TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN MÔN LẬP TRÌNH TRỰC QUAN

Tên đề tài: THIẾT KẾ GIAO DIỆN TRANG WEB QUẢN LÝ CHI TIÊU

Giảng viên hướng dẫn: TS. HỒNG TRUNG DŨNG
Học viên thực hiện: TRƯƠNG LÊ PHƯỚC LỘC
MSSV: 2311560589
Khoá: 2023
Ngành/ chuyên ngành: CÔNG NGHỆ THÔNG TIN

Tp HCM, tháng 09 năm 2024

TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN MÔN LẬP TRÌNH TRỰC QUAN

Tên đề tài: THIẾT KẾ GIAO DIỆN TRANG WEB QUẢN LÝ CHI TIÊU

Giảng viên hướng dẫn: TS. HỒNG TRUNG DŨNG
Học viên thực hiện: TRƯƠNG LÊ PHƯỚC LỘC
MSSV: 2311560589
Khoá: 2023
Ngành/ chuyên ngành: CÔNG NGHỆ THÔNG TIN

Tp HCM, tháng 09 năm 2024

LỜI CẢM ƠN

Tôi xin chân thành cảm ơn Thầy hướng dẫn TS. Hồng Trung Dũng người đã hướng dẫn tận tình, đóng góp ý kiến chuyên môn cho bài tiểu luận của tôi. Thầy đã động viên tinh thần giúp tôi cố gắng hoàn thành những nghiên cứu đặt ra. Thầy cũng cung cấp một số tài liệu liên quan đến tiểu luận mà chúng tôi đang nghiên cứu và ân cần nhắc nhở tôi đến tiến độ thực hiện bài tiểu luận này.

Tôi cũng chân thành gởi lời cảm ơn đến quý Thầy đã giảng dạy, hướng dẫn tôi trong suốt thời gian qua, từ đó giúp trang bị các kiến thức cho tôi trong năm học vừa qua, từ các kiến thức cơ bản đến các vấn đề chuyên sâu.

Tôi xin gửi lời cảm ơn đến anh, chị, bạn bè, đồng nghiệp bằng nhiều hình thức khác nhau đã giúp đỡ tôi trong quá trình học tập tại trường cũng như trong thời gian hoàn thành tiểu luận.

Đặc biệt, xin gởi lời cảm ơn đến gia đình đã động viên tinh thần cũng như chia sẻ khó khăn trong thời gian qua.

Do kiến thức bản thân còn nhiều hạn chế, trong quá trình thực hiện và hoàn thiện bài tiểu luận này không tránh khỏi những sai sót, kính mong Thầy cho lời nhận xét và góp ý.

Xin chân thành cảm ơn!

Học viên thực hiện

Trương Lê Phước Lộc

LỜI MỞ ĐẦU

Trong thời đại số hóa ngày nay, quản lý chi tiêu cá nhân đã trở thành một trong những kỹ năng sống vô cùng quan trọng. Mỗi người chúng ta đều phải đối mặt với vô số khoản chi tiêu hàng ngày, từ tiền thuê nhà, hóa đơn, ăn uống, cho đến các khoản mua sắm và giải trí. Nếu không có một hệ thống quản lý tài chính hiệu quả, việc kiểm soát dòng tiền và đạt được các mục tiêu tài chính sẽ trở nên cực kỳ khó khăn.

May mắn thay, với sự phát triển chóng mặt của công nghệ, ngày nay chúng ta có thể sử dụng các trang web và ứng dụng quản lý chi tiêu trực tuyến để giải quyết vấn đề này một cách dễ dàng hơn. Những công cụ này không chỉ giúp chúng ta theo dõi và phân tích các khoản chi tiêu một cách chi tiết, mà còn cung cấp nhiều tính năng thông minh như lập ngân sách, cảnh báo chi tiêu, phân tích xu hướng và đề xuất giải pháp tiết kiệm. Tuy nhiên, để một trang web quản lý chi tiêu thực sự hiệu quả, việc thiết kế giao diện người dùng (UI) đóng vai trò vô cùng quan trọng.

Giao diện cần phải trực quan, dễ sử dụng và tối ưu hóa cho người dùng, giúp họ dễ dàng theo dõi, phân tích và quản lý tình hình tài chính cá nhân. Từ cấu trúc layout, lựa chọn màu sắc, typography, cho đến các tính năng và phân tích dữ liệu - mỗi yếu tố đều cần được thiết kế một cách cẩn thận để mang lại trải nghiệm người dùng tuyệt vời. Nếu giao diện không thân thiện với người dùng, người dùng sẽ nhanh chóng cảm thấy bị choáng ngợp và có thể từ bỏ việc quản lý tài chính một cách kỷ luật.

Trong tiểu luận này, tôi sẽ đi sâu khám phá các nguyên tắc thiết kế giao diện tối ưu cho trang web quản lý chi tiêu, từ những yếu tố cơ bản cho đến những tính năng nâng cao, nhằm giúp người dùng kiểm soát tài chính cá nhân một cách hiệu quả và đạt được các mục tiêu tài chính quan trọng. Tôi đã vận dụng các kiến thức đã học trong thiết kế UI, từ cấu trúc layout, lựa chọn màu sắc, typography, cho đến các tính năng như phân tích dữ liệu, lập ngân sách và đề xuất giải pháp tiết kiệm. Mục tiêu là mang lại một trải nghiệm người dùng tuyệt vời, giúp họ quản lý tài chính cá nhân một cách dễ dàng và hiệu quả hơn.

BM-ChT-

TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
TRUNG TÂM KHẢO THÍ

KỲ THI KẾT THÚC HỌC PHẦN
HỌC KỲ 2 NĂM HỌC 2023 - 2024

PHIẾU CHẤM THI TIỂU LUẬN/ĐỒ ÁN

Môn thi: Lập trình trực quan — Lớp học phần: 110107616101

Học viên thực hiện: Trương Lê Phước Lộc — Tham gia đóng góp: 100%

Ngày thi:..... Phòng thi: .....

Đề tài tiểu luận/báo cáo của học viên: **Thiết kế giao diện trang web quản lý chi tiêu.**

Phần đánh giá của giảng viên (căn cứ trên thang rubrics của môn học):

Tiêu chí (theo CĐR HP)	**Đánh giá của GV**	**Điểm tối đa**	**Điểm đạt được**
Cấu trúc của báo cáo	....		
Nội dung			
- Các nội dung thành phần	....		
- Lập luận	....		
- Kết luận	.....		
Trình bày	.....		
TỔNG ĐIỂM			

Giảng viên chấm thi
(ký, ghi rõ họ tên)

TS. Hồng Trung Dũng

MỤC LỤC

LỜI CẢM ƠN iii
LỜI MỞ ĐẦU iv
DANH MỤC HÌNH ẢNH viii
KÝ HIỆU CÁC CỤM TỪ VIẾT TẮT ix
CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN QUẢN LÝ CHI TIÊU 1
1.1 Giới thiệu chung 1
1.2 Tính cấp thiết của đề tài 2
1.3 Mục đích nghiên cứu của đề tài 3
1.4 Phương pháp nghiên cứu của đề tài 3
1.5 Phạm vi nghiên cứu của đề tài 4
1.6 Đối tượng nghiên cứu của đề tài 4
1.7 Bố cục khi trình bày 4
CHƯƠNG 2: TỔNG QUAN THIẾT KẾ GIAO DIỆN NGƯỜI DÙNG VÀ ỨNG DỤNG CHO THIẾT KẾ GIAO DIỆN TRANG WEB QUẢN LÝ CHI TIÊU 6
2.1 Khái niệm và vai trò của thiết kế giao diện người dùng (UI) 6
2.2 Các nguyên tắc cơ bản trong thiết kế UI 6
2.2.1 Sự đơn giản hóa và tính trực quan 6
2.2.2 Tính thông suốt và dễ sử dụng 6
2.2.3 Tính thống nhất và tính nhất quán 6
2.2.4 Tính phản hồi và tính tương tác 6
CHƯƠNG 3: KHAI THÁC SỬ DỤNG REACT JS VÀ TAILWIND CSS TRONG THIẾT KẾ GIAO DIỆN WEBSITE 7
3.1 Giới thiệu đôi nét về thiết kế giao diện website hiện đại 7
3.1.1 Bối cảnh và tầm quan trọng của việc xây dựng giao diện website hiện đại 7
3.1.2 Vai trò của ReactJS và Tailwind CSS trong thiết kế và phát triển giao diện website 7
3.2 Xây dựng giao diện website với ReactJS 7
3.2.1 Tổng quan về ReactJS 7
3.2.2 Ứng dụng ReactJS trong thiết kế giao diện 8
3.3 Tối ưu giao diện với Tailwind CSS 8
3.3.1 Tổng quan về Tailwind CSS 8
3.3.2 Ứng dụng Tailwind CSS trong thiết kế giao diện 8
3.4 Ưu điểm và thách thức khi sử dụng ReactJS và Tailwind CSS 8

3.5 Triển vọng và xu hướng phát triển của các công nghệ này trong tương lai 9
3.6 Gợi ý hướng phát triển và ứng dụng trong các dự án website khác 9
3.7 Cài đặt cấu hình cho dự án 9
3.7.1 Cài đặt Node.js 9
3.7.2 Tạo dự án React mới 15
CHƯƠNG 4: THIẾT KẾ GIAO DIỆN TRANG WEB CHO ỨNG DỤNG QUẢN LÝ CHI TIÊU 16
4.1 Cài đặt các thư viện cần thiết cho ứng dụng 16
4.2 Viết code cho ứng dụng 16
4.3 Kết quả 27
CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 30
5.1 Kết quả đạt được 30
5.2 Hướng phát triển của đề tài 30
DANH MỤC TÀI LIỆU THAM KHẢO 32

DANH MỤC HÌNH ẢNH

Hình 3. 1 Giao diện web chính thức của Node.js 9
Hình 3. 2 Phiên bản phù hợp với hệ điều hành để tải xuống 10
Hình 3. 3 Giao diện cài đặt Node.js 10
Hình 3. 4 Giao diện thỏa thuận cấp phép người dùng cuối 11
Hình 3. 5 Giao diện chọn thư mục đích để cài đặt 11
Hình 3. 6 Giao diện thiết lập tùy chỉnh 12
Hình 3. 7 Giao diện cài đặt các công cụ thiết yếu 12
Hình 3. 8 Giao diện cài đặt Node.js 13
Hình 3. 9 Giao diện tiến trình cài đặt 13
Hình 3. 10 Giao diện cài đặt thành công 14
Hình 3. 11 Giao diện kiểm tra các phiên bản được cài đặt 14
Hình 4. 1 Giao diện chính của ứng dụng quản lý thu chi 27
Hình 4. 2 Giao diện chức năng quy đổi tiền tệ 27
Hình 4. 3 Giao diện chức năng thêm giao dịch mới 28
Hình 4. 4 Giao diện chức năng cập nhật giao dịch 28
Hình 4. 5 Giao diện chức năng hiển thị lịch sử giao dịch, nhập và xuất file Excel 29
Hình 4. 6 Giao diện chức năng hiển thị biểu đồ mức độ chi tiêu 29

KÝ HIỆU CÁC CỤM TỪ VIẾT TẮT

Chữ viết tắt	Ý nghĩa
UI	User Interface
UX	User Experience
JS	JavaScript
CSS	Cascading Style Sheets

CHƯƠNG 1: GIỚI THIỆU BÀI TOÁN QUẢN LÝ CHI TIÊU

1.1 Giới thiệu chung

Quản lý chi tiêu hiệu quả là một trong những kỹ năng tài chính cơ bản mà mỗi cá nhân và gia đình cần nắm vững. Bài toán quản lý chi tiêu liên quan đến việc theo dõi, phân tích và kiểm soát các khoản thu, chi của cá nhân hoặc gia đình nhằm đạt được các mục tiêu tài chính cụ thể.

Tầm quan trọng của bài toán quản lý chi tiêu:

- Giúp chúng ta nắm rõ tình hình tài chính của mình, từ đó đưa ra các quyết định đầu tư, tiết kiệm và chi tiêu phù hợp.
- Phát hiện và loại bỏ các khoản chi tiêu không cần thiết, giúp tiết kiệm chi phí.
- Lập kế hoạch tài chính chi tiết cho ngắn hạn, trung hạn và dài hạn, như mua nhà, nghỉ hưu, v.v.
- Đảm bảo cân bằng giữa thu nhập và chi tiêu, tránh tình trạng mất cân bằng tài chính.
- Cung cấp dữ liệu để phân tích, đánh giá và đưa ra các quyết định tài chính phù hợp.

Các bước cơ bản trong bài toán quản lý chi tiêu:

- Thu thập và phân loại thông tin về các khoản thu, chi.
- Phân tích, đánh giá và tối ưu hóa các khoản chi tiêu.
- Lập và theo dõi ngân sách chi tiêu.
- Lập kế hoạch tài chính dài hạn, như mục tiêu tiết kiệm, đầu tư, trả nợ.
- Kiểm soát và điều chỉnh kế hoạch tài chính linh hoạt theo tình hình thực tế.

Các công cụ hữu ích để quản lý chi tiêu hiệu quả:

- Phần mềm quản lý tài chính cá nhân hoặc gia đình.
- Bảng tính Excel để lập và theo dõi ngân sách.
- Các ứng dụng di động để ghi chép, theo dõi các khoản thu chi.
- Sổ ghi chép tay truyền thống.

Giải quyết bài toán quản lý chi tiêu đòi hỏi sự kỷ luật, kiên trì và linh hoạt điều chỉnh các kế hoạch tài chính. Tuy nhiên, những nỗ lực này sẽ giúp chúng ta đạt được những mục tiêu tài chính quan trọng, từ đó cải thiện chất lượng cuộc sống.

1.2 Tính cấp thiết của đề tài

Trong thời đại công nghệ số hiện đại, việc quản lý chi tiêu cá nhân và gia đình trở nên ngày càng phức tạp. Với sự gia tăng của các khoản thu, chi, nhu cầu ghi chép, theo dõi và phân tích tình hình tài chính cá nhân đang trở nên hết sức cần thiết. Tuy nhiên, nhiều người vẫn chưa có công cụ hỗ trợ phù hợp để quản lý chi tiêu một cách hiệu quả.

Nhu cầu quản lý tài chính cá nhân ngày càng tăng cùng với sự gia tăng của các khoản thu nhập từ lương, kinh doanh, đầu tư, v.v., cùng với các khoản chi tiêu hàng ngày như tiền nhà, điện, nước, thực phẩm, giải trí, v.v., việc theo dõi và kiểm soát chi tiêu cá nhân và gia đình trở nên ngày càng phức tạp.

Nhu cầu lập kế hoạch tài chính dài hạn như mua nhà, nghỉ hưu, chi phí chăm sóc sức khỏe, giáo dục con cái, v.v. càng trở nên quan trọng. Người dùng cần tối ưu hóa chi tiêu và tích lũy để đạt được các mục tiêu tài chính như tích lũy tài sản, giảm nợ, tích lũy vốn để khởi nghiệp, nghỉ hưu sớm, v.v.

Việc thiếu các công cụ quản lý chi tiêu hiệu quả, nhiều người vẫn sử dụng sổ ghi chép tay hoặc bảng tính Excel để quản lý chi tiêu, điều này rất tốn thời gian và dễ bị sơ sót. Các ứng dụng quản lý chi tiêu hiện có trên thị trường thường không tích hợp đầy đủ các tính năng cần thiết như liên kết tài khoản ngân hàng, phân tích chi tiêu, lập kế hoạch tài chính, v.v.

Giao diện người dùng của các ứng dụng hiện có thường chưa được thiết kế để tối ưu hóa trải nghiệm người dùng, khiến người dùng không sử dụng thường xuyên. Một giao diện dễ sử dụng, thân thiện với người dùng sẽ khuyến khích người dùng quản lý chi tiêu thường xuyên. Các tính năng phân tích, lập kế hoạch tài chính hiệu quả sẽ giúp người dùng đạt được mục tiêu tài chính. Tích hợp các tính năng hữu ích như nhắc nhở, báo cáo sẽ nâng cao trải nghiệm người dùng và khuyến khích sử dụng thường xuyên.

Vì vậy, nhằm đáp ứng nhu cầu ngày càng tăng của người dùng trong việc quản lý tài chính cá nhân và gia đình một cách hiệu quả. Một giao diện thân thiện, tích hợp một số tính

năng cần thiết sẽ giúp người dùng dễ dàng theo dõi, phân tích và lập kế hoạch tài chính một cách chi tiết, từ đó đạt được các mục tiêu tài chính quan trọng. Đây cũng chính là lý do tại sao đề tài "Thiết Kế Giao Diện Trang Web Quản Lý Chi Tiêu" lại trở nên vô cùng cấp thiết.

1.3 Mục đích nghiên cứu của đề tài

Xác định nhu cầu và thói quen quản lý chi tiêu của người dung. Nghiên cứu về các thói quen, phương pháp quản lý chi tiêu của người dùng hiện tại. Khảo sát nhu cầu, mong muốn và các đặc điểm người dùng mong muốn có trong một ứng dụng quản lý chi tiêu. Phân tích các điểm mạnh, điểm yếu, cơ hội và thách thức trong việc quản lý chi tiêu của người dùng.

Nghiên cứu và áp dụng các nguyên tắc, tiêu chuẩn thiết kế giao diện người dùng tối ưu. Xây dựng các giao diện người dùng dựa trên nhu cầu và phản hồi của người dùng. Thử nghiệm, đánh giá và cải thiện giao diện người dùng để đạt được trải nghiệm tối ưu.

Nghiên cứu và phân tích các tính năng cần thiết cho một ứng dụng quản lý chi tiêu hiệu quả. Thiết kế và tích hợp các tính năng như xem giá vàng thế giới, đổi ngoại tệ, phân tích chi tiêu, lập kế hoạch tài chính, v.v.

Tiến hành thử nghiệm ứng dụng với người dùng mục tiêu. Thu thập phản hồi và đánh giá hiệu quả ứng dụng trong việc hỗ trợ quản lý chi tiêu. Đề xuất các cải tiến, nâng cấp để nâng cao hiệu quả của ứng dụng.

Thông qua việc nghiên cứu và thiết kế giao diện trang web quản lý chi tiêu, đề tài hướng đến mục tiêu cuối cùng là xây dựng một ứng dụng có giao diện người dùng thân thiện, tích hợp đầy đủ các tính năng cần thiết, đáp ứng nhu cầu quản lý tài chính cá nhân và gia đình một cách hiệu quả.

1.4 Phương pháp nghiên cứu của đề tài

Nghiên cứu tài liệu và khảo sát nhu cầu người dung: Tìm hiểu và phân tích các nghiên cứu, báo cáo liên quan đến quản lý chi tiêu cá nhân và gia đình. Tiến hành khảo sát, phỏng vấn người dùng mục tiêu để thu thập thông tin về nhu cầu, thói quen, mong muốn trong việc quản lý chi tiêu. Phân tích dữ liệu khảo sát để xác định các đặc điểm, yêu cầu chính của người dùng.

Thiết kế giao diện người dùng sử dụng ReactJS và Tailwind CSS: Sử dụng ReactJS, một thư viện JavaScript phổ biến, để xây dựng giao diện người dùng trang web. Áp dụng Tailwind CSS, một framework CSS tiện ích, để tạo ra giao diện người dùng có tính thẩm mỹ cao, responsive và dễ dàng tùy chỉnh. Thực hiện các vòng lặp thiết kế, thử nghiệm và cải tiến giao diện để đạt được trải nghiệm người dùng tối ưu.

Tích hợp các tính năng quản lý chi tiêu: Nghiên cứu và phân tích các tính năng cần thiết cho ứng dụng quản lý chi tiêu. Thiết kế và tích hợp các tính năng như xem giá vàng thế giới, đổi ngoại tệ, phân tích chi tiêu, lập kế hoạch tài chính, v.v. vào giao diện người dùng. Đảm bảo các tính năng đáp ứng được trải nghiệm người dùng.

Thử nghiệm và đánh giá hiệu quả ứng dụng: Tiến hành thử nghiệm ứng dụng với người dùng mục tiêu. Thu thập phản hồi và đánh giá hiệu quả ứng dụng trong việc hỗ trợ quản lý chi tiêu. Đề xuất các cải tiến, nâng cấp để nâng cao hiệu quả của ứng dụng..

1.5 Phạm vi nghiên cứu của đề tài

Nghiên cứu nhu cầu và thói quen quản lý chi tiêu cá nhân và gia đình. Thiết kế giao diện người dùng trang web sử dụng ReactJS và Tailwind CSS để tối ưu trải nghiệm người dùng. Tích hợp các tính năng quản lý chi tiêu như xem giá vàng thế giới, đổi ngoại tệ, phân tích chi tiêu, lập kế hoạch tài chính. Thử nghiệm và đánh giá hiệu quả ứng dụng trong việc hỗ trợ quản lý chi tiêu cá nhân và gia đình.

1.6 Đối tượng nghiên cứu của đề tài

Người dùng: Cá nhân, các nhóm có nhu cầu quản lý chi tiêu, tài chính cá nhân.

Giao diện website: Sử dụng ReactJS, Tailwind CSS. Tính năng: Xem giá vàng thế giới, đổi ngoại tệ, phân tích chi tiêu, lập kế hoạch tài chính.

1.7 Bố cục khi trình bày

Tiểu luận bao gồm 05 chương:

Chương 1: Giới Thiệu Bài Toán Quản Lý Chi Tiêu

Chương 2: Tổng Quan Thiết Kế Giao Diện Người Dùng Và Ứng Dụng Cho Thiết Kế Giao Diện Trang...

Chương 3: Khai Thác Sử Dụng React Js Và Tailwind Css Trong Thiết Kế Giao Diện Website

Chương 4: Thiết Kế Giao Diện Trang Web Cho Ứng Dụng Quản Lý Chi Tiều

Chương 5: Kết Luận Và Hướng Phát Triển

CHƯƠNG 2: TỔNG QUAN THIẾT KẾ GIAO DIỆN NGƯỜI DÙNG VÀ ỨNG DỤNG CHO THIẾT KẾ GIAO DIỆN TRANG WEB QUẢN LÝ CHI TIÊU

2.1 Khái niệm và vai trò của thiết kế giao diện người dùng (UI)

UI (User Interface) là giao diện giao tiếp giữa người dùng và thiết bị/phần mềm. Nó bao gồm các thành phần trực quan như nút bấm, menu, icons, màu sắc, phông chữ, bố cục, v.v.

Vai trò chính của UI là tạo ra một trải nghiệm người dùng (UX) tốt, dễ sử dụng, hiệu quả và đẹp mắt. Một UI tốt giúp người dùng dễ dàng tương tác với hệ thống, tăng năng suất và sự hài lòng của người dùng.

2.2 Các nguyên tắc cơ bản trong thiết kế UI

2.2.1 Sự đơn giản hóa và tính trực quan

Giao diện cần được thiết kế đơn giản, dễ hiểu, tập trung vào các chức năng then chốt.

Sử dụng các biểu tượng, hình ảnh trực quan để truyền tải thông tin dễ dàng.

2.2.2 Tính thông suốt và dễ sử dụng

Thiết kế giao diện phải tuân theo logic tự nhiên, tránh gây nhầm lẫn cho người dùng.

Các tính năng, thao tác cần được bố trí một cách nhất quán, phù hợp với suy nghĩ của người dùng.

2.2.3 Tính thống nhất và tính nhất quán

Giao diện cần được thiết kế với phong cách, màu sắc, font chữ thống nhất trên toàn hệ thống.

Các thao tác, cách hiển thị thông tin cũng cần tuân thủ các tiêu chuẩn nhất quán.

2.2.4 Tính phản hồi và tính tương tác

Giao diện cần cung cấp phản hồi rõ ràng cho mọi hành động của người dùng.

Tạo cảm giác tương tác trực tiếp, liền mạch giữa người dùng và hệ thống.

CHƯƠNG 3: KHAI THÁC SỬ DỤNG REACT JS VÀ TAILWIND CSS TRONG THIẾT KẾ GIAO DIỆN WEBSITE

3.1 Giới thiệu đôi nét về thiết kế giao diện website hiện đại

3.1.1 Bối cảnh và tầm quan trọng của việc xây dựng giao diện website hiện đại

Trong thời đại công nghệ số, website đóng vai trò quan trọng trong hoạt động kinh doanh và truyền thông của các doanh nghiệp.

Giao diện website ảnh hưởng trực tiếp đến trải nghiệm người dùng và góp phần tạo nên ấn tượng đầu tiên về thương hiệu.

Việc xây dựng giao diện website hiện đại, thiết kế website phù hợp trên tất cả các thiết bị, mọi độ phân giải màn hình và tối ưu trải nghiệm người dùng trở nên ngày càng quan trọng.

3.1.2 Vai trò của ReactJS và Tailwind CSS trong thiết kế và phát triển giao diện website

ReactJS là một thư viện JavaScript phổ biến, giúp xây dựng giao diện website động, hiệu suất cao và dễ bảo trì.

Tailwind CSS là một framework CSS utility-first, giúp tối ưu hóa quy trình thiết kế giao diện website một cách nhanh chóng và hiệu quả.

Kết hợp sử dụng ReactJS và Tailwind CSS mang lại nhiều lợi ích trong việc xây dựng giao diện website hiện đại.

3.2 Xây dựng giao diện website với ReactJS

3.2.1 Tổng quan về ReactJS

ReactJS là một thư viện JavaScript được phát triển bởi Facebook, dành cho việc xây dựng giao diện người dùng.

Các đặc điểm chính của ReactJS: component-based, virtual DOM, one-way data binding, JSX syntax.

Quy trình phát triển giao diện website bằng ReactJS bao gồm: thiết kế components, quản lý state và props, sử dụng React Router.

3.2.2 Ứng dụng ReactJS trong thiết kế giao diện

Các thành phần giao diện (components) trong ReactJS: header, navbar, product list, shopping cart, v.v.

Quản lý state và props để tối ưu hiệu suất giao diện, ví dụ: lưu trữ trạng thái giỏ hàng, quản lý dữ liệu sản phẩm.

Sử dụng React Router để xây dựng các trang web động, cho phép người dùng dễ dàng chuyển đổi giữa các trang.

3.3 Tối ưu giao diện với Tailwind CSS

3.3.1 Tổng quan về Tailwind CSS

Tailwind CSS là một framework CSS utility-first, giúp tối ưu hóa quy trình thiết kế giao diện.

Các đặc điểm chính của Tailwind CSS: utility-first approach, responsive design, customizable.

3.3.2 Ứng dụng Tailwind CSS trong thiết kế giao diện

Sử dụng các class utility (Ví dụ: text-gray-500, bg-blue-600, py-2, etc.) để định kiểu nhanh chóng.

Xây dựng các components với Tailwind CSS, ví dụ: button, card, form field.

Tùy chỉnh và mở rộng Tailwind CSS để phù hợp với thiết kế, ví dụ: thêm custom color, typography, v.v.

Thiết kế các thành phần giao diện (Header, Product List, Cart, Checkout) với Tailwind CSS.

Tối ưu layout, typography, color scheme, responsive design bằng Tailwind CSS.

Tích hợp Tailwind CSS vào quy trình phát triển ReactJS, ví dụ: sử dụng Tailwind directives trong file CSS.

3.4 Ưu điểm và thách thức khi sử dụng ReactJS và Tailwind CSS

Ưu điểm: hiệu suất cao, dễ bảo trì, tối ưu trải nghiệm người dùng, tăng tốc độ phát triển.

Thách thức: yêu cầu kiến thức về JavaScript, React, CSS, cần thời gian học hỏi.

3.5 Triển vọng và xu hướng phát triển của các công nghệ này trong tương lai

ReactJS và Tailwind CSS đang ngày càng phổ biến và được ưa chuộng trong phát triển giao diện website. Sự kết hợp giữa framework front-end như ReactJS và utility-first CSS như Tailwind CSS sẽ tiếp tục phát triển.

3.6 Gợi ý hướng phát triển và ứng dụng trong các dự án website khác

Áp dụng ReactJS và Tailwind CSS vào các dự án website khác như: blog, portfolio, web app, v.v. Tích hợp các tính năng nâng cao như state management, animation, server-side rendering. Nghiên cứu và ứng dụng các công nghệ mới liên quan như Next.js, Gatsby, Emotion, v.v.

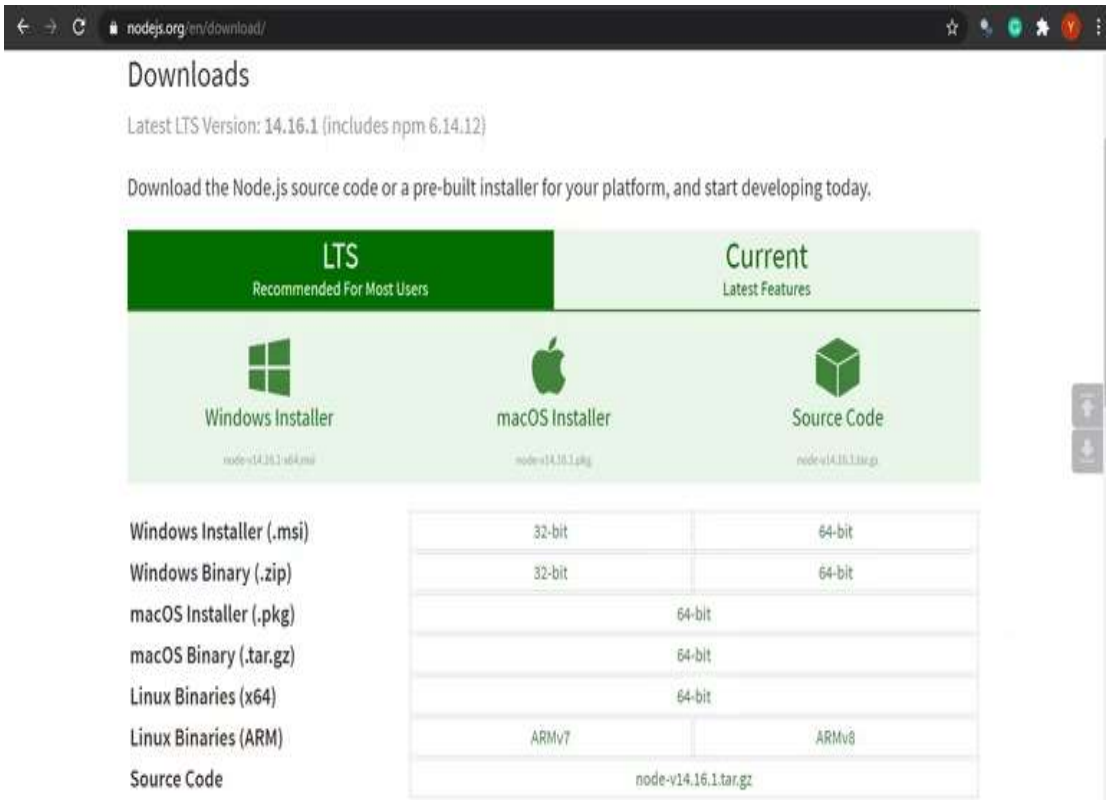
3.7 Cài đặt cấu hình cho dự án

3.7.1 Cài đặt Node.js

Bước 1: Truy cập trang web chính thức của Node.js

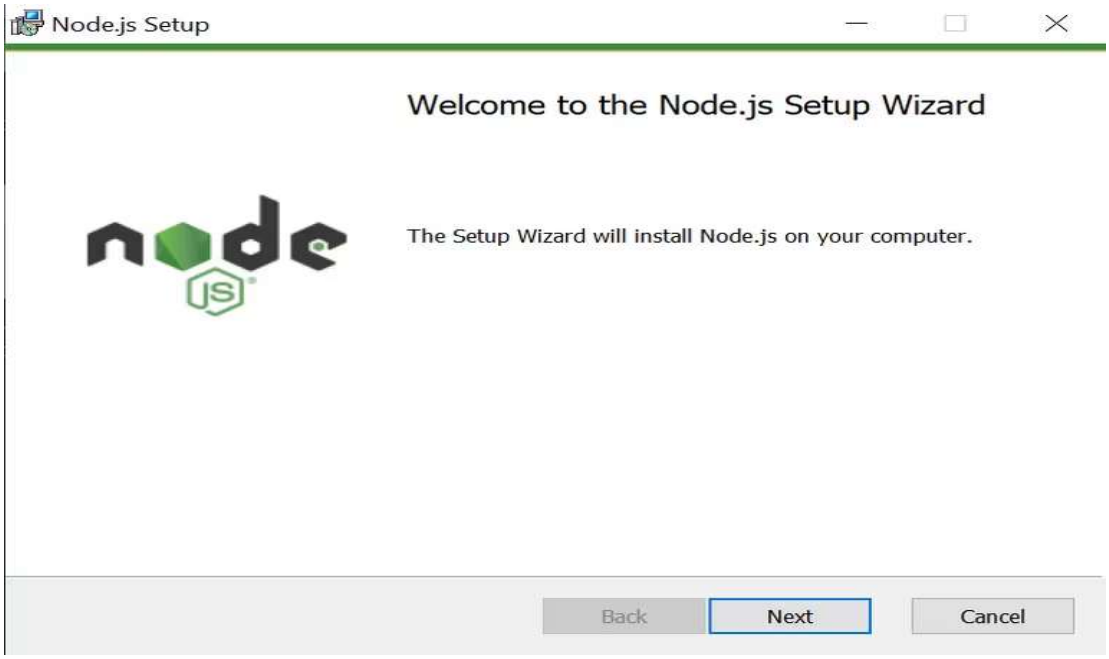
Hình 3. 1 Giao diện web chính thức của Node.js

Bước 2: Bấm vào phiên bản phù hợp với hệ điều hành để tải xuống



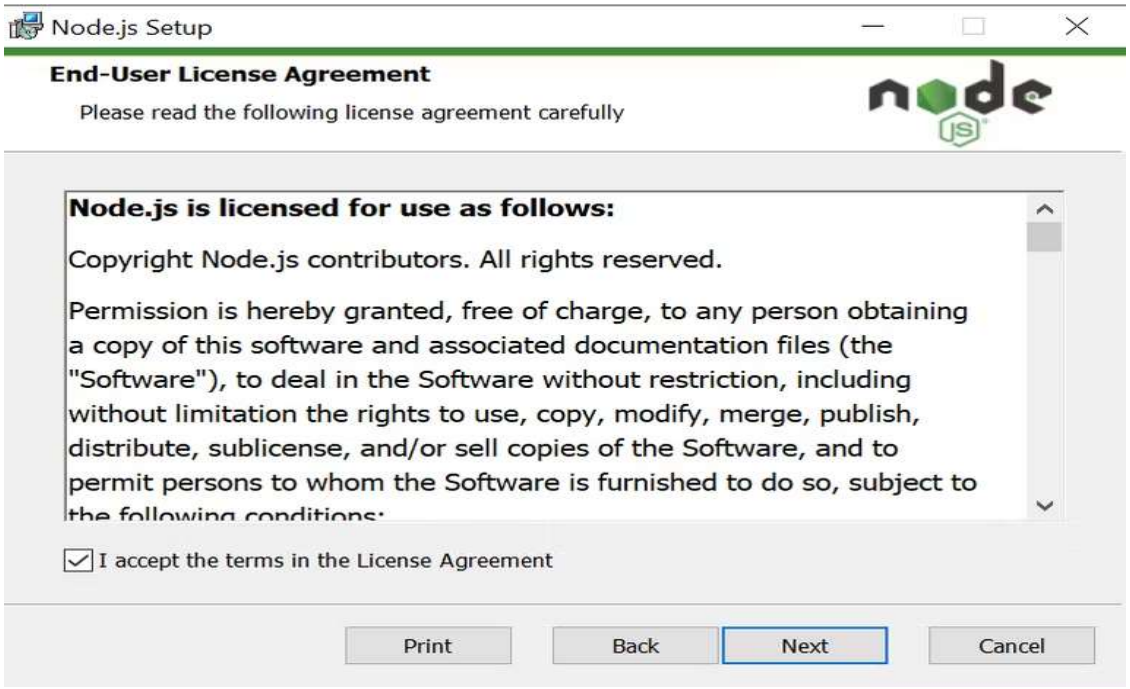
Hình 3. 2 Phiên bản phù hợp với hệ điều hành để tải xuống

Bước 3: Thực thi tệp tin cài đặt .msi



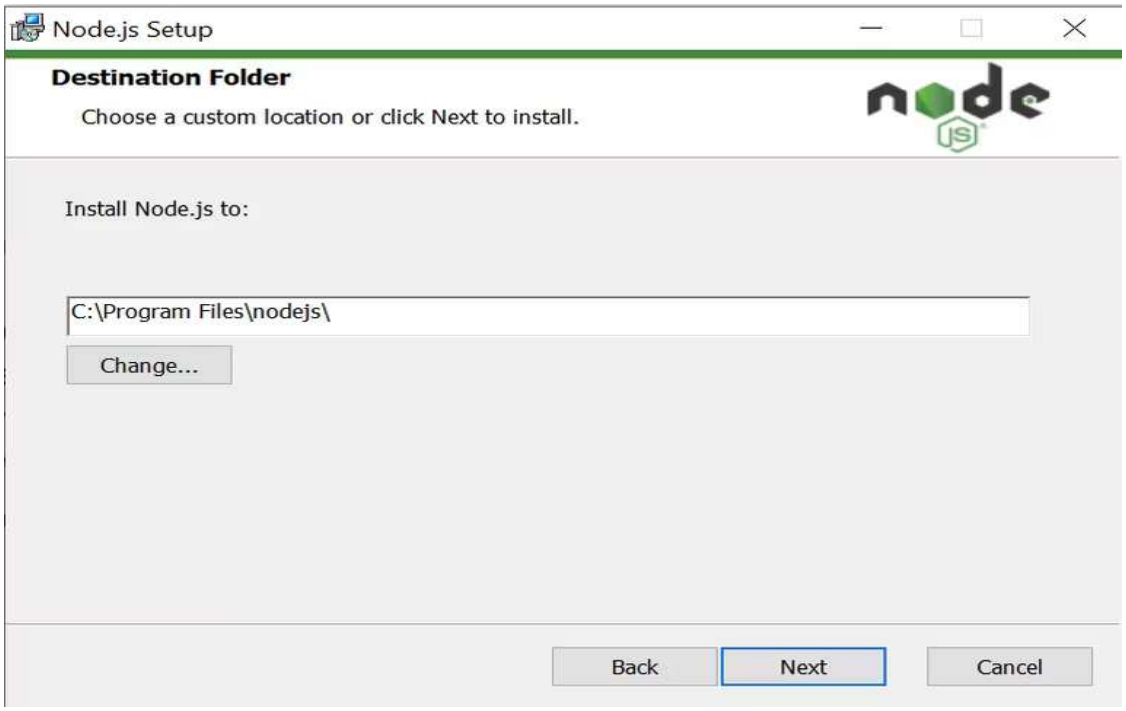
Hình 3. 3 Giao diện cài đặt Node.js

Bước 4: Đọc thỏa thuận cấp phép người dùng cuối.



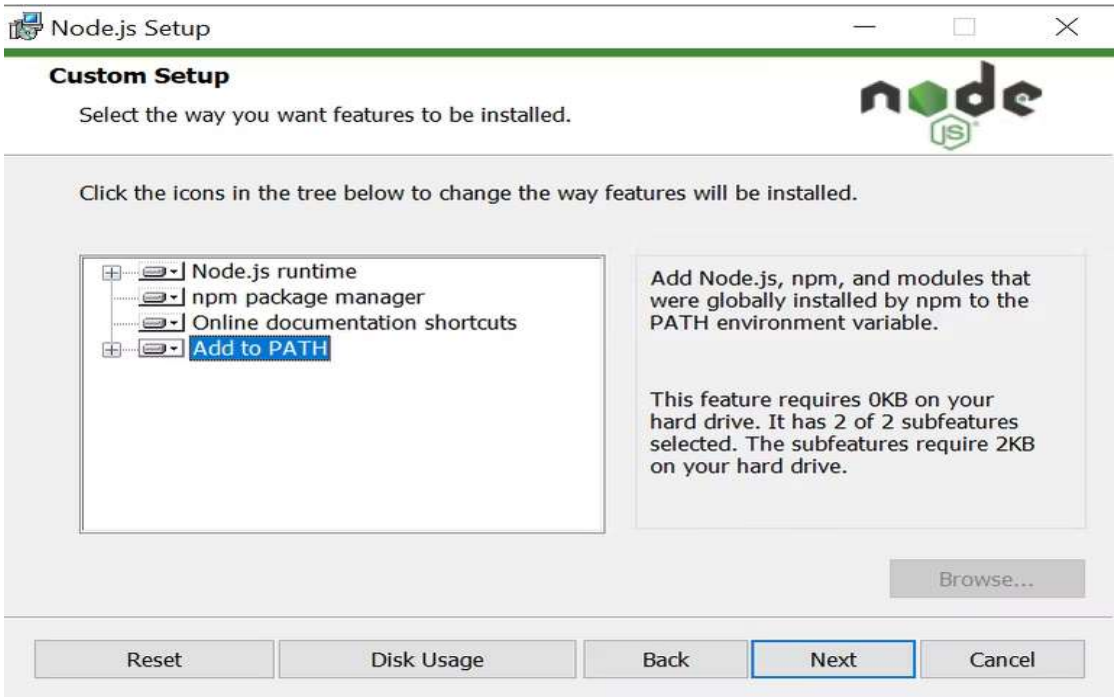
Hình 3. 4 Giao diện thỏa thuận cấp phép người dùng cuối

Bước 5: Chọn thư mục đích.



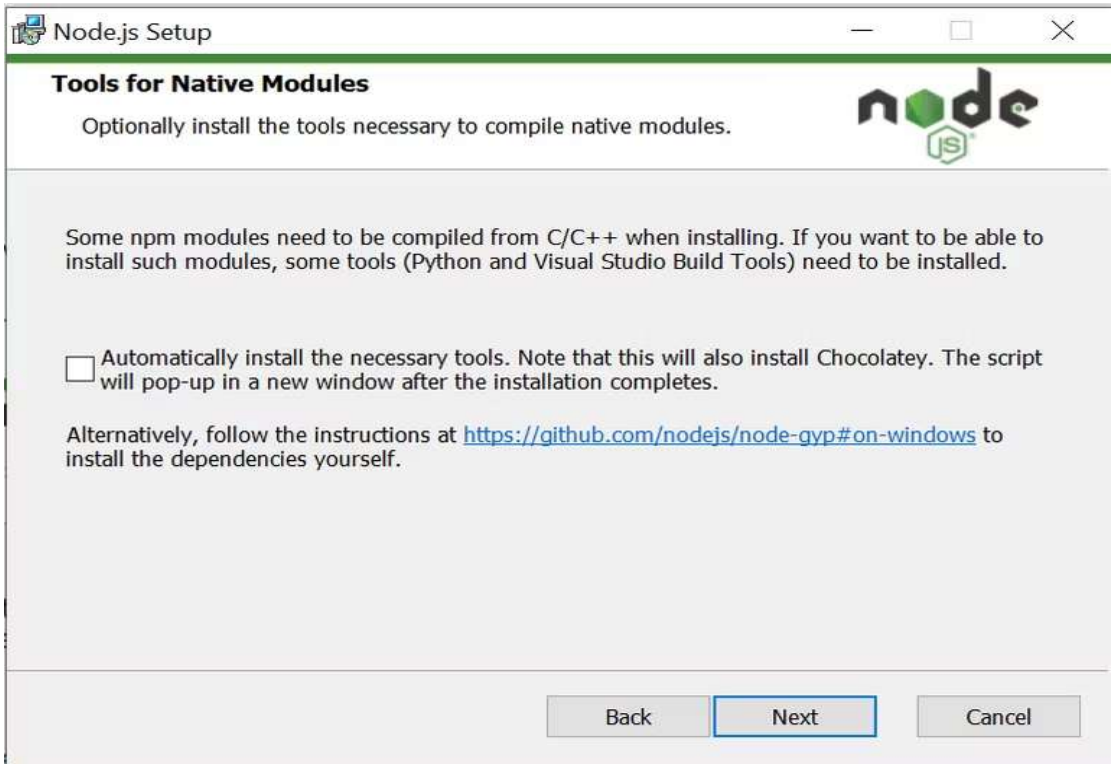
Hình 3. 5 Giao diện chọn thư mục đích để cài đặt

Bước 6: Thiết lập tùy chỉnh



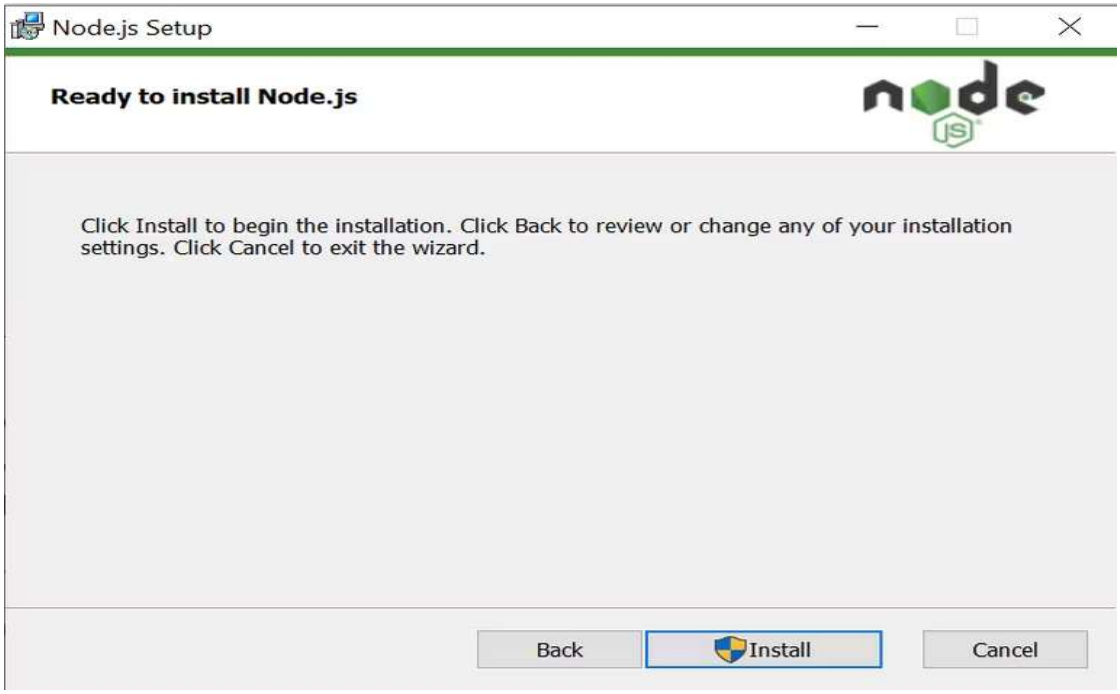
Hình 3. 6 Giao diện thiết lập tùy chỉnh

Bước 7: Công cụ cho mô-đun gốc

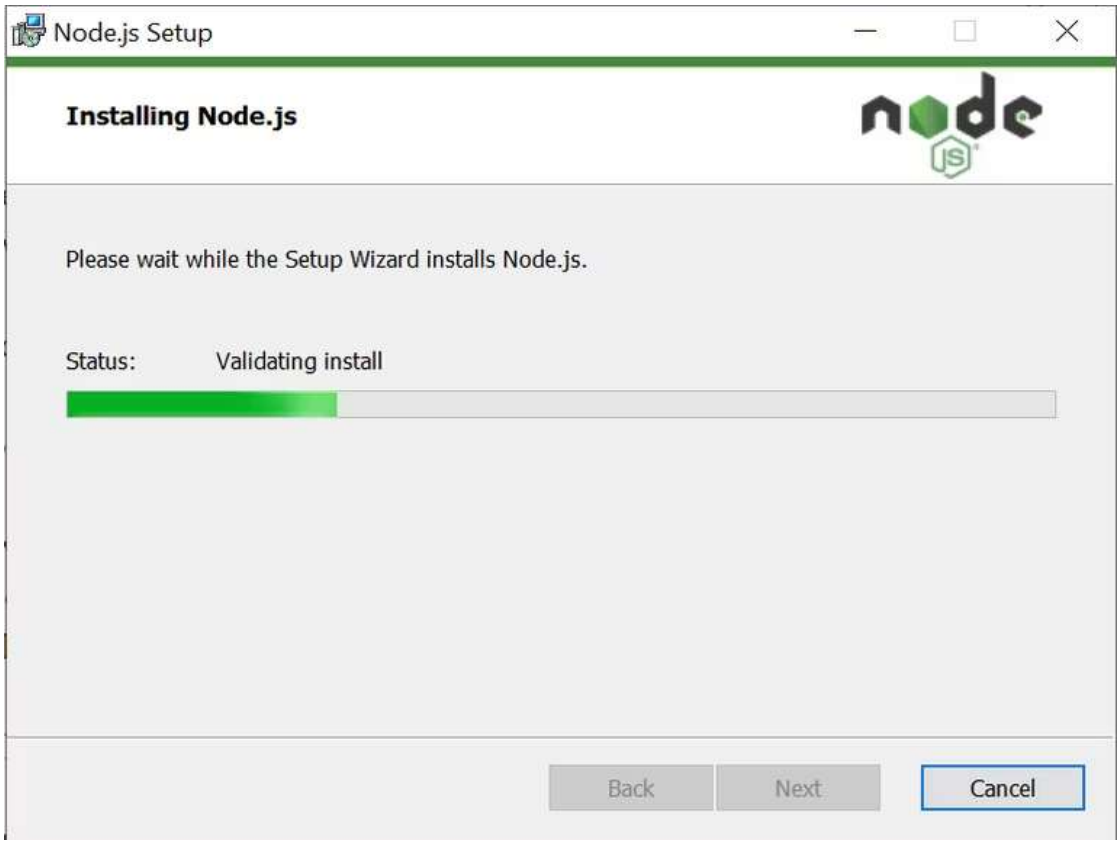


Hình 3. 7 Giao diện cài đặt các công cụ thiết yếu

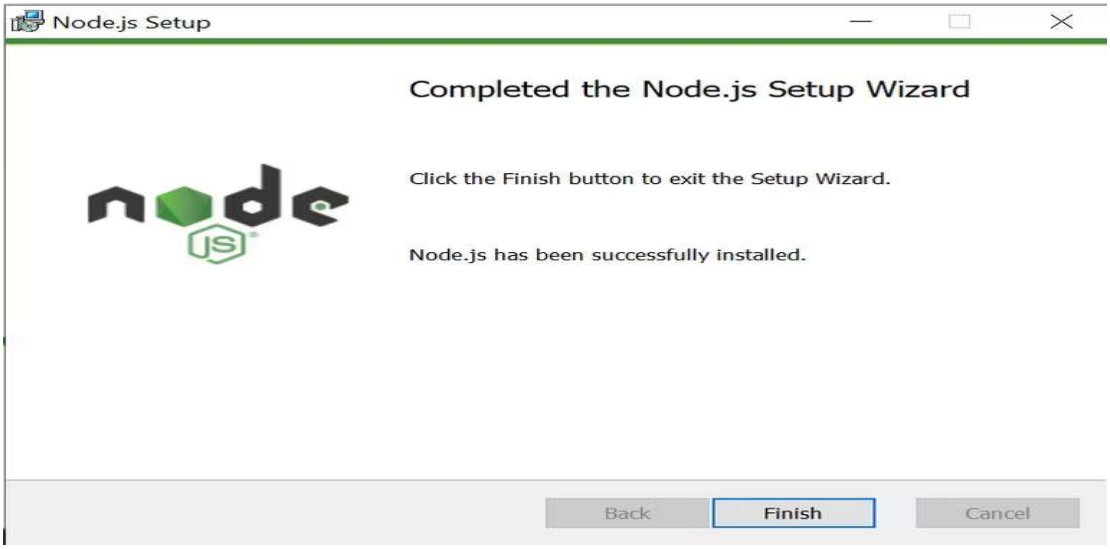
Bước 8: Tiến hành cài đặt Node.js



Hình 3. 8 Giao diện cài đặt Node.js



Hình 3. 9 Giao diện tiến trình cài đặt



Hình 3. 10 Giao diện cài đặt thành công

Bước 9: Xác nhận rằng Node.js và npm đã được cài đặt chính xác

- Để kiểm tra xem đã cài đặt Node.js đúng cách trên hệ thống chưa, chạy lệnh sau trong terminal:

node --version

- Để kiểm tra xem đã cài đặt npm đúng cách trên hệ thống chưa, chạy lệnh sau trong terminal:

npm --version

- Phiên bản đã cài đặt của Node.js và npm được hiển thị trong terminal.

Hình 3. 11 Giao diện kiểm tra các phiên bản được cài đặt

3.7.2 Tạo dự án React mới

Bước 1: Tạo 1 thư mục chứa dự án nằm trong ổ đĩa D:/. Đặt tên là "quan-li-thu-chi".

Bước 2: Mở terminal hoặc command prompt.

Bước 3: Điều hướng đến thư mục dự án.

cd quan-ly-thu-chi

Bước 4: Cài đặt Tailwind CSS

npm install -D tailwindcss

npx tailwindcss init

Lệnh này sẽ tạo ra một tệp tailwind.config.js trong dự án.

Bước 5: Chạy thử dự án

npm start

CHƯƠNG 4: THIẾT KẾ GIAO DIỆN TRANG WEB CHO ỨNG DỤNG QUẢN LÝ CHI TIÊU

4.1 Cài đặt các thư viện cần thiết cho ứng dụng

Điều hướng đến thư mục dự án. Tiến hành nhập các lệnh sau:

 # Cài đặt React và React DOM

npm install react react-dom

 # Cài đặt Axios

npm install axios

 # Cài đặt XLSX

npm install xlsx

 # Cài đặt Chart.js

npm install chart.js

 # Cài đặt react-chartjs-2

npm install react-chartjs-2

4.2 Viết code cho ứng dụng

```
import React, { createContext, useReducer, useContext, useState, useEffect }
from 'react';
import axios from 'axios';
import * as XLSX from 'xlsx';
import { Bar } from 'react-chartjs-2';
import 'chart.js/auto';
import './index.css';

// Initial State
const initialState = {
  transactions: []
};

// Reducer
const AppReducer = (state, action) => {
  switch (action.type) {
    case 'ADD_TRANSACTION':
      return {
        ...state,
        transactions: [...state.transactions, action.payload]
```

```
      };
    case 'UPDATE_TRANSACTION':
      return {
        ...state,
        transactions: state.transactions.map(transaction =>
          transaction.id === action.payload.id
            ? { ...transaction, ...action.payload }
            : transaction
        )
      };
    case 'DELETE_TRANSACTION':
      return {
        ...state,
        transactions: state.transactions.filter(
          transaction => transaction.id !== action.payload.id
        )
      };
    case 'SET_TRANSACTIONS':
      return {
        ...state,
        transactions: action.payload
      };
    default:
      return state;
  }
};

// Create Context
const GlobalContext = createContext(initialState);

// Provider Component
const GlobalProvider = ({ children }) => {
  const [state, dispatch] = useReducer(AppReducer, initialState);

  // Actions
  function addTransaction(transaction) {
    dispatch({
      type: 'ADD_TRANSACTION',
      payload: transaction
    });
  }

  function updateTransaction(updatedTransaction) {
    dispatch({
      type: 'UPDATE_TRANSACTION',
      payload: updatedTransaction
```

```
    });
  }

  function deleteTransaction(id) {
    dispatch({
      type: 'DELETE_TRANSACTION',
      payload: { id }
    });
  }

  function setTransactions(transactions) {
    dispatch({
      type: 'SET_TRANSACTIONS',
      payload: transactions
    });
  }

  return (
    <GlobalContext.Provider
      value={{
        transactions: state.transactions,
        addTransaction,
        updateTransaction,
        deleteTransaction,
        setTransactions
      }}
    >
      {children}
    </GlobalContext.Provider>
  );
};

// Balance Component
const Balance = () => {
  const { transactions } = useContext(GlobalContext);
  const total = transactions.reduce((acc, transaction) => acc +
transaction.amount, 0).toFixed(2);

  return (
    <div className="bg-white p-4 rounded-lg shadow mb-4">
      <h3 className="text-xl font-semibold text-gray-700">Tổng số dư</h3>
      <h2 className="text-2xl font-bold">{total} VND</h2>
    </div>
  );
};
```

```
// IncomeExpenses Component
const IncomeExpenses = () => {
  const { transactions } = useContext(GlobalContext);

  const income = transactions
    .filter(transaction => transaction.amount > 0)
    .reduce((acc, transaction) => acc + transaction.amount, 0)
    .toFixed(2);

  const expense = transactions
    .filter(transaction => transaction.amount < 0)
    .reduce((acc, transaction) => acc + Math.abs(transaction.amount), 0)
    .toFixed(2);

  return (
    <div className="bg-white p-4 rounded-lg shadow mb-4">
      <div className="flex justify-between">
        <div>
          <h4 className="text-lg font-semibold text-green-700">Thu nhập</h4>
          <p className="text-2xl font-bold text-green-700">{income} VND</p>
        </div>
        <div>
          <h4 className="text-lg font-semibold text-red-700">Chi tiêu</h4>
          <p className="text-2xl font-bold text-red-700">{expense} VND</p>
        </div>
      </div>
    </div>
  );
};

// AddTransaction Component
const AddTransaction = ({ currentTransaction, onClose }) => {
  const { addTransaction, updateTransaction } = useContext(GlobalContext);
  const [text, setText] = useState(currentTransaction ? currentTransaction.text
: '');
  const [amount, setAmount] = useState(currentTransaction ?
currentTransaction.amount : 0);

  const onSubmit = e => {
    e.preventDefault();

    const newTransaction = {
      id: currentTransaction ? currentTransaction.id : Math.floor(Math.random()
* 100000000),
      text,
      amount: +amount,
```

```
      date: new Date().toLocaleDateString('vi-VN')
    };

    if (currentTransaction) {
      updateTransaction(newTransaction);
    } else {
      addTransaction(newTransaction);
    }
    setText('');
    setAmount(0);
    if (typeof onClose === 'function') {
      onClose(); // Hide form after update or add
    }
  };

  return (
    <div className="bg-white p-4 rounded-lg shadow mb-4">
      <h3 className="text-xl font-semibold text-gray-700">{currentTransaction ?
'Cập nhật giao dịch' : 'Thêm giao dịch mới'}</h3>
      <form onSubmit={onSubmit}>
        <div className="mb-4">
          <label className="block text-gray-700">Mô tả</label>
          <input
            type="text"
            value={text}
            onChange={(e) => setText(e.target.value)}
            className="w-full border border-gray-300 p-2 rounded"
            placeholder="Nhập mô tả..."
          />
        </div>
        <div className="mb-4">
          <label className="block text-gray-700">
            Số tiền (âm - chi tiêu, dương - thu nhập)
          </label>
          <input
            type="number"
            value={amount}
            onChange={(e) => setAmount(e.target.value)}
            className="w-full border border-gray-300 p-2 rounded"
            placeholder="Nhập số tiền..."
          />
        </div>
        <button className="bg-blue-500 text-white px-4 py-2 rounded hover:bg-
blue-600 transition">
          {currentTransaction ? 'Cập nhật giao dịch' : 'Thêm giao dịch'}
        </button>
```

```
      </form>
    </div>
  );
};

// ExpenseList Component
const ExpenseList = () => {
  const { transactions, deleteTransaction } = useContext(GlobalContext);
  const [editingTransaction, setEditingTransaction] = useState(null);

  const handleDelete = (id) => {
    deleteTransaction(id);
  };

  const handleEdit = (transaction) => {
    setEditingTransaction(transaction);
  };

  const handleCloseEditForm = () => {
    setEditingTransaction(null);
  };

  return (
    <div className="bg-white p-4 rounded-lg shadow mb-4">
      <h3 className="text-xl font-semibold text-gray-700">Lịch sử giao
dịch</h3>
      <ul>
        {transactions.map(transaction => (
          <li key={transaction.id} className={`mb-2 p-2 rounded
${transaction.amount < 0 ? 'bg-red-100' : 'bg-green-100'}`}>
            {transaction.text} <span>{transaction.amount} VND</span>
            <button onClick={() => handleEdit(transaction)} className="ml-4
text-blue-600 hover:text-blue-800">
              Sửa
            </button>
            <button onClick={() => handleDelete(transaction.id)} className="ml-
4 text-red-600 hover:text-red-800">
              Xóa
            </button>
          </li>
        ))}
      </ul>
      {editingTransaction && (
        <AddTransaction currentTransaction={editingTransaction}
onClose={handleCloseEditForm} />
      )}
```

```
    </div>
  );
};

// ImportExport Components
const ExportExcel = () => {
  const { transactions } = useContext(GlobalContext);

  const handleExport = () => {
    const worksheet = XLSX.utils.json_to_sheet(transactions);
    const workbook = XLSX.utils.book_new();
    XLSX.utils.book_append_sheet(workbook, worksheet, 'Transactions');
    XLSX.writeFile(workbook, 'danhsachthuchi.xlsx');
  };

  return (
    <button
      onClick={handleExport}
      className="bg-green-500 text-white px-4 py-2 rounded hover:bg-green-600
transition w-full max-w-xs"
    >
      Xuất Excel
    </button>
  );
};

const ImportExcel = () => {
  const { setTransactions } = useContext(GlobalContext);

  const handleImport = (e) => {
    const file = e.target.files[0];
    const reader = new FileReader();

    reader.onload = (event) => {
      const data = event.target.result;
      const workbook = XLSX.read(data, { type: 'binary' });
      const sheetName = workbook.SheetNames[0];
      const worksheet = workbook.Sheets[sheetName];
      const json = XLSX.utils.sheet_to_json(worksheet);
      setTransactions(json);
    };

    reader.readAsBinaryString(file);
  };

  return (
```

```
    <div className="w-full max-w-xs">
      <input type="file" onChange={handleImport} className="hidden" id="file-
upload" />
      <label
        htmlFor="file-upload"
        className="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-600
transition cursor-pointer block text-center"
      >
        Nhập Excel
      </label>
    </div>
  );
};

// IncomeExpenseChart Component
const IncomeExpenseChart = () => {
  const { transactions } = useContext(GlobalContext);

  const dates = transactions.map((transaction) => transaction.date);
  const incomeData = transactions.map((transaction) => (transaction.amount > 0
? transaction.amount : 0));
  const expenseData = transactions.map((transaction) => (transaction.amount < 0
? Math.abs(transaction.amount) : 0));

  const data = {
    labels: dates,
    datasets: [
      {
        label: 'Thu nhập',
        data: incomeData,
        backgroundColor: 'rgba(75, 192, 192, 0.2)',
        borderColor: 'rgba(75, 192, 192, 1)',
        borderWidth: 1
      },
      {
        label: 'Chi tiêu',
        data: expenseData,
        backgroundColor: 'rgba(255, 99, 132, 0.2)',
        borderColor: 'rgba(255, 99, 132, 1)',
        borderWidth: 1
      },
    ],
  };

  const options = {
    scales: {
```

```
      x: {
        beginAtZero: true,
      },
      y: {
        beginAtZero: true,
      },
    },
  };

  return (
    <div className="bg-white p-4 rounded-lg shadow mt-4">
      <h3 className="text-xl mb-4 font-semibold text-gray-700">Biểu đồ thu
chi</h3>
      <Bar data={data} options={options} />
    </div>
  );
};

// CurrencyRates Component
const CurrencyRates = () => {
  const [rates, setRates] = useState({});
  const [amount, setAmount] = useState(1);
  const [baseCurrency, setBaseCurrency] = useState('USD');
  const [targetCurrency, setTargetCurrency] = useState('VND');
  const [convertedAmount, setConvertedAmount] = useState(null);

  useEffect(() => {
    const fetchRates = async () => {
      try {
        const response = await axios.get(`https://api.exchangerate-
api.com/v4/latest/${baseCurrency}`);
        setRates(response.data.rates);
      } catch (error) {
        console.error('Lỗi khi lấy dữ liệu tỷ giá tiền tệ:', error);
      }
    };

    fetchRates();
  }, [baseCurrency]);

  const handleConvert = () => {
    if (rates[targetCurrency]) {
      setConvertedAmount((amount * rates[targetCurrency]).toFixed(2));
    } else {
      setConvertedAmount('Không thể chuyển đổi');
    }
```

```
  };

  return (
    <div className="bg-blue-100 p-4 rounded-lg shadow mb-4">
      <h3 className="text-xl font-semibold text-blue-700 mb-2">Quy đổi tiền
tệ</h3>
      <div className="flex flex-col mb-4">
        <input
          type="number"
          value={amount}
          onChange={(e) => setAmount(e.target.value)}
          className="w-full border border-blue-300 p-2 rounded mb-2"
          placeholder="Nhập số tiền..."
        />
        <div className="flex mb-2">
          <select
            value={baseCurrency}
            onChange={(e) => setBaseCurrency(e.target.value)}
            className="border border-blue-300 p-2 rounded mr-2"
          >
            {Object.keys(rates).map((currency) => (
              <option key={currency} value={currency}>
                {currency}
              </option>
            ))}
          </select>
          <span className="mx-2">đổi sang</span>
          <select
            value={targetCurrency}
            onChange={(e) => setTargetCurrency(e.target.value)}
            className="border border-blue-300 p-2 rounded"
          >
            {Object.keys(rates).map((currency) => (
              <option key={currency} value={currency}>
                {currency}
              </option>
            ))}
          </select>
        </div>
        <button
          onClick={handleConvert}
          className="bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-600
transition"
        >
          Chuyển đổi
        </button>
```

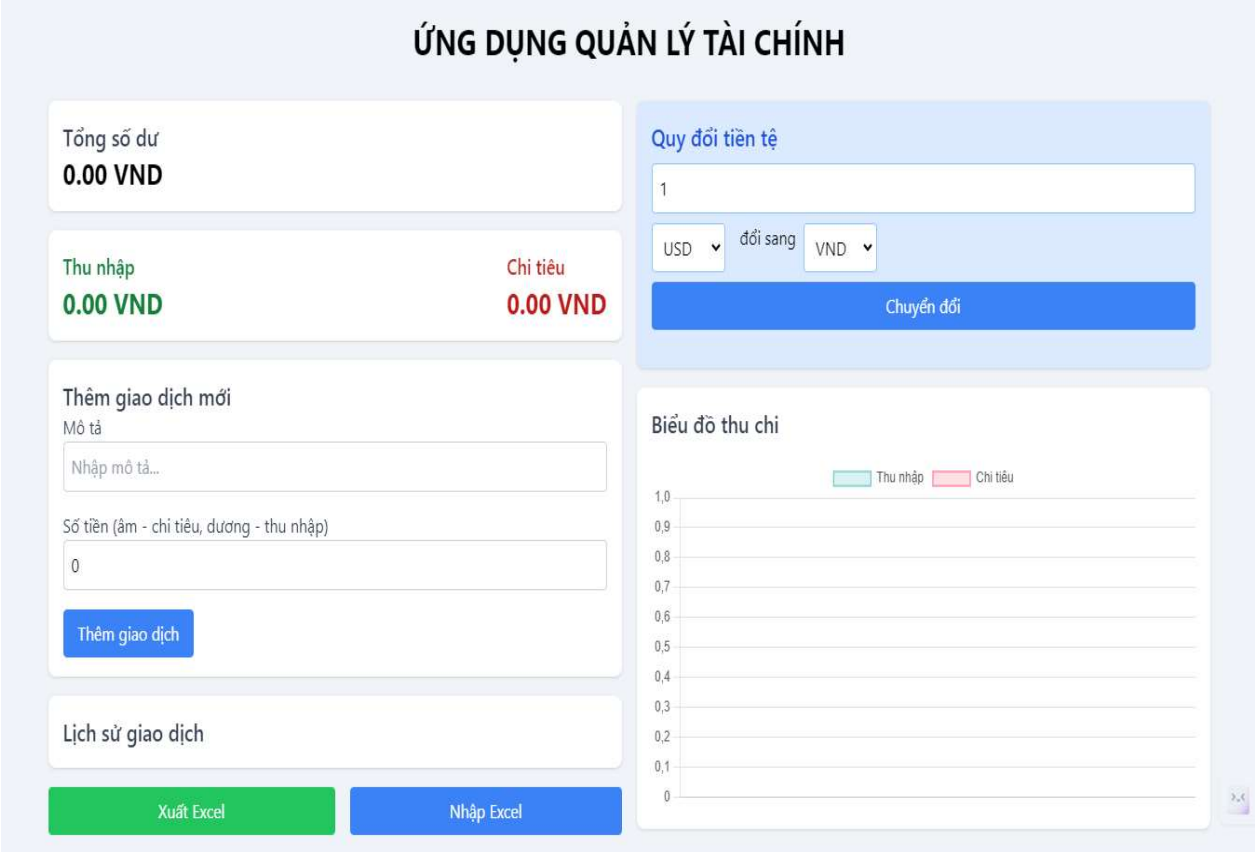
```
      </div>
      {convertedAmount && (
        <p className="text-lg text-blue-900">
          {amount} {baseCurrency} = {convertedAmount} {targetCurrency}
        </p>
      )}
    </div>
  );
};

// App Component
const App = () => {
  return (
    <GlobalProvider>
      <div className="container mx-auto p-4">
        <h1 className="text-3xl font-bold text-center mb-8">ỨNG DỤNG QUẢN LÝ
TÀI CHÍNH</h1>
        <div className="flex flex-col md:flex-row md:space-x-4">
          <div className="w-full md:w-1/2">
            <Balance />
            <IncomeExpenses />
            <AddTransaction />
            <ExpenseList />
            <div className="flex space-x-4 mt-4">
              <ExportExcel />
              <ImportExcel />
            </div>
          </div>
          <div className="w-full md:w-1/2">
            <CurrencyRates />
            <IncomeExpenseChart />
          </div>
        </div>
      </div>
    </GlobalProvider>
  );
};

export default App;
```

4.3 Kết quả

+ Giao diện chính:

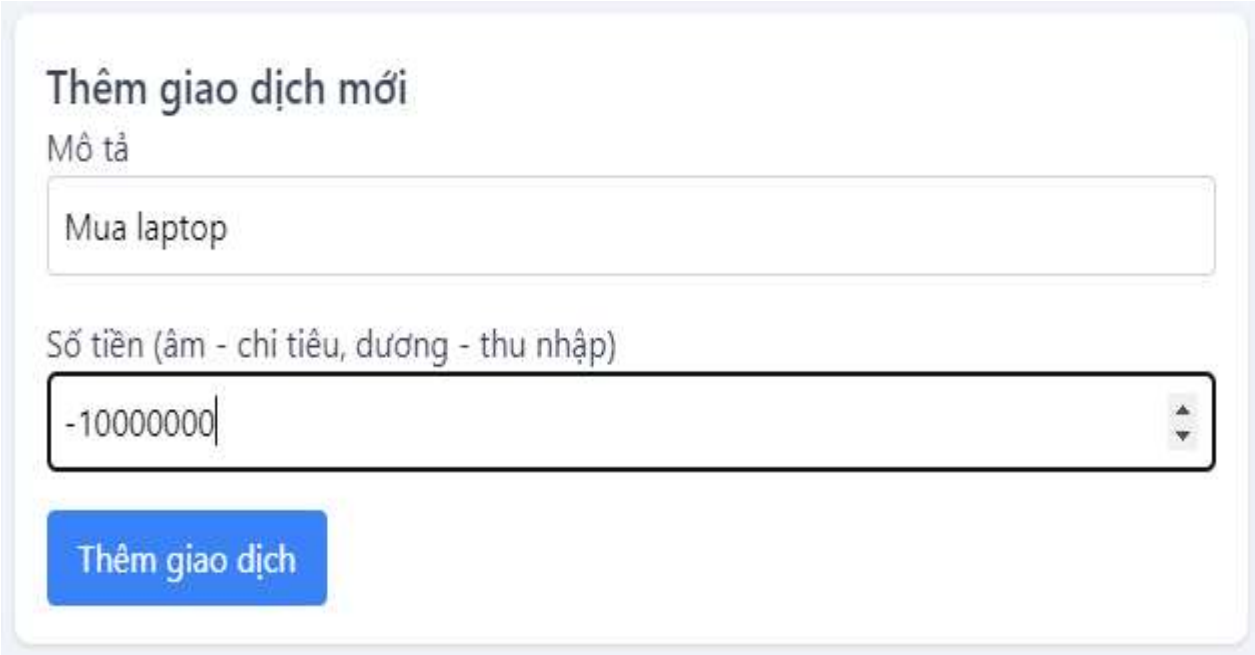


Hình 4. 1 Giao diện chính của ứng dụng quản lý thu chi

+ Chức năng quy đổi tiền tệ:

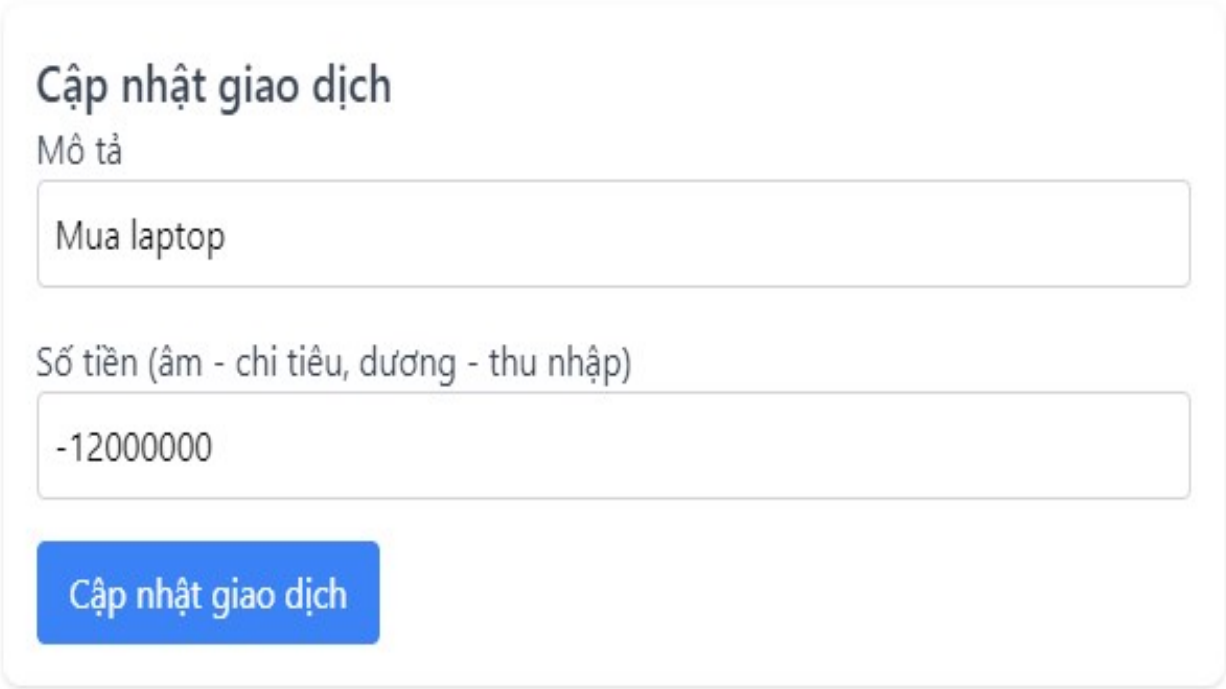
Hình 4. 2 Giao diện chức năng quy đổi tiền tệ

+ Chức năng thêm giao dịch mới:



Hình 4. 3 Giao diện chức năng thêm giao dịch mới

+ Chức năng cập nhật giao dịch:

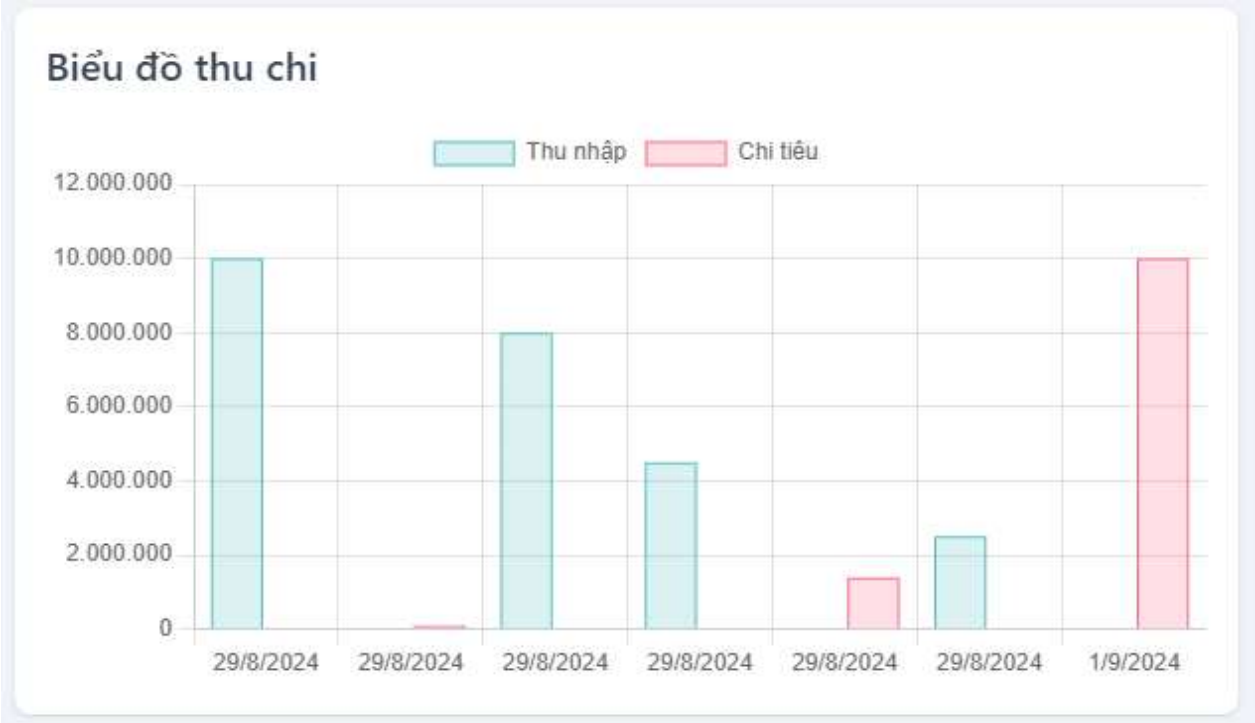


Hình 4. 4 Giao diện chức năng cập nhật giao dịch

+ Chức năng hiển thị lịch sử giao dịch, xuất và nhập file excel quản lý chi tiêu:

Hình 4. 5 Giao diện chức năng hiển thị lịch sử giao dịch, nhập và xuất file Excel

+ Chức năng hiển thị biểu đồ mức độ chi tiêu:



Hình 4. 6 Giao diện chức năng hiển thị biểu đồ mức độ chi tiêu

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1 Kết quả đạt được

Qua quá trình nghiên cứu và phát triển, ứng dụng quản lý tài chính đã đạt được những kết quả sau:

- Giao diện trực quan, thân thiện: Ứng dụng sở hữu giao diện đơn giản, dễ sử dụng, giúp người dùng nhanh chóng làm quen và thực hiện các thao tác.
- Các tính năng cơ bản hoàn chỉnh: Ứng dụng đã triển khai đầy đủ các tính năng cần thiết để quản lý tài chính cá nhân như:
  + Theo dõi thu chi: Ghi nhận chi tiết các khoản thu nhập và chi tiêu.
  + Chuyển đổi tiền tệ: Hỗ trợ chuyển đổi giữa các loại tiền tệ khác nhau.
  + Biểu đồ thống kê: Trực quan hóa dữ liệu chi tiêu, giúp người dùng dễ dàng phân tích và đưa ra quyết định.
  + Lịch sử giao dịch: Lưu trữ đầy đủ lịch sử giao dịch, tiện cho việc tra cứu và đối chiếu.
- Tính năng xuất nhập dữ liệu: Cho phép người dùng xuất dữ liệu ra file Excel để lưu trữ hoặc nhập dữ liệu từ file Excel vào ứng dụng.
- Khả năng tùy chỉnh: Người dùng có thể tùy chỉnh các danh mục chi tiêu, đơn vị tiền tệ theo nhu cầu cá nhân.

5.2 Hướng phát triển của đề tài

Để hoàn thiện hơn và đáp ứng nhu cầu ngày càng cao của người dùng, ứng dụng có thể được phát triển theo các hướng sau:

- Mở rộng tính năng:
  + Lập kế hoạch ngân sách: Cho phép người dùng đặt mục tiêu chi tiêu cho từng tháng, từng năm và theo dõi tiến độ thực hiện.
  + Cảnh báo chi tiêu: Gửi thông báo khi chi tiêu vượt quá ngân sách đã đặt.
  + Tích hợp với các dịch vụ tài chính: Liên kết với các ứng dụng ngân hàng, ví điện tử để tự động cập nhật giao dịch, giúp người dùng quản lý tài chính một cách liền mạch.

    + Phân tích dữ liệu nâng cao: Sử dụng các thuật toán học máy để phân tích hành vi chi tiêu, đưa ra các gợi ý tiết kiệm và đầu tư.
- Cải thiện hiệu năng:
    + Tối ưu hóa tốc độ tải trang: Giảm thiểu thời gian chờ đợi khi người dùng truy cập ứng dụng.
    + Nâng cao khả năng bảo mật: Bảo vệ dữ liệu người dùng bằng các biện pháp mã hóa và xác thực mạnh mẽ.
- Cá nhân hóa:
    + Tùy chỉnh giao diện: Cho phép người dùng tùy chỉnh giao diện theo sở thích cá nhân.
    + Đề xuất cá nhân: Dựa trên lịch sử giao dịch và hành vi của người dùng để đưa ra các đề xuất phù hợp.

Các yếu tố khác cần xem xét:

- Công nghệ: Nghiên cứu và áp dụng các công nghệ mới để nâng cao hiệu năng và trải nghiệm người dùng như:
    + Cloud computing: Lưu trữ dữ liệu trên đám mây để đảm bảo tính an toàn và khả năng truy cập mọi lúc mọi nơi.
    + Artificial Intelligence: Áp dụng các thuật toán AI để phân tích dữ liệu và đưa ra các quyết định thông minh.
- Phản hồi người dùng: Thu thập và phân tích ý kiến phản hồi của người dùng để cải thiện sản phẩm.
- Xu hướng thị trường: Theo dõi các xu hướng mới trong lĩnh vực quản lý tài chính để cập nhật và nâng cấp sản phẩm.

DANH MỤC TÀI LIỆU THAM KHẢO

[1] Facebook Inc. (2013). React: A JavaScript library for building user interfaces. Available at: *https://reactjs.org/*.

[2] Dan Abramov, Andrew Clark. (2015). Introducing Redux: A predictable state container for JavaScript apps. Available at: *https://redux.js.org/*.

[3] Axios. (2017). Axios: Promise based HTTP client for the browser and Node.js. Available at: *https://axios-http.com/*.

[4] SheetJS LLC. (2014). XLSX: Spreadsheet format library for JavaScript. Available at: *https://github.com/SheetJS/sheetjs*.

[5] Chart.js Contributors. (2013). Chart.js: Simple yet flexible JavaScript charting. Available at: *https://www.chartjs.org/*.

[6] Tailwind Labs. (2017). Tailwind CSS: A utility-first CSS framework for rapidly building custom designs. Available at: *https://tailwindcss.com/*.

[7] React Spring Contributors. (2018). React Spring: A spring-physics-based animation library for React applications. Available at: *https://react-spring.dev/*.

[8] React Router Team. (2014). React Router: Declarative routing for React.js. Available at: *https://reactrouter.com/*.

[9] React Hook Form Contributors. (2019). React Hook Form: Performant, flexible and extensible forms with easy-to-use validation. Available at: *https://react-hook-form.com/*.

[10] Moment.js Contributors. (2011). Moment.js: Parse, validate, manipulate, and display dates in JavaScript. Available at: *https://momentjs.com/*.

[11] TypeScript Contributors. (2012). TypeScript: JavaScript with syntax for types. Available at: *https://www.typescriptlang.org/*.