**­­­­­­­­**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**ĐỒ ÁN NGHIÊN CỨU TỐT NGHIỆP II**

**Trang mạng xã hội chia sẻ kiến thức lập trình**

**Hoàng Thọ Tùng - 20184325**

tung.ht184325@sis.hust.edu.vn

**Ngành Công nghệ thông tin ICT**

**Chuyên ngành Công nghệ phần mềm**

|  |  |
| --- | --- |
| **Giảng viên hướng dẫn:** | Th.s Lê Đức Trung |
| **Bộ môn:** | Công nghệ phần mềm |
| **Viện:** | Công nghệ thông tin – Truyền thông |
| **HÀ NỘI, 03/2023** | |

# Giới thiệu đề tài

Hiện nay, thế giới đang trong thời đại công nghệ 4.0, với xu thế hội nhập, Việt Nam chúng ta đang rất cần nguồn nhân lực ở các ngành liên quan đến công nghệ cao. Khoảng 10 năm trở lại đây, ngành Công Nghệ Thông Tin (CNTT) ở Việt Nam đã trở thành một trong những ngành nghề hot nhất, với nhu cầu tuyển dụng rất nhiều, và mức lương bình quân là cao hơn so với mặt bằng chung, các bạn sinh viên mới ra trường ngành CNTT luôn được các nhà tuyển dụng săn đón. Bởi vậy mà ngày càng có nhiều các em học sinh cấp 3 lựa chọn CNTT là mục tiêu hướng tới ở bậc đại học, thậm chí là có nhiều những bạn sinh viên trái ngành cũng theo đuổi đam mê CNTT. Tuy nhiên CNTT là một ngành rộng, không dễ học, giai đoạn mới bắt đầu học luôn là một thách thức đối với các bạn sinh viên, và nhiều bạn đã cảm thấy chán nản, mất phương hướng, thậm chí bỏ ngành. Ở Việt Nam, CNTT vẫn còn là một ngành khá mới, số lượng tài liệu, sách, video hướng dẫn bằng tiếng Việt không nhiều. Lấy kinh nghiệm từ chính bản thân em, lúc mới bắt đầu học CNTT cũng rất vất vả, khả năng tiếng Anh chưa đủ tốt, nhiều thắc mắc không biết phải hỏi ai, chỉ có thể mày mò ở các blog, diễn đàn, hội nhóm CNTT Việt Nam. Em đã học được rất nhiều điều từ những người đi trước đó, và em cảm thấy rất biết ơn họ đã sẵn sàng bỏ ra thời gian, công sức giải đáp thắc mắc, chia sẻ kiến thức của mình để xây dựng cộng đồng CNTT Việt Nam.

Được truyền cảm hứng từ anh Loda - loda.me, anh Xuân Thu – xuanthu.lab cùng với rất nhiều những blog IT Việt Nam khác, em ấp ủ ý tưởng được góp một phần công sức vào việc phát triển cộng đồng, giúp đỡ cho các mầm non IT tiếp cận với kiến thức lập trình một cách có hệ thống, dễ dàng, thuần Việt và miễn phí. Nhận thấy việc Việt Nam có rất nhiều những blog kiến thức hay và có hệ thống, nhưng lại đơn lẻ và không có sự tương tác; hoặc những diễn đàn có tương tác, nhưng kiến thức lại lẻ tẻ, dàn trải, hơi nâng cao và không phù hợp với các bạn mới, em có ý định xây dựng một website (toby.me) dưới dạng trang mạng xã hội, đối tượng hướng đến là các bạn sinh viên Việt Nam, cho phép mọi người cùng chia sẻ, trao đổi với nhau, và kiến thức sẽ được hệ thống theo từng chủ để - là những chủ để cơ bản của ngành CNTT.

# Introduction

Currently, the world is in the age of 4.0 technology, with the trend of integration, Vietnam is in great need of human resources in the industries related to high technology. In the last 10 years or so, Information Technology (IT) in Vietnam has become one of the hottest occupations, with a lot of recruitment needs, and the average salary is higher than the general ground, new graduates are always welcomed by employers. As a result, more and more high school students choose IT as their goal for college, and even many non-IT college students pursue their passion for IT. However, IT is a broad discipline, not easy to learn, the beginning of learning has always been a challenge for students, and many of them have felt depressed, disoriented, even dropped out. In Vietnam, IT is still a fairly new industry, the number of materials, books, videos in Vietnamese is not much. Taking experience from myself, at the beginning of learning IT was also very hard, my English skills were not good enough, many questions did not find who to ask, could only search in Vietnamese speaking blogs, forums, IT groups. I have learned a lot from previous people, and I feel very grateful that they are willing to spend time and effort to answer questions and share their knowledge to build a strong Vietnamese IT community.

Inspired by Mr. Loda - loda.me, Mr. Xuan Thu – xuanthu.lab along with many other Vietnamese IT blogs, I inculcated the idea of contributing a great deal to community development, helping IT cloves to access programming knowledge systematically, easily, purely Vietnamese and free of charge. Realizing that Vietnam has a lot of good and systematic knowledge blogs, but not linked with others and without interaction; or interactive forums, but the knowledge is odd, spread, slightly advanced and not suitable for newbies, I intend to build a website (toby.me) in the form of a social network, the target audience is Vietnamese students, allowing people to share and exchange with each other, and the knowledge will be systematized according to each topic - are the basic topics of IT.

# Kết quả đạt được

Đề tài Đồ án nghiên cứu tốt nghiệp II của em lựa chọn là với định hướng phát triển thành đồ án tốt nghiệp, cho nên em mong muốn được tự mình thực hiện tất cả các bước trong quy trình phát triển phần mềm, bao gồm: phân tích yêu cầu, thiết kế giao diện người dùng, thiết kế hệ thống, cơ sở dữ liệu, thực hiện viết code, deploy hệ thống lên các máy chủ remote, và vận hành.

Kết quả đạt được sau môn Đồ án nghiên cứu tốt nghiệp II: nắm vững kiến thức về mô hình Client-Server, Restful API, Java SpringBoot, ReactJS Typescript; hoàn thiện website với các chức năng cơ bản.

Table of Contents

[Giới thiệu đề tài 2](#_Toc129616694)

[Introduction 3](#_Toc129616695)

[Kết quả đạt được 4](#_Toc129616696)

[Chương 1: Phân tích yêu cầu 7](#_Toc129616697)

[1.1 Tổng quan chức năng 7](#_Toc129616698)

[1.1.1 Biểu đồ use case tổng quan 7](#_Toc129616699)

[1.1.2 Biểu đồ use case phân rã 8](#_Toc129616700)

[1.1.3 Quy trình nghiệp vụ 9](#_Toc129616701)

[1.2 Yêu cầu phi chức năng 10](#_Toc129616702)

[Chương 2: Công nghệ sử dụng 11](#_Toc129616703)

[2.1 Restful API 11](#_Toc129616704)

[2.2 Server 12](#_Toc129616705)

[2.2.1 Java và Spring Boot 12](#_Toc129616706)

[2.2.2 MySQL 12](#_Toc129616707)

[2.3 Client 13](#_Toc129616708)

[2.3.1 ReactJS 13](#_Toc129616709)

[2.3.2 Typescript 13](#_Toc129616710)

[2.4 Xác thực 14](#_Toc129616711)

[2.4.1 JWT 14](#_Toc129616712)

[2.4.2 Oauth2 15](#_Toc129616713)

[Chương 3: Phát triển và triển khai ứng dụng 17](#_Toc129616714)

[3.1 Thiết kế tổng quan 17](#_Toc129616715)

[3.2 Thiết kế gói 18](#_Toc129616716)

[3.2.1 Domain 18](#_Toc129616717)

[3.3 Thiết kế chi tiết 19](#_Toc129616718)

[3.3.1 Thiết kế giao diện 19](#_Toc129616719)

[3.3.2 Thiết kế lớp 22](#_Toc129616720)

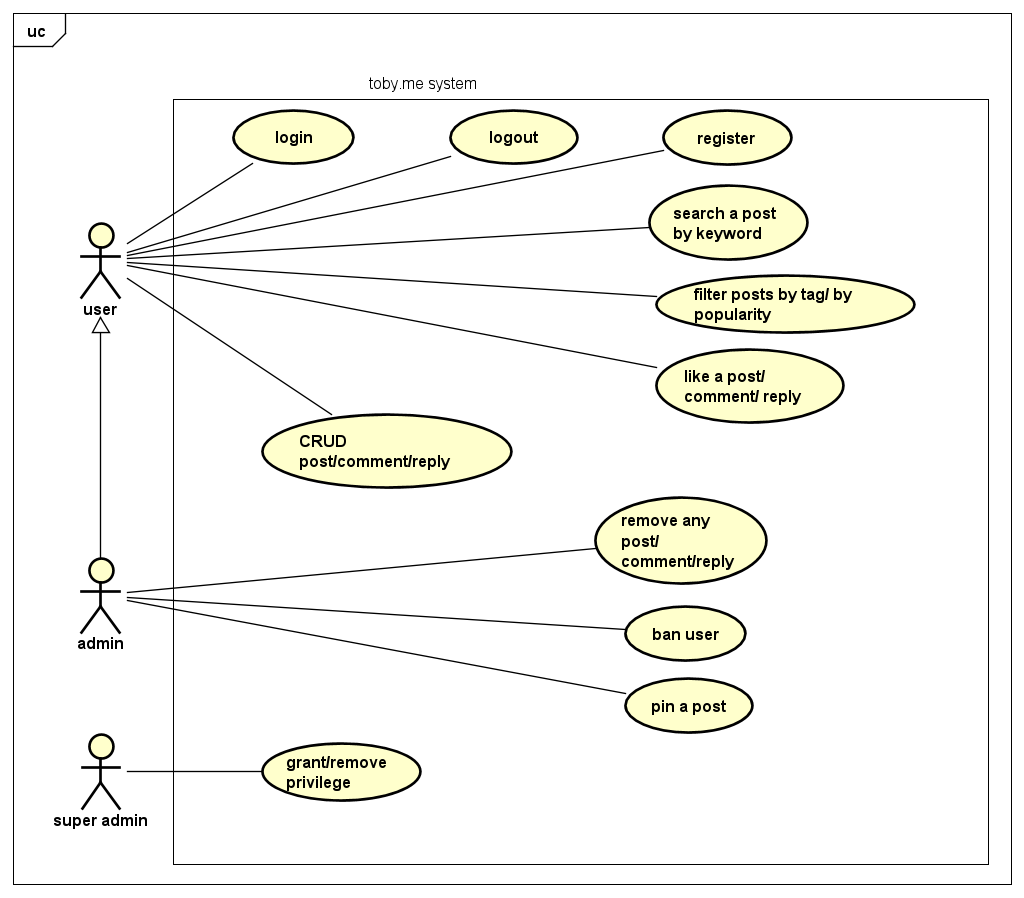
[3.3.3 Thiết kế cơ sở dữ liệu 22](#_Toc129616721)

# Phân tích yêu cầu

## Tổng quan chức năng

Hệ thống có 3 tác nhân chính: người dùng (user), quản trị viên (admin) và super admin (người quản trị hệ thống).

### Biểu đồ use case tổng quan



Trong hệ thống, người dùng (user) có thể làm những việc sau: đăng kí, đăng nhập (có thể đăng nhập bằng Facebook, Google), đăng xuất, lựa chọn các chủ đề, xem/tìm kiếm theo chủ để - từ khóa, lọc/thích/tạo/sửa bài viết, xem/thích/trả lời/sửa một bình luận, xem/thích/sửa một câu trả lời của bình luận. Người dùng không bắt buộc cần phải đăng kí, đăng nhập vào website mà vẫn có thể xem các nội dung. Người dùng cũng chính là những người góp sức xây dựng nội dung cho website.

Quản trị viên (admin) chính là những người dùng bình thường, nhưng được cấp các quyền để ghim bài viết (giống như đăng một bài thông báo mọi người sẽ thấy đầu tiên khi vào website), gỡ bỏ một bài viết, bình luận, trả lời không phù hợp. Quản trị viên là những người có trách nhiệm duy trì website vận hành một cách ngăn nắp, sạch sẽ, ngăn chặn những kẻ quấy phá, đồng thời nhiệt tình xây dựng cộng đồng.

Super admin là người quản trị hệ thống, có thể can thiệp trực tiếp vào code, cơ sở dữ liệu, cấp quyền/xóa quyền quản trị viên cho những người dùng phù hợp/không phù hợp.

### Biểu đồ use case phân rã

Người dùng không cần đăng nhập vào website vẫn có thể xem/tìm kiếm/lọc các nội dung trên website, tuy nhiên sẽ không thể tương tác với người dùng khác trên website. Dưới đây là những use case không cần điều kiện đăng nhập:

Diagram

Description automatically generated

Để có thể tương tác với những người dùng khác, người dùng cần phải đăng nhập vào hệ thống. Dưới đây là những use case cần điều kiện đăng nhập:

Diagram

Description automatically generated

### Quy trình nghiệp vụ

Các use case ở đây tuy nhiều nhưng đều là những use case đơn giản, chỉ gồm 1 bước, nên em minh họa 1 use case của hệ thống. Biểu đồ hoạt động của use case: “write a post”:

Diagram

Description automatically generated

## Yêu cầu phi chức năng

Hệ thống có các tính năng hoạt động theo thời gian thực nên cần có khả năng vận hành ổn định, dữ liệu được toàn vẹn và được backup đầy đủ. Ngoài ra, bởi đây là một website dưới dạng mạng xã hội, nhiều người dùng, nên hệ thống cần được thiết kế đảm bảo khả năng cân bằng tải, dễ mở rộng, để dự phòng trường hợp số người truy cập đông gây quá tải cho server.

# Công nghệ sử dụng

Với đề tài này, em lựa chọn mô hình Client-Server. Client và Server sẽ giao tiếp với nhau thông qua các JSON Restful API.

## Restful API

RESTful API là một tiêu chuẩn dùng trong việc thiết kế API cho các ứng dụng web (thiết kế Web services) để tiện cho việc quản lý các resource. Nó chú trọng vào tài nguyên hệ thống (tệp văn bản, ảnh, âm thanh, video, hoặc dữ liệu động…), bao gồm các trạng thái tài nguyên được định dạng và được truyền tải qua HTTP.

API (Application Programming Interface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác. API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như JSON hay XML.

REST (REpresentational State\*\* T\*\*ransfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau.

Chức năng quan trọng nhất của REST là quy định cách sử dụng các HTTP method (như GET, POST, PUT, DELETE…) và cách định dạng các URL cho ứng dụng web để quản các resource. RESTful không quy định logic code ứng dụng và không giới hạn bởi ngôn ngữ lập trình ứng dụng, bất kỳ ngôn ngữ hoặc framework nào cũng có thể sử dụng để thiết kế một RESTful API.

REST hoạt động chủ yếu dựa vào giao thức HTTP. Các hoạt động cơ bản nêu trên sẽ sử dụng những phương thức HTTP riêng.

* GET (SELECT): Trả về một Resource hoặc một danh sách Resource.
* POST (CREATE): Tạo mới một Resource.
* PUT (UPDATE): Cập nhật thông tin cho Resource.
* DELETE (DELETE): Xoá một Resource.

Những phương thức hay hoạt động này thường được gọi là CRUD tương ứng với Create, Read, Update, Delete – Tạo, Đọc, Sửa, Xóa.

Hiện tại đa số lập trình viên viết RESTful API giờ đây đều chọn JSON là format chính thức nhưng cũng có nhiều người chọn XML làm format, tùy vào mục đích và nhu cầu sử dụng.

## Server

### Java và Spring Boot

Java là một ngôn ngữ lập trình hướng đối tượng (OOP). Lập trình hướng đối tượng, với khả năng đưa mọi thứ về các lớp, các đối tượng, đặc trưng, phương thức, để chúng tương tác với nhau như những sự vật ngoài đời thực. Hệ thống lập trình hướng đối tượng đã rất phát triển, với các nguyên lý SOLID, rất nhiều loại design partern giúp cho OOP dễ xây dựng, dễ bảo trì, mở rộng. Code OOP có vẻ dài dòng và cồng kềnh nhưng đọc rất dễ hiểu, dễ hình dung. Java là một ngôn ngữ rất chuẩn mực về hướng đối tượng, nó kế thừa tất cả những ưu điểm trên, ngoài ra, hiệu năng của Java rất tốt và ổn định. Chính vì lý do đó mà Java được ưu tiên để dựng các hệ thống lớn và phức tạp như ngân hàng, chính phủ điện tử. Ở đề tài này, em lựa chọn Java làm ngôn ngữ phía server.

Spring Boot là một dự án phát triển bởi Java trong hệ sinh thái Spring framework. Nó giúp cho các lập trình viên chúng ta đơn giản hóa quá trình lập trình một ứng dụng với Spring, chỉ tập trung vào việc phát triển tác vụ cho ứng dụng. Sử dụng các trình quản lý gói/thư viện Maven/Gradle, cùng với Tomcat Server được tích hợp sẵn, khiến cho framework này trở thành lựa chọn tốt nhất cho việc xây dựng các ứng dụng chạy trên nền web.

### MySQL

MySQL là một hệ thống quản trị cơ sở dữ liệu phổ biến được sử dụng bởi hàng triệu website và ứng dụng trên toàn thế giới. MySQL được phát triển bởi công ty MySQL AB (hiện thuộc sở hữu của Oracle Corporation) vào những năm 1990. MySQL được thiết kế để cung cấp hiệu suất cao, độ tin cậy và tính linh hoạt cho các ứng dụng web, cơ sở dữ liệu và hệ thống quản trị dữ liệu. Trong đoạn văn này, chúng ta sẽ tìm hiểu về ưu điểm và nhược điểm của MySQL.

Ưu điểm của MySQL:

* Miễn phí và mã nguồn mở: MySQL là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở và được phát hành miễn phí dưới giấy phép GPL. Điều này có nghĩa là bất kỳ ai cũng có thể sử dụng, sửa đổi và phân phối mã nguồn của MySQL mà không cần phải trả bất kỳ khoản phí nào.
* Độ tin cậy cao: MySQL được thiết kế để cung cấp độ tin cậy cao, bảo vệ dữ liệu của bạn khỏi sự cố hệ thống và sự cố phần cứng. Hệ thống này cũng cung cấp tính năng khôi phục dữ liệu để giúp khôi phục dữ liệu nhanh chóng trong trường hợp có sự cố.
* Hiệu suất cao: MySQL là một hệ thống quản trị cơ sở dữ liệu tối ưu hóa hiệu suất. Nó có thể xử lý hàng nghìn truy vấn mỗi giây và được sử dụng rộng rãi trong các ứng dụng web yêu cầu hiệu suất cao.
* Khả năng mở rộng: MySQL được thiết kế để có thể mở rộng dễ dàng. Bạn có thể thêm nhiều máy chủ cơ sở dữ liệu để tăng hiệu suất và đảm bảo khả năng xử lý các tải trọng lớn.

## Client

### ReactJS

ReactJS là một thư viện UI phát triển tại Facebook để hỗ trợ việc xây dựng những thành phần (components) UI có tính tương tác cao, có trạng thái và có thể sử dụng lại được. ReactJS được sử dụng tại Facebook trong production, và www.instagram.com được viết hoàn toàn trên ReactJS.

Một trong những điểm hấp dẫn của ReactJS là thư viện này không chỉ hoạt động trên phía client, mà còn được render trên server và có thể kết nối với nhau. ReactJS so sánh sự thay đổi giữa các giá trị của lần render này với lần render trước và cập nhật ít thay đổi nhất trên DOM. ReactJS có rất nhiều những gói hỗ trợ, giúp việc lập trình giao diện web, xử lý request với server trở nên dễ dàng hơn. So với các framework FrontEnd khác như AngularJS, VueJS thì ReactJS nhẹ hơn, ít các thành phần hơn và dễ dàng tiếp cận.

### Typescript

TypeScript là một ngôn ngữ lập trình mã nguồn mở được phát triển bởi Microsoft. Nó là một phiên bản cải tiến của JavaScript, có thêm tính năng kiểu dữ liệu tĩnh. TypeScript được thiết kế để giúp các nhà phát triển tạo ra các ứng dụng lớn và phức tạp một cách dễ dàng hơn, đồng thời giảm thiểu các lỗi cú pháp và logic. Dưới đây là những ưu điểm và nhược điểm của TypeScript.

Ưu điểm:

* Kiểu dữ liệu tĩnh: TypeScript cho phép khai báo kiểu dữ liệu tĩnh cho biến, hàm và đối tượng. Điều này giúp kiểm tra các lỗi cú pháp và logic trong quá trình phát triển ứng dụng, giúp tăng tính ổn định và hiệu suất của ứng dụng.
* Mã nguồn rõ ràng và dễ hiểu: TypeScript sử dụng cú pháp gần giống với JavaScript nên mã nguồn của nó rất dễ hiểu và dễ bảo trì.
* Hỗ trợ các tính năng mới của ECMAScript: TypeScript hỗ trợ tất cả các tính năng mới của ECMAScript như async/await, destructuring, template literals,... giúp cho các nhà phát triển có thể sử dụng những tính năng mới này một cách dễ dàng hơn.

Nhược điểm:

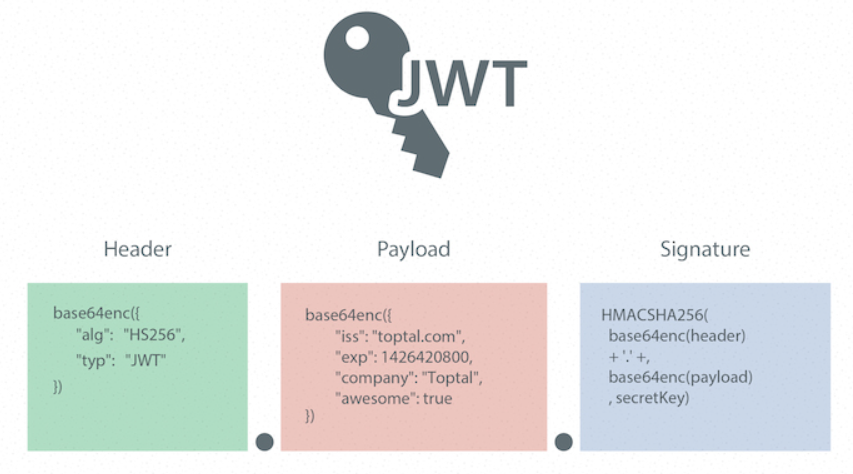
* Học hỏi khá phức tạp: Với các lập trình viên chưa có kinh nghiệm về kiểu dữ liệu tĩnh, TypeScript có thể khá phức tạp và tốn thời gian để học hỏi.
* Không thể chạy trực tiếp trên trình duyệt: TypeScript phải được biên dịch thành JavaScript trước khi chạy được trên trình duyệt. Điều này có thể tốn thêm thời gian và công sức để phát triển ứng dụng.

Tóm lại, TypeScript là một ngôn ngữ lập trình rất hữu ích trong việc phát triển các ứng dụng lớn và phức tạp. Tuy nhiên, nó cũng có một số nhược điểm nhất định mà các nhà phát triển cần lưu ý khi sử dụng.

## Xác thực

### JWT

JWT là một phương tiện đại diện cho các yêu cầu chuyển giao giữa hai bên Client – Server , các thông tin trong chuỗi JWT được định dạng bằng JSON . Trong đó chuỗi Token phải có 3 phần là header , phần payload và phần signature được ngăn bằng dấu “.”



JSON Web Token là 1 cách thức khá hay để truyền thông tin an toàn giữa các thành viên với nhau, nhờ vào phần signature của nó. Phía người nhận có thể biết được người gửi là ai thông qua phần signature. Và chữ ký được tạo ra bằng việc kết hợp cả phần header, payload lại nên thông qua đó ta có thể xác nhận được chữ ký có bị giả mạo hay không.

Ở đề tài của em, em sử dụng JWT cho quá trình xác thực, phân quyền. Khi người dùng đã đăng nhập vào hệ thống, hệ thống sẽ cấp cho họ một chuỗi JWT, và những request tiếp theo từ phía người dùng gửi lên server sẽ cần phải chứa thêm mã JWT, server sẽ trích xuất thông tin từ chuỗi JWT này để xác minh thông tin người dùng. Điều này cho phép người dùng được cấp quyền truy cập vào các url, service, và resource mà mã Token đó cho phép. Phương pháp này không bị ảnh hưởng bởi Cross-Origin Resource Sharing (CORS) do nó không sử dụng cookie.

### Oauth2

OAuth2 đã thay thế OAuth 1.0 vào năm 2012 và hiện là tiêu chuẩn ngành trên thực tế để ủy quyền trực tuyến. OAuth 2.0 cho phép các ứng dụng chia sẻ tài nguyên với nhau mà không cần xác thực qua username và password như cách truyền thống. Điều này giúp hạn chế những phiền toái khi phải đăng nhập username và password ở quá nhiều nơi, hoặc quá nhiều tài khoản mà không nhớ hết.

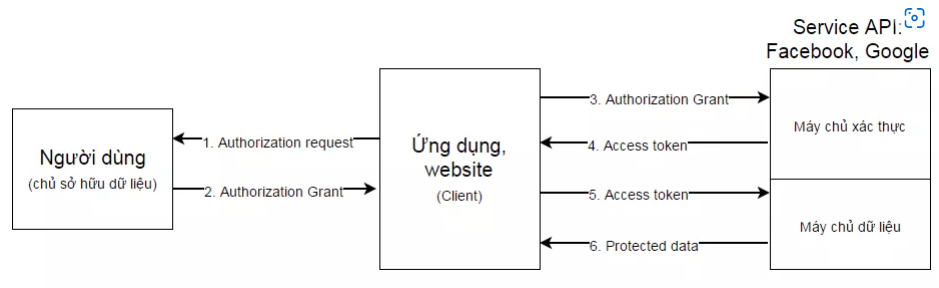
Mặc dù web là nền tảng chính cho OAuth 2, thông số kỹ thuật cũng mô tả cách xử lý loại quyền truy cập được ủy quyền này đối với các loại ứng dụng khách khác (ứng dụng dựa trên trình duyệt, ứng dụng web phía máy chủ, ứng dụng gốc/di động, thiết bị được kết nối,…

OAuth xác định bao gồm 4 thành phần chính:

* Resource Owner (Chủ sở hữu tài nguyên): Chủ sở hữu tài nguyên là người dùng có khả năng cấp quyền truy cập vào tài khoản của họ. Ví dụ: Bạn muốn chia sẻ các thông tin như email, ngày sinh, giới tính, địa chỉ lên một trang web nào đó có thể đăng nhập bằng Facebook. Những thông tin này chính là tài nguyên cần chia sẻ và bạn chính là Resource Owner.
* Client: Client là ứng dụng muốn truy cập vào tài khoản của người dùng. Trước khi có thể làm như vậy, client phải được người dùng ủy quyền và ủy quyền phải được xác thực bởi API.
* Resource Server: Lưu trữ các tài khoản người dùng được bảo vệ.
* Authorization Server: Authorization Server xác minh danh tính của người dùng sau đó cấp mã access token thông báo truy cập cho ứng dụng. Đôi khi Authorization Server cũng chính là resource server.

Token là một đoạn mã được sinh ra ngẫu nhiên bởi Authorization Server khi có yêu cầu được gửi đến từ Client. Có 2 loại token: Access token và Refresh token.

Cách hoạt động của Oauth2:

* Client yêu cầu ủy quyền (Authorization Request) từ Authorization Server, cung cấp client ID và secret làm nhận dạng. Đồng thời, cũng cung cấp các scope và endpoint URI (redirect URI) để gửi Access Token hoặc Authorization Code.
* Authorization Server xác thực client và xác minh rằng scope yêu cầu được cho phép.
* Resource owner tương tác với Authorization Serve để cấp quyền truy cập.
* Authorization Server chuyển hướng trở lại client bằng Authorization Code hoặc Access Token, tùy thuộc vào loại. Refresh Token cũng có thể được return.
* Với Access Token, client yêu cầu quyền truy cập vào tài nguyên từ máy Resource server.

Ở đề tài của em, là một website hướng đến đối tượng trẻ, những người có sẵn các loại tài khoản mạng xã hội, họ có rất nhiều các loại tài khoản nên khá ngại đăng kí những tài khoản mới. Bởi vậy, website có thêm 2 lựa chọn đăng nhập bằng tài khoản Google, hoặc Facebook.

# Phát triển và triển khai ứng dụng

## Thiết kế tổng quan

Hệ thống chia làm 3 package chính: domain, exception, security

Diagram

Description automatically generated

* Domain: xử lý các logic chính của hệ thống.
* Exception: xử lý các lỗi trong quá trình runtime.
* Security: phụ trách xác thực, phân quyền cho hệ thống.

## Thiết kế gói

### Domain

Diagram

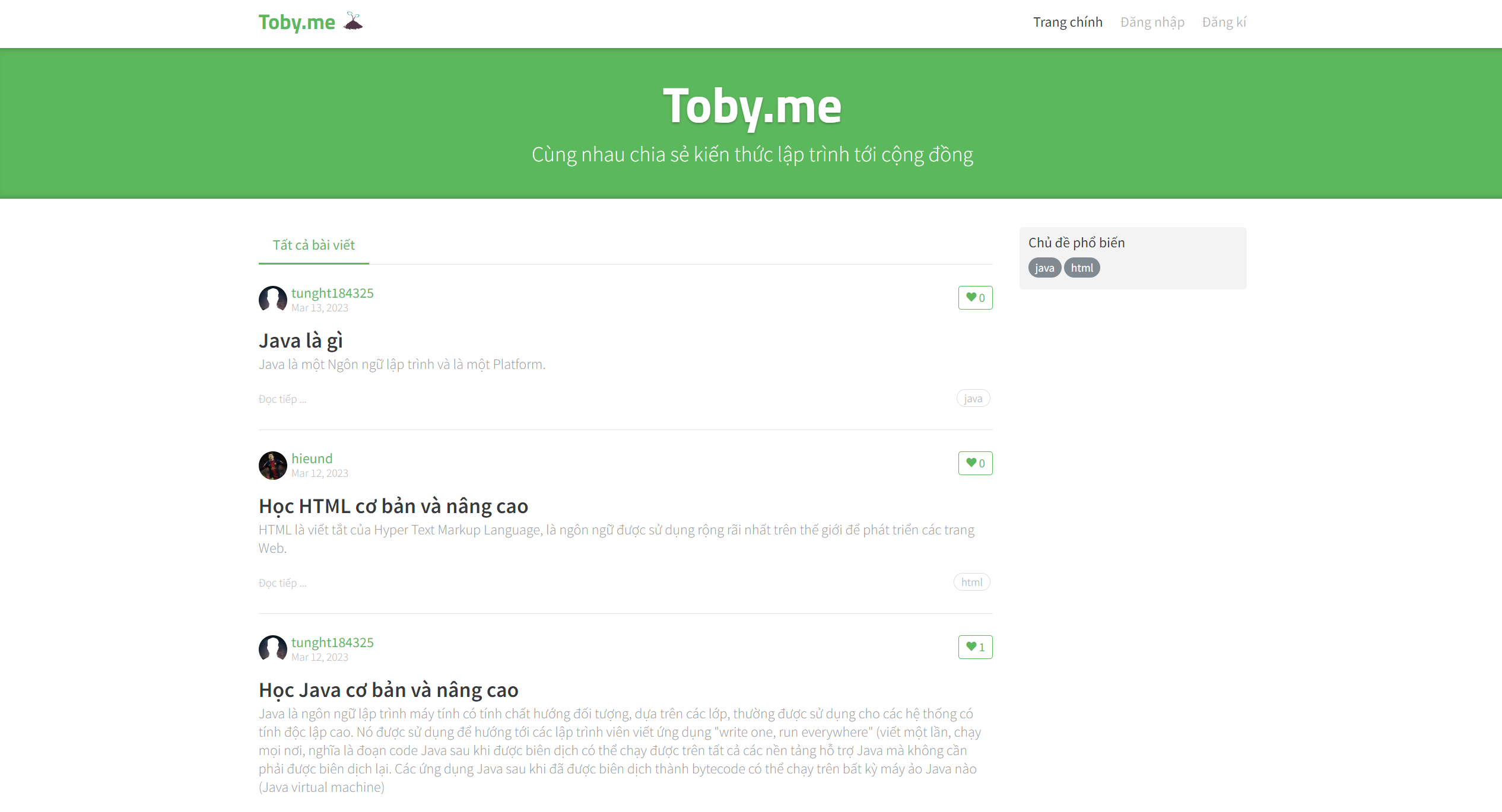
Description automatically generated

* article, profile, user, tag là các nhóm thực thể của hệ thống; common chứa các thuộc tính chung của các thực thể.
* Với gói article, bao gồm các gói con:
  + Controller: điều hướng yêu cầu của người dùng tới service xử lý phù hợp, sau đó nhận kết quả từ service và trả về cho người dùng.
  + Entity: các thực thể, quan hệ
  + Service: xử lý logic hệ thống
  + Repository: tương tác với cơ sở dữ liệu
  + Model: các đối tượng giữ các tham số request của người dung
  + Dto: Data Transfer Object, dùng để đón/trả dữ liệu từ/về người dùng. Giúp bảo nguyên tắc che giấu thông tin của server, client chỉ nhận được đúng phần thông tin mình cần chứ không thể biết tất cả.
* Các gói profile, user, tag tương tự gói article.

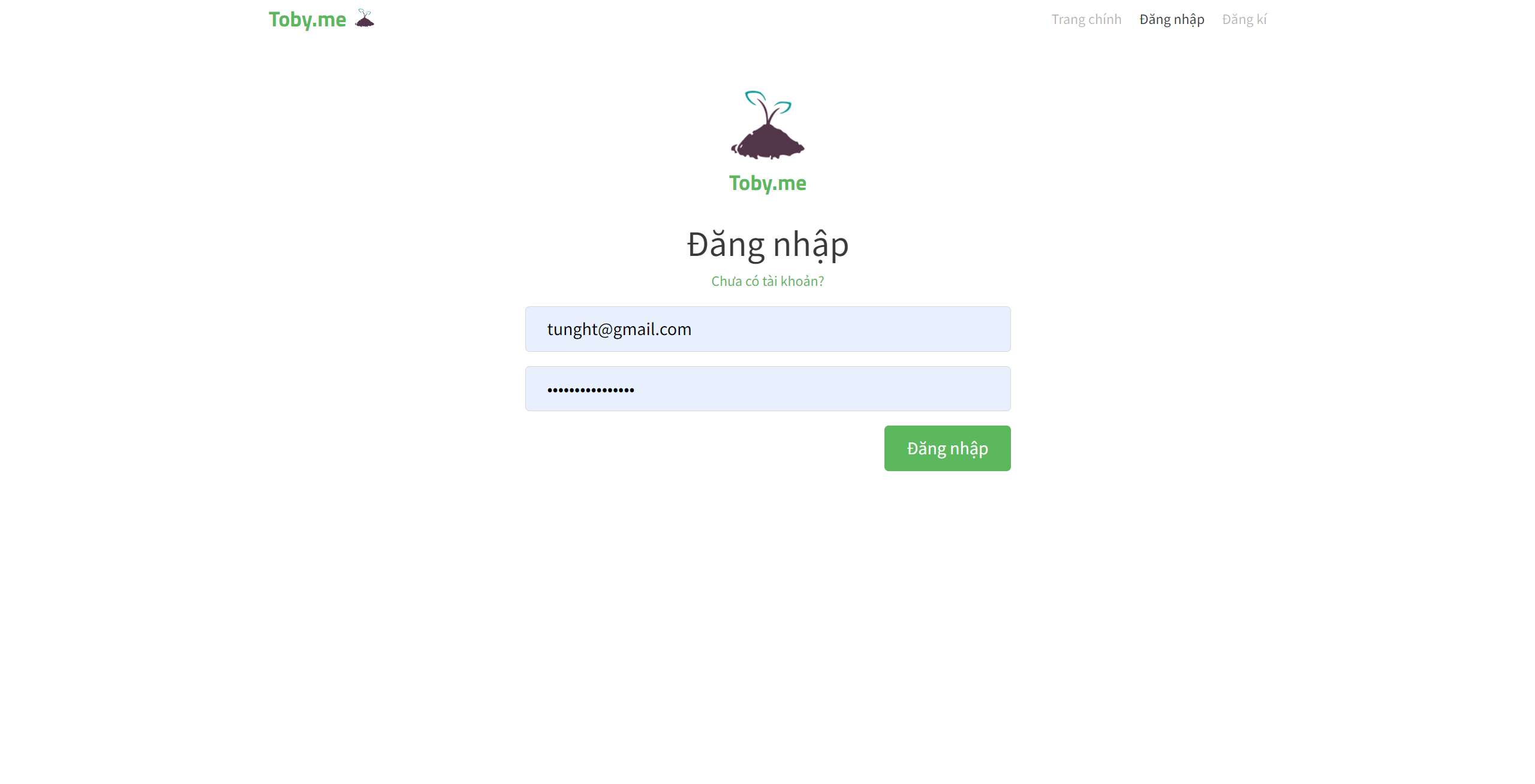
## Thiết kế chi tiết

### Thiết kế giao diện

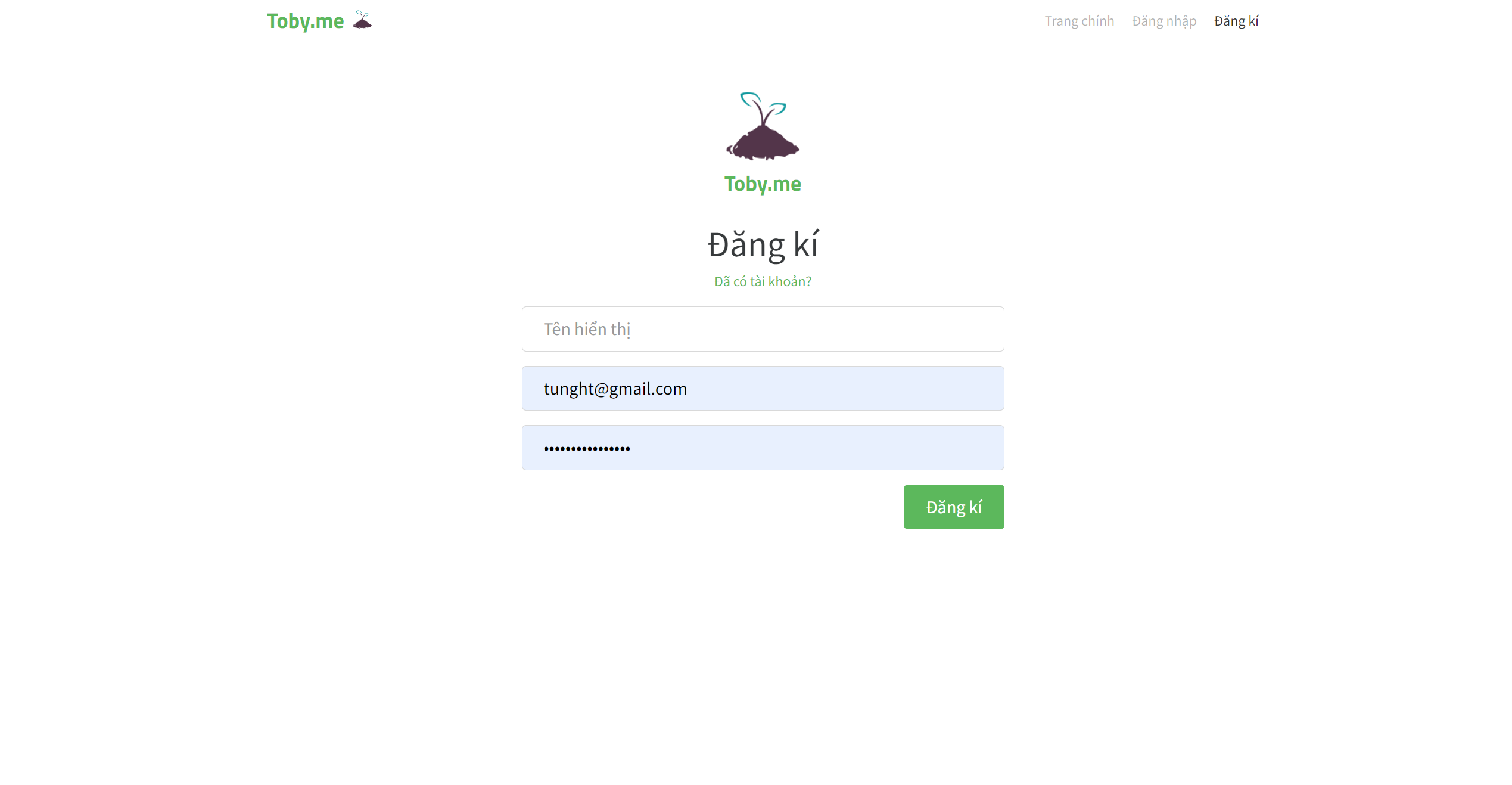
Trang chính:



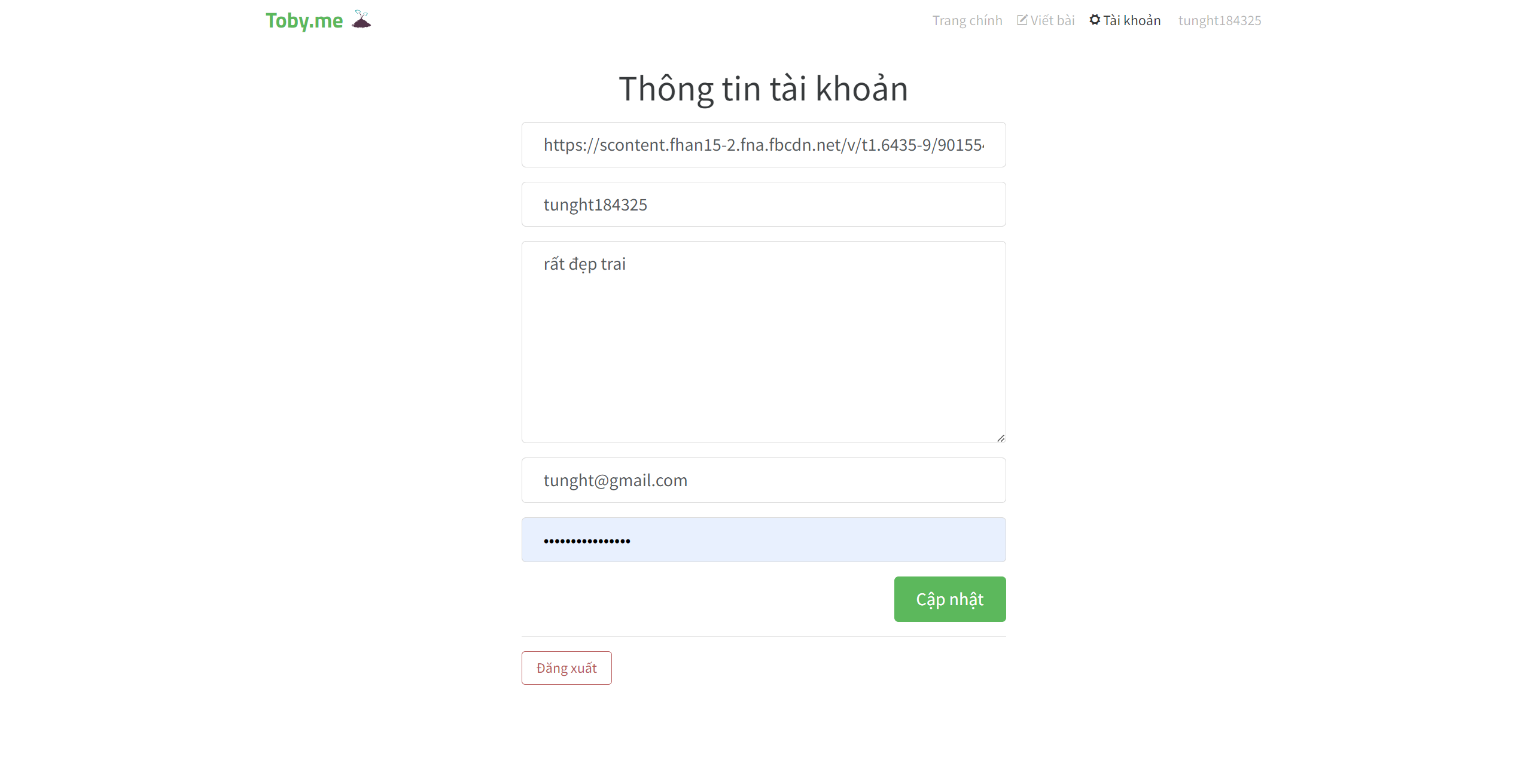
Màn hình đăng nhập:



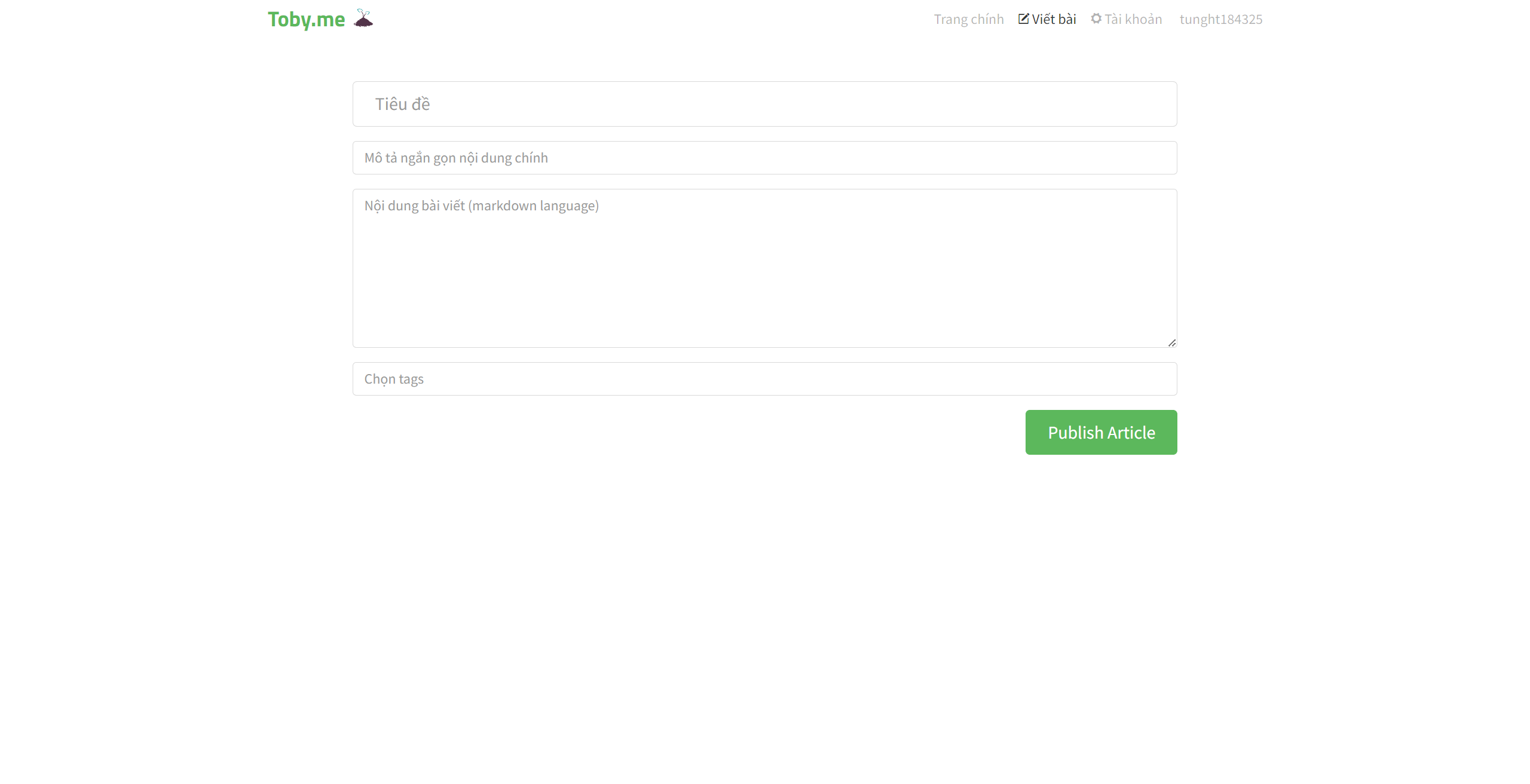
Màn hình đăng kí:



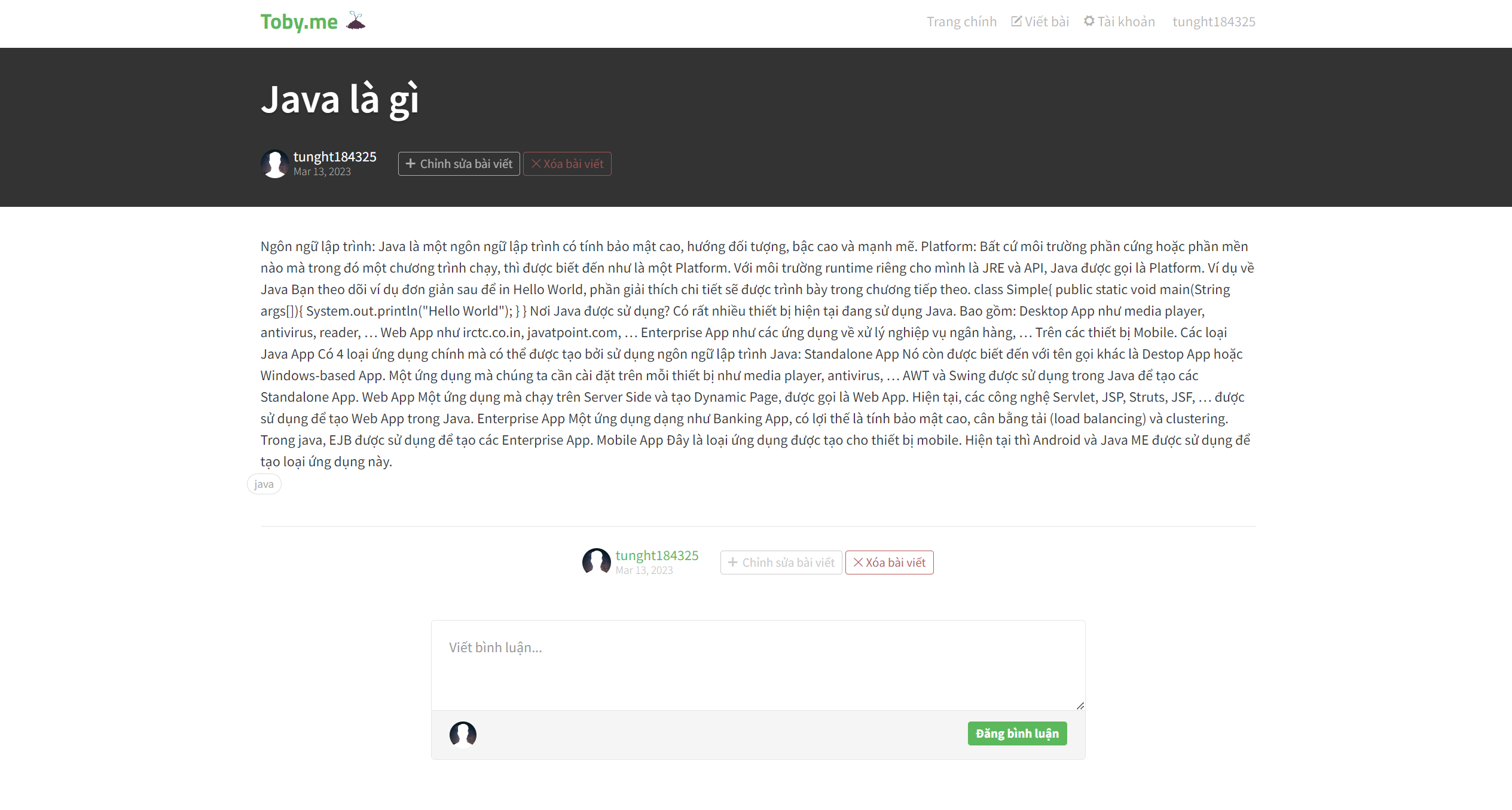
Quản lý tài khoản:



Tạo, sửa bài viết:



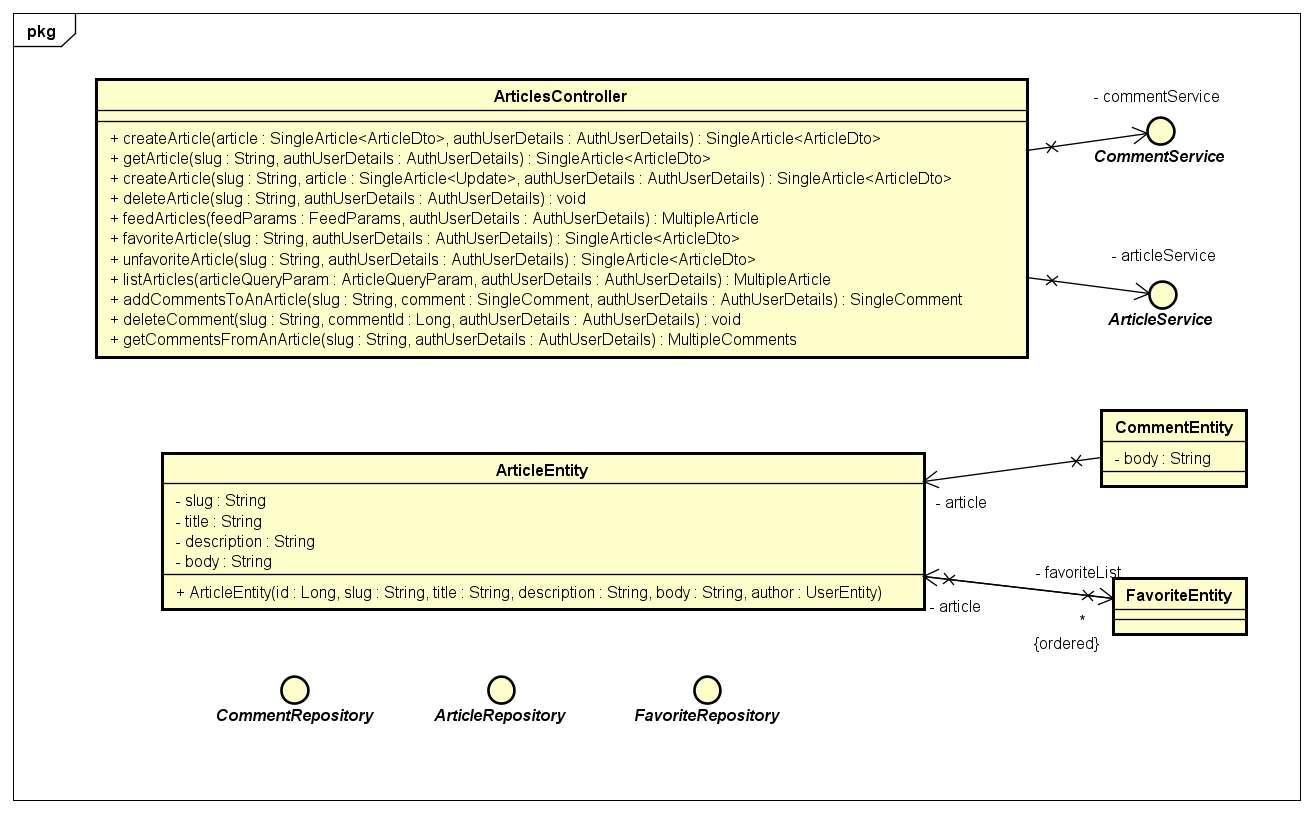
Xem chi tiết bài viết:



### Thiết kế lớp

Theo kinh nghiệm của em làm một số dự án SpringBoot client-server, việc thiết kế lớp trước gần như rất khó thực hiện. Bởi vì thiết kế lớp của 1 ứng dụng SpringBoot khá đơn giản, chỉ bao gồm những package như em đã trình bày phần trên. Tuy nhiên trong quá trình viết code, sẽ xảy ra nhiều vấn đề, cũng như việc thiết kế các class cũng phụ thuộc vào việc thiết kế API, khi đó mình sẽ thường phải thêm các class phụ để xử lý.

Thiết kế lớp của gói article:



### Thiết kế cơ sở dữ liệu

Thiết kế database: có thể thay đổi trong quá trình code. Bởi vì, việc ánh xạ các quan hệ hướng đối tượng sang quan hệ giữa các thực thể trong CSDL quan hệ không quá chính xác (ví dụ trong OOP có quan hệ kế thừa, nhưng trong CSDL quan hệ thì không có mối quan hệ nào là kế thừa). Một database được chuẩn hóa kĩ đôi khi gây ra khó khăn trong quá trình viết code.Waterfall chart

Description automatically generated