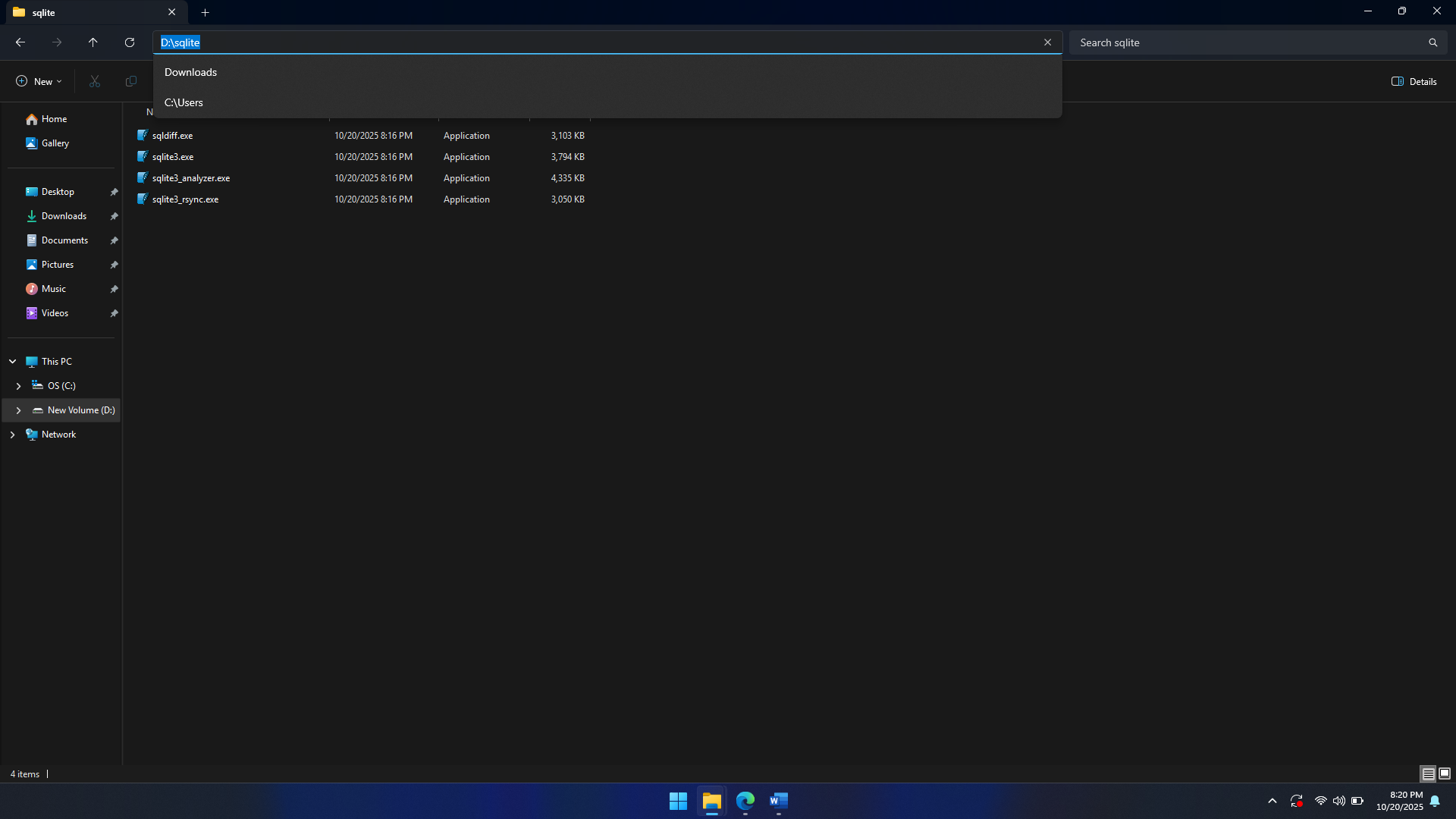
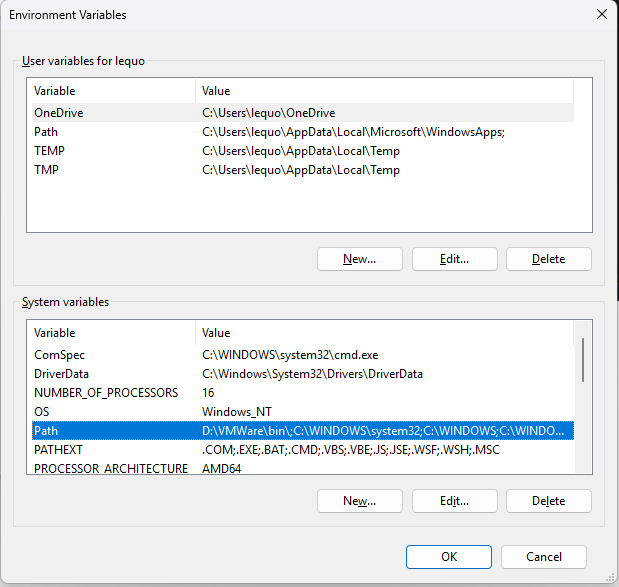
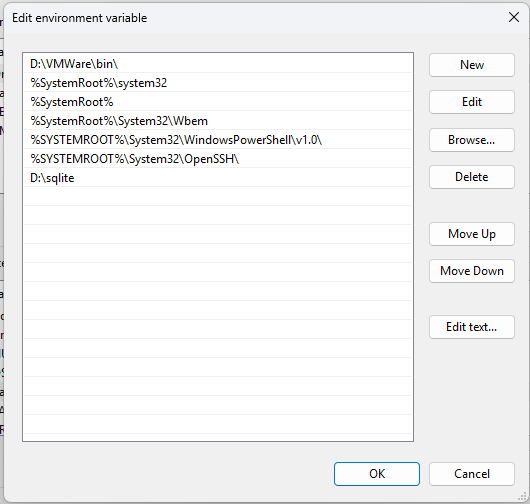
1. **Triển khai hệ thống**
2. ***Hệ quản trị cơ sở dữ liệu***
3. ***Chuẩn bị***

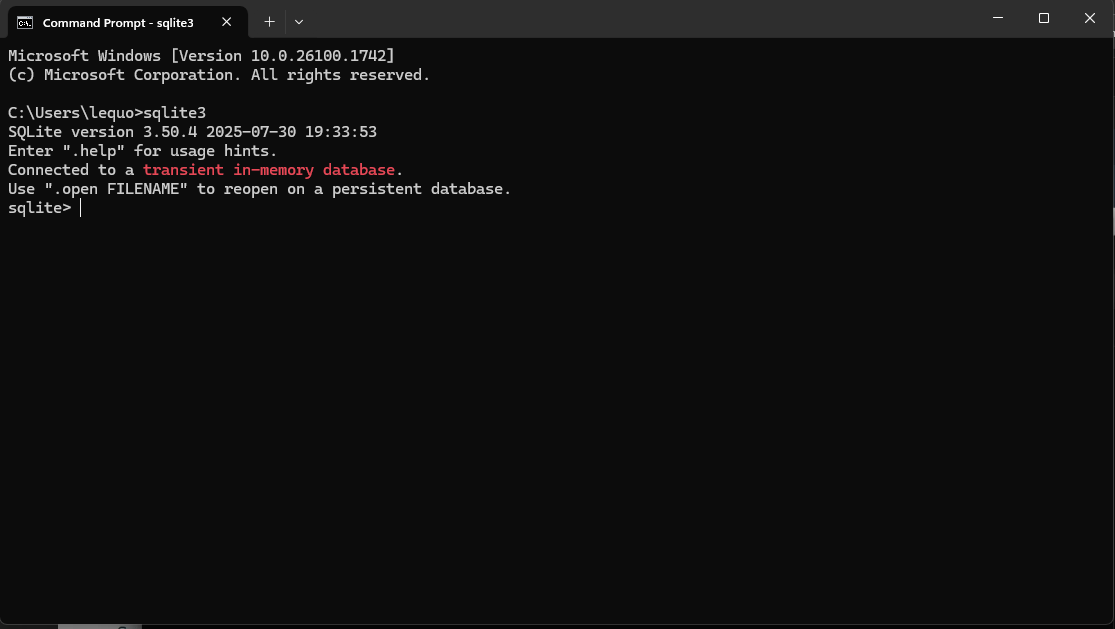
* Tải hệ quản trị cơ sở dữ liệu tại: <https://sqlite.org/2025/sqlite-tools-win-x64-3500400.zip>

1. ***Các bước cài đặt SQLite***

* ***Bước 1****: Giải nén folder sqlite.zip vừa tải ở trên*
* ***Bước 2****: Chuyển folder sqlite vừa giải nén sang một ổ bất kì (ổ C, …)*
* ***Bước 3****: Nhấn vào folder sqlite và copy đường dẫn của folder sqlite*
* *Bước 4: Nhấn phím* Window *, Gõ* ***edit the system environment variables,*** *Chọn* ***Environment Variables…****,*

**

* ***Bước 5****: Tại phần* ***System variables****, Nháy đúp chuột phần* ***Path****, chọn* ***New*** *và dán đường dẫn folder sqlite*
* ***Bước 6:*** *Nhấn phím* Window *, Gõ* ***Terminal,*** *chọn Terminal và gõ* ***sqlite3***

**

1. ***Các truy vấn SQL chính***
2. ***CRUD (Create – Read – Update – Delete)***

* ***Create***

CREATE TABLE "auth\_user" (

"id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "password" varchar(128) NOT NULL,

"last\_login" datetime NULL,

"is\_superuser" bool NOT NULL,

"username" varchar(150) NOT NULL UNIQUE,

"last\_name" varchar(150) NOT NULL,

"email" varchar(254) NOT NULL,

"is\_staff" bool NOT NULL,

"is\_active" bool NOT NULL,

"date\_joined" datetime NOT NULL,

"first\_name" varchar(150) NOT NULL

)

* ***Read***

SELECT \* FROM auth\_user

* ***Update***

UPDATE `auth\_user`

SET `username` = ?

WHERE `rowid` IS ? AND `username` IS ?, ('lequochuyupdate', 2,'lequochuy')

* ***Delete***

DELETE FROM `auth\_user` WHERE (`rowid` IN (?)), [2]

1. ***Các truy vấn phức tạp (JOIN, GROUP BY)***

* ***JOIN (Liệt kê 5 tài liệu được yêu thích nhất)***

SELECT d.title AS TenTaiLieu, COUNT(f.id) AS SoLuotYeuThich

FROM app\_document AS d

JOIN app\_document\_favorited\_by AS f ON d.id = f.document\_id

GROUP BY d.title

ORDER BY SoLuotYeuThich DESC

LIMIT 5;

* ***GROUP BY (Tìm người dùng đã tải lên nhiều tài liệu nhất)***

SELECT u.username AS TenNguoiDung, COUNT(d.id) AS SoTaiLieuTaiLen

FROM auth\_user AS u

JOIN app\_document AS d ON u.id = d.user\_id

GROUP BY u.username

ORDER BY SoTaiLieuTaiLen DESC

LIMIT 1;

1. ***Backend***
2. ***Ngôn ngữ sử dụng***

* *Sử dụng ngôn ngữ lập trình Python*
* *Python là một ngôn ngữ lập trình bậc cao, mã nguồn mở và đa nền tảng, được thiết kế để dễ học và dễ sử dụng. Với cú pháp rõ ràng, ngắn gọn và dễ đọc, Python đã trở thành một trong những ngôn ngữ lập trình phổ biến nhất trên thế giới.*
* *Đặc điểm chính của Python*
* *Đa mô hình lập trình: Python hỗ trợ nhiều phong cách lập trình như lập trình hướng đối tượng, lập trình thủ tục, và lập trình chức năng.*
* *Kiểu động: Không cần khai báo kiểu dữ liệu, Python tự động xác định kiểu dựa trên giá trị.*
* *Thư viện phong phú: Python được mô tả là ngôn ngữ "bao gồm pin" vì có thư viện tiêu chuẩn toàn diện, hỗ trợ nhiều tác vụ như xử lý dữ liệu, phát triển web, và học máy.*
* *Đa nền tảng: Chạy được trên nhiều hệ điều hành như Windows, macOS, Linux.*
* *Ứng dụng của Python*
* *Phát triển web: Sử dụng các framework như Django, Flask.*
* *Khoa học dữ liệu và học máy: Phân tích dữ liệu, xây dựng mô hình AI/ML với các thư viện như Pandas, NumPy, TensorFlow.*
* *Tự động hóa: Viết script để tự động hóa các tác vụ lặp đi lặp lại.*
* *Phát triển phần mềm: Tạo ứng dụng desktop hoặc công cụ hỗ trợ*

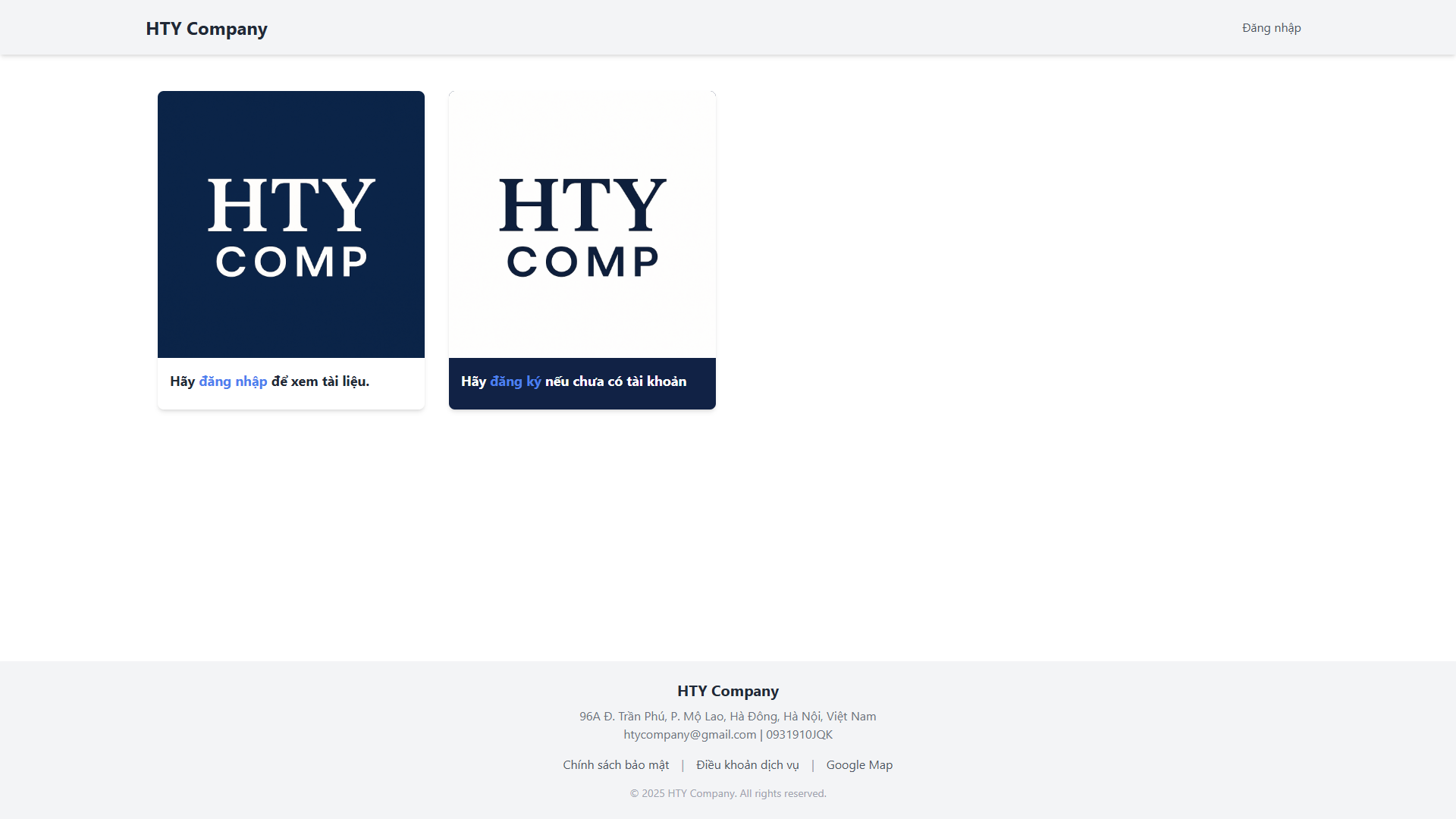
1. ***Framework***

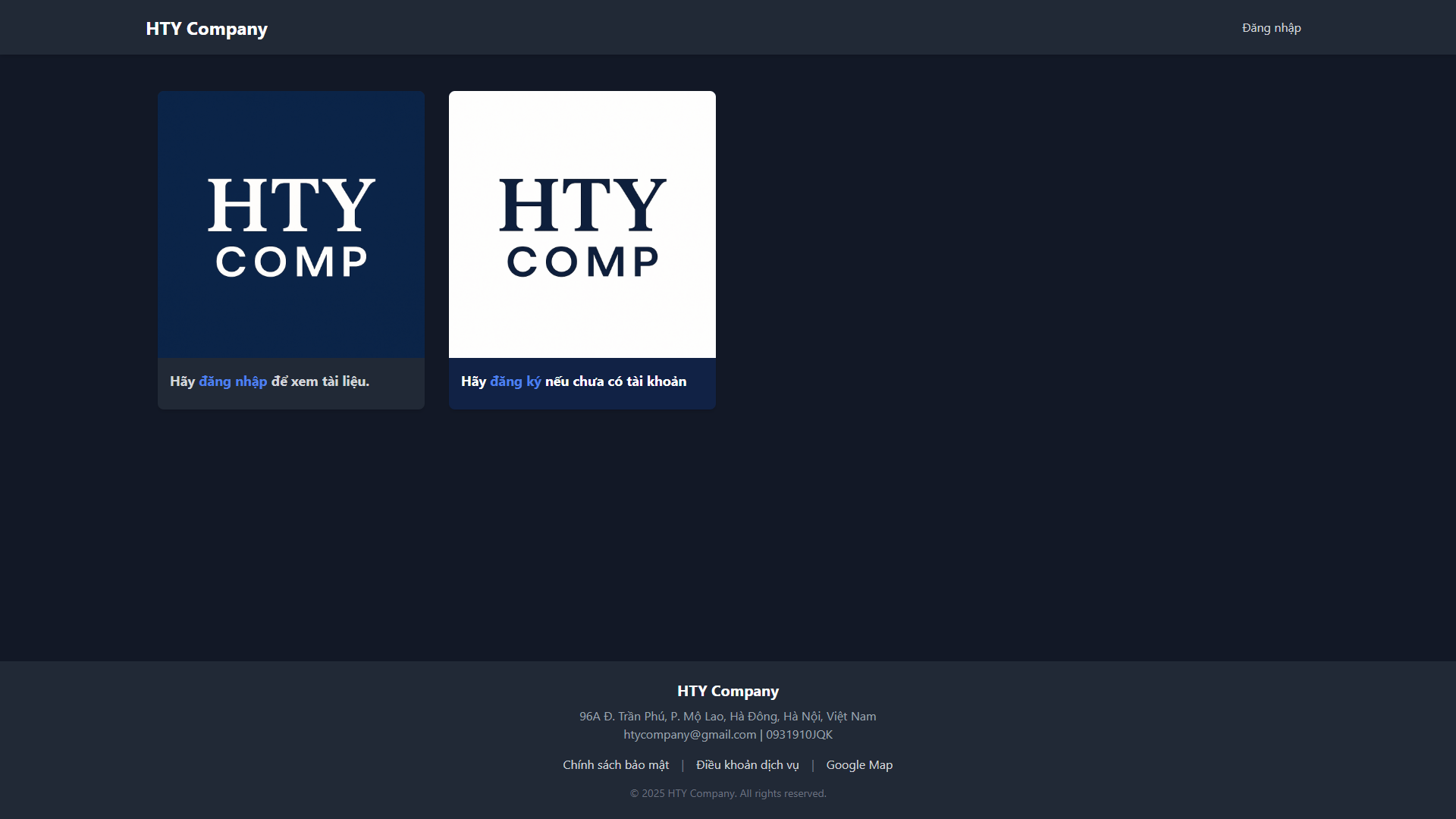
* *Sử dụng framework Django để phát triển web*
* *Django là phần mềm bạn có thể sử dụng để phát triển các ứng dụng web một cách nhanh chóng và hiệu quả. Hầu hết các ứng dụng web có một số chức năng phổ biến, như xác thực, truy xuất thông tin từ cơ sở dữ liệu và quản lý cookie.*

1. ***Frontend***
2. ***Công nghệ sử dụng***

* *Sử dụng HTML, CSS, Javascript để làm giao diện cho ứng dụng we****b***
* *Sử dụng thêm Framework Tailwind CSS*
* *Tailwind CSS là một utility-first CSS framework, nó cũng giống như Bootstrap, nó có những class built-in mà chúng ta có thể dùng. Tailwind CSS có nhiều các class bao gồm các thuộc tính CSS khác nhau và quan trọng, chúng ta có thể dễ dàng mở rộng tạo mới ra những class bằng chính những class của nó.*

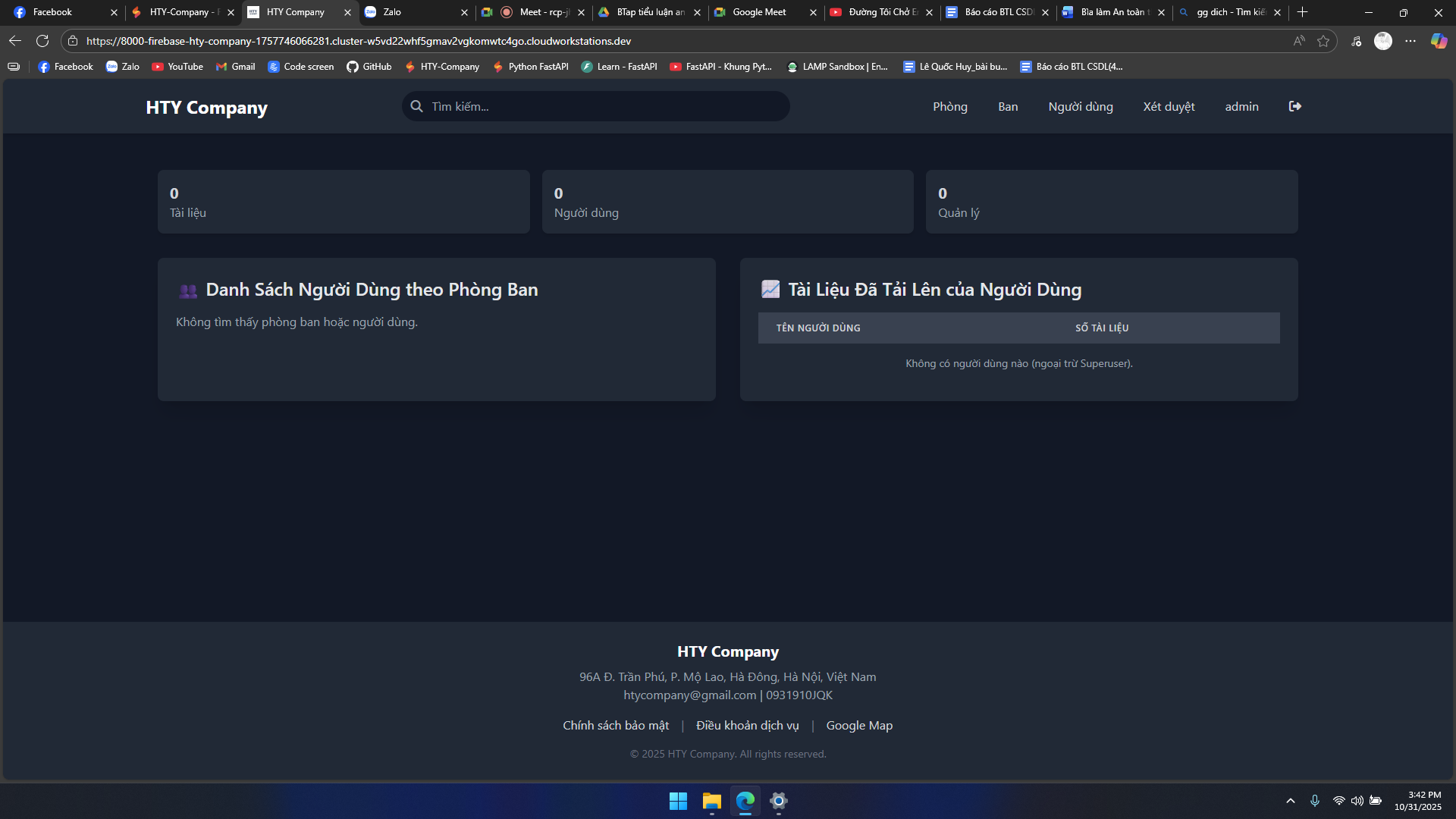
1. ***Giao diện***

* ***Do sử dụng theme thay đổi theo theme của thiết bị nên ứng dụng web sẽ có 2 giao diện là sáng và tối***
* ***Giao diện sáng: ***
* ***Giao diện tối:***

******

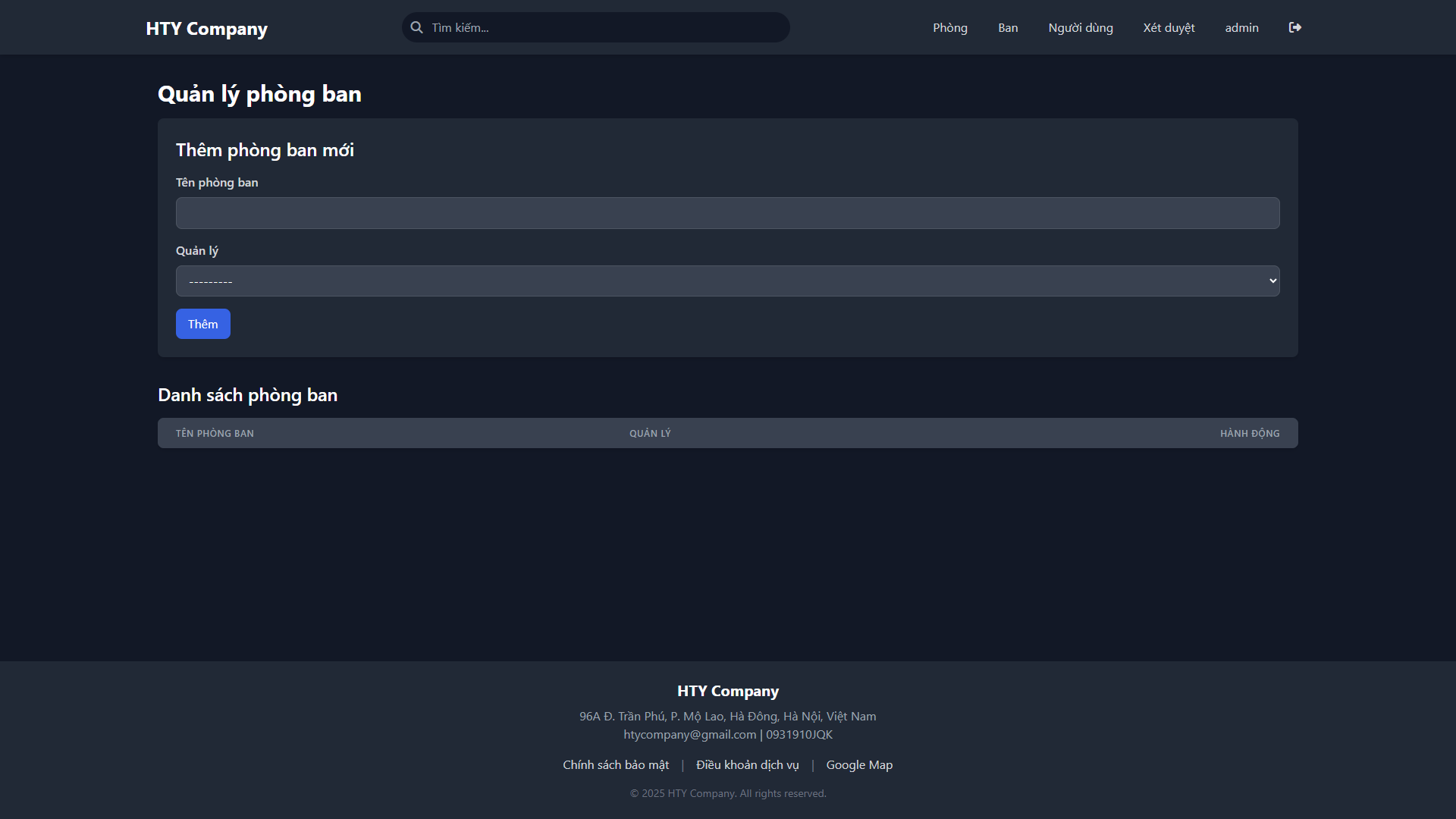
1. **Demo & Kết quả thử nghiệm**
2. **Trang dành cho quản trị viên**
3. **Trang home**

* Trang home của quản trị viên sẽ có các thành phần bao gồm: số tài liệu mà người dùng đã tải lên, số người dùng (bao gòm quản lý và nhân viên), số quản lý của từng phòng ban, danh sách người dùng theo phòng ban và số tài liệu đã tải lên của người dùng
* Sau khi một nhân viên dăng kí thành công thì sẽ được xét duyệt bởi quản trị viên hoặc quản lý. Sau khi đã được duyệt thì các thành phần ở trên sẽ hiển thị.



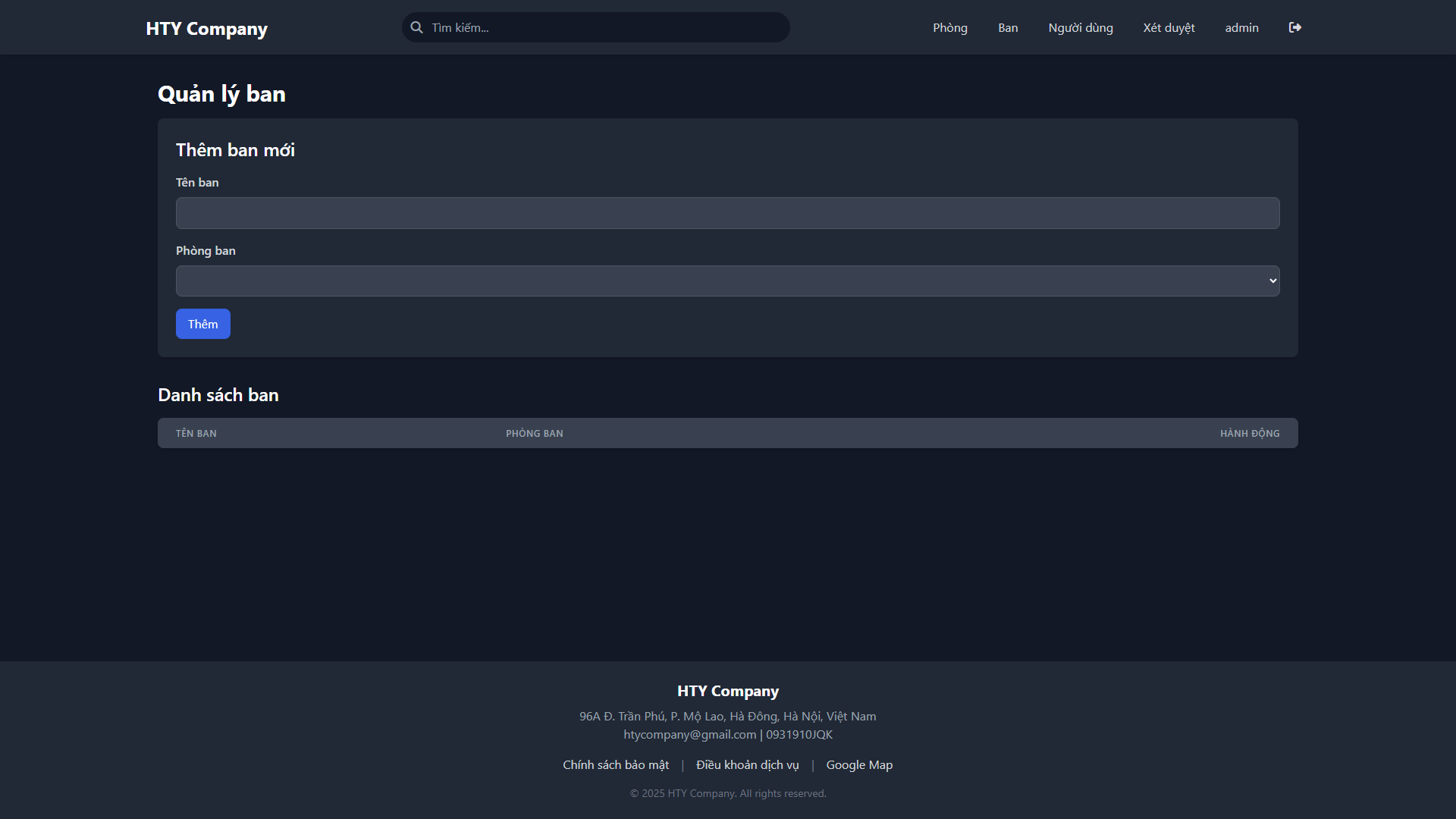
1. **Trang quản lý phòng**

* Bao gồm e-form để quản trị viên có thể thêm phòng của công ty
* Sau khi điền 1 phòng ban thì có thể điền thêm người quản lý, nếu không có người quản lý thì sẽ mặc định N/A, còn nếu có thì sẽ hiện tên người dùng cùng tên phòng ban
* Quản trị có thể chính sửa, xóa các phòng hoặc thay đổi quản lý



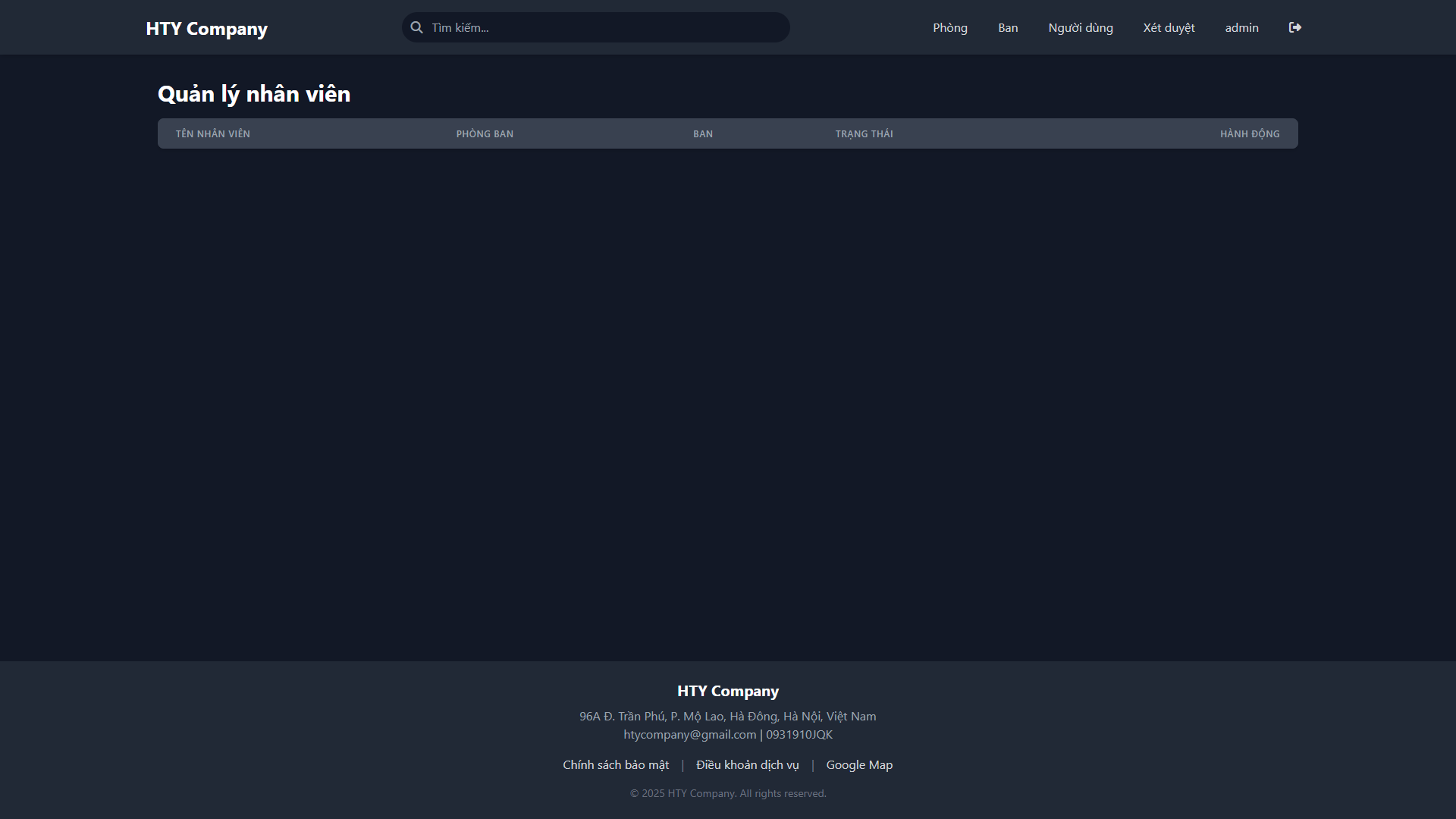
1. **Trang quản lý ban**

* Cũng tương tự như trang quản lý phòng nhưng phần phòng ban sẽ hiển thị các phòng đã thêm trước đó. Sau khi thêm thành công thì hiện thị danh sách các ban, phòng ban
* Quản trị viên có thể chỉnh sửa, xóa các ban



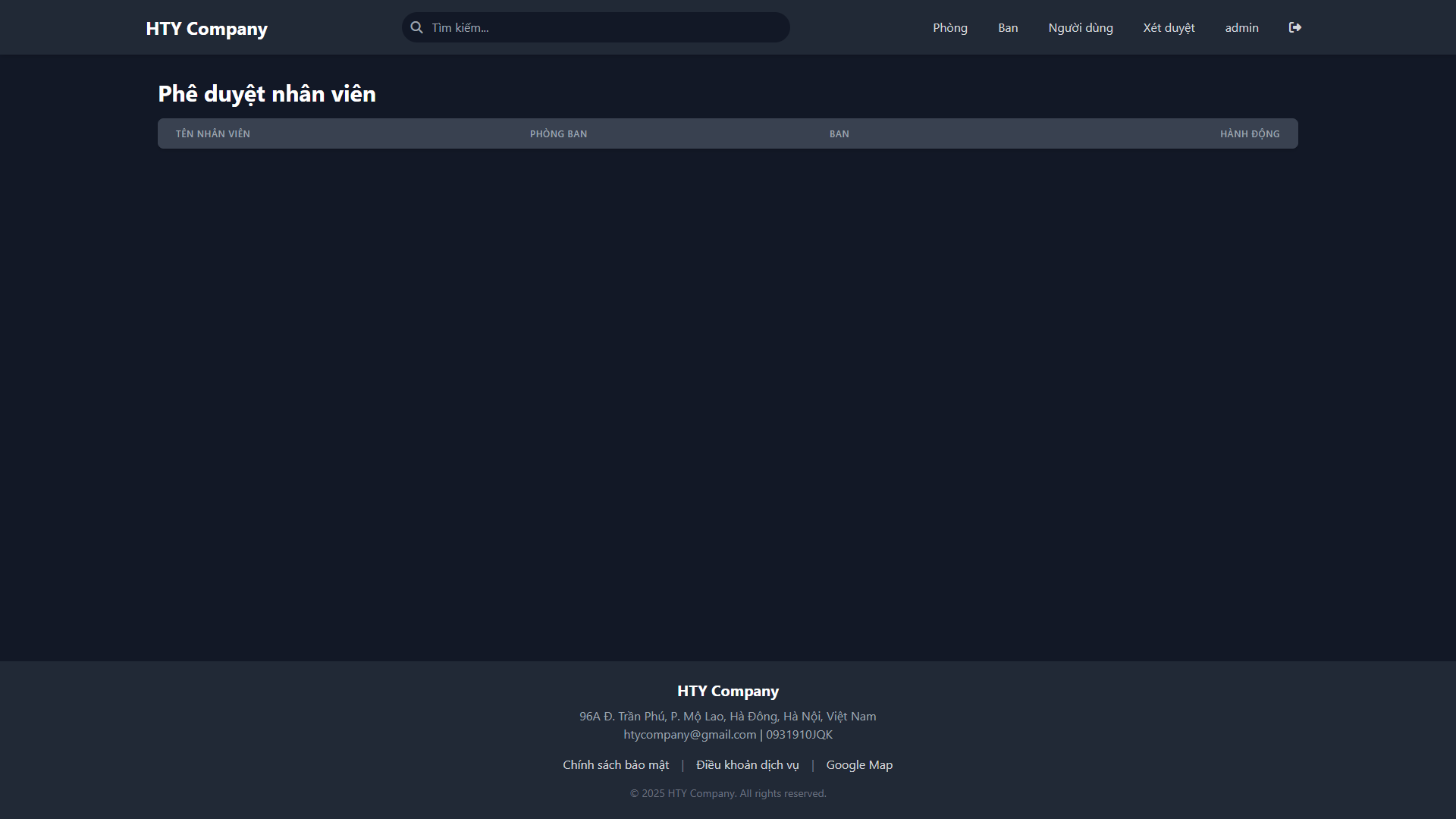
1. **Trang quản lý người dùng**

* Hiển trị danh sách các nhân viên thuộc phòng, ban và tên của nhân và trạng thái đã được xét duyệt
* Quản trị có thể chỉnh sửa, xóa, thay đỏi tên,....



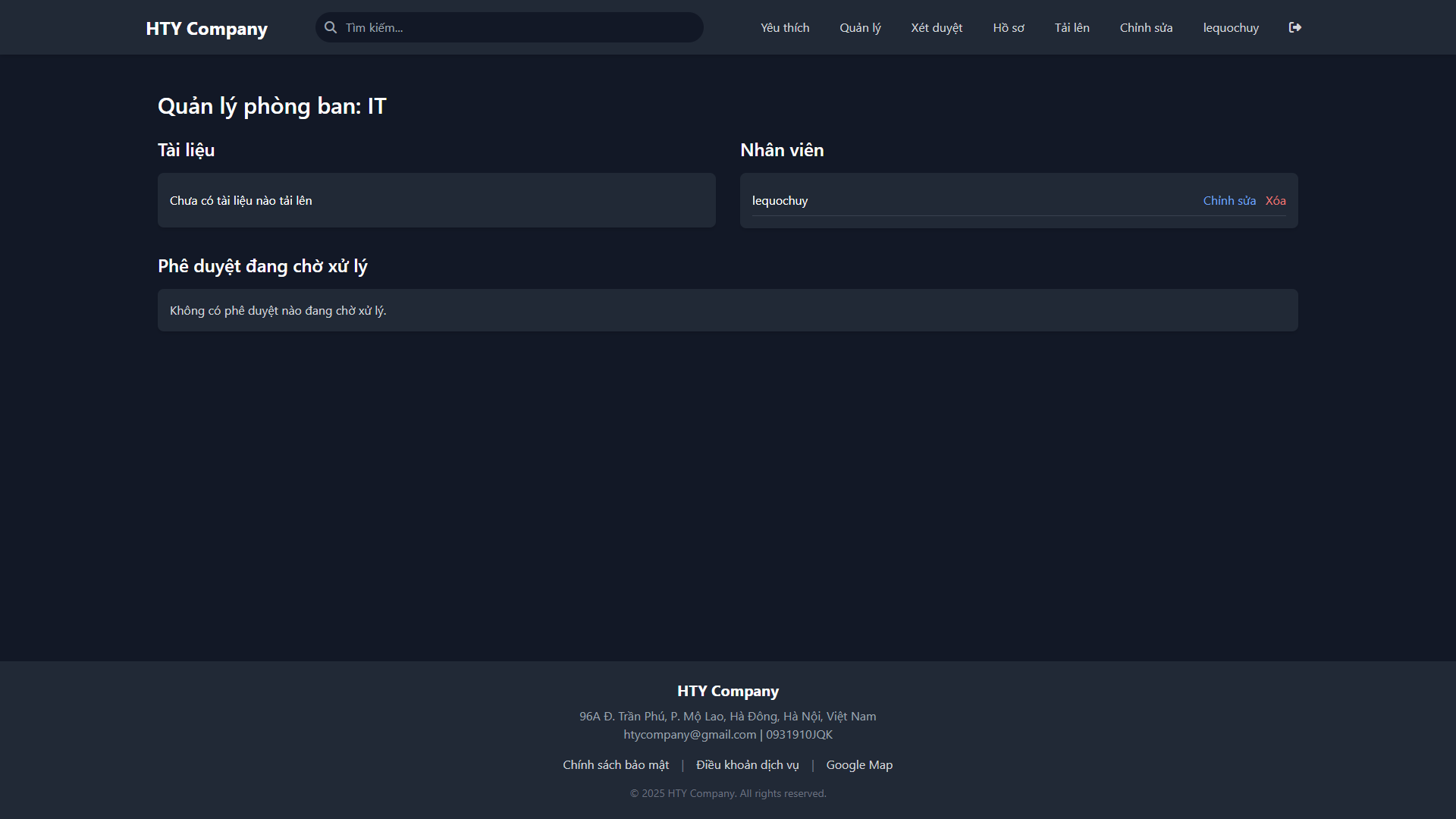
1. **Trang xét duyệt**

* Hiển thị các nhân viên sau khi đăng kí thành công sẽ được đưa vào danh sách xét duyệt
* Quản trị có thể từ chối nhân viên sau khi đăng kí nếu nhân viên đó không thuộc công ty



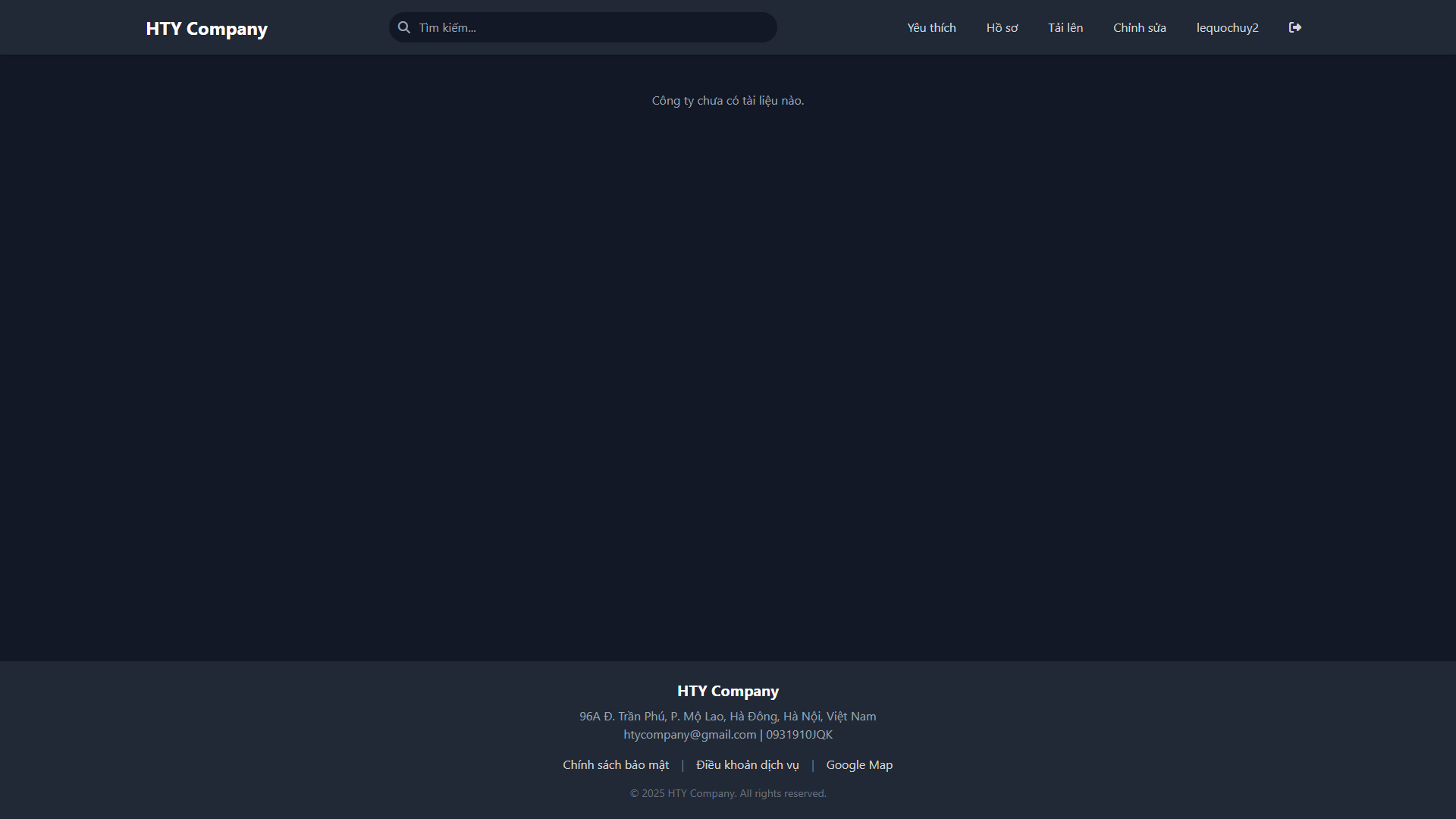
1. **Trang dành cho quản lý**

* Trang danh cho quản lý sẽ có giao diện gần gióng giao diện dành cho nhân viên nhưng sẽ có thêm các quyền khác như là quản lý nhân viên, quản lý các thư mục mà nhân viên tải lên và xét duyệt.
* Trang xét duyệt có giao diện tương tự như trang xét duyệt dành cho quản trị viên
* Trang quản lý sẽ có thành phần như tài liệu, nhân viên thuộc phòng ban đó và các nhân viên đang chờ xét duyệt
* Quản lý phòng ban có quyền chỉnh sửa, xóa các tài liệu và nhân viên thuộc phòng ban đó



1. **Trang dành cho nhân viên**

* Sau khi đăng kí và được xét duyệt thành công, trang dành cho nhân viên sẽ có các trang như sau:
* Trang home: Hiển thị các tài liệu mà các nhân viên cùng thuộc phòng ban tải lên
* Trang Yêu thích : Dùng để lưu các tài liệu
* Trang Hồ sơ: Dùng thay đổi mật khẩu của nhân viên đó
* Trang tải lên: Tải các tài liệu của nhân viên.
* Trang chỉnh sửa: Chỉnh sửa tài liệu và xóa tài liệu
* Ngoài ra, nhân viên có thêm các chức năng khác như là tìm kiếm tài liệu và tải xuống các tài liệu

****

1. **Đánh giá & Hạn chế**
2. ***Đánh giá***

* **Cấu trúc dự án tốt:** Ứng dụng được xây dựng trên nền tảng Django, một framework mạnh mẽ và có cấu trúc rõ ràng. Việc phân chia thành các ứng dụng (app), cùng với việc quản lý các tệp tĩnh (staticfiles), templates, và models tuân thủ theo các quy ước tốt nhất của Django, giúp cho việc bảo trì và phát triển sau này trở nên dễ dàng hơn.
* **Thiết kế Cơ sở dữ liệu (CSDL) có phân tích:**
* Bạn đã dành thời gian để phân tích và thiết kế CSDL, thể hiện qua các tệp Bảng database.dbml, Phân tích mức độ chuẩn hóa, và Rằng buộc toàn vẹn.
* Việc phân tích mức độ chuẩn hóa (đạt 1NF và có thể hơn) và xác định các ràng buộc toàn vẹn cho thấy sự chú trọng vào chất lượng và tính nhất quán của dữ liệu.
* **Toàn vẹn dữ liệu được đảm bảo:** Như trong tệp Rằng buộc toàn vẹn, ứng dụng đã áp dụng các quy tắc để đảm bảo dữ liệu hợp lệ ở nhiều cấp độ:
* **Toàn vẹn miền:** Sử dụng đúng kiểu dữ liệu (CharField, DateTimeField), và các ràng buộc như unique=True.
* **Toàn vẹn thực thể:** Mỗi bảng đều có khóa chính.
* **Toàn vẹn tham chiếu:** Sử dụng ForeignKey để liên kết các bảng, đảm bảo mối quan hệ giữa các dữ liệu.
* **Logic nghiệp vụ:** Có các quy tắc được định nghĩa ở tầng ứng dụng (trong views.py hoặc models.py) để xử lý các logic phức tạp.
* **Tập trung vào các tính năng truy vấn phức tạp:** Tệp 3 truy vấn phức tạp.sql cho thấy ứng dụng không chỉ dừng lại ở các chức năng CRUD (Tạo, Đọc, Cập nhật, Xóa) cơ bản mà còn có các tính năng phân tích dữ liệu, ví dụ như tìm tài liệu phổ biến nhất hay người dùng đóng góp nhiều nhất.
* **Giao diện người dùng hiện đại:** Việc sử dụng tailwind.config.js cho thấy bạn đang dùng Tailwind CSS, một framework CSS hiện đại giúp xây dựng giao diện người dùng một cách nhanh chóng và nhất quán.

1. ***Hạn chế***

* Hiệu năng và Khả năng mở rộng:
* Các truy vấn phức tạp như trong ví dụ có thể trở nên chậm khi lượng dữ liệu (tài liệu, người dùng) tăng lên. Cần xem xét việc tối ưu hóa truy vấn, sử dụng select\_related và prefetch\_related của Django ORM, hoặc đánh index cho các cột thường xuyên được lọc và sắp xếp.
* Chưa thấy các cơ chế caching (bộ nhớ đệm) phía server (ví dụ: Redis, Memcached) được triển khai. Việc cache các kết quả truy vấn tốn kém hoặc các trang ít thay đổi sẽ cải thiện đáng kể hiệu năng.
* Bảo mật:
* Django cung cấp sẵn các cơ chế bảo vệ chống lại các lỗ hổng phổ biến (XSS, CSRF, SQL Injection qua ORM), đó là một điểm cộng lớn.
* Tuy nhiên, nếu có sử dụng các truy vấn SQL thô (raw SQL), cần đảm bảo rằng chúng được параметри hóa đúng cách để tránh SQL Injection.
* Cần kiểm tra kỹ logic phân quyền trong views.py, đặc biệt là các chức năng quản lý, để đảm bảo người dùng thông thường không thể truy cập vào các tài nguyên của quản trị viên.
* Kiểm thử (Testing):
* Dự án có tệp app/tests.py, nhưng không rõ mức độ bao phủ của các bài kiểm thử (test coverage). Một bộ kiểm thử đầy đủ cho các models, views, và logic nghiệp vụ là rất quan trọng để đảm bảo ứng dụng hoạt động đúng như mong đợi và tránh phát sinh lỗi khi có sự thay đổi trong mã nguồn.
* Giao diện người dùng (Frontend):
* Hiện tại, dự án dường như đang sử dụng Django Templates để render giao diện. Đối với các trang có nhiều tương tác phức tạp phía người dùng, cách tiếp cận này có thể trở nên khó quản lý. Cân nhắc việc tích hợp một thư viện JavaScript (như Alpine.js để bổ sung cho Tailwind CSS) hoặc một framework frontend (như React, Vue) nếu ứng dụng yêu cầu độ tương tác cao.
* Quy trình triển khai (Deployment):
* Chưa có thông tin về quy trình CI/CD (Continuous Integration/Continuous Deployment). Việc thiết lập một quy trình tự động để kiểm thử và triển khai ứng dụng sẽ giúp giảm thiểu lỗi do con người và tăng tốc độ phát triển.

1. **Kết luận & hướng phát triển**
2. ***Kết luận***

* Ứng dụng web đang ở một trạng thái rất tốt với một nền tảng vững chắc.
* Điểm mạnh cốt lõi:
* Cấu trúc Backend rõ ràng: Việc sử dụng Django và tuân thủ các quy ước của nó giúp dự án có khả năng bảo trì và mở rộng cao.
* Thiết kế CSDL bài bản: Bạn đã đầu tư thời gian vào việc phân tích và chuẩn hóa dữ liệu (đạt 3NF), đồng thời xác định các ràng buộc toàn vẹn. Đây là một lợi thế cực kỳ lớn, giúp đảm bảo tính nhất quán và giảm thiểu các vấn đề về dữ liệu trong tương lai.
* Logic nghiệp vụ được xác định: Các truy vấn phức tạp đã được phác thảo, cho thấy bạn đã suy nghĩ sâu về các tính năng thực sự mang lại giá trị cho người dùng, thay vì chỉ dừng lại ở các chức năng cơ bản***.***

1. **Hướng phát triển**
2. ***Tối ưu hiệu năng***

Khi dữ liệu người dùng và tài liệu tăng lên, hiệu năng sẽ là yếu tố quyết định trải nghiệm người dùng.

* **Tối ưu hóa Truy vấn ORM:**
  + Thay vì viết SQL thô, hãy tận dụng tối đa Django ORM. Sử dụng select\_related (cho quan hệ một-nhiều) và prefetch\_related (cho quan hệ nhiều-nhiều hoặc quan hệ ngược) để giảm số lượng truy vấn đến CSDL. Ví dụ, khi lấy danh sách tài liệu và thông tin tác giả, hãy dùng Document.objects.select\_related('author').all().
* **Đánh Index cho CSDL:**
  + Trong models.py, thêm db\_index=True vào các trường thường xuyên được sử dụng để lọc, sắp xếp hoặc trong các mệnh đề JOIN (ví dụ: title, created\_at, các khóa ngoại).
* **Triển khai Caching:**
  + Sử dụng Redis hoặc Memcached để cache (lưu vào bộ nhớ đệm) các kết quả của những truy vấn tốn kém (như các truy vấn phức tạp bạn đã viết) hoặc các trang ít thay đổi. Django có hệ thống cache tích hợp rất mạnh mẽ.

1. ***Nâng cao trải nghiệm người dùng***

* Tăng tính tương tác với JavaScript:
* Tích hợp một thư viện JavaScript nhẹ như Alpine.js hoặc HTMX. Chúng kết hợp rất tốt với Tailwind CSS và Django templates, cho phép bạn thêm các tính năng động (như tìm kiếm real-time, sắp xếp bảng mà không cần tải lại trang) một cách dễ dàng mà không cần đến một framework frontend cồng kềnh.
* Xây dựng chức năng tìm kiếm mạnh mẽ:
* Thay vì dùng filter(title\_\_icontains=...), hãy cân nhắc tích hợp một công cụ tìm kiếm chuyên dụng như Elasticsearch hoặc sử dụng tính năng Full-Text Search có sẵn của PostgreSQL. Điều này sẽ cho kết quả tìm kiếm nhanh, chính xác và liên quan hơn rất nhiều.

1. ***Mở rộng tính năng***

* Hệ thống phiên bản (Versioning): Cho phép người dùng tải lên các phiên bản mới của một tài liệu hiện có. Hệ thống nên lưu lại các phiên bản cũ để có thể xem lại hoặc khôi phục khi cần.
* Hệ thống bình luận (Commenting): Cho phép người dùng thảo luận ngay bên dưới mỗi tài liệu.
* Thông báo (Notifications): Xây dựng hệ thống thông báo (ví dụ: sử dụng Django Channels cho thông báo real-time) để báo cho người dùng khi tài liệu của họ được yêu thích, bình luận, hoặc khi có tài liệu mới trong phòng ban của họ.

1. ***Đảm bảo chất lượng và quy trình chuyên nghiệp***

* Tăng độ bao phủ kiểm thử (Test Coverage): Viết thêm các unit test cho models, views và các logic nghiệp vụ phức tạp trong tests.py. Mục tiêu là để bạn tự tin rằng việc thêm tính năng mới không làm hỏng các tính năng cũ.
* Thiết lập CI/CD (Continuous Integration/Continuous Deployment):
* Sử dụng các công cụ như GitHub Actions hoặc GitLab CI để tự động chạy các bài kiểm thử mỗi khi bạn đẩy code mới lên.
* Cấu hình để tự động triển khai (deploy) ứng dụng lên server khi các bài kiểm thử thành công. Điều này giúp giảm thiểu sai sót của con người và tăng tốc độ phát hành sản phẩm.