

### **NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

[illegible]

Vĩnh Long, ngày.... tháng....năm....

## Giảng viên hướng dẫn

(Ký và ghi rõ họ tên)

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

*Vĩnh Long, ngày.... tháng....năm....*

**Thành viên hội đồng**  
*(Ký và ghi rõ họ tên)*

## LỜI MỞ ĐẦU



Trong bối cảnh công nghệ thông tin ngày càng phát triển mạnh mẽ, việc quản lý công việc cá nhân một cách khoa học và hiệu quả đóng vai trò quan trọng trong học tập, công việc và cuộc sống hằng ngày. Đặc biệt, nhu cầu sử dụng các công cụ số để theo dõi, sắp xếp và nhắc nhở công việc ngày càng trở nên phổ biến và thiết thực.

Xuất phát từ thực tiễn đó, nhóm chúng em đã thực hiện đề tài: **“Xây dựng website quản lý công việc cá nhân”**. Mục tiêu của đề tài là xây dựng một hệ thống quản lý công việc đơn giản, thân thiện với người dùng, giúp cá nhân dễ dàng tạo, theo dõi và cập nhật các công việc hằng ngày mọi lúc, mọi nơi.

Báo cáo này trình bày đầy đủ quá trình thực hiện từ phân tích yêu cầu, thiết kế giao diện, xây dựng hệ thống bằng NodeJS (phía backend) và ReactJS (phía frontend), đến kiểm thử và đánh giá hiệu quả sử dụng. Qua đề tài, nhóm không chỉ củng cố kiến thức lập trình web mà còn rèn luyện kỹ năng giải quyết vấn đề thực tiễn và làm việc nhóm hiệu quả.

## LỜI CẢM ƠN



Chúng tôi xin bày tỏ lòng biết ơn sâu sắc đến Thầy Nguyễn Bảo Ân, giảng viên môn Công nghệ phần mềm tại Trường Đại học Trà Vinh, người đã tận tình hướng dẫn và đồng hành cùng chúng tôi trong suốt quá trình thực hiện đề tài.

Trong suốt thời gian triển khai đề án, thầy đã không ngừng hỗ trợ chúng tôi bằng những chỉ dẫn cụ thể, hướng dẫn nhưng công nghệ hiện đại mới. Nhờ sự hướng dẫn tận tình của thầy, chúng tôi đã có định hướng rõ ràng hơn trong quá trình phân tích, thiết kế và xây dựng hệ thống, cũng như nắm vững được các kiến thức chuyên môn liên quan đến lĩnh vực phát triển phần mềm.

Mặc dù đã cố gắng hoàn thành tốt nhất trong khả năng của mình, nhưng do hạn chế về thời gian và kinh nghiệm thực tế, đề tài chắc chắn vẫn còn một số thiếu sót. Chúng tôi rất mong nhận được những ý kiến đóng góp quý báu của thầy để đề tài ngày càng được hoàn thiện hơn.

Chúng tôi xin chân thành cảm ơn!

## **MỤC LỤC**

CHƯƠNG 1: GIỚI THIỆU .....	1
1.1. Lý do chọn đề tài.....	1
1.2. Tên dự án và chủ đề .....	1
1.3. Mục tiêu đề tài .....	2
CHƯƠNG 2: PHÂN TÍCH YÊU CẦU .....	3
2.1. Các yêu cầu chức năng.....	3
2.2. Các yêu cầu phi chức năng .....	4
CHƯƠNG 3: THIẾT KẾ HỆ THỐNG .....	6
3.1. Kiến trúc tổng thể hệ thống.....	6
3.1.1. Frontend: .....	6
3.1.2. Backend:.....	6
3.1.3. Cơ sở dữ liệu: .....	6
3.1.4. Docker: .....	6
3.2. Thiết kế cơ sở dữ liệu.....	7
3.2.1. Mô hình thực thể - quan hệ (ERD).....	7
3.2.2. Mô tả bảng dữ liệu .....	8
3.2.2.1. Bảng users – Quản lý người dùng .....	8
3.2.2.2. Bảng jobs – Lưu thông tin công việc .....	8
3.2.2.3. Bảng typejobs – Quản lý loại công việc.....	9
3.2.2.4. Bảng logs – Ghi lại lịch sử thay đổi công việc.....	9
3.3. Thiết kế API.....	9
3.3.1. Xác thực người dùng (Auth) .....	9
3.3.2. Quản lý công việc (Job) .....	10
3.3.3. Quản lý loại công việc (Typejob).....	10
3.3.4. Nhật ký thay đổi (Logs) .....	11
3.3.1. Các hình ảnh Swagger.....	11
3.4. Thiết kế giao diện (UI/UX).....	12

CHƯƠNG 4: TRIỂN KHAI VÀ CÔNG NGHỆ SỬ DỤNG .....	23
4.1. Công nghệ sử dụng .....	23
4.2. Quy trình CI/CD với GitHub Actions.....	24
4.3. Cấu hình Docker và Docker Compose .....	24
4.4. Quy trình triển khai ứng dụng.....	25
CHƯƠNG 5: QUẢN LÝ DỰ ÁN.....	26
5.1. Phương pháp quản lý dự án .....	26
5.2. Phân công nhiệm vụ.....	26
CHƯƠNG 6: KIỂM THỬ.....	28
6.1. Chiến lược kiểm thử.....	28
6.2. Công cụ sử dụng .....	29
6.3. Kết quả kiểm thử API .....	29
6.4. Kết luận kiểm thử.....	32
CHƯƠNG 7: ĐÁNH GIÁ VÀ KẾT LUẬN.....	33
7.1. Những khó khăn gặp phải .....	33
7.2. Bài học rút ra.....	33
7.2. Đề xuất và cải tiến trong tương lai.....	33

## **DANH MỤC HÌNH ẢNH**

Hình 3.1 Sơ đồ kiến trúc hệ thống .....	7
Hình 3.2 Mô hình ERD .....	7
Hình 3.3 Swagger xác thực người dùng .....	11
Hình 3.4 Swagger quản lý công việc.....	12
Hình 3.5 Swagger quản lý loại công việc .....	12
Hình 3.6 Figma đăng nhập .....	13
Hình 3.7 Figma tạo tài khoản .....	13
Hình 3.8 Figma đổi mật khẩu .....	14
Hình 3.9 Figma quên mật khẩu.....	14
Hình 3.10 Figma đặt lại mật kh .....	15
Hình 3.11 Figma tổng quan.....	15
Hình 3.12 Figma loại công việc.....	16
Hình 3.13 Figma công việc .....	16
Hình 3.14 Figma lịch sử thay đổi.....	17
Hình 3.15 Giao diện đăng nhập hệ thống .....	17
Hình 3.16 Giao diện tạo khoản mới .....	18
Hình 3.17 Giao diện quên mật khẩu.....	18
Hình 3.18 Giao diện đặt lại mật khẩu.....	19
Hình 3.19 Giao diện đổi mật khẩu .....	19
Hình 3.20 Giao diện tổng quan.....	20
Hình 3.21 Giao diện loại công việc.....	21

Hình 3.22 Giao diện quản lý công việc .....	21
Hình 3.23 Giao diện nhật ký thay đổi .....	22
Hình 6.1 Test trên GitHub Action.....	28
Hình 6.2 Kiểm thử API đăng ký người dùng bằng Postman .....	30
Hình 6.3 Kiểm thử API đăng nhập người dùng bằng Postman .....	31
Hình 6.4 Kiểm thử API tạo loại công việc bằng Postman.....	31
Hình 6.5 Kiểm thử API tạo công việc bằng Postman .....	32



## **CHƯƠNG 1: GIỚI THIỆU**

### **1.1. Lý do chọn đề tài**

Trong thời đại số hóa hiện nay, con người ngày càng phải đối mặt với khối lượng công việc và thông tin lớn mỗi ngày. Việc quản lý thời gian và sắp xếp công việc một cách hiệu quả đóng vai trò quan trọng trong học tập, làm việc và sinh hoạt cá nhân. Tuy nhiên, nhiều người vẫn gặp khó khăn trong việc ghi nhớ, theo dõi và thực hiện các công việc đúng thời hạn do thiếu công cụ hỗ trợ phù hợp.

Mặc dù trên thị trường đã có nhiều ứng dụng quản lý công việc, nhưng phần lớn đều yêu cầu người dùng trả phí, giao diện phức tạp hoặc chưa hỗ trợ đầy đủ tiếng Việt, gây khó khăn cho người dùng phổ thông. Do đó, nhu cầu xây dựng một hệ thống quản lý công việc đơn giản, dễ sử dụng, miễn phí và tối ưu cho người Việt là hoàn toàn cần thiết.

Từ thực tiễn đó, nhóm quyết định chọn đề tài "Xây dựng website quản lý công việc cá nhân" với mục tiêu:

- Giải quyết vấn đề quản lý công việc cá nhân một cách tiện lợi, hiệu quả và dễ tiếp cận.
- Tận dụng các công nghệ hiện đại như NodeJS (backend) và ReactJS (frontend) để xây dựng hệ thống theo hướng modular, dễ mở rộng.
- Góp phần nâng cao kỹ năng lập trình, làm việc nhóm, tư duy phân tích và giải quyết vấn đề thực tế cho các thành viên trong nhóm.

Đề tài không chỉ mang tính thực tiễn cao mà còn phù hợp với xu hướng phát triển của công nghệ phần mềm hiện đại, đặc biệt là trong việc ứng dụng mô hình web đa tầng, RESTful API và giao diện tương tác người dùng.

### **1.2. Tên dự án và chủ đề**

Tên dự án: Website Quản lý Công việc Cá nhân

Dự án tập trung vào việc phát triển một ứng dụng web hỗ trợ người dùng quản lý công việc cá nhân, nhằm nâng cao hiệu suất làm việc và khả năng tổ chức thời gian một cách khoa học, với các chức năng cơ bản.

### **1.3. Mục tiêu đề tài**

Xây dựng một ứng dụng web hỗ trợ người dùng quản lý công việc cá nhân một cách hiệu quả, khoa học và trực quan. Ứng dụng giúp người dùng dễ dàng tạo, sắp xếp và theo dõi các công việc theo lịch trình cụ thể, từ đó nâng cao khả năng quản lý thời gian và hiệu suất làm việc. Bên cạnh đó, đề tài cũng hướng đến việc áp dụng các công nghệ hiện đại trong phát triển phần mềm như kiến trúc MVC, RESTful API, MongoDB, React và triển khai thực tế thông qua Docker, CI/CD, quen với quy trình phát triển phần mềm chuyên nghiệp.

## **CHƯƠNG 2: PHÂN TÍCH YÊU CẦU**

### **2.1. Các yêu cầu chức năng**

Hệ thống cần cung cấp đầy đủ các chức năng thiết yếu để hỗ trợ người dùng quản lý công việc cá nhân một cách hiệu quả, trực quan và thuận tiện. Các chức năng được thiết kế nhằm đảm bảo người dùng có thể dễ dàng tổ chức, theo dõi và điều chỉnh công việc của mình trong suốt quá trình sử dụng ứng dụng. Cụ thể, bao gồm:

- **Đăng ký, đăng nhập, đăng xuất tài khoản người dùng:** ứng dụng cho phép người dùng tạo tài khoản mới bằng email và mật khẩu, đăng nhập để truy cập vào hệ. Mỗi tài khoản có không gian quản lý công việc độc lập nhằm đảm bảo tính cá nhân hóa, bảo mật và thuận tiện trong theo dõi tiến độ cá nhân.
- **Tạo công việc mới:** dễ dàng tạo một công việc mới với các thông tin chi tiết như: tiêu đề, mô tả nội dung công việc, thời gian bắt đầu – kết thúc, trạng thái ban đầu, loại công việc.
- **Chỉnh sửa công việc:** người dùng cập nhật các thông tin đã tạo trước đó như thay đổi tiêu đề, mô tả, loại công việc, thời gian thực hiện hoặc trạng thái hiện tại. Việc cập nhật này là cần thiết trong trường hợp công việc thay đổi nội dung hoặc lịch trình.
- **Xóa công việc:** Khi một công việc không còn cần thiết, người dùng có thể xóa nó khỏi hệ thống. Hành động xóa này là vĩnh viễn nhằm loại bỏ dữ liệu không còn sử dụng, giúp giao diện luôn gọn gàng và dễ quản lý.
- **Phân loại công việc:** Mỗi công việc có thể được gán vào một loại cụ thể. Việc phân loại giúp người dùng dễ dàng lọc công việc theo mục đích hoặc nhóm nhiệm vụ, đồng thời hỗ trợ thống kê và đánh giá mức độ phân bố thời gian.
- **Cập nhật trạng thái công việc:** Một công việc có thể chuyển đổi qua các trạng thái: Chưa thực hiện, Đang thực hiện, Đã hoàn thành. Việc cập nhật trạng thái giúp người dùng nắm bắt tiến độ và ưu tiên xử lý công việc một cách linh hoạt

- **Hiện thị công việc dưới dạng lịch và thông báo:** chế độ lịch giúp người dùng hình dung tổng thể kế hoạch theo ngày/tuần/tháng, từ đó dễ dàng điều phối thời gian. Thông báo số ngày trước khi đến hạn chót.
- **Lưu lại lịch sử thay đổi của công việc:** Mỗi khi có sự thay đổi về trạng thái, thời gian hoặc nội dung công việc, hệ thống sẽ ghi nhận các hành động đó kèm thời điểm diễn ra. Chức năng này mang lại tính minh bạch, hỗ trợ người dùng kiểm tra quá trình thay đổi cũng như phục vụ cho mục đích đánh giá lại sau này.

## **2.2. Các yêu cầu phi chức năng**

- **Hiệu năng:** Hệ thống cần đảm bảo phản hồi nhanh với các thao tác như đăng nhập, tạo công việc, chỉnh sửa và hiển thị dữ liệu. Thời gian phản hồi cho các hành động phổ biến không nên vượt quá 1 giây trong điều kiện mạng bình thường. Giao diện phải tải nhanh và hoạt động mượt mà kể cả khi có nhiều công việc được hiển thị.
- **Tính sẵn sàng:** Ứng dụng cần hoạt động ổn định mọi lúc, hạn chế tối đa thời gian ngừng hoạt động. Các thành phần của hệ thống được triển khai trên nền tảng điện toán đám mây để đảm bảo luôn sẵn sàng phục vụ người dùng và có khả năng mở rộng khi cần thiết.
- **Tính bảo mật:** Hệ thống phải bảo vệ thông tin cá nhân của người dùng, bao gồm việc mã hóa mật khẩu, xác thực và phân quyền truy cập. Các tệp tin được tải lên phải được kiểm soát định dạng và dung lượng để đảm bảo an toàn.
- **Khả năng mở rộng:** Ứng dụng được xây dựng theo kiến trúc phân tầng, có thể dễ dàng bổ sung thêm các chức năng mới trong tương lai như nhắc việc, chia sẻ công việc, đồng bộ với lịch cá nhân,...
- **Tính dễ sử dụng:** Giao diện thân thiện, đơn giản, dễ thao tác cho cả người dùng phổ thông. Các nút chức năng được bố trí hợp lý, hỗ trợ tương tác trực quan như kéo thả trong lịch, chọn nhanh trạng thái hoặc phân loại công việc.

- **Khả năng bảo trì và đảm bảo toàn vẹn dữ liệu:** Mã nguồn được tổ chức rõ ràng theo từng chức năng, thuận tiện cho việc bảo trì và phát triển sau này. Đồng thời, hệ thống cần đảm bảo dữ liệu không bị mất mát, trùng lặp hay sai lệch; mọi thao tác đầu vào phải được kiểm tra và lưu trữ đúng định dạng.

## **CHƯƠNG 3: THIẾT KẾ HỆ THỐNG**

### **3.1. Kiến trúc tổng thể hệ thống**

Hệ thống được xây dựng theo mô hình **Client-Server**, bao gồm các thành phần chính: giao diện người dùng (frontend), máy chủ xử lý (backend), cơ sở dữ liệu và hạ tầng triển khai. Các thành phần này được tổ chức và triển khai tách biệt nhằm tăng tính linh hoạt, khả năng bảo trì và mở rộng về sau.

#### **3.1.1. Frontend:**

Giao diện người dùng được phát triển bằng React.js, giúp xây dựng các thành phần UI hiện đại và phản hồi nhanh. Giao diện có nhiệm vụ hiển thị dữ liệu công việc, lịch trình, trạng thái và cho phép người dùng tương tác như thêm, sửa, xóa công việc. Giao diện này được triển khai trên nền tảng Netlify, một dịch vụ hosting hỗ trợ CI/CD giúp tự động cập nhật phiên bản mới sau mỗi lần đẩy code lên Git. Netlify cũng tối ưu tốc độ tải trang và phân phối qua CDN, đảm bảo trải nghiệm người dùng tốt trên nhiều thiết bị.

#### **3.1.2. Backend:**

Phần xử lý logic nghiệp vụ được xây dựng với Node.js kết hợp với Express.js, đóng vai trò là cầu nối giữa frontend và cơ sở dữ liệu. Backend chịu trách nhiệm xác thực người dùng, xử lý yêu cầu tạo/chỉnh sửa/xóa công việc, phân loại, cập nhật trạng thái và ghi nhận lịch sử thay đổi. Các API được thiết kế theo chuẩn RESTful để đảm bảo khả năng mở rộng, tái sử dụng. Dịch vụ backend được triển khai trên nền tảng Render, giúp duy trì hoạt động ổn định và dễ cấu hình môi trường triển khai thực tế.

#### **3.1.3. Cơ sở dữ liệu:**

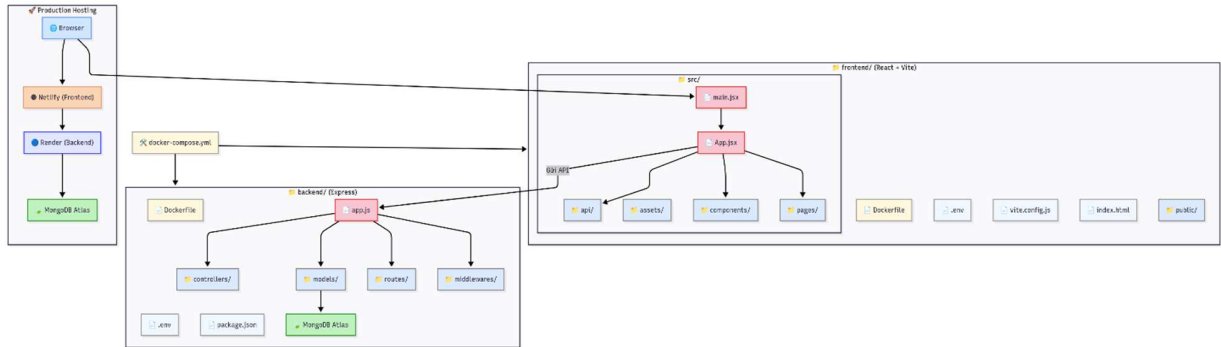
Hệ thống sử dụng MongoDB làm hệ quản trị cơ sở dữ liệu NoSQL, phù hợp với dạng dữ liệu linh hoạt của công việc và người dùng. Dữ liệu được lưu trữ thành các collection như: users, jobs, typejobs, logs,... Hệ thống có thể sử dụng MongoDB Atlas.

#### **3.1.4. Docker:**

Công nghệ Docker được áp dụng để đóng gói và triển khai backend. Docker cho phép môi trường phát triển và môi trường triển khai có thể đồng nhất, giảm rủi ro lỗi

phát sinh do sự khác biệt hệ thống. Backend được đóng gói thành container, có thể dễ dàng khởi chạy bằng Docker Compose, kèm theo cấu hình biến môi trường.

Tổng thể hệ thống có khả năng hoạt động ổn định, triển khai linh hoạt và dễ dàng tích hợp các tính năng mở rộng trong tương lai như thông báo đẩy, chia sẻ công việc nhóm, hoặc tích hợp lịch bên ngoài.

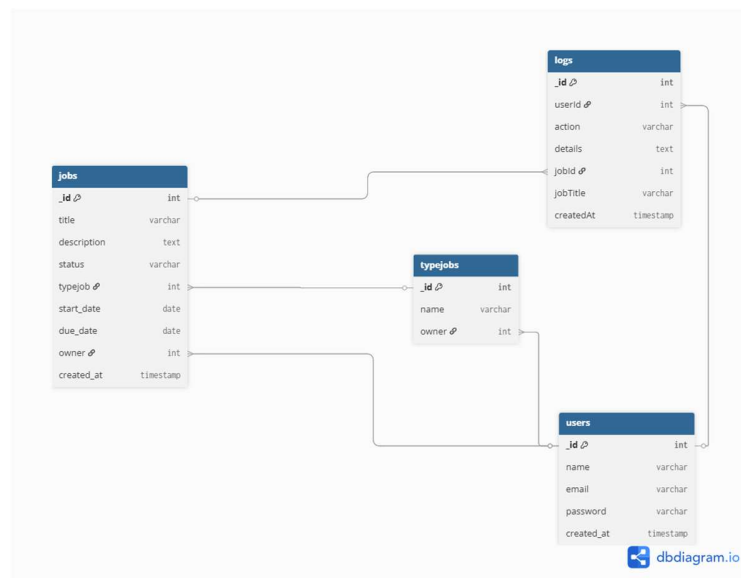


Hình 3.1 Sơ đồ kiến trúc hệ thống

## 3.2. Thiết kế cơ sở dữ liệu

### 3.2.1. Mô hình thực thể - quan hệ (ERD)

Hệ thống được thiết kế với 4 bảng dữ liệu chính: users, jobs, typejobs, và logs. Mỗi quan hệ giữa các bảng thể hiện qua mô hình ERD sau:



Hình 3.2 Mô hình ERD

### 3.2.2. Mô tả bảng dữ liệu

#### 3.2.2.1. Bảng users – Quản lý người dùng

Tên trường	Kiểu dữ liệu	Mô tả
_id	int	Khóa chính, định danh người dùng
name	varchar	Họ và tên người dùng
email	varchar	Email đăng nhập (duy nhất)
password	varchar	Mật khẩu được mã hóa
created_at	timestamp	Thời điểm tạo tài khoản

#### 3.2.2.2. Bảng jobs – Lưu thông tin công việc

Tên trường	Kiểu dữ liệu	Mô tả
_id	int	Khóa chính
title	varchar	Tiêu đề công việc
description	text	Mô tả chi tiết
status	varchar	Trạng thái (chưa bắt đầu, đang làm...)
typejob	int	Khóa ngoại đến typejobs
start_date	date	Ngày bắt đầu
due_date	date	Hạn hoàn thành
owner	int	Khóa ngoại đến users
created_at	timestamp	Ngày tạo công việc



3.2.2.3. Bảng typejobs – Quản lý loại công việc

Tên trường	Kiểu dữ liệu	Mô tả
_id	int	Khóa chính
name	varchar	Tên loại công việc (ví dụ: học tập...)
owner	int	Khóa ngoại đến users

3.2.2.4. Bảng logs – Ghi lại lịch sử thay đổi công việc

Tên trường	Kiểu dữ liệu	Mô tả
_id	int	Khóa chính
userId	int	Người thực hiện hành động
action	varchar	Loại hành động (xóa, sửa, cập nhật...)
details	text	Nội dung chi tiết
jobId	int	Công việc bị ảnh hưởng
jobTitle	varchar	Tiêu đề công việc tại thời điểm đó
createdAt	timestamp	Thời điểm xảy ra hành động

### 3.3. Thiết kế API

Hệ thống được xây dựng theo mô hình RESTful API, chia thành các nhóm chính tương ứng với các tài nguyên: Xác thực người dùng (Auth), Công việc (Job), Loại công việc (Typejob) và Lịch sử hoạt động (Logs). Dưới đây là mô tả các endpoint chính và cấu trúc request/response cơ bản.

3.3.1. Xác thực người dùng (Auth)

Phương thức	Đường dẫn	Mô tả
POST	/auth/register	Đăng ký tài khoản
POST	/auth/login	Đăng nhập

POST	/auth/change-password	Đổi mật khẩu (yêu cầu token)
POST	/auth/forgot-password	Gửi mã xác thực đến email
POST	/auth/reset-password	Đặt lại mật khẩu bằng mã

### 3.3.2. Quản lý công việc (Job)

Phương thức	Đường dẫn	Mô tả
GET	/job/jobs	Lấy tất cả công việc của người dùng
GET	/job/jobs/filter?typejob=1	Lọc công việc theo loại công việc
GET	/job/jobs/type/name/<name>	Lọc công việc theo tên loại
POST	/job/jobs	Tạo công việc mới
PUT	/job/jobs/:id	Cập nhật công việc
DELETE	/job/jobs/:id	Xoá công việc

### 3.3.3. Quản lý loại công việc (Typejob)

Phương thức	Đường dẫn	Mô tả
GET	/typejob	Lấy danh sách loại công việc
POST	/typejob	Tạo mới loại công việc
PUT	/typejob/:id	Cập nhật loại công việc
DELETE	/typejob/:id	Xoá loại công việc

### 3.3.4. Nhật ký thay đổi (Logs)

Phương thức	Đường dẫn	Mô tả
GET	/logs	Xem lịch sử thay đổi công việc (tạo, sửa, xóa)

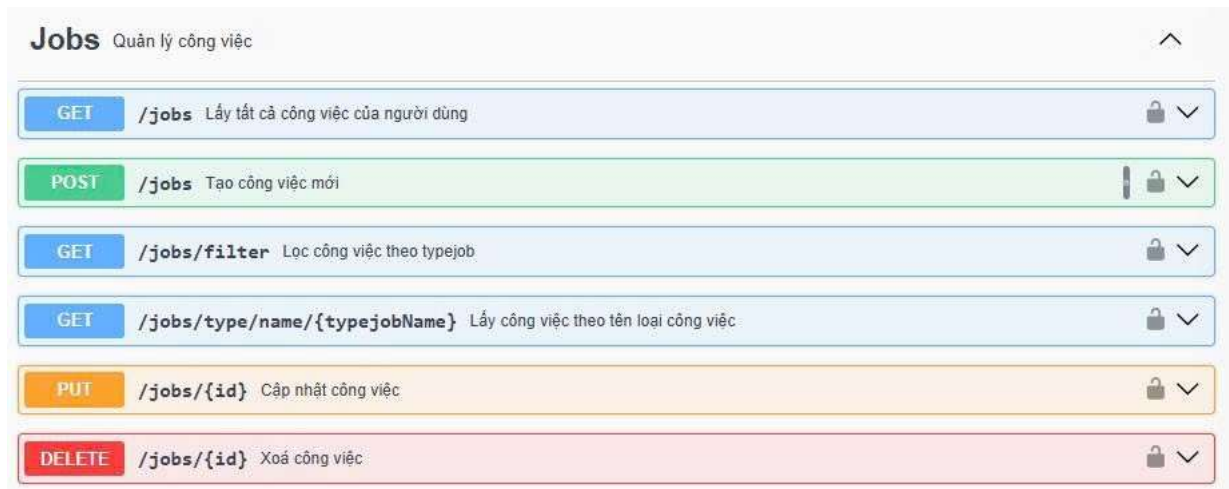
#### 3.3.1. Các hình ảnh Swagger

Giao diện Swagger UI của API Quản Lý Công Việc Cá Nhân, hiển thị các chức năng xác thực người dùng gồm: Đăng ký người dùng mới, Đăng nhập, Đổi mật khẩu, Gửi mã xác nhận đặt lại mật khẩu, và Đặt lại mật khẩu bằng mã xác nhận, tất cả đều hoạt động tại server <http://localhost:3000/api>



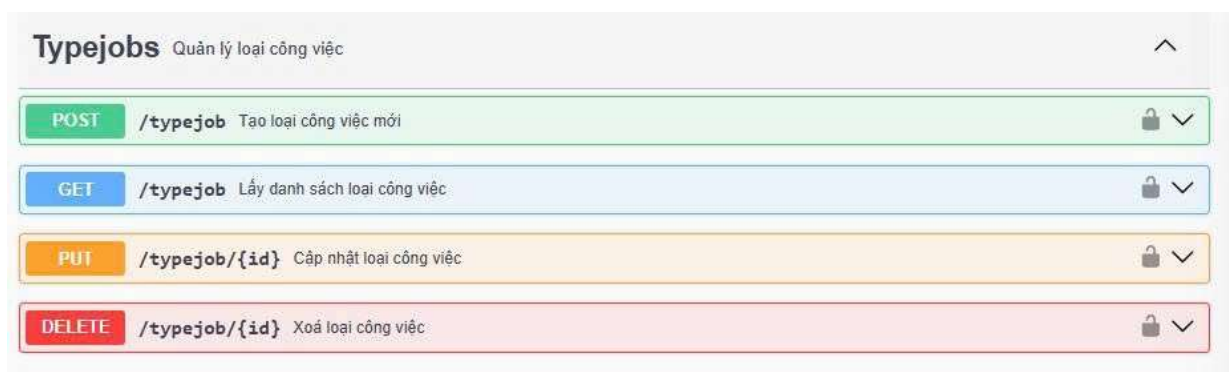
Hình 3.3 Swagger xác thực người dùng

Giao diện Swagger UI phân quản lý công việc trong API Quản Lý Công Việc Cá Nhân, hiển thị các chức năng sau: Lấy tất cả công việc của người dùng, Tạo công việc mới, Lọc công việc theo loại, Lấy công việc theo tên loại công việc, Cập nhật công việc và Xóa công việc.



Hình 3.4 Swagger quản lý công việc

Giao diện Swagger UI phân quản lý loại công việc trong API Quản Lý Công Việc Cá Nhân, hiển thị các chức năng sau: Tạo loại công việc mới, Lấy danh sách loại công việc, Cập nhật loại công việc và Xóa loại công việc.

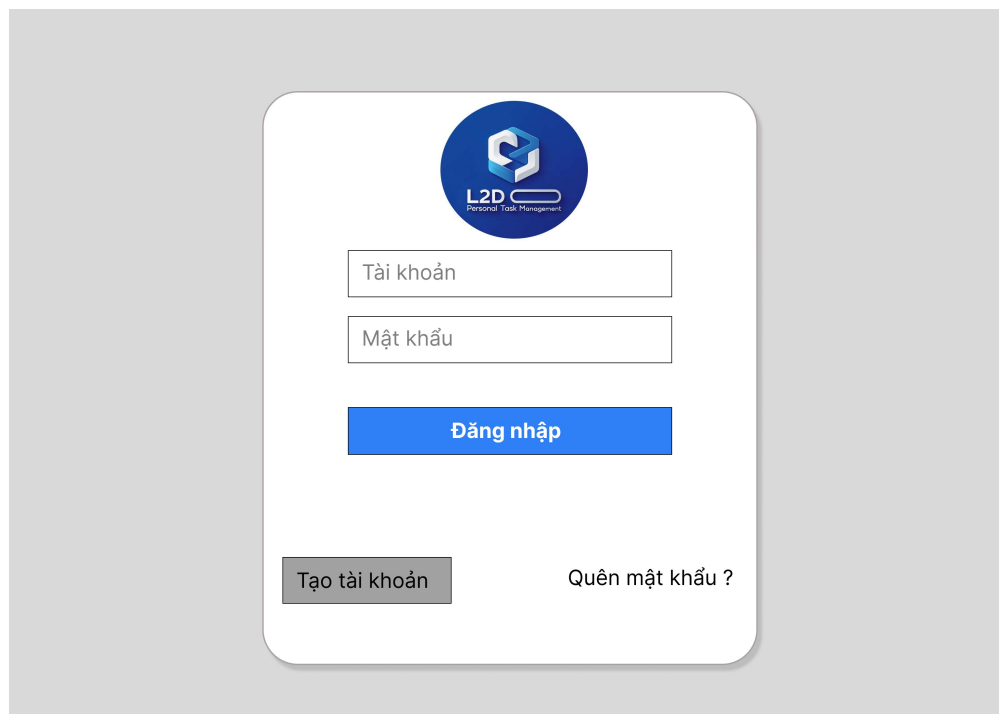


Hình 3.5 Swagger quản lý loại công việc

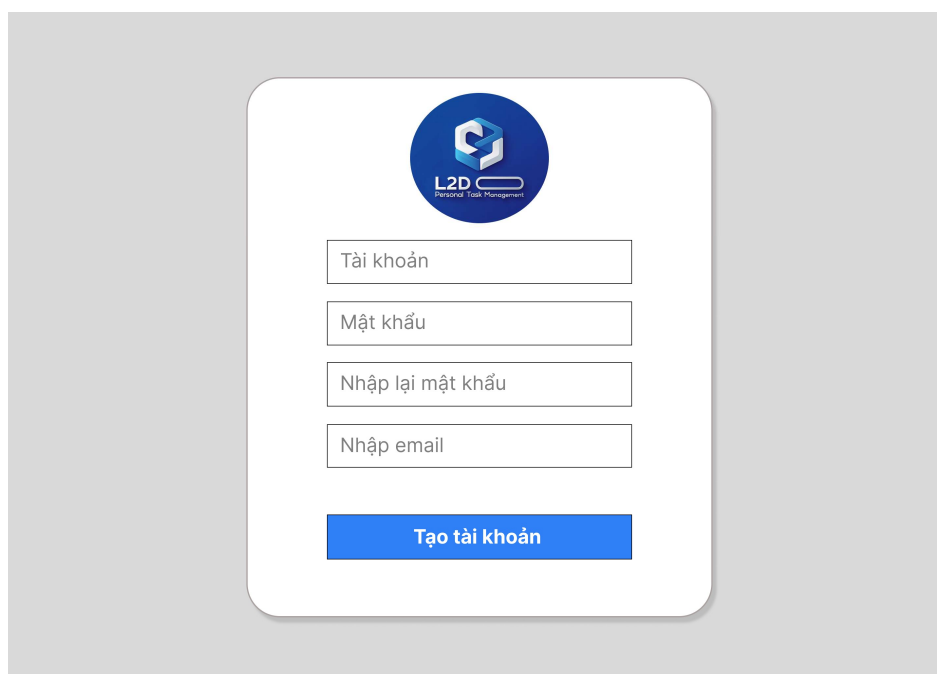
### 3.4. Thiết kế giao diện (UI/UX)

Giao diện người dùng được thiết kế với mục tiêu đơn giản, trực quan và dễ sử dụng, nhằm mang lại trải nghiệm tốt nhất cho người dùng trong quá trình quản lý công việc cá nhân. Quy trình thiết kế tuân theo nguyên tắc UX cơ bản như nhất quán, phản hồi nhanh, và tối ưu cho thao tác thường xuyên. Mỗi thành phần trên giao diện đều được bố trí rõ ràng, hỗ trợ người dùng dễ dàng tạo, xem, chỉnh sửa và theo dõi công việc cũng như loại công việc của mình.

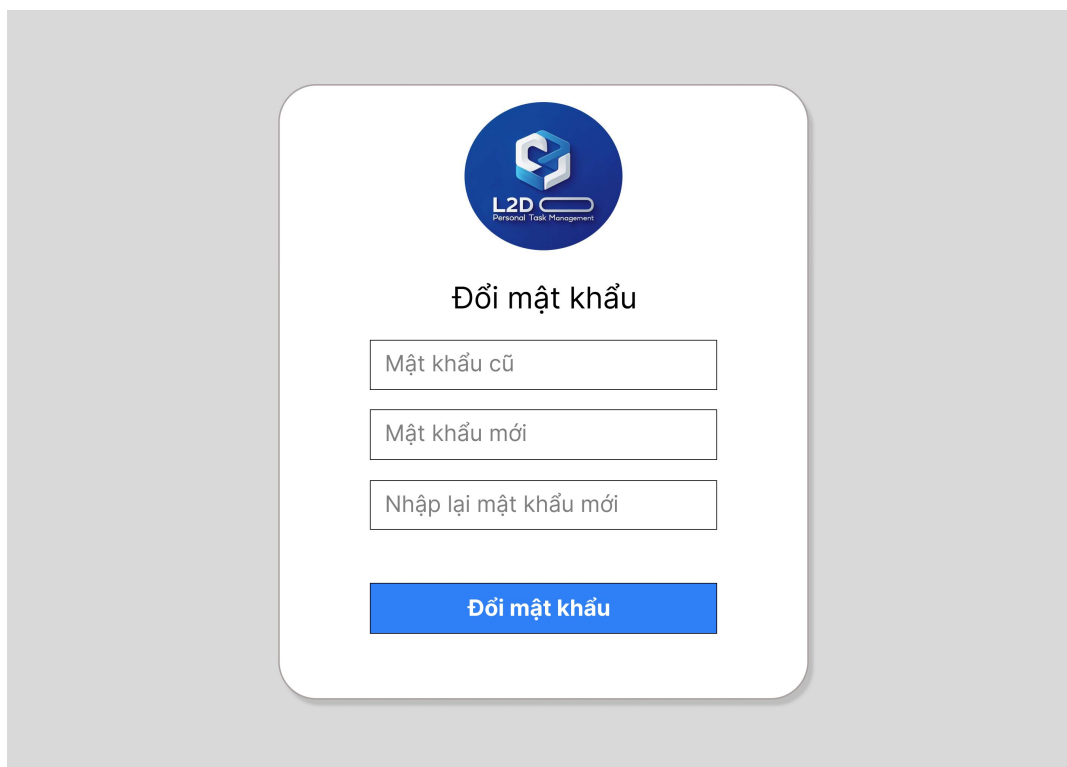
Phần dưới bao gồm bản thiết kế bằng Figma và giao diện thực tế sau khi triển khai để so sánh giữa ý tưởng và sản phẩm hoàn thiện.




Hình 3.6 Figma đăng nhập



Hình 3.7 Figma tạo tài khoản





**Đổi mật khẩu**

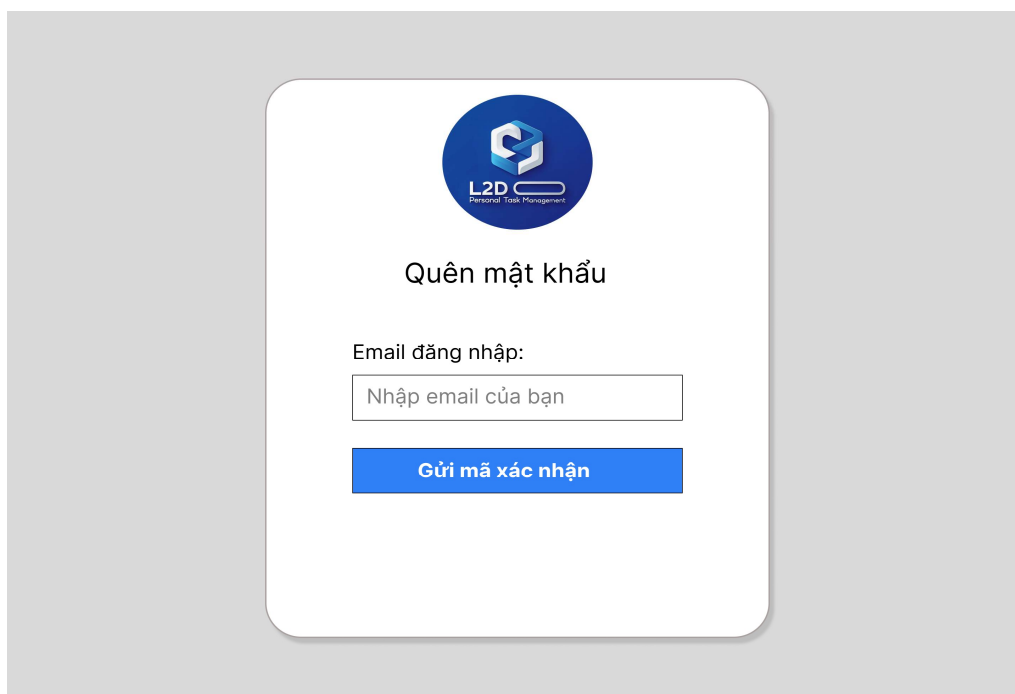
Mật khẩu cũ


Mật khẩu mới

Nhập lại mật khẩu mới

**Đổi mật khẩu**

Hình 3.8 Figma đổi mật khẩu





**Quên mật khẩu**

Email đăng nhập:

Nhập email của bạn

**Gửi mã xác nhận**

Hình 3.9 Figma quên mật khẩu

The image shows a Figma design for a password reset form. At the top is the L2D logo with the text "L2D Personal Task Management". Below the logo is the title "Đặt lại mật khẩu". There are two input fields: "Mật khẩu mới" (New password) and "Nhập lại mật khẩu mới" (Repeat new password). At the bottom is a blue button labeled "Đổi mật khẩu" (Change password).

Hình 3.10 Figma đặt lại mật kh

The image shows a Figma design for a dashboard. On the left is a dark blue sidebar with the L2D logo and the text "L2D Personal Task Management". Below the logo are four menu items: "Tổng Quan", "Loại Công Việc", "Công Việc", and "Lịch sử thay đổi". At the bottom of the sidebar are three items: "Tài Khoản", "Đăng xuất", and "Đổi mật khẩu". The main content area is titled "TỔNG QUAN". Below the title are five colored buttons: "Loại công việc" (0), "Số lượng công việc" (0), "Chưa thực hiện" (0), "Đang thực hiện" (0), and "Đã hoàn thành" (0). Below these buttons is a calendar for "Tháng 5 / 2025". The calendar has columns for days of the week (CN, T2, T3, T4, T5, T6, T7) and rows for dates (1-31). Below the calendar are two sections: "Trạng thái công việc" (Task status) with a blue circular progress indicator, and "Thông báo công việc" (Task notification) with four input fields.

Hình 3.11 Figma tổng quan

**L2D**  
Personal Task Management

Tổng Quan  
Loại Công Việc  
Công Việc  
Lịch sử thay đổi

Tài Khoản  
Đăng xuất  
Đổi mật khẩu

## LOẠI CÔNG VIỆC

Nhập tên công việc mới

+ Thêm

Thiết kế

Phân tích

Hình 3.12 Figma loại công việc

**L2D**  
Personal Task Management

Tổng Quan  
Loại Công Việc  
Công Việc  
Lịch sử thay đổi

Tài Khoản  
Đăng xuất  
Đổi mật khẩu

## CÔNG VIỆC

### Thêm công việc mới

Tên công việc

Loại công việc

Ngày bắt đầu

Ngày kết thúc

Trạng thái

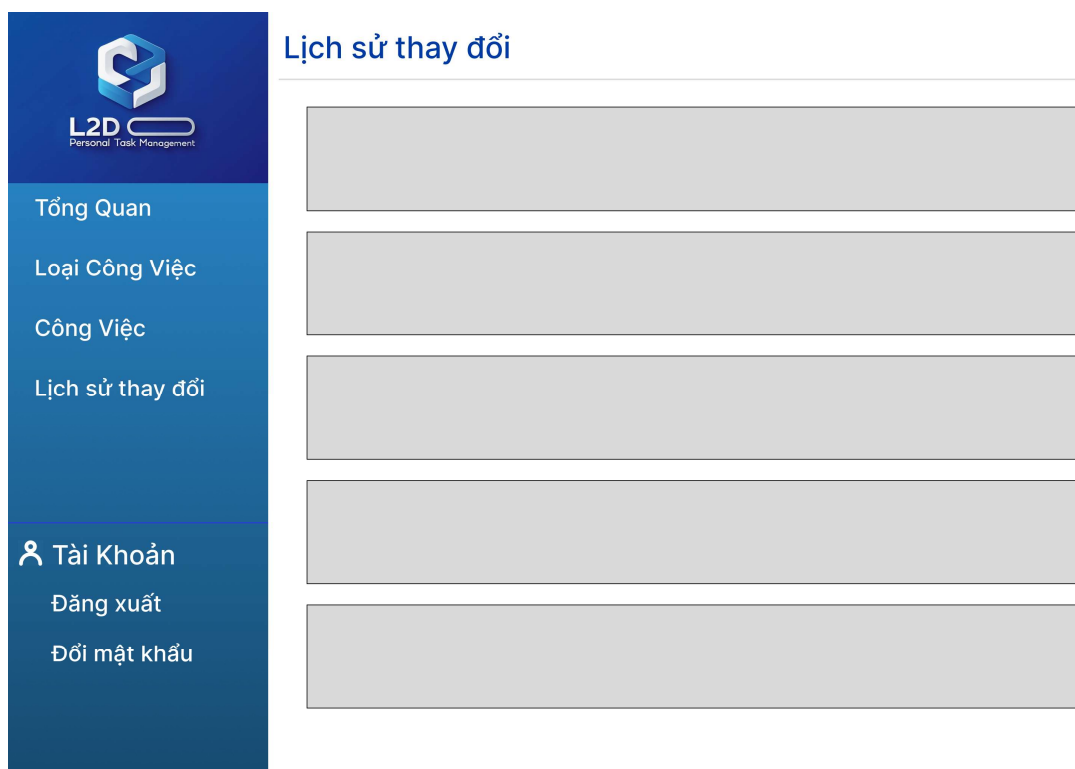
File đính kèm

Mô tả

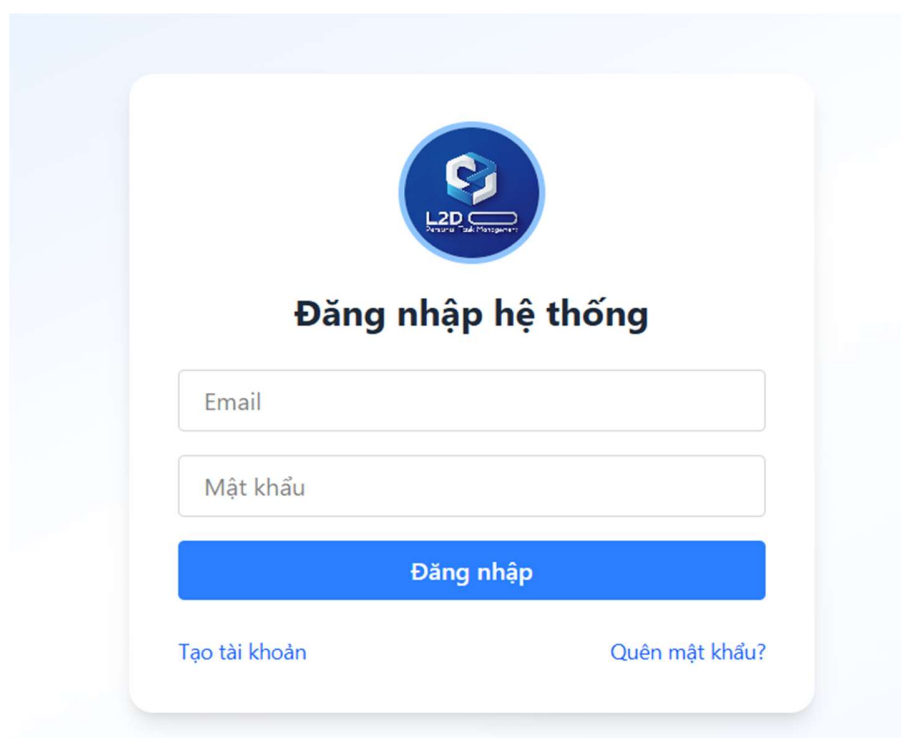
+ Thêm công việc

Hình 3.13 Figma công việc

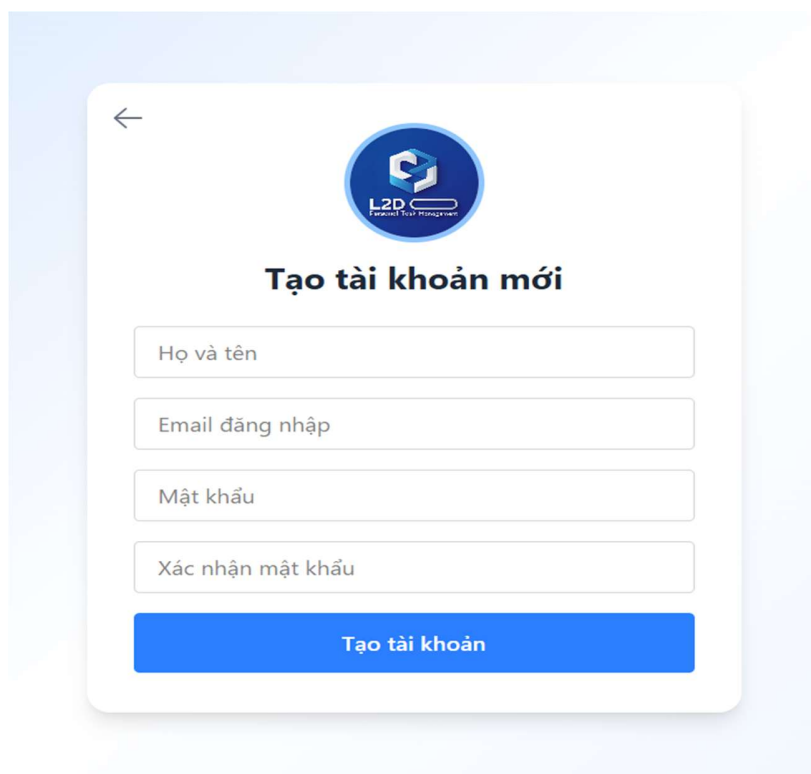





Hình 3.14 Figma lịch sử thay đổi



Hình 3.15 Giao diện đăng nhập hệ thống



←



**Tạo tài khoản mới**

Họ và tên

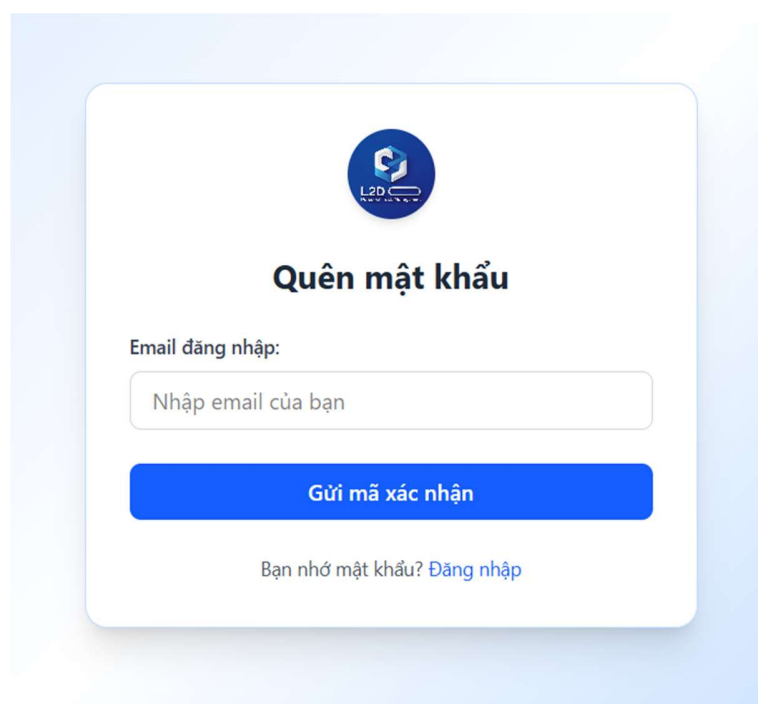
Email đăng nhập


Mật khẩu

Xác nhận mật khẩu

**Tạo tài khoản**

Hình 3.16 Giao diện tạo khoản mới





**Quên mật khẩu**

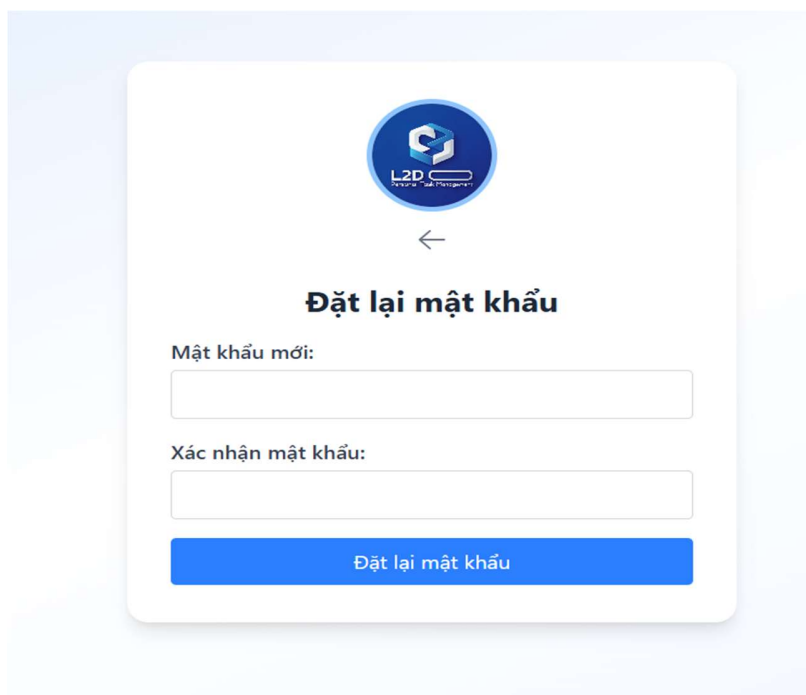
Email đăng nhập:


Nhập email của bạn

**Gửi mã xác nhận**

Bạn nhớ mật khẩu? [Đăng nhập](#)

Hình 3.17 Giao diện quên mật khẩu





←

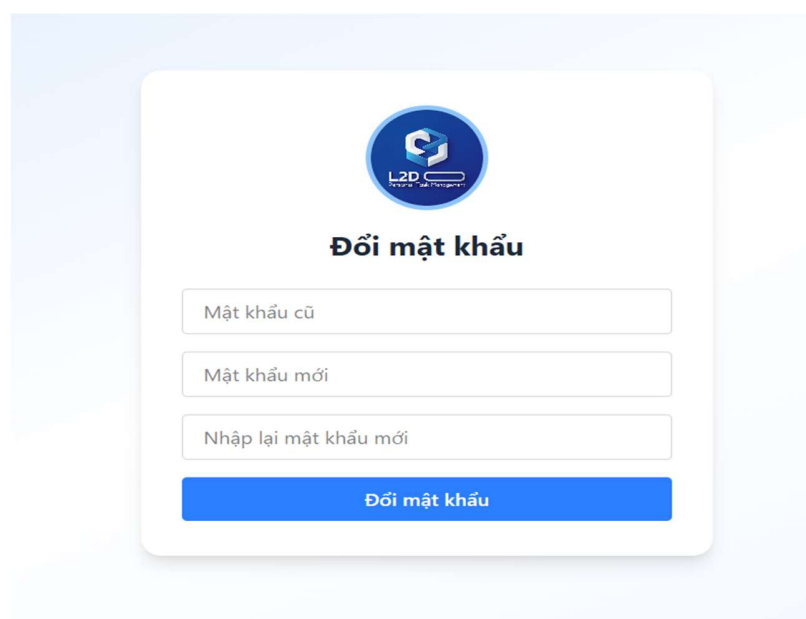
**Đặt lại mật khẩu**


Mật khẩu mới:

Xác nhận mật khẩu:

Đặt lại mật khẩu

Hình 3.18 Giao diện đặt lại mật khẩu





**Đổi mật khẩu**

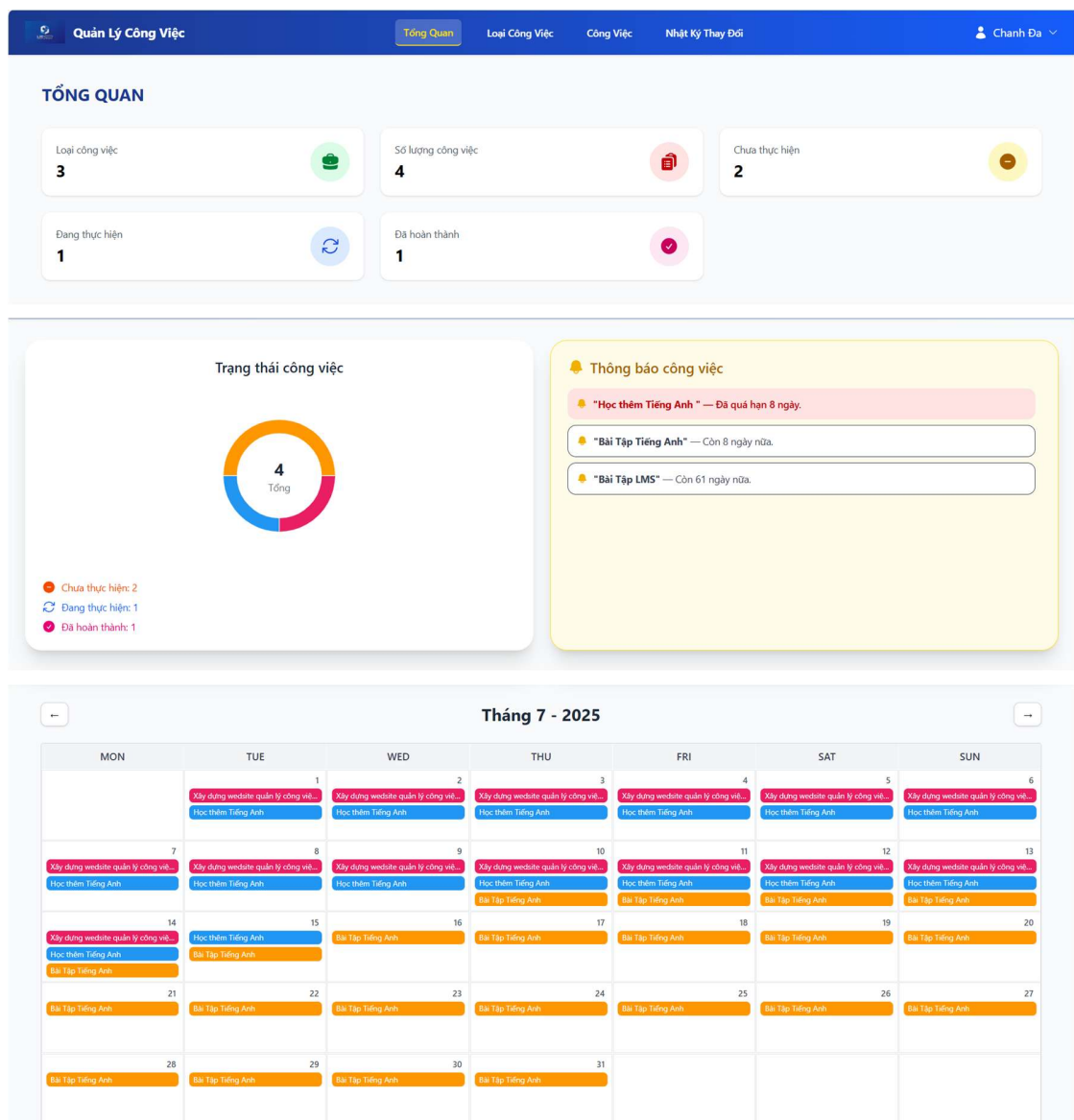
Mật khẩu cũ

Mật khẩu mới

Nhập lại mật khẩu mới

Đổi mật khẩu

Hình 3.19 Giao diện đổi mật khẩu



Hình 3.20 Giao diện tổng quan

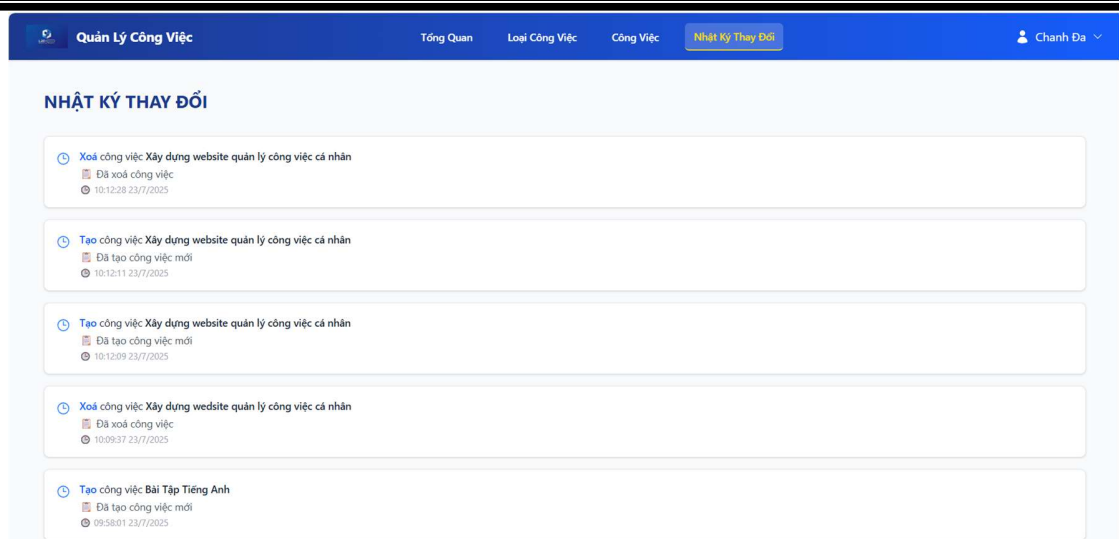
The screenshot shows the 'Loại Công Việc' (Job Type) management page. At the top, there is a navigation bar with 'Quản Lý Công Việc' and tabs for 'Tổng Quan', 'Loại Công Việc', 'Công Việc', and 'Nhật Ký Thay Đổi'. The user 'Chanh Đa' is logged in. The main section is titled 'LOẠI CÔNG VIỆC'. It features a search bar with the placeholder 'Nhập tên loại công việc mới' and a '+ Thêm' button. Below the search bar are three filter buttons: 'Đồ Án', 'Học thêm', and 'Bài Tập'.

Hình 3.21 Giao diện loại công việc

The screenshot shows the 'QUẢN LÝ CÔNG VIỆC' (Job Management) page. The navigation bar is the same as in the previous image. The main section is titled 'QUẢN LÝ CÔNG VIỆC'. It features a 'THÊM CÔNG VIỆC MỚI' (Add New Job) form. The form has the following fields: 'Tên công việc' (Job Name) with the value 'Xây dựng website quản lý công việc cá nhân', 'Loại công việc' (Job Type) with the value 'Đồ Án', 'Ngày bắt đầu' (Start Date) with the value '01/06/2025', 'Ngày kết thúc' (End Date) with the value '14/07/2025', 'Trạng thái' (Status) with the value 'Đã hoàn thành', and 'Mô tả' (Description) with the value 'Kết thúc môn công nghệ phần mềm'. There is a '+ Thêm công việc' button at the bottom right of the form. Below the form, there are filters: '-- Chọn theo loại --', '-- Chọn theo trạng thái --', and a date picker 'dd/mm/yyyy'. There are four job cards displayed in a grid:

- Học thêm Tiếng Anh**: Loại: Học thêm, Từ: 2025-07-01 → 2025-07-15, Trạng thái: Đang thực hiện, Học thêm Tiếng Anh.
- Bài Tập LMS**: Loại: Bài Tập, Từ: 2025-09-08 → 2025-09-22, Trạng thái: Chưa thực hiện, Bài Tập LMS.
- Bài Tập Tiếng Anh**: Loại: Bài Tập, Từ: 2025-07-10 → 2025-07-31, Trạng thái: Chưa thực hiện, Bài tập của học thêm Tiếng Anh.
- Xây dựng website quản lý công việc cá nhân**: Loại: Đồ Án, Từ: 2025-06-01 → 2025-07-14, Trạng thái: Đã hoàn thành, Kết thúc môn công nghệ phần mềm.

Hình 3.22 Giao diện quản lý công việc



Hình 3.23 Giao diện nhật ký thay đổi

## **CHƯƠNG 4: TRIỂN KHAI VÀ CÔNG NGHỆ SỬ DỤNG**

### **4.1. Công nghệ sử dụng**

Trong quá trình phát triển hệ thống quản lý công việc cá nhân, nhóm đã lựa chọn các công nghệ phổ biến, hiện đại và phù hợp với mục tiêu của đề án, bao gồm:

#### **Ngôn ngữ lập trình và Framework**

- Frontend:
  - Ngôn ngữ: JavaScript
  - Framework/Thư viện: React.js, Tailwind CSS, React Router DOM, Heroicons
- Backend:
  - Ngôn ngữ: JavaScript (Node.js)
  - Framework: Express.js
  - Thư viện hỗ trợ: Mongoose (kết nối MongoDB), Multer (xử lý upload tệp), JSON Web Token (xác thực người dùng), Nodemailer (gửi email khi quên mật khẩu)

#### **Cơ sở dữ liệu**

- MongoDB Atlas: Lưu trữ toàn bộ dữ liệu của hệ thống như thông tin người dùng, công việc, loại công việc, lịch sử thay đổi...

#### **Triển khai docker:**

- Docker: Đóng gói backend, frontend để dễ triển khai và đồng nhất môi trường giữa các máy.
- Dotenv: Quản lý biến môi trường trong các file .env frontend và .env.docker, .env backend

## **4.2. Quy trình CI/CD với GitHub Actions**

Hệ thống sử dụng GitHub Actions để thiết lập quy trình kiểm tra và triển khai tự động mã nguồn, đảm bảo chất lượng và giảm thiểu lỗi khi cập nhật.

- Một workflow tên là lint.yml được cấu hình tại thư mục .github/workflows/.
- Workflow tự động chạy mỗi khi có thay đổi được đẩy lên nhánh main.
- Các bước chính của workflow:
  - Kiểm tra lint code theo quy tắc ESLint.
  - Xác nhận project có thể build thành công.

Quy trình CI/CD giúp tự động hóa quá trình kiểm tra chất lượng mã nguồn, đồng thời đẩy nhanh quá trình phát hành phiên bản mới cho người dùng.

## **4.3. Cấu hình Docker và Docker Compose**

Hệ thống sử dụng Docker để đóng gói cả frontend và backend, giúp đảm bảo môi trường nhất quán trong quá trình phát triển, kiểm thử và triển khai.

### **Cấu hình cụ thể:**

- Dockerfile Backend:
  - Đặt tại thư mục backend/.
  - Mô tả quá trình build backend từ mã nguồn Node.js, cài đặt các thư viện, và chạy server Express.
- Dockerfile Frontend:
  - Đặt tại thư mục frontend/.
  - Mô tả quá trình build ứng dụng React, sử dụng npm run build để tạo thư mục dist hoặc build, sau đó có thể dùng Nginx hoặc các dịch vụ cloud như Netlify để phục vụ frontend.
- docker-compose.yml:
  - Dùng để chạy toàn bộ hệ thống backend bằng lệnh docker-compose up -d.



- Do MongoDB sử dụng dịch vụ cloud (MongoDB Atlas) nên không cần định nghĩa container MongoDB hay volume trong docker-compose.
- Có thể bổ sung service frontend nếu cần chạy frontend từ container trong môi trường development hoặc staging.

**Biến môi trường:**

- Frontend và backend có các file .env riêng biệt để cấu hình địa chỉ API, cổng chạy ứng dụng, v.v.
- Backend có thêm file .env.docker dùng riêng cho môi trường Docker.

Việc chia tách và sử dụng đúng biến môi trường giữa local và Docker giúp bảo mật thông tin, đồng thời đảm bảo ứng dụng hoạt động chính xác trên từng môi trường.

#### **4.4. Quy trình triển khai dựn dụng**

Hệ thống được triển khai trên các nền tảng cloud miễn phí với khả năng CI/CD linh hoạt:

- Frontend:
  - Được deploy trên Netlify.
  - Mỗi lần push code lên nhánh main, Netlify tự động thực hiện build và phát hành phiên bản mới lên domain công khai.
- Backend:
  - Được deploy trên Render.
  - Backend được build từ Dockerfile.
  - Render tự động nhận cập nhật mới từ GitHub khi có thay đổi và thực hiện quá trình build/deploy.

Thông qua việc sử dụng CI/CD và Docker, việc triển khai trở nên dễ dàng, tiết kiệm thời gian và giúp hệ thống luôn ở trạng thái sẵn sàng phục vụ người dùng.

## **CHƯƠNG 5: QUẢN LÝ DỰ ÁN**

### **5.1. Phương pháp quản lý dự án**

Để đảm bảo quá trình phát triển diễn ra hiệu quả, nhóm đã áp dụng mô hình quản lý dự án Agile với phương pháp Scrum. Scrum cho phép chia nhỏ dự án thành các chu kỳ phát triển ngắn (sprint), giúp dễ dàng kiểm soát tiến độ, điều chỉnh linh hoạt và cải thiện liên tục sau mỗi vòng lặp.

Các thành phần chính trong quy trình Scrum:

- **Product Backlog:** Danh sách các tính năng, yêu cầu và nhiệm vụ được xác định từ ban đầu, đóng vai trò như kho yêu cầu tổng thể cho toàn bộ dự án.
- **Sprint Backlog:** Tập hợp các công việc cụ thể được chọn ra từ Product Backlog để thực hiện trong mỗi Sprint.
- **Sprint:** Dự án được chia thành 5 sprint, mỗi sprint kéo dài từ 1 đến 2 tuần. Sau mỗi sprint, nhóm tiến hành đánh giá lại tiến độ và điều chỉnh kế hoạch nếu cần.

Công cụ hỗ trợ quản lý được sử dụng là Jira Software, một nền tảng quản lý dự án phổ biến với khả năng lập kế hoạch, theo dõi và báo cáo tiến độ chi tiết. Trong quá trình sử dụng Jira:

- Nhóm đã tạo đầy đủ các task và user story tương ứng cho từng yêu cầu cụ thể.
- Các công việc được phân chia rõ ràng, cập nhật theo tiến độ thực tế.
- Jira không tích hợp trực tiếp với GitHub mà chỉ dùng cho mục đích lập kế hoạch và theo dõi.

### **5.2. Phân công nhiệm vụ**

Dự án được thực hiện bởi nhóm gồm các thành viên, mỗi người phụ trách một hoặc nhiều phần việc cụ thể như sau:

<b>Thành viên</b>	<b>Vai trò &amp; Nhiệm vụ chính</b>
Huỳnh Hữu Lộc	Toàn bộ backend: Thiết lập server backend, kết nối mongodb, các chức năng của website như thêm, xóa, sửa công việc, loại công việc. Triển khai lên host.
Trần Quốc Đạm	Xây dựng frontend các trang tổng quan, loại công việc, công việc và nhật ký thay đổi với các chức năng phù hợp. Chức năng đăng ký, đăng nhập và quên mật khẩu.
Triệu Chanh Đa	Triển khai Docker, kết hợp chỉnh sửa giao diện. Thiết kế figma, viết báo cáo.

## CHƯƠNG 6: KIỂM THỬ

### 6.1. Chiến lược kiểm thử

Chiến lược kiểm thử trong dự án bao gồm kết hợp kiểm thử thủ công, kiểm thử tự động và kiểm thử tích hợp liên tục thông qua các công cụ:

- Kiểm thử thủ công: Kiểm tra giao diện, luồng nghiệp vụ chính, xác thực và phân quyền bằng tay trên trình duyệt và Postman.
- Kiểm thử API: Gửi các request đến các endpoint để kiểm tra phản hồi, mã trạng thái và tính đúng đắn của dữ liệu trả về.
- Kiểm thử tự động: Sử dụng GitHub Actions để kiểm tra định dạng mã nguồn và đảm bảo quá trình build không bị lỗi mỗi khi có thay đổi đẩy lên nhánh main.

61 workflow runs			Event	Status	Branch	Actor
✓ Merge pull request #16 from QuocDam279/FE/chanhda	main	17 hours ago	17x	...		
Lint Code with ESLint v9 #10: Commit 8b3d417 pushed by QuocDam279						
✓ Fe/chanhda	FE/chanhda	17 hours ago	27x	...		
Lint Code with ESLint v9 #10: Pull request #16 opened by QuocDam279						
✓ Fix trạng thái dư icon không sử dụng	FE/chanhda	17 hours ago	13x	...		
Lint Code with ESLint v9 #10: Commit c1ae10e pushed by QuocDam279						
✗ Đổi giao diện component tổng quan và menu trái	FE/chanhda	17 hours ago	19x	...		
Lint Code with ESLint v9 #10: Commit 116a70c pushed by QuocDam279						
✓ Merge pull request #15 from QuocDam279/FE/chanhda	main	2 days ago	22x	...		
Lint Code with ESLint v9 #10: Commit 6d5b12b pushed by QuocDam279						
✓ Fe/chanhda	FE/chanhda	2 days ago	16x	...		
Lint Code with ESLint v9 #10: Pull request #15 opened by QuocDam279						
✓ Fix action 3	FE/chanhda	2 days ago	27x	...		
Lint Code with ESLint v9 #10: Commit 299aef7 pushed by QuocDam279						
✗ Fix lỗi action 2	FE/chanhda	2 days ago	17x	...		
Lint Code with ESLint v9 #10: Commit d7c7b8d pushed by QuocDam279						
✗ Fix lỗi action	FE/chanhda	2 days ago	15x	...		
Lint Code with ESLint v9 #10: Commit 956d5b1 pushed by QuocDam279						
✗ Sửa giao diện PQuenmk	FE/chanhda	2 days ago	17x	...		
Lint Code with ESLint v9 #10: Commit 91a1a1d pushed by QuocDam279						
✗ Chỉnh sửa giao diện hiển đại hơn	FE/chanhda	2 days ago	16x	...		
Lint Code with ESLint v9 #10: Commit 10b140d pushed by QuocDam279						
✓ Merge pull request #14 from QuocDam279/FE/chanhda	main	2 days ago	17x	...		
Lint Code with ESLint v9 #10: Commit f1cc07c pushed by QuocDam279						
✓ Fix lỗi gửi mật khẩu, thêm file .env.docker	FE/chanhda	2 days ago	17x	...		
Lint Code with ESLint v9 #10: Pull request #14 opened by QuocDam279						
✓ Fix lỗi gửi mật khẩu, thêm file .env.docker	FE/chanhda	2 days ago	15x	...		
Lint Code with ESLint v9 #10: Commit 4378920 pushed by QuocDam279						

Hình 6.1 Test trên GitHub Action

## 6.2. Công cụ sử dụng

Công cụ	Mục đích sử dụng
Postman	Kiểm thử thủ công các API RESTful
Swagger UI	Giao diện kiểm thử và tài liệu hóa API
GitHub Actions	Tự động hóa kiểm thử và kiểm tra lint mã nguồn
DevTools	Kiểm tra giao diện, sự kiện và lỗi trình duyệt

Bảng 6.1 Công cụ sử dụng

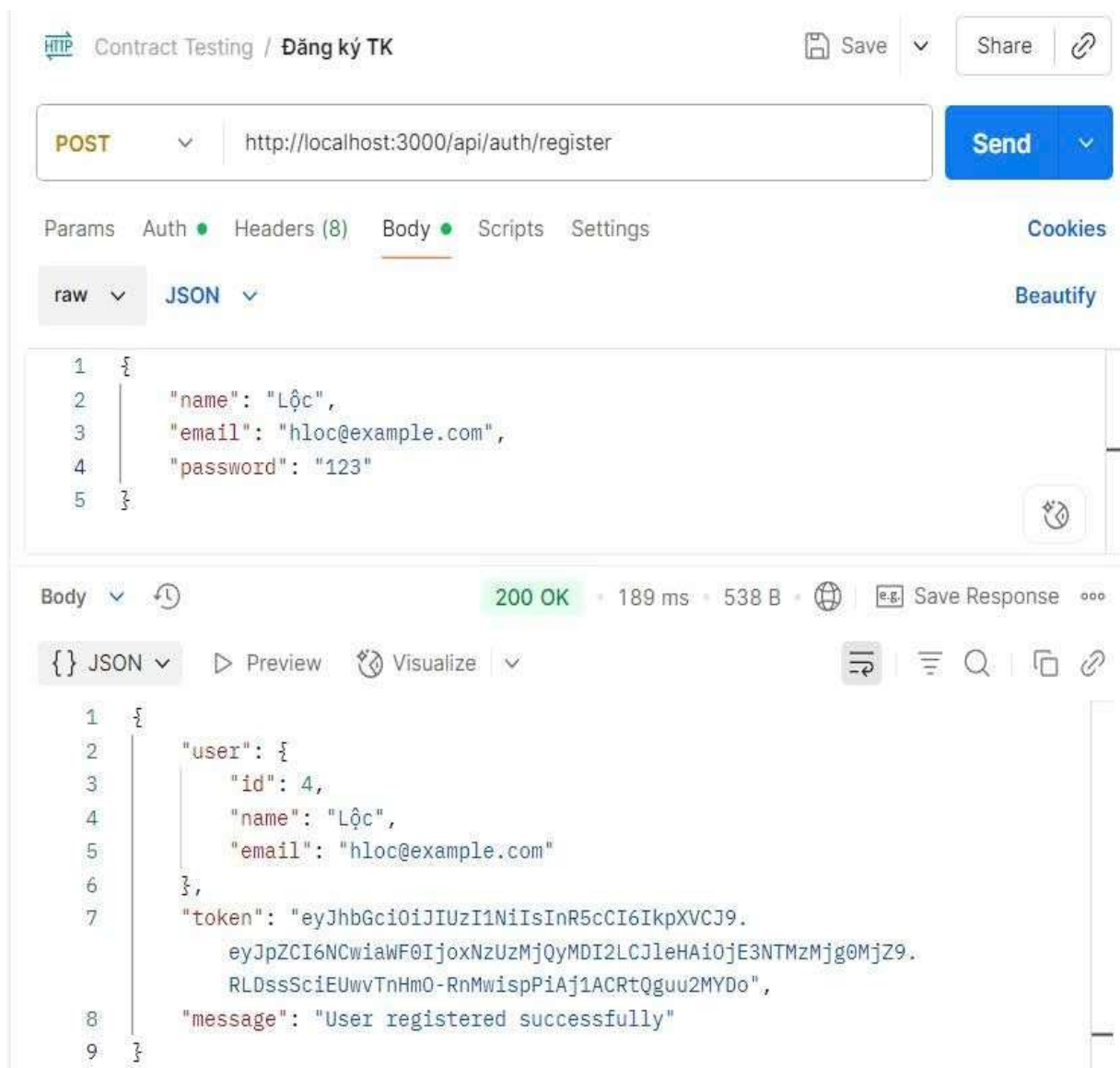
## 6.3. Kết quả kiểm thử API

Đã thực hiện kiểm thử tất cả các endpoint chính của hệ thống:

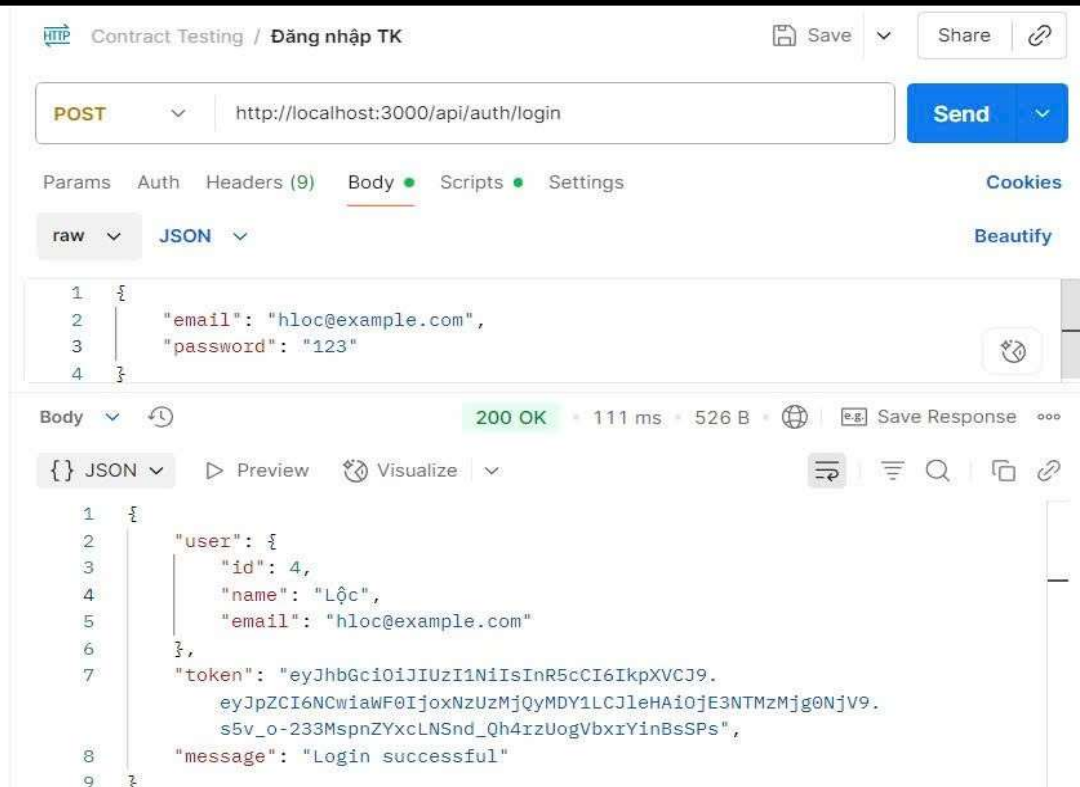
Endpoint	Phương thức	Mô tả	Kết quả
/api/auth/register	POST	Đăng ký tài khoản	Thành công
/api/auth/login	POST	Đăng nhập, nhận JWT	Thành công
/api/job	POST	Tạo công việc mới	Thành công
/api/job/:id	PUT	Cập nhật công việc	Thành công
/api/job/:id	DELETE	Xóa công việc	Thành công
/api/typejob	GET/POST	Lấy danh sách/Thêm loại công việc	Thành công
/api/logs	GET	Xem lịch sử thay đổi	Thành công

Bảng 6.2 Kết quả kiểm thử

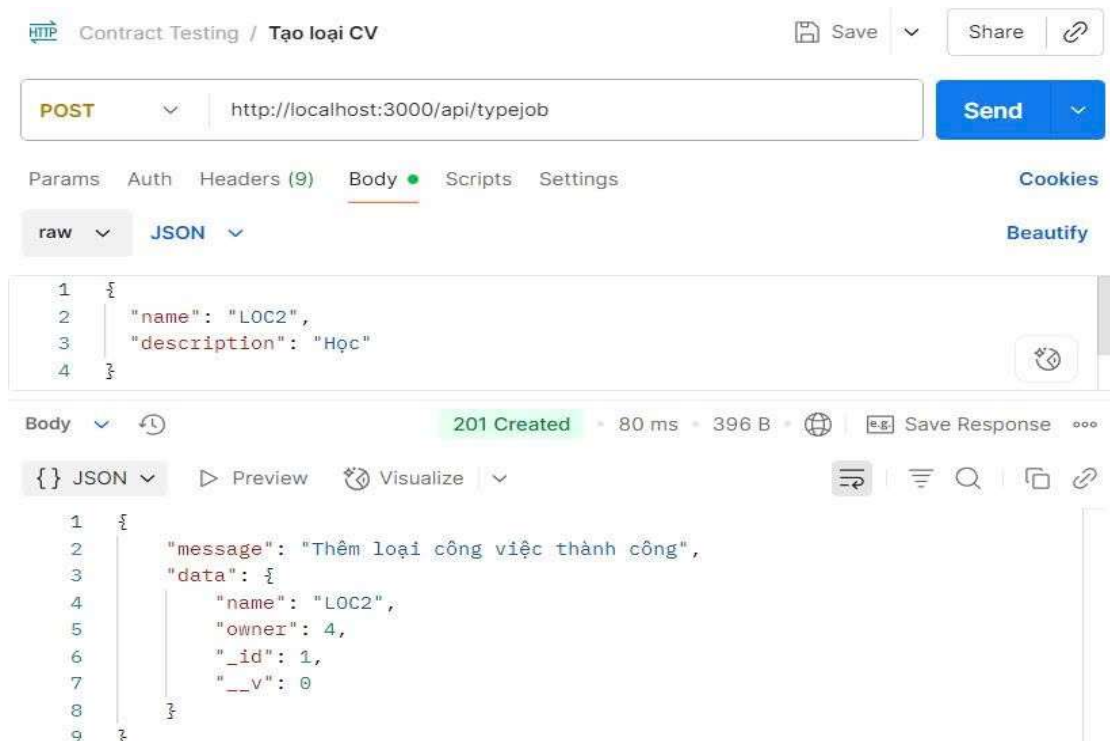
### Một số hình ảnh kiểm thử Postman



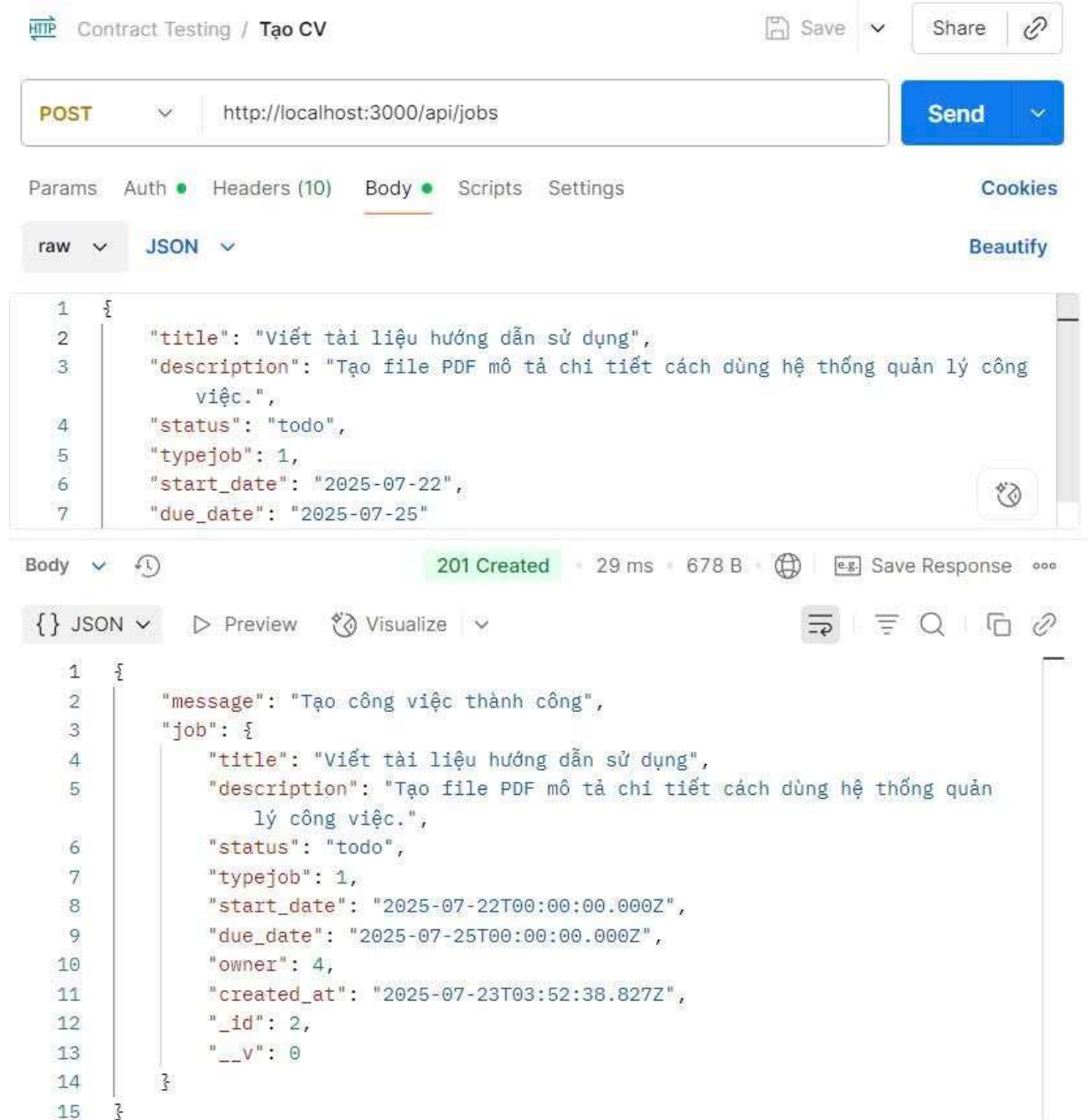
Hình 6.2 Kiểm thử API đăng ký người dùng bằng Postman



Hình 6.3 Kiểm thử API đăng nhập người dùng bằng Postman



Hình 6.4 Kiểm thử API tạo loại công việc bằng Postman



Hình 6.5 Kiểm thử API tạo công việc bằng Postman

#### 6.4. Kết luận kiểm thử

- Hệ thống đã vượt qua tất cả các ca kiểm thử chức năng chính.
- Không phát hiện lỗi nghiêm trọng trong quá trình kiểm thử API và UI.
- Các chức năng bảo mật và phân quyền hoạt động đúng như kỳ vọng.
- CI/CD hoạt động hiệu quả giúp phát hiện lỗi sớm.



## **CHƯƠNG 7: ĐÁNH GIÁ VÀ KẾT LUẬN**

### **7.1. Những khó khăn gặp phải**

Trong quá trình phát triển hệ thống quản lý công việc cá nhân, nhóm đã gặp phải một số khó khăn điển hình như:

- Thiếu kinh nghiệm ban đầu với các công nghệ như Docker, CI/CD, JWT khiến việc cấu hình ban đầu gặp nhiều lỗi và mất thời gian để tra cứu và sửa lỗi.
- Triển khai hệ thống lên môi trường thực tế (Render, Netlify) gặp một số vấn đề về cấu hình môi trường, dẫn đến lỗi CORS, lỗi kết nối cơ sở dữ liệu do sai định dạng URI hoặc biến môi trường không đồng bộ.
- Phân chia công việc trong nhóm đôi khi chưa hiệu quả, một số thành viên gặp khó khăn trong việc phối hợp và đồng bộ tiến độ.

### **7.2. Bài học rút ra**

- Hiểu rõ hơn về quy trình phát triển phần mềm hiện đại, từ việc thiết kế hệ thống, triển khai backend/frontend, cho đến quản lý mã nguồn và triển khai.
- Học được cách sử dụng hiệu quả các công cụ như Git, GitHub, Docker, MongoDB Atlas, Swagger, Postman, CI/CD và cách tích hợp chúng trong một dự án thực tế.
- Tăng kỹ năng làm việc nhóm, phân công công việc, sử dụng Jira để quản lý sprint và theo dõi tiến độ.

### **7.3. Đề xuất và cải tiến trong tương lai**

- Tối ưu giao diện UI/UX để nâng cao trải nghiệm người dùng, đặc biệt là trên thiết bị di động.
- Thêm tính năng thống kê theo thời gian để hỗ trợ người dùng đánh giá hiệu suất cá nhân.
- Tích hợp tính năng gửi email thông báo khi có công việc đến hạn hoặc có thay đổi quan trọng.
- Hoàn thiện kiểm thử tự động (unit test) cho backend nhằm nâng cao độ tin cậy và giúp phát hiện lỗi sớm hơn.

## **TÀI LIỆU THAM KHẢO**

- [1] Mozilla, “JWT (JSON Web Token),” *MDN Web Docs*. [Online]. Available: [https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication#json\\_web\\_tokens\\_jwt](https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication#json_web_tokens_jwt). [Accessed: Jul. 23, 2025].
- [2] MongoDB Inc., “MongoDB Atlas Documentation,” *MongoDB*. [Online]. Available: <https://www.mongodb.com/docs/atlas/>. [Accessed: Jul. 23, 2025].
- [3] Docker Inc., “Docker Documentation,” *Docker Docs*. [Online]. Available: <https://docs.docker.com/>. [Accessed: Jul. 23, 2025].
- [4] Swagger, “OpenAPI Specification,” *Swagger*. [Online]. Available: <https://swagger.io/specification/>. [Accessed: Jul. 23, 2025].
- [5] ReactJS, “React – A JavaScript library for building user interfaces,” *ReactJS.org*. [Online]. Available: <https://reactjs.org/>. [Accessed: Jul. 23, 2025].
- [6] ExpressJS, “Express - Node.js web application framework,” *ExpressJS.com*. [Online]. Available: <https://expressjs.com/>. [Accessed: Jul. 23, 2025].