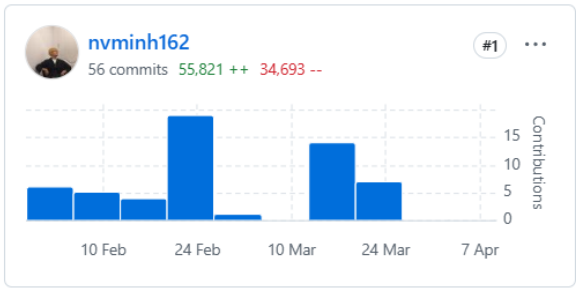
NHÓM THỰC HIỆN:**Nhóm 11.....**

THÀNH VIÊN NHÓM 11

STT	Họ và tên	MSSV	Vai trò	Ghi chú
53	Nguyễn Văn Minh	22003405	Nhóm trưởng	Có tham gia
51	Bùi Quang Minh	22664411	Thành viên	Có tham gia
52	Nguyễn Tấn Minh	22001075	Thành viên	Có tham gia
54	Trần Công Minh	22638121	Thành viên	Có tham gia
55	Trần Vũ Uyên My	22002045	Thành viên	Có tham gia

ĐÁNH GIÁ HIỆU SUẤT LÀM VIỆC TRONG NHÓM

STT	Họ và tên	Ý thức	Chất lượng	Thời gian	Tương tác	Hoàn thành	Tổng
53	Nguyễn Văn Minh	100%	100%	100%	100%	100%	100%
51	Bùi Quang Minh	65%	50%	80%	0%	65%	65%
52	Nguyễn Tấn Minh	80%	80%	80%	0%	80%	80%
54	Trần Công Minh	70%	50%	80%	0%	70%	70%
55	Trần Vũ Uyên My	65%	50%	80%	0%	65%	65%



NHIỆM VỤ

STT	Họ và tên	Nhiệm vụ
53	Nguyễn Văn Minh	<ul style="list-style-type: none"> - Thiết lập toàn bộ cấu trúc dự án - Cài đặt các thư viện cần thiết cho ứng dụng - Phân chia components dùng chung tăng tính tái sử dụng - Xây dựng layout toàn bộ website - Chịu trách nhiệm trang: <ul style="list-style-type: none"> + Trang chủ (Home Page) + Tìm vé máy bay (Flight Page) + Tìm khách sạn (Hotel Page) + Form tìm kiếm (Search Modal) + Form thông báo (Notify Modal) + Trang xác thực (Login, SignUp, Forgot Page) + Không tìm thấy 404 (404 Page) + Chi tiết đặt phòng (Hotel Detail Page) + Chi tiết đặt du thuyền (Cruise Detail Page) - Quản lý state Redux toolkit (Auth, SignUp, Forgot), custom Hook - Viết báo cáo và testcase toàn bộ cho dự án - Viết full xử lý Backend, tạo API kết nối cơ sở dữ liệu CRUD với MongoDB, lưu trữ cloud MongoDB Atlas thời gian thực.
51	Bùi Quang Minh	<ul style="list-style-type: none"> - Trang phân công và chịu trách nhiệm: <ul style="list-style-type: none"> + Về chúng tôi (About Page) + Điều khoản (Terms Page) + Chính sách riêng tư (Privacy Page)
52	Nguyễn Tấn Minh	<ul style="list-style-type: none"> - Tạo các hàm Utils hỗ trợ tái sử dụng trong dự án - Giám sát thời gian và hỗ trợ về kỹ thuật cho toàn dự án - Vẽ sơ đồ Sitemap - Trang phân công và chịu trách nhiệm: <ul style="list-style-type: none"> + Tìm du thuyền (Cruise Page) + Doanh nghiệp (Business Page) + Đầu trang (Header Layout) + Chân trang (Footer Layout)
54	Trần Công Minh	<ul style="list-style-type: none"> - Trang phân công và chịu trách nhiệm: <ul style="list-style-type: none"> + Hướng dẫn dùng (User Manual Page) + Thanh toán (Payment Page) + Liên hệ (Contact Page)
55	Trần Vũ Uyên My	<ul style="list-style-type: none"> - Trang phân công và chịu trách nhiệm: <ul style="list-style-type: none"> + Quy định (Regulations Page) + Câu hỏi thường gặp (FAQ Page) + Blog (Blog Page)

[illegible]

Mục Lục

CHƯƠNG 1.	GỚI THIỆU ĐỀ TÀI.....	6
1.1	Mục đích	6
1.1.1	Giới Thiệu Tổng Quan (Tên ứng dụng là TourX)	6
1.1.2	Mục Đích.....	6
1.2	Cơ sở lý thuyết và các thông tin kỹ thuật:	7
1.2.1	Các Kiến Thức Lý Thuyết Vận Dụng cho Việc Hiện Thực Website:	7
1.2.2	Lý Do và Ưu Điểm của Việc Áp Dụng:	8
CHƯƠNG 2.	XÂY DỰNG ỨNG DỤNG.....	9
2.1	Khả năng tương thích đa thiết bị (Responsive)	9
2.2	Sitemap	9
2.3	Giao diện	10
2.3.1	Trang chủ	10
2.3.2	Tìm du thuyền.....	10
2.3.3	Tìm vé máy bay.....	11
2.3.4	Tìm khách sạn	11
2.3.5	Trang chi tiết dịch vụ	12
CHƯƠNG 3.	KIỂM THỬ ỨNG DỤNG BẰNG VITEST.....	13
3.1	Test Case 01: Validate Contact Form Submission	13
3.2	Test Case 02: Validate Contact Form Empty Fields	13
3.3	Test Case 03: Validate Email Format in Contact Form	14
3.4	Test Case 04: Validate Phone Number Format in Contact Form	14
3.5	Test Case 05: Check if CruiseForm renders and displays correct content.....	14
3.6	Test Case 06: Test users can enter yacht name into search box	15
3.7	Test Case 07: Check if the "location" and "price" dropdowns are displaying correctly.....	15
3.8	Test Case 08: Is "Search" displayed and navigated correctly?	15
3.9	Test Case 09: Test when clicking "Search", the page will scroll to the top	16
3.10	Test Case 10: Check FlightForm renders and displays correct content.....	16
3.11	Test Case 11: Check the departure and destination points, they must not be the same.	16
3.12	Test Case 12: Check that the departure date is not less than the current date and the return date is after the departure date.....	17
3.13	Test Case 13: FlightForm Passenger Count Test.....	17
3.14	Test Case 14: FlightForm Search Button Test	17
3.15	Test Case 15: CruiseForm Input Placeholder Test.....	18
3.16	Test Case 16: CruiseForm Dropdown Render Test.....	18
3.17	Test Case 17: CruiseForm Search Button Test	18
3.18	Test Case 18: CruiseForm Scroll Behavior Test	19

3.19	Test Case 19: CruiseForm Link Navigation Test	19
3.20	Test Case 20: Kiểm tra Button render đúng thẻ khi có to hoặc href	19
CHƯƠNG 4.	KẾT LUẬN – HƯỚNG PHÁT TRIỂN	20
4.1	Kết quả đạt được	20
4.2	Hạn chế của ứng dụng web.....	20
4.3	Hướng phát triển	20
CHƯƠNG 5.	TÀI LIỆU THAM KHẢO.....	21
5.1	Giáo trình – Sách	21
5.2	Website	21
CHƯƠNG 6.	Mã nguồn dự án và sản phẩm triển khai	21
6.1	Giới thiệu dự án (Introduce)	21
6.2	Mã nguồn dự án (Source Code).....	21
6.3	Sản phẩm triển khai (Deployment Production)	Error! Bookmark not defined.

CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

1.1 Mục đích

1.1.1 Giới Thiệu Tổng Quan (Tên ứng dụng là TourX)

TourX là một nền tảng trực tuyến chuyên cung cấp các dịch vụ và thông tin liên quan đến du lịch, khám phá và trải nghiệm văn hóa. Trang web này được thiết kế nhằm mục đích kết nối người dùng với các hoạt động du lịch độc đáo, tour du lịch, và các dịch vụ lưu trú tại nhiều điểm đến hấp dẫn trong và ngoài nước. Với giao diện thân thiện và dễ sử dụng, TourX hướng đến việc mang lại trải nghiệm tối ưu cho người dùng, từ khâu tìm kiếm thông tin đến đặt chỗ và thanh toán.

TourX không chỉ là một công cụ đặt tour thông thường mà còn là nơi chia sẻ những bài viết, đánh giá, và kinh nghiệm du lịch từ cộng đồng. Điều này giúp người dùng có cái nhìn toàn diện hơn về các điểm đến, đồng thời tạo ra một không gian tương tác giữa những người yêu thích du lịch. Trang web cũng tích hợp các công nghệ hiện đại để cá nhân hóa trải nghiệm, giúp người dùng dễ dàng tìm kiếm các gói dịch vụ phù hợp với nhu cầu và sở thích cá nhân.

Với sứ mệnh mang đến những chuyến đi ý nghĩa và đáng nhớ, TourX đang dần khẳng định vị thế của mình trong lĩnh vực du lịch trực tuyến, trở thành một địa chỉ tin cậy cho những ai đam mê khám phá và trải nghiệm.

1.1.2 Mục Đích

1.1.2.1 Mục Đích Chung

TourX được xây dựng với mục đích chung là trở thành một nền tảng toàn diện, kết nối người dùng với các dịch vụ du lịch chất lượng, đồng thời cung cấp thông tin hữu ích và đáng tin cậy về các điểm đến, hoạt động du lịch, và văn hóa địa phương. Trang web hướng đến việc tạo ra một cộng đồng yêu thích du lịch, nơi người dùng có thể chia sẻ kinh nghiệm, đánh giá, và khám phá những hành trình mới. Mục tiêu lớn nhất của TourX là mang lại trải nghiệm du lịch trọn vẹn, từ giai đoạn lên kế hoạch đến khi kết thúc chuyến đi, thông qua việc tích hợp công nghệ hiện đại và dịch vụ chuyên nghiệp.

1.1.2.2 Mục Đích Cụ Thể Đối Với Các Đối Tượng Sử Dụng

- Đối với khách du lịch cá nhân:
 - o Cung cấp thông tin chi tiết và đa dạng về các điểm đến, tour du lịch, và hoạt động khám phá.
 - o Hỗ trợ người dùng dễ dàng tìm kiếm, so sánh, và đặt các gói dịch vụ du lịch phù hợp với nhu cầu và ngân sách.
 - o Tạo điều kiện để khách hàng chia sẻ đánh giá, kinh nghiệm, và nhận tư vấn từ cộng đồng du lịch.
- Đối với nhóm du lịch hoặc gia đình:

- Đề xuất các gói tour nhóm, dịch vụ lưu trú, và hoạt động phù hợp với nhu cầu của nhiều người.
- Hỗ trợ quản lý và tổ chức chuyến đi một cách thuận tiện thông qua các công cụ đặt chỗ và thanh toán trực tuyến.
- Đối với đối tác du lịch (công ty lữ hành, khách sạn, nhà cung cấp dịch vụ):
 - Cung cấp một kênh quảng bá hiệu quả để giới thiệu dịch vụ và sản phẩm đến với khách hàng tiềm năng.
 - Hỗ trợ quản lý đơn đặt, phản hồi từ khách hàng, và nâng cao chất lượng dịch vụ thông qua các công cụ phân tích và đánh giá.
- Đối với cộng đồng yêu thích du lịch:
 - Tạo ra một không gian mở để chia sẻ kinh nghiệm, bài viết, và đánh giá về các điểm đến.
 - Khuyến khích sự tương tác và kết nối giữa những người có cùng đam mê du lịch.
- Đối với nhà phát triển và quản trị trang web:
 - Xây dựng một hệ sinh thái du lịch trực tuyến bền vững, có khả năng mở rộng và phát triển trong tương lai.
 - Thu thập dữ liệu và phản hồi từ người dùng để không ngừng cải thiện chất lượng dịch vụ và trải nghiệm người dùng.

1.2 Cơ sở lý thuyết và các thông tin kỹ thuật:

1.2.1 Các Kiến Thức Lý Thuyết Vận Dụng cho Việc Hiện Thực Website:

Để xây dựng và vận hành một trang web như TourX, các kiến thức lý thuyết sau đây đã được áp dụng:

- Thiết kế giao diện người dùng (UI - User Interface):
 - Nguyên tắc thiết kế trực quan, đảm bảo giao diện dễ sử dụng, thân thiện với người dùng.
 - Áp dụng các quy tắc về màu sắc, bố cục, và typography để tạo sự hài hòa và thu hút.
- Trải nghiệm người dùng (UX - User Experience):
 - Tối ưu hóa quy trình tương tác của người dùng, từ tìm kiếm thông tin đến đặt chỗ và thanh toán.
 - Đảm bảo tính nhất quán và dễ dàng điều hướng trên mọi thiết bị (desktop, mobile, tablet).
- Công nghệ phát triển web:
 - Sử dụng các ngôn ngữ lập trình như JavaScript để xây dựng giao diện front-end.

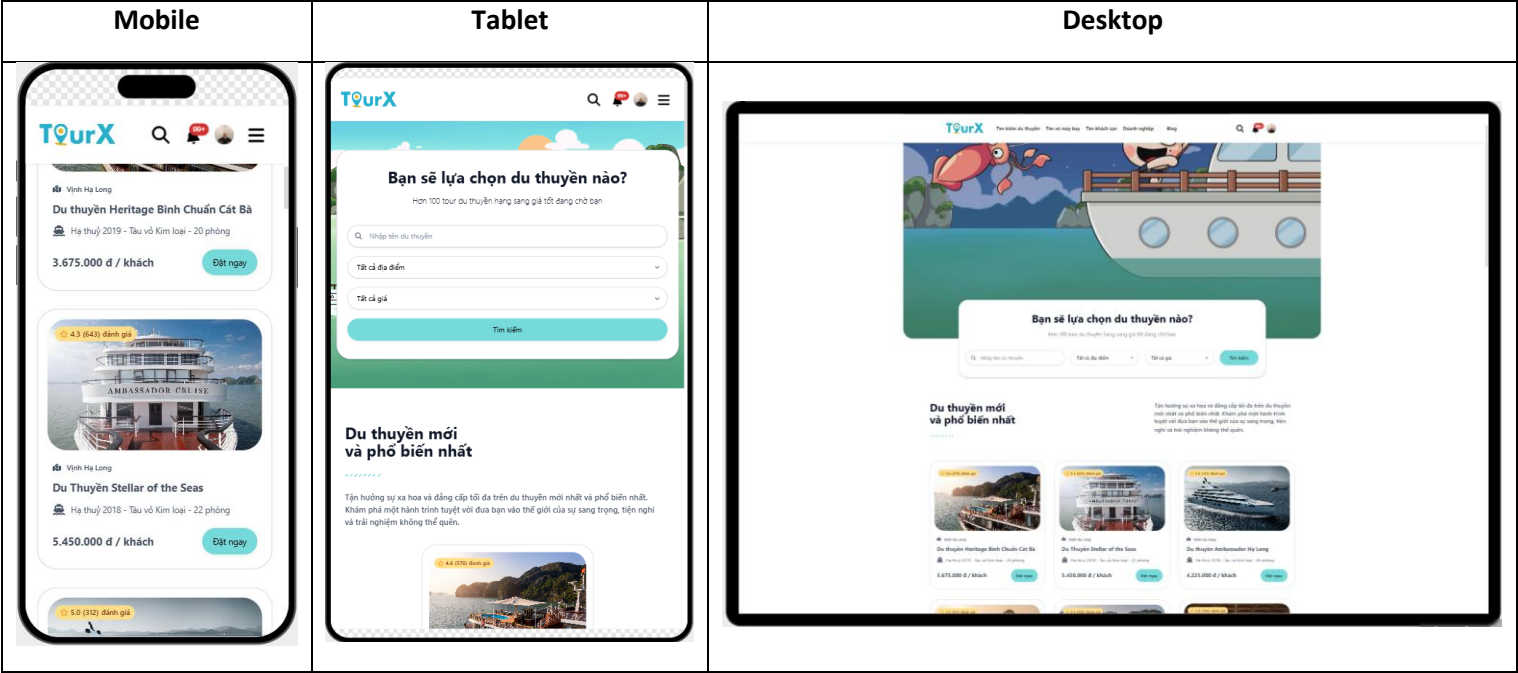
- Áp dụng các framework và thư viện hiện đại như React, TailwindCSS để tăng hiệu suất và khả năng tương tác.

1.2.2 Lý Do và Ưu Điểm của Việc Áp Dụng:

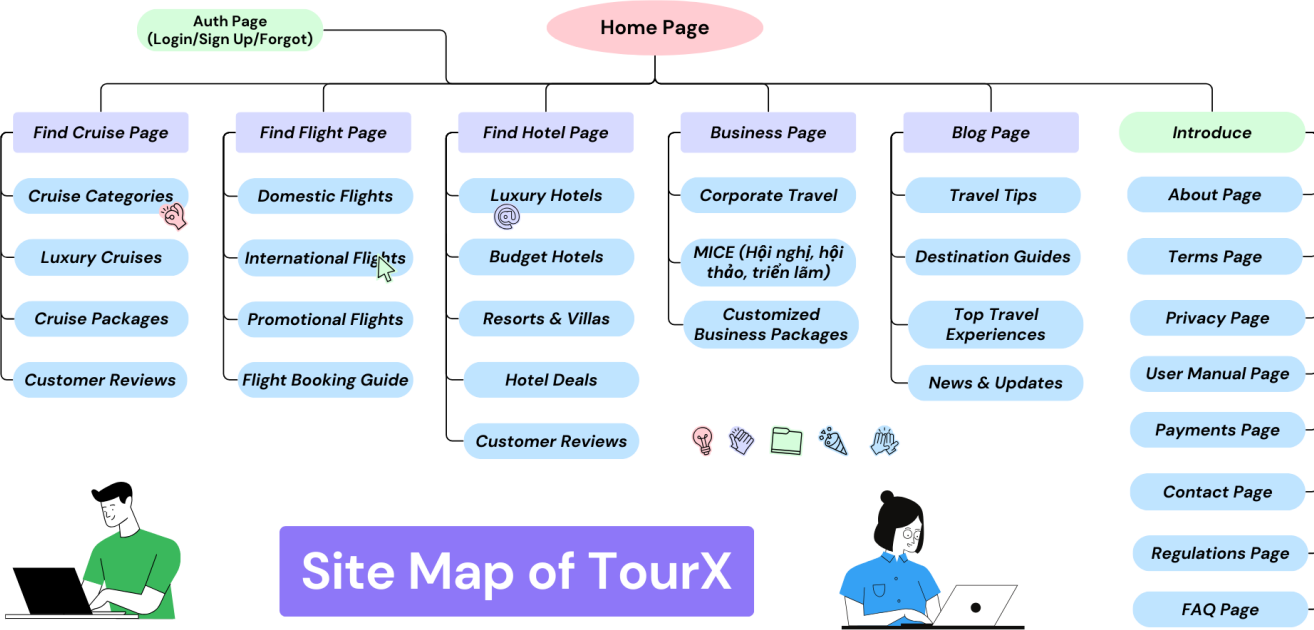
- Lý do áp dụng các kiến thức lý thuyết:
 - Đáp ứng nhu cầu người dùng: Các kiến thức về UI/UX giúp tạo ra một trang web dễ sử dụng, thu hút và giữ chân người dùng.
 - Nâng cao hiệu suất: Các kỹ thuật tối ưu hóa và quản lý cơ sở dữ liệu giúp trang web hoạt động ổn định và hiệu quả, ngay cả khi lượng truy cập lớn.
 - Đảm bảo an toàn thông tin: Bảo mật là yếu tố quan trọng để xây dựng lòng tin với người dùng, đặc biệt khi xử lý thông tin cá nhân và thanh toán trực tuyến.
- Ưu điểm của việc áp dụng:
 - Tăng trải nghiệm người dùng: Giao diện thân thiện và quy trình đơn giản giúp người dùng dễ dàng tìm kiếm và đặt chỗ, từ đó tăng tỷ lệ chuyển đổi.
 - Khả năng mở rộng: Công nghệ hiện đại và cơ sở dữ liệu được thiết kế tốt cho phép trang web dễ dàng mở rộng khi cần thiết.
 - Tối ưu hóa chi phí: Việc áp dụng các công nghệ phù hợp giúp giảm thiểu chi phí vận hành và bảo trì trong dài hạn.

CHƯƠNG 2. XÂY DỰNG ỨNG DỤNG

2.1 Khả năng tương thích đa thiết bị (Responsive)

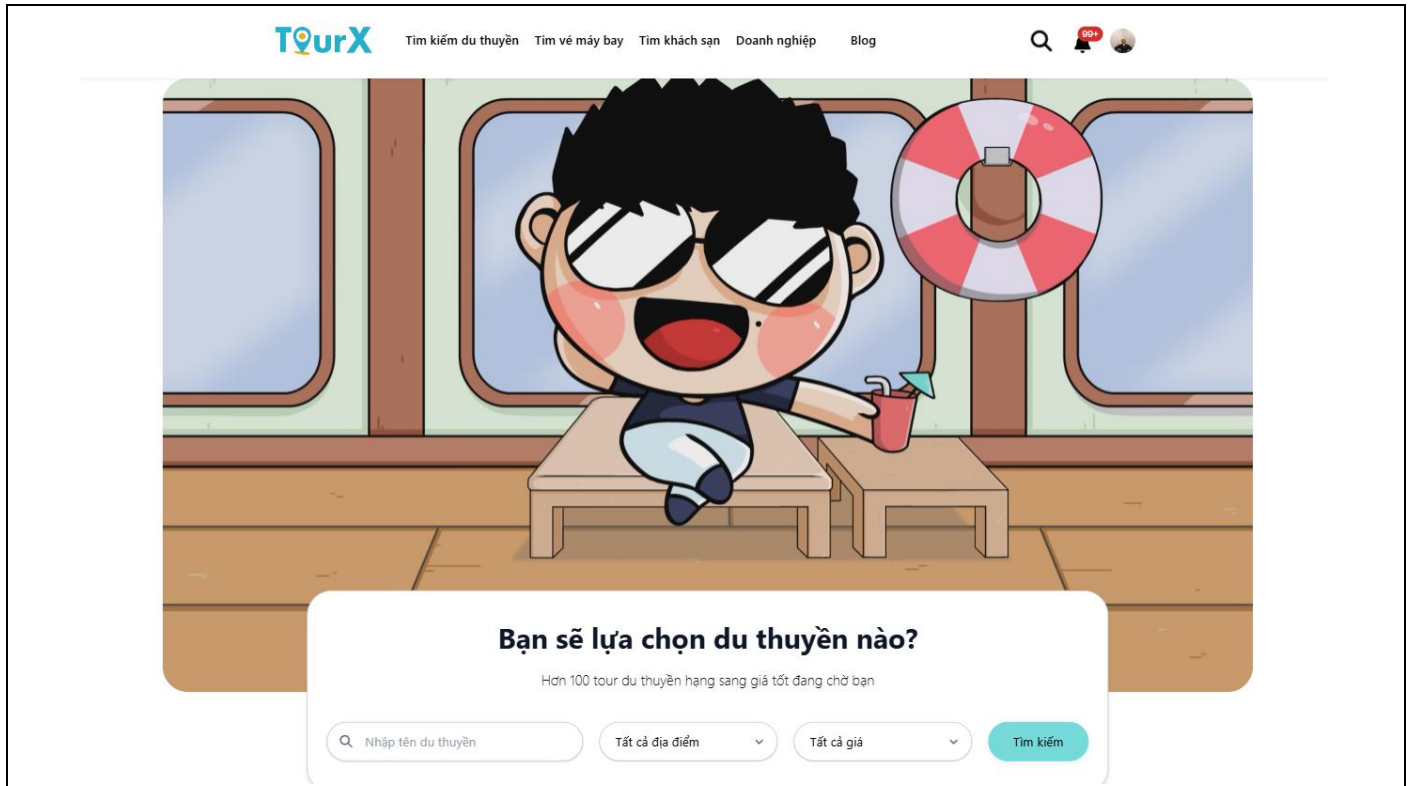


2.2 Sitemap

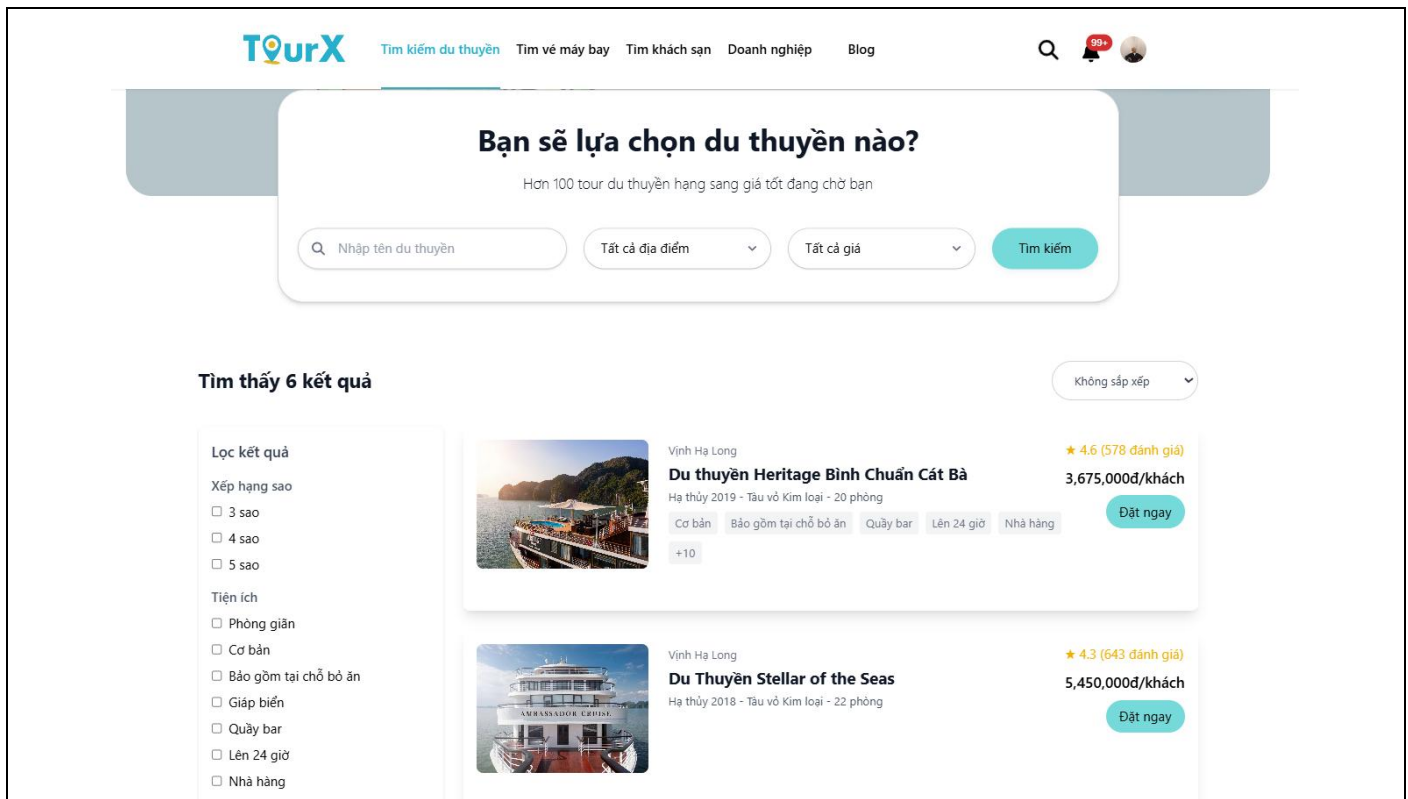


2.3 Giao diện

2.3.1 Trang chủ



2.3.2 Tìm du thuyền



2.3.3 Tìm vé máy bay

TourX

Tìm kiếm du thuyền

Tim vé máy bay

Tim khách sạn

Doanh nghiệp

Blog

🔍

🛎️ 99+

👤

Mở cánh cửa khám phá cùng TourX

TourX - Đặt chân lên đỉnh mây với một bước nhảy

☒ Một chiều

☐ Khứ hồi

☐ Vé rẻ nhất tháng

Điểm đi

Vui lòng nhập điểm đi

Điểm đến

Vui lòng nhập điểm đến

Ngày đi

20/03/2025

Người lớn

0

Trẻ em

0

Em bé

0

Tim kiếm

Đánh giá từ những người đã trải nghiệm


Khách hàng chia sẻ về những kỷ niệm tuyệt vời trên chuyến du lịch với chúng tôi.

“




Lần đầu chị đặt vé bay đi nước ngoài bên em và cảm thấy rất hài lòng! Chị cảm ơn bên em tư vấn cho chị chuyển bay, giá bay đẹp, thời gian chuyển hợp lý, không bị mệt. Chắc chắn chị sẽ đặt vé bên em nhiều lần nữa.

CHỊ GIANG -

2.3.4 Tìm khách sạn

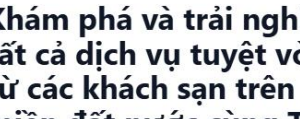


[Tìm kiếm du thuyền](#)
[Tìm vé máy bay](#)
[Tìm khách sạn](#)
[Doanh nghiệp](#)
[Blog](#)

Khám phá và trải nghiệm tất cả dịch vụ tuyệt vời nhất từ các khách sạn trên mọi miền đất nước cùng TourX.

Không gian nghỉ dưỡng sang trọng, tiện nghi và hiện đại cùng dịch vụ chuyên nghiệp, TourX tự hào mang đến những trải nghiệm hoàn hảo cho kỳ nghỉ của bạn, giúp bạn tận hưởng từng khoảnh khắc một cách đáng nhớ và trọn vẹn nhất!



4.6 (764) đánh giá

Hạ Long


Citadines Marina Hạ Long

Phòng 326

1.950.000 Đ / PHÒNG

1.550.000 đ / phòng

Đặt ngay



4.0 (32) đánh giá

Hạ Long

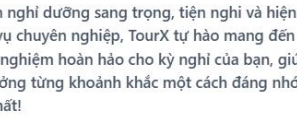
Anya Hotel Quy Nhơn

Phòng 580

2.000.000 Đ / PHÒNG

1.600.000 đ / phòng

Đặt ngay



4.9 (124) đánh giá

Hạ Long

Grand Tourane Hotel Đà Nẵng

Phòng 354

2.700.000 Đ / PHÒNG

2.300.000 đ / phòng

Đặt ngay

2.3.5 Trang chi tiết dịch vụ

[Tìm kiếm du thuyền](#)

[Tìm vé máy bay](#)

[Tìm khách sạn](#)

[Doanh nghiệp](#)

[Blog](#)

Grand Tourane Hotel Đà Nẵng

★ 4.9 (124 đánh giá) [Hạ Long](#) [Xem bản đồ và vị trí](#)

2.300.000 đ/đêm

Đặc điểm nổi bật

Wifi miễn phí

Nhà hàng

Quầy bar

Lễ tân 24 giờ

Bữa sáng miễn phí

Bồn tắm

- ✓ Khách sạn được thiết kế với phong cách hiện đại và sang trọng
- ✓ Phòng ngủ tiện nghi sang trọng với tầm nhìn panorama ra biển và thành phố
- ✓ Đặc biệt hơn, khách sạn có hồ bơi vô cực trên tầng thượng là địa điểm checkin yêu thích của mọi du khách
- ✓ Vị trí đắc địa, chỉ cách bãi biển 5 phút đi bộ và gần các điểm tham quan nổi tiếng

Thông tin khách sạn

Xây dựng	2019
Số phòng	120
Nhận phòng	14:00
Trả phòng	12:00
Quản lý	Citadines Marina Hạ Long

Các loại phòng & giá

Xoá lựa chọn

Phòng Deluxe

28 m² Tối đa: 1

1.200.000 đ
/ĐÊM

—

0

+

Phòng Premium

32 m² Tối đa: 1

1.450.000 đ
/ĐÊM

—

0

+

CHƯƠNG 3. KIỂM THỬ ỨNG DỤNG BẰNG VITEST

3.1 Test Case 01: Validate Contact Form Submission

```
import { render, screen, fireEvent, waitFor } from "@testing-library/react";
import { test, expect, vi } from "vitest";
import ContactForm from "../ContactForm"; // Đường dẫn có thể thay đổi tùy theo dự án

// Mock emailjs để không gọi API thật
vi.mock("@emailjs/browser", () => ({
  send: vi.fn(() => Promise.resolve("Email sent successfully")),
}));

test("Validate Contact Form Submission", async () => {
  const mockOnSuccess = vi.fn();
  render(<ContactForm onSuccess={mockOnSuccess} />);

  // Lấy các input
  const nameInput = screen.getByPlaceholderText("Nhập họ và tên");
  const emailInput = screen.getByPlaceholderText("Nhập email");
  const phoneInput = screen.getByPlaceholderText("Nhập số điện thoại");
  const messageInput = screen.getByPlaceholderText("Nhập yêu cầu của bạn");
  const submitButton = screen.getByRole("button", { name: /Gửi yêu cầu đến TourX/i });

  // Kiểm tra input thay đổi giá trị đúng
  fireEvent.change(nameInput, { target: { value: "Nguyễn Văn A" } });
  fireEvent.change(emailInput, { target: { value: "email@example.com" } });
  fireEvent.change(phoneInput, { target: { value: "0987654321" } });
  fireEvent.change(messageInput, { target: { value: "Tôi muốn đặt tour" } });

  expect(nameInput.value).toBe("Nguyễn Văn A");
  expect(emailInput.value).toBe("email@example.com");
  expect(phoneInput.value).toBe("0987654321");
  expect(messageInput.value).toBe("Tôi muốn đặt tour");

  // Gửi form
  fireEvent.click(submitButton);

  // Đợi xử lý API giả lập và kiểm tra onSuccess được gọi
  await waitFor(() => expect(mockOnSuccess).toHaveBeenCalled());

  // Kiểm tra các input có reset không
  expect(nameInput.value).toBe("");
  expect(emailInput.value).toBe("");
  expect(phoneInput.value).toBe("");
  expect(messageInput.value).toBe("");
});
```

PASS

src/__tests__/ContactForm.test.jsx > Validate Contact Form Submission

Test Files 1 passed (1)

Tests 1 passed (1)

3.2 Test Case 02: Validate Contact Form Empty Fields

```
import { render, screen, fireEvent } from "@testing-library/react";
import { test, expect } from "vitest";
import ContactForm from "../ContactForm"; // Đường dẫn có thể thay đổi tùy theo dự án

test("Validate Contact Form Empty Fields", async () => {
  render(<ContactForm />);

  // Lấy nút submit
  const submitButton = screen.getByRole("button", { name: /Gửi yêu cầu đến TourX/i });

  // Click submit mà không nhập gì
  fireEvent.click(submitButton);

  // Kiểm tra thông báo lỗi cho từng field
  expect(screen.getByText("Vui lòng nhập họ và tên")).toBeInTheDocument();
  expect(screen.getByText("Vui lòng nhập email")).toBeInTheDocument();
  expect(screen.getByText("Vui lòng nhập số điện thoại")).toBeInTheDocument();
  expect(screen.getByText("Vui lòng nhập nội dung")).toBeInTheDocument();
});
```

PASS

src/__tests__/ContactForm.test.jsx > Validate Contact Form Empty Fields

Test Files 1 passed (1)

Tests 1 passed (1)

3.3 Test Case 03: Validate Email Format in Contact Form

<pre>import { render, screen, fireEvent } from "@testing-library/react"; import { test, expect } from "vitest"; import ContactForm from "../ContactForm"; // Đường dẫn có thể thay đổi tùy theo dự án test("Validate Email Format in Contact Form", async () => { render(<ContactForm />); // Lấy input email và nút submit const emailInput = screen.getByPlaceholderText("Nhập email"); const submitButton = screen.getByRole("button", { name: /Gửi yêu cầu đến TourX/i }); // Nhập email không hợp lệ fireEvent.change(emailInput, { target: { value: "invalid-email" } }); // Click submit fireEvent.click(submitButton); // Kiểm tra thông báo lỗi email không hợp lệ expect(screen.getByText("Email không hợp lệ")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/ContactForm.test.jsx > Validate Email Format in Contact Form</p> <p>Test Files 1 passed (1)</p> <p>Tests 1 passed (1)</p>
---	---

3.4 Test Case 04: Validate Phone Number Format in Contact Form

<pre>import { render, screen, fireEvent } from "@testing-library/react"; import { test, expect } from "vitest"; import ContactForm from "../ContactForm"; // Đường dẫn có thể thay đổi tùy theo dự án test("Validate Phone Number Format in Contact Form", async () => { render(<ContactForm />); // Lấy input số điện thoại và nút submit const phoneInput = screen.getByPlaceholderText("Nhập số điện thoại"); const submitButton = screen.getByRole("button", { name: /Gửi yêu cầu đến TourX/i }); // Nhập số điện thoại không hợp lệ (chứa chữ) fireEvent.change(phoneInput, { target: { value: "123abc456" } }); // Click submit fireEvent.click(submitButton); // Kiểm tra thông báo lỗi số điện thoại không hợp lệ expect(screen.getByText("Số điện thoại phải là số")).toBeInTheDocument(); });</pre>	<p>FAIL src/__tests__/ContactForm.test.jsx > Fail case: Accepts invalid phone number format</p> <ul style="list-style-type: none">● Fail case: Accepts invalid phone number format <p>Unable to find an element with the text: Số điện thoại phải là số.</p>
---	---

3.5 Test Case 05: Check if CruiseForm renders and displays correct content

<pre>import { render, screen } from "@testing-library/react"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("CruiseForm renders without crashing", () => { render(<CruiseForm to="/search" />); expect(screen.getByText("Bạn sẽ lựa chọn du thuyền nào?")).toBeInTheDocument(); expect(screen.getByPlaceholderText("Nhập tên du thuyền")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormRender.test.jsx > CruiseForm renders without crashing</p> <p>Test Files 1 passed (1)</p> <p>Tests 1 passed (1)</p>
---	--

3.6 Test Case 06: Test users can enter yacht name into search box

<pre>CruiseFormInput.test.jsx import { render, screen, fireEvent } from "@testing-library/react"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("User can type into the search input", () => { render(<CruiseForm to="/search" />); const searchInput = screen.getByPlaceholderText("Nhập tên du thuyền") fireEvent.change(searchInput, { target: { value: "Elite Cruise" } }); expect(searchInput.value).toBe("Elite Cruise"); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormInput.test.jsx > User can type into the search input</p> <p>Test Files 1 passed (1)</p> <p>Tests 1 passed (1)</p>
--	---

3.7 Test Case 07: Check if the "location" and "price" dropdowns are displaying correctly

<pre>CruiseFormSelect.test.jsx x import { render, screen } from "@testing-library/react"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("Select menus should render", () => { render(<CruiseForm to="/search" />); expect(screen.getByText("địa điểm")).toBeInTheDocument(); expect(screen.getByText("giá")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormSelect.test.jsx > Select menus should render</p> <p>Test Files 1 passed (1)</p> <p>Tests 1 passed (1)</p>
---	---

3.8 Test Case 08: Is "Search" displayed and navigated correctly?

<pre>CruiseFormButton.test.jsx import { render, screen } from "@testing-library/react"; import { MemoryRouter } from "react-router-dom"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("Search button should render and navigate correctly", () => { render(<MemoryRouter> <CruiseForm to="/search" /> </MemoryRouter>); const searchButton = screen.getByRole("link", { name: /Tìm kiếm/i }); expect(searchButton).toBeInTheDocument(); expect(searchButton).toHaveAttribute("href", "/search"); });</pre>	<p>PASS src/__tests__/CruiseFormButton.test.jsx > Search button should render and navigate correctly</p> <p>Test Files 1 passed (1)</p> <p>Tests 1 passed (1)</p>
---	--

3.9 Test Case 09: Test when clicking "Search", the page will scroll to the top

<pre>CruiseFormScroll.test.jsx import { render, screen, fireEvent } from "@testing-library/react"; import { MemoryRouter } from "react-router-dom"; import { test, expect, vi } from "vitest"; import CruiseForm from "../CruiseForm"; test("Clicking the search button should scroll to top", () => { window.scrollTo = vi.fn(); // Mock scrollTo render(<MemoryRouter> <CruiseForm to="/search" /> </MemoryRouter>); const searchButton = screen.getByRole("link", { name: /Tìm kiếm/i }); fireEvent.click(searchButton); expect(window.scrollTo).toHaveBeenCalledWith({ top: 0, behavior: "smooth" }); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormScroll.test.js</p> <p>x > Clicking the search button should scroll to top</p> <p>Test Files 1 passed (1)</p> <p>Tests 1 passed (1)</p>
---	--

3.10 Test Case 10: Check FlightForm renders and displays correct content

<pre>FlightFormRender.test.jsx x import { render, screen } from "@testing-library/react"; import { test, expect } from "vitest"; import FlightForm from "../FlightForm"; test("FlightForm renders without crashing", () => { render(<FlightForm />); expect(screen.getByText("Mô cánh cửa khám phá cùng TourX")).toBeInTheDocument(); expect(screen.getByText("Một chiều")).toBeInTheDocument(); expect(screen.getByText("Khứ hồi")).toBeInTheDocument(); expect(screen.getByText("Tìm kiếm")).toBeInTheDocument(); });</pre>	<p>PASS src/__tests__/FlightFormRender.test.jsx</p> <p>> FlightForm renders without crashing</p>
---	---

3.11 Test Case 11: Check the departure and destination points, they must not be the same.

<pre>FlightFormLocation.test.jsx x import { render, screen, fireEvent } from "@testing-library/react"; import { test, expect } from "vitest"; import FlightForm from "../FlightForm"; test("User selects departure and arrival locations", () => { render(<FlightForm />); const departureInput = screen.getByPlaceholderText("Vui lòng nhập điểm đi"); const arrivalInput = screen.getByPlaceholderText("Vui lòng nhập điểm đến"); fireEvent.change(departureInput, { target: { value: "Hà Nội" } }); fireEvent.change(arrivalInput, { target: { value: "Hồ Chí Minh" } }); expect(departureInput.value).toBe("Hà Nội"); expect(arrivalInput.value).toBe("Hồ Chí Minh"); // Kiểm tra nếu người dùng chọn điểm đi và đến giống nhau fireEvent.change(arrivalInput, { target: { value: "Hà Nội" } }); fireEvent.click(screen.getByText("Tìm kiếm")); expect(screen.getByText("Điểm đi và điểm đến không thể trùng nhau!")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/FlightFormLocation.test.jsx ></p> <p>User selects departure and arrival locations</p>
--	---

3.12 Test Case 12: Check that the departure date is not less than the current date and the return date is after the departure date.

<pre>FlightFormDate.test.jsx x import { render, screen, fireEvent } from "@testing-library/react"; import { test, expect } from "vitest"; import dayjs from "dayjs"; import FlightForm from "../FlightForm"; test("Validate departure and return dates", () => { render(<FlightForm />); const departureInput = screen.getByLabelText("Ngày đi"); // Chọn ngày đi nhỏ hơn ngày hiện tại fireEvent.change(departureInput, { target: { value: dayjs().subtract(2, "day").format("YYYY-MM-DD") } }); fireEvent.click(screen.getByText("Tìm kiếm")); expect(screen.getByText("Ngày đi không được nhỏ hơn ngày hiện tại!")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/FlightFormDate.test.jsx > Validate departure and return dates</p>
---	---

3.13 Test Case 13: FlightForm Passenger Count Test

<pre>FlightFormPassengers.test.jsx x import { render, screen, fireEvent } from "@testing-library/react"; import { test, expect } from "vitest"; import FlightForm from "../FlightForm"; test("Validate passenger count", () => { render(<FlightForm />); const adultsInput = screen.getByLabelText("Người lớn"); // Đặt số lượng người lớn về 0 và bấm tìm kiếm fireEvent.change(adultsInput, { target: { value: "0" } }); fireEvent.click(screen.getByText("Tìm kiếm")); expect(screen.getByText("Phải có ít nhất một người lớn!")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/FlightFormPassengers.test.js x > Validate passenger count</p>
---	---

3.14 Test Case 14: FlightForm Search Button Test

<pre>FlightFormSearch.test.jsx import { render, screen, fireEvent } from "@testing-library/react"; import { test, expect, vi } from "vitest"; import FlightForm from "../FlightForm"; // Mock toast để kiểm tra thông báo vi.mock("react-toastify", () => ({ toast: { success: vi.fn(), error: vi.fn(), }, })); test("Search button validates the form and shows success message if valid", () => { render(<FlightForm />); const departureInput = screen.getByPlaceholderText("Vui lòng nhập điểm đi"); const arrivalInput = screen.getByPlaceholderText("Vui lòng nhập điểm đến"); const adultsInput = screen.getByLabelText("Người lớn"); fireEvent.change(departureInput, { target: { value: "Hà Nội" } }); fireEvent.change(arrivalInput, { target: { value: "Hồ Chí Minh" } }); fireEvent.change(adultsInput, { target: { value: "1" } }); fireEvent.click(screen.getByText("Tìm kiếm")); expect(screen.getByText("Form hợp lệ, tiến hành tìm kiếm...")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/FlightFormSearch.test.jsx > Search button validates the form and shows success message if valid</p>
--	---

3.15 Test Case 15: CruiseForm Input Placeholder Test

<pre>import { render, screen } from "@testing-library/react"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("Search input should have correct placeholder", () => { render(<CruiseForm />); const searchInput = screen.getByPlaceholderText("Nhập tên du thuyền"); expect(searchInput).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormPlaceholder.test.jsx > Search input should have correct placeholder</p>
--	---

3.16 Test Case 16: CruiseForm Dropdown Render Test

<pre>import { render, screen } from "@testing-library/react"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("Dropdowns for location and price should render", () => { render(<CruiseForm />); expect(screen.getByText("địa điểm")).toBeInTheDocument(); expect(screen.getByText("giá")).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormDropdown.test.jsx > Dropdowns for location and price should render</p>
---	--

3.17 Test Case 17: CruiseForm Search Button Test

<pre>import { render, screen } from "@testing-library/react"; import { MemoryRouter } from "react-router-dom"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("Search button should render correctly", () => { render(<MemoryRouter> <CruiseForm to="/search" /> </MemoryRouter>); const searchButton = screen.getByRole("link", { name: /Tìm kiếm/ }); expect(searchButton).toBeInTheDocument(); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormSearchButton.test.jsx > Search button should render correctly</p>
---	---

3.18 Test Case 18: CruiseForm Scroll Behavior Test

<pre>CruiseFormScroll.test.jsx x import { render, screen, fireEvent } from "@testing-library/react"; import { MemoryRouter } from "react-router-dom"; import { test, expect, vi } from "vitest"; import CruiseForm from "../CruiseForm"; test("Clicking the search button should scroll to top", () => { window.scrollTo = vi.fn(); // Mock scrollTo render(<MemoryRouter> <CruiseForm to="/search" /> </MemoryRouter>); const searchButton = screen.getByRole("Link", { name: /Tìm kiếm/i }); fireEvent.click(searchButton); expect(window.scrollTo).toHaveBeenCalledWith({ top: 0, behavior: "smooth" }); });</pre>	<p>PASS</p> <p>src/__tests__/CruiseFormSearchButton.test.jsx > Search button should render correctly</p>
---	---

3.19 Test Case 19: CruiseForm Link Navigation Test

<pre>CruiseFormNavigation.test.jsx x import { render, screen } from "@testing-library/react"; import { MemoryRouter } from "react-router-dom"; import { test, expect } from "vitest"; import CruiseForm from "../CruiseForm"; test("Search button should navigate to the correct page", () => { render(<MemoryRouter> <CruiseForm to="/search" /> </MemoryRouter>); const searchButton = screen.getByRole("Link", { name: /Tìm kiếm/i }); expect(searchButton).toHaveAttribute("href", "/search"); });</pre>	<p>PASS src/__tests__/CruiseFormNavigation.test.jsx > Search button should navigate to the correct page</p>
--	--

3.20 Test Case 20: Kiểm tra Button render đúng thẻ khi có to hoặc href

<pre>ButtonLink.test.jsx import { render, screen } from "@testing-library/react"; import { MemoryRouter } from "react-router-dom"; import { test, expect } from "vitest"; import Button from "../Button"; test("Button should render as a Link when 'to' prop is provided", () => { render(<MemoryRouter> <Button to="/about">About Us</Button> </MemoryRouter>); const linkElement = screen.getByRole("Link", { name: /About Us/i }); // Kiểm tra nút có được render dưới dạng <Link> không expect(linkElement).toBeInTheDocument(); expect(linkElement).toHaveAttribute("href", "/about"); }); test("Button should render as an anchor tag when 'href' prop is provided", () => { render(<Button href="https://example.com">Go to Example</Button>); const anchorElement = screen.getByRole("Link", { name: /Go to Example/i }); // Kiểm tra nút có được render dưới dạng <a> không expect(anchorElement).toBeInTheDocument(); expect(anchorElement).toHaveAttribute("href", "https://example.com"); });</pre>	<p>PASS src/__tests__/ButtonLink.test.jsx > Button should render as a Link when 'to' prop is provided</p> <p>PASS src/__tests__/ButtonLink.test.jsx > Button should render as an anchor tag when 'href' prop is provided</p> <p>Test Files 1 passed (1)</p> <p>Tests 2 passed (2)</p>
---	---

CHƯƠNG 4. KẾT LUẬN – HƯỚNG PHÁT TRIỂN

4.1 Kết quả đạt được

Sau quá trình nghiên cứu, phát triển và triển khai, ứng dụng web TourX đã đạt được những kết quả đáng kể như sau:

- Xây dựng thành công một nền tảng du lịch trực tuyến: TourX đã trở thành một trang web cung cấp dịch vụ du lịch toàn diện, từ tìm kiếm thông tin, đặt tour, đến thanh toán và chia sẻ kinh nghiệm.
- Giao diện thân thiện và dễ sử dụng: Nhờ áp dụng các nguyên tắc trong thiết kế UI/UX của môn học “*Tư duy thiết kế và trải nghiệm người dùng*” và “*Phát triển giao diện ứng dụng*”, trang web mang lại trải nghiệm mượt mà và thuận tiện cho người dùng trên mọi thiết bị.
- Xây dựng cộng đồng du lịch sôi động: Tính năng chia sẻ đánh giá và kinh nghiệm đã tạo ra một cộng đồng người dùng tích cực, góp phần nâng cao giá trị của trang web.
- Hợp tác với nhiều đối tác du lịch: TourX đã thiết lập được mạng lưới đối tác đa dạng, từ công ty lữ hành đến khách sạn và nhà cung cấp dịch vụ du lịch.

4.2 Hạn chế của ứng dụng web

Mặc dù đạt được nhiều kết quả tích cực, TourX vẫn còn một số hạn chế cần khắc phục:

- Hạn chế về tính năng cá nhân hóa: Mặc dù đã có tính năng gợi ý tour, nhưng hệ thống vẫn chưa thực sự cá nhân hóa mạnh mẽ dựa trên hành vi và sở thích của người dùng.
- Bảo mật dữ liệu: Mặc dù đã áp dụng các biện pháp bảo mật cơ bản, nhưng vẫn cần nâng cao hơn nữa để đảm bảo an toàn tuyệt đối cho thông tin người dùng.

4.3 Hướng phát triển

Để khắc phục những hạn chế và tiếp tục phát triển, TourX có thể tập trung vào các hướng đi sau:

- Cải thiện hiệu suất và tốc độ tải trang:
 - Tối ưu hóa mã nguồn, sử dụng công nghệ caching và CDN để tăng tốc độ tải trang.
 - Nén hình ảnh và tài nguyên để giảm thiểu dung lượng tải xuống.
- Nâng cao trải nghiệm người dùng trên thiết bị di động:
 - Phát triển ứng dụng di động riêng để mang lại trải nghiệm tốt hơn cho người dùng smartphone.
 - Tối ưu hóa giao diện responsive để đảm bảo tương thích với mọi thiết bị.
- Tăng cường tính năng cá nhân hóa:
 - Áp dụng trí tuệ nhân tạo (AI) và học máy (Machine Learning) để phân tích hành vi người dùng và đưa ra gợi ý phù hợp.
 - Xây dựng hệ thống đề xuất tour dựa trên sở thích, lịch sử tìm kiếm, và đánh giá của người dùng.
- Nâng cao bảo mật và quyền riêng tư:

- Triển khai các biện pháp bảo mật nâng cao như xác thực hai yếu tố (2FA) và mã hóa dữ liệu nhạy cảm.
- Tuân thủ các tiêu chuẩn bảo mật quốc tế như GDPR để đảm bảo quyền riêng tư của người dùng.
- Mở rộng hệ thống đối tác và dịch vụ:
 - Hợp tác với nhiều đối tác du lịch quốc tế để cung cấp các gói tour đa dạng và hấp dẫn hơn.
 - Phát triển thêm các dịch vụ đi kèm như đặt vé máy bay, thuê xe, và bảo hiểm du lịch.
- Cải thiện quản lý nội dung người dùng:
 - Xây dựng hệ thống kiểm duyệt tự động bằng AI để lọc và quản lý đánh giá, bài viết từ người dùng.
 - Khuyến khích người dùng đóng góp nội dung chất lượng bằng các chương trình khuyến mãi hoặc điểm thưởng.
- Phát triển cộng đồng du lịch:
 - Tổ chức các sự kiện trực tuyến và offline để kết nối người dùng, tạo ra một cộng đồng du lịch gắn kết.
 - Xây dựng các chuyên mục chia sẻ kinh nghiệm, tips du lịch, và hướng dẫn chi tiết về các điểm đến.

CHƯƠNG 5. TÀI LIỆU THAM KHẢO

5.1 Giáo trình – Sách

- *Hệ thống và công nghệ Web* – (ThS Nguyễn Thị Hồng Lương)
- *Tư duy thiết kế và trải nghiệm người dùng* – (ThS Trần Thế Trung)
- *Phát triển giao diện ứng dụng* – (ThS Từ Thị Xuân Hiền)

5.2 Website

Tên website tham khảo	Nguồn
Mixivivu	https://mixivivu.com/

CHƯƠNG 6. Mã nguồn dự án và sản phẩm triển khai

6.1 Giới thiệu dự án (Introduce, Front-end, Back-end)

- [Introduce 1 - Project](#)
- [Introduce 2 - Front end](#)
- [Introduce 3 - Backend](#)

6.2 Mã nguồn dự án (Source Code)

- <https://github.com/nvminh162/tourx-app>