**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**🕯✡🕮🕮✡🕯**



**BÁO CÁO TIỂU LUẬN CHUYÊN NGÀNH**

**Ngành: CÔNG NGHỆ THÔNG TIN**

**Chuyên ngành: HỆ THỐNG THÔNG TIN**

**ĐỀ TÀI**

**TÌM HIỀU VỀ MICROSERVICE VỚI ANGULARJS VÀ SPRING FRAMEWORK**

**GVHD: TS. Nguyễn Thành Sơn**

**SVTH MSSV**

**Nguyễn Đặng Nhật Minh 15110081**

**Lê Thị Phương Nha 15110090**

**Tp. Hồ Chí Minh, tháng 9 năm 2018**

**CHƯƠNG 1**

**TỔNG QUAN VỀ KIẾN TRÚC MICROSERVICE, ANGULARJS**

**VÀ SPRING FRAMEWORK**

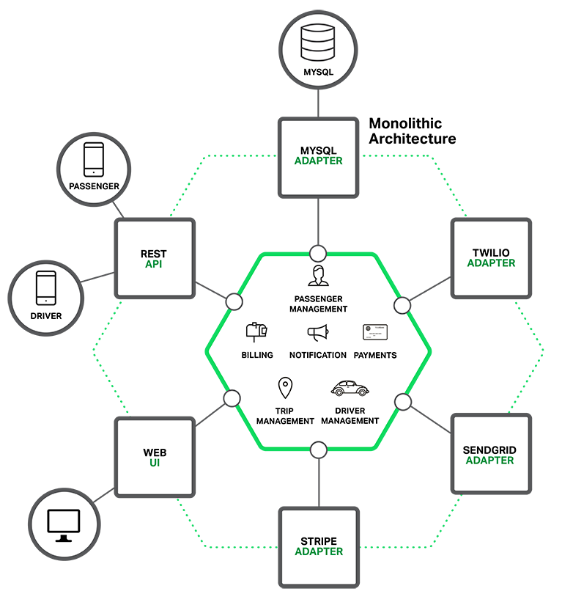
* 1. **Kiến trúc Microservice**

**1.1.1. Tổng quan về kiến trúc Microservice**

Microservice là kiến trúc với nhiều khối dịch vụ nhỏ. Chính xác là một kiểu xây dựng hệ thống, biến ứng dụng thành một hệ sinh thái tích hợp được nhiều công nghệ. Cách xây dựng này còn dễ dàng trong việc áp dụng phương pháp Agile (Scrum), giúp cho các doanh nghiệp biến các phần mềm có cấu trúc phức tạp trở nên dễ dàng hơn.

Có thể hiểu đơn giản là 1 ứng dụng sẽ có cả web và mobile, hệ thống cần thiết kế backend chung cho cả web và mobile kết nối vào backend đó. Xong web sẽ có frontend lại được chia thêm 2 loại cho người dùng sử dụng (client) và cho người quản lý (admin). Cũng có thể tương tự với mobile. Nghĩa là sẽ hệ thống trên sẽ có 1 quản lý database (backend) kết nối với hẳn 4 loại giao diện gồm web app và mobile app. Hoặc nhiều ngôn ngữ backend khác nhau như C#, Java, PHP kết nối với 1 web app.

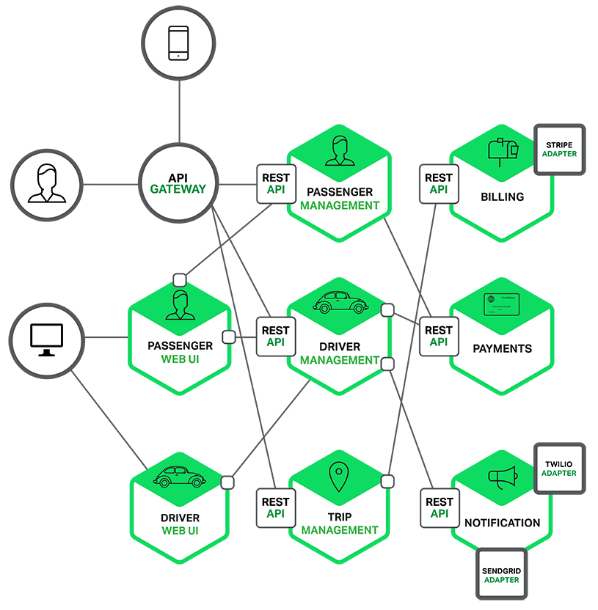
Một ví dụ cho thấy sự khác biệt giữa việc sử dụng hay không sử dụng Microservice trong việc phát triển ứng dụng:



Tưởng tượng bạn nhận một yêu cầu từ khách hàng phải xây dựng một dịch vụ gọi taxi qua di động cạnh tranh với Uber. Sau buổi phân tích yêu cầu, bạn chọn các công nghệ đại loại như Spring Boot, .Net, MEAN, MERN, Rails hay Maven. Dự án này sẽ có kiến trúc chia khối lục giác (hexagol architecture) hoặc ít hơn. Kiến trúc đa diện giúp ứng dụng chuyên biệt mô hình dữ liệu và đầu vào, đầu ra.

Trong lõi của ứng dụng là business logic thể hiện cho từng dịch vụ, sự kiện có trong hệ thống. Xung quanh lõi là các adapter, api, ui như cơ sở dữ liệu, web service hay frontend của web

Mặc dù cấu trúc module hợp lý. Nhưng ứng dụng sẽ được đóng gói thành 1 khối(monolithic). Và khi chạy phải tùy thuộc vào ngôn ngữ lập trình hay thư viện framework. Ví dụ như Java đóng gói thành file WAR, triển khai deploy trên application server như Tomcat. Hay 1 số ngôn ngữ khác đóng gói theo cấu trúc thư mục đa cấp, điều này làm cho ứng dụng kém tính linh hoạt, đa dạng, chỉ sử dụng 1 ngôn ngữ để lập trình . Điều này khiến code khó tái sừ dụng, trở nên cồng kềnh sau một thời gian do phải đối mặt với hàng tá gói thư viện JAR trong ứng dụng hàng triệu dòng code, biến ứng dụng của bạn trở thành một thanh niên phải bán hành vì bản thân không thể tự mình ăn hết. Đấy là điều mà bất kì lập trình viên nào khi đi làm cần phải lưu ý vì code khó đồng nghĩa với việc mất tiền.



Thực chất nhiều tập đoàn như Amazon, Ebay, Netflix đã giải quyết vấn đề ứng dụng dụng bằng kiến trúc microservices (nhiều dịch vụ nhỏ).

Với mỗi dịch vụ nhỏ thực hiện một tập các chức năng chuyên biệt, có kiến trúc đa diện lõi là business logic kết nối với các adapter khác nhau. Các dịch vụ giao tiếp với nhau qua API cho dịch vụ nhỏ khác hay ứng dụng client gọi tới. Khi vận hành mỗi dịch vụ nhỏ sẽ được chạy trên một máy ảo (virtual machine) hoặc Docker container (ảo hóa tầng ứng dụng). Nói cách khác là chia nhỏ từng dịch vụ sẽ giảm độ phức tạp của hệ thống lớn, dễ quản lý, bảo trì, nâng cấp và tái sử dụng code. Tiết kiệm thời gian nghĩa là tiết kiệm tiền .

**1.1.2. Tại sao lại sử dụng kiến trúc Microservice?**

Thứ nhất, giúp đơn giản hóa hệ thống. Với tổng số chức năng không đổi, kiến trúc Microservices chia nhỏ hệ thống ra làm nhiều dịch vụ nhỏ lẽ dể dàng quản lý và triển khai từng phần so với kiến trúc nguyên khối. Mỗi dịch vụ thì được định nghĩa để giao tiếp với các dịch vụ khác thông qua RPC (Remote Procedure Call) hay message-driven API. Kiến trúc Microservices thúc đẩy việc phân tách rạch ròi các dịch vụ nhỏ, việc khó có thể làm nếu xây dựng ứng dụng bằng kiến trúc một khối. Trên hết với mỗi dịch vụ nhỏ, chúng ta sẽ có thời gian phát triển nhanh hơn, dễ nắm bắt cũng như bảo trì hơn.

Thứ hai, kiến trúc này cho phép việc mỗi dịch vụ được phát triển độc lập bởi những team khác nhau. Cũng như cho phép developer có thể tự do chọn lựa technology stack cho mỗi dịch vụ mình phát triển. Dĩ nhiên tự do lựa chọn nhưng không phải là tạo ra một mớ hỗn độn về technology, điều này thì chằng có một dự án hay sản phẩm nào mong muốn cả. Tuy nhiên, sự tự do này có nghĩa là các developer không còn phải bắt buộc phải sử dụng các công nghệ lỗi thời có thể đã tồn tại vào lúc bắt đầu dự án. Khi viết một dịch vụ mới, họ có tùy chọn của việc sử dụng công nghệ bắt kịp với xu thế. Hơn nữa, vì dịch vụ là tương đối nhỏ, việc viết lại một dịch vụ cũ dựa trên nền tảng công nghệ mới hơn là hoàn toàn khả thi.

Thứ ba, kiến trúc Microservices cho phép mỗi dịch vụ có thể được triển khai một cách độc lập. Cùng với đó thì việc triển khai hệ thống theo kiểu continuous deployment là hoàn toàn có thể.

Cuối cùng, kiến trúc Microservices cho phép mỗi dịch vụ có thể thực hiện việc scale một cách độc lập. Bạn có thể scale dễ dàng bằng cách tăng số instance phục vụ cho mỗi dịch vụ lên và phân tải bằng load balancer. Ngoài ra bạn cũng có thể tối ưu chi phí vận hành dịch vụ bằng cách triển khai mỗi dịch vụ lên server có resource thích hơp.

* 1. **AngularJS Framework**

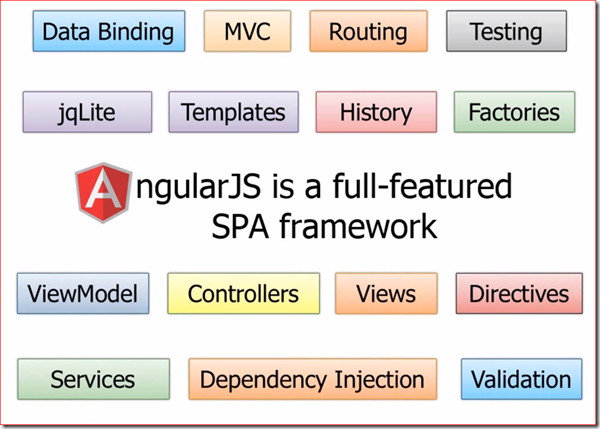
**1.2.1. Lịch sử phát triển**

Dự án AngularJS được bắt đầu từ năm 2009, do lập trình viên Misko Hevery tại Google viết ra. Misko và nhóm lúc này đang tham gia vào một dự án của Google tên là Google Feedback. Với AngularJS, Misko đã rút ngắn số dòng code front-end từ 17000 dòng còn chỉ khoảng 1500 dòng. Với sự thành cong đó, đội ngũ của dự án Google Feedback quyết định phát triển AngularJS theo hướng mã nguồn mở.

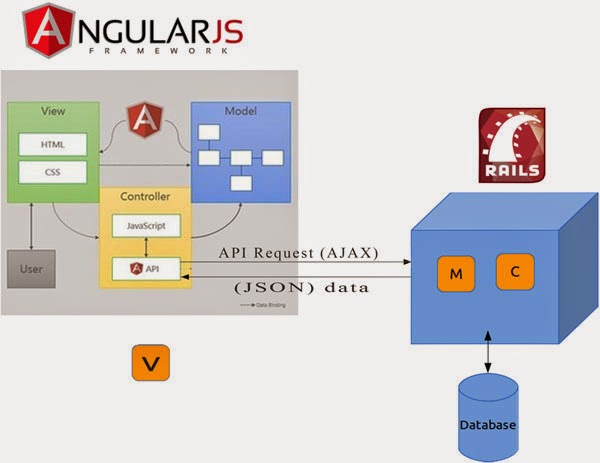
Trước năm 2000, một công nghệ HTML hỗ trợ tốt cho các trang web tĩnh. Khi bạn xây dựng một trang web với PHP, Node/Express, hay Ruby thì nó cũng chỉ là một trang web tĩnh với nội dung được thay đổi khi bạn gửi request về máy chủ, máy chủ sẽ render một trang với nội dung tương ứng. Tuy nhiên mọi thứ đã thay đổi nhiều từ sự phát triển của HTML5, nhất là khi có sự chống lưng từ những ông lớn như Google, Yahoo, Facebook, và sự tập hợp đông đảo của cộng đồng mã nguồn mở. Trong đó kể đến AngularJS giúp HTML động hơn.

**1.2.2. AngularJS là gì?**

Angular là một bộ Javascript Framework rất mạnh và thường được sử dụng để xây dựng project **Single Page Application** (SPA). Nó hoạt động dựa trên các thuộc tính mở rộng HTML (các atributes theo quy tắc của Angular). Một framework có cấu trúc cho các ứng dụng web động. Nó cho phép bạn sử dụng HTML như là ngôn ngữ mẫu và cho phép bạn mở rộng cú pháp của HTML để diễn đạt các thành phần ứng dụng của bạn một cách rõ ràng và xúc tích. Hai tính năng cốt lõi: Data binding (liên kết dữ liệu) và Dependency injection giúp loại bỏ phần lớn code mà bạn thường phải viết. Nó xảy ra trong tất cả các trình duyệt, làm cho nó trở thành đối tác lý tưởng của bất kỳ công nghệ Server nào. Đây là một Framework mã nguồn mở hoàn toàn miễn phí và được hàng ngàn các lập trình viên trên thế giới ưa chuộng và sử dụng.



Phân tích sơ đồ làm việc với AngularJS: Web của bạn sẽ kết nối với server API, lấy về result dưới dạng JSON, AngularJS sẽ đóng vai trò dât-driven để biến JSON đó thành những Models. Bạn có thể nhanh chóng update lên HTML content mà không cần phải refresh lại trang.



Nói qua cơ chế hoạt động SPA – Single-page Application, ở lần request đầu tiên, hệ thống sẽ trả về tất cả code xử lý cũng như code hiển thị của toàn bộ trang web, ở những yêu cầu tiếp theo client chỉ phải requessts những phần nào mình cần, và server sẽ trả về dữ liệu dưới dạng thô, giúp rút ngắn thời gian truyền tải, giúp nâng cao trải nghiệm của người dùng hơn.

Angular thích hợp sử dụng nhất cho những ứng dụng có 1 trang SPA (đó là một trang web hay một ứng dụng web, mà tất cả những thao tác xử lý của trang web đều được diễn ra trên một trang duy nhất. Với mục đích chính là nâng cao trải nghiệm người dùng tốt hơn, hiệu quả hơn, thoải mái hơn.).

Một ví dụ đơn giản dễ hiểu là trang Facebook, Nếu bạn đang lướt News feed, không cần load lại cả trang web, tất cả những gì thay đổi chỉ là bảng tin mới nhất được cập nhật, các khung xung quanh vẫn không đổi. Bạn sẽ cảm giác đang trượt tuyết trên một làn băng duy nhất và không phải bất công làm chuyển làn băng khác.

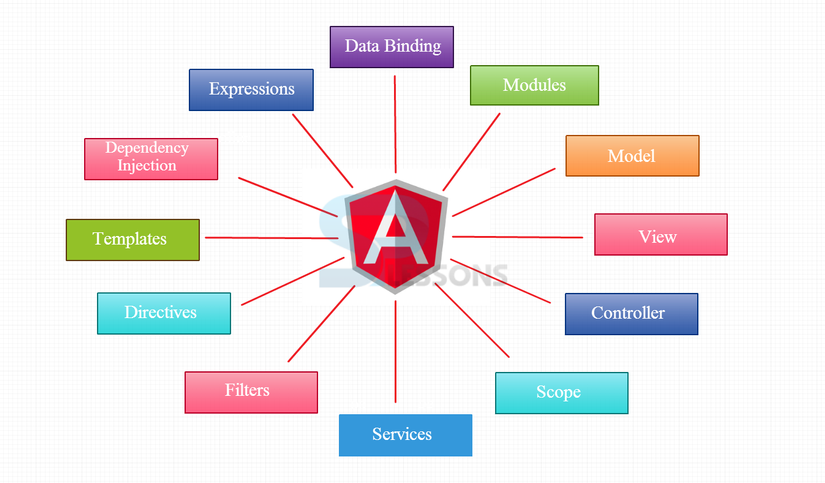
.Để học được Angular thì bạn phải có kiến thức cơ bản về javascript, biết cách xử lý object, string,… Hay thậm chí nếu như bạn biết chuyên sâu Javascript thì đó là một lợi thế rất nhiều. Vì bản chất Angular hoạt động dạng Single Page, sử dụng dạng API để lấy dữ liệu, vì vậy bạn phải biết các kỹ thuật xử lý DHTML, Ajax.

**1.2.3. Các ứng dụng chung của AngularJS trong phát triển ứng dụng web**

* AngularJS là một Framework phát triển mạnh mẽ dựa trên JavaScript để tạo các ứng dụng RICH Internet Application (RIA).
* AngularJS thường dùng để phát triển frontend (giao diện khách hàng) thông qua các API để gọi data, sử dụng mô hình MVC rất mạnh mẽ
* Angular là mã nguồn mở, hoàn toàn miễn phí và được phát triển bởi hàng ngàn các lập trình viên trên thế giới.
* Các ứng dụng được viết bởi AngularJS tương thích với nhiều phiên bản trình duyệt web. AngularJS tự động xử lý mã JavaScript để phù hợp với mỗi trình duyệt.
* AngularJS cung cấp cho lập trình viên những tùy chọn để viết các ứng dụng client-side trong mô hình MVC (Model View Controller) một cách rõ ràng.

Nhìn chung, AngularJS là một framework để tạo các ứng dụng lớn, các hiệu năng cao trong khi giữ cho chúng có thể dễ dàng duy trì.

**1.2.4. Các tính năng chính của AngularJS**



* **Data-binding**: Tự động đồng bộ hóa dữ liệu giữa Model và View.
* **Module**: dùng để định nghĩa một ứng dụng AngularJS. Module giống như một chiếc hộp để chứa các thành phần khác của ứng dụng bên trong nó.Module còn được gọi là một container của ứng dụng.
* **Scope**: Là những đối tượng hướng đến model. Nó là cầu nối giữa Controller và View.
* **Controller**: Là những hàm JavaScrip xử lí kết hợp với Scope.
* **Service**: AngularJS sử dụng các API được xây dựng từ các web service để thao tác với Database.
* **Filters**: Bộ lọc để lọc ra các thành phần của một mảng và trả về một mảng mới.
* **Directives**: Đánh dấu vào các yếu tố của DOM, nghĩa là sẽ tạo ra các thẻ HTML tùy chỉnh.
* **Template**: Các rendered view với các thông tin từ controller và model có thể sử dụng trong các file riêng biệt hoặc với view với trang sử dụng Partials.
* **Routing**: Chuyển đổi giữa các action trong Controller.
* **Model View Controller**: Mô hình chia thành phần riêng biệt thành Model, View, Controller. Đây là một mô hình khá hay nhưng trong Angular thì nó được chế biến lại một chút thành Model View View Model.
* **Deep Linking**: Liên kết sâu cho phép mã hóa trạng thái của ứng dụng trong các URL để nó có thể đánh dấu được với công cụ tìm kiếm.
* **Dependency Injection**: Giúp các nhà phát triển tạo ứng dụng dễ dàng hơn.

**1.2.5. Tại sao phải sử dụng AngularJS?**

* Tiết kiệm thời gian viết code của bạn
* Nó có template cho phép bạn viết các thẻ đặc biệt trong HTML mà sau đó sẽ được thay thế bằng dữ liệu để giảm bớt công việc của bạn: Angular Tags {{prevItem}}{{nextItem}}
* Có đầy đủ chức năng của AJAX để tải về dữ liệu bên ngoài và có thể dễ dàng tải về dữ liệu JSON. Nó tự động với DOM.
* Sử dụng kiến trúc MVC
* Angular được đưa ra bởi Google. Họ đã đưa những giải pháp javascript vào việc phát triển ứng dụng web một cách tinh tế và đáng tin cậy.

**1.3. Spring Framework**

**1.3.1. Tổng quan về Spring**

Có thể nói rằng Spring Framework ra đời là nhờ sự thành công của **EJB**. EJB (Enterprise JavaBeans) là một kiến trúc phát triển dành cho các ứng dụng doanh nghiệp đòi hỏi tính mạnh mẽ và khả năng mở rộng cao. EJB là một thành phần quan trọng của nền tảng Java EE. EJB được tạo ra bởi hai gã khổng lồ công nghệ là IBM và Sun vào những năm cuối thập niên 90. Dưới sự phát triển của hai tập đoàn này, EJB đã trở nên nặng nề hơn. Tiêu chuẩn EJB cũng nhanh chóng được thông qua. Sau tất cả, EJB được ca ngợi là giải pháp tốt nhất để xây dựng các ứng dụng phân tán dành cho doanh nghiệp.

Nhưng rồi những chỉ trích về chi tiết kỹ thuật của EJB đã tăng nhanh còn hơn sự nổi tiếng của nó. EJB là một framework lớn với độ phức tạp cao. Các nhà phát triền khi làm việc với EJB đã nhận ra rằng nó không hoạt động như những gì nó đã đề ra. EJB chỉ tuyệt vời trên bàn giấy, nhưng ra thực tế đó lại là một chuyện khác.

EJB vẫn được tiếp tục phát triển, dễ sử dụng hơn nhiều so với buổi ban đầu. Nhưng những chỉ trích vẫn còn đó, nỗi thất vọng trong cộng đồng Java ngày càng tăng lên, đòi hỏi cần có một giải pháp thay thế.

**1.3.2. Sự ra đời của Spring**

Vào năm 2002, Spring Framework phát hành phiên bản đầu tiên bởi Rod Johnson. Việc xây dựng các ứng dụng doanh nghiệp trở nên đơn giản và dễ dàng hơn. Rod Johnson đã cho xuất bản một cuốn sách được phổ biến rộng rãi có nhan đề "J2EE Development without EJB". Điều này đã cho Spring Framework rất nhiều động lực để thay thế EJB.

Theo đà phát triển đó, Spring đã trở thành framework mã nguồn mở phổ biến nhất để xây dựng các ứng dụng doanh nghiệp. Cách tiếp cận thực tế ban đầu của Rod Johnson tiếp tục được phát triển và hướng tới một bộ công cụ hoàn chỉnh dành cho xây dựng các ứng dụng doanh nghiệp. Theo một số nguồn, trên 50% các ứng dụng web Java hiện nay đang sử dụng Spring.

Để ngăn chặn sự phức tạp trong phát triển các ứng dụng, Spring Framework thường dựa trên các quan điểm như sau:

* Đơn giản hóa công việc phát triển thông qua việc sử dụng các POJO (Plain Old Java Object)
* Nới lỏng ràng buộc giữa các thành phần thông qua việc sử dụng Dependency Injection
* Giảm thiểu các mã boilerplate thông qua việc sử dụng template và aspect ...

Trên thực tế Spring Framework là một tập hợp của nhiều dự án con. Spring Core như chúng ta đã tìm hiểu là nền tảng của các dự án trong Spring Framework. Một dự án sẽ đảm nhận một chức năng riêng trong việc xây dựng các ứng dụng doanh nghiệp. Chúng ta sẽ cùng điểm qua các dự án lớn trong Spring.

**1.3.4. Các dự án trong Spring**

Trên thực tế Spring Framework là một tập hợp của nhiều dự án con. Spring Core như chúng ta đã tìm hiểu là nền tảng của các dự án trong Spring Framework. Một một dự án sẽ đảm nhận một chức năng riêng trong việc xây dựng các ứng dụng doanh nghiệp. Chúng ta sẽ cùng điểm qua các dự án lớn trong Spring.

**Spring MVC**

Spring MVC được thiết kế dành cho việc xây dựng các ứng dụng nền tảng web. Đây là một dự án chúng ta không thể bỏ qua khi xây dựng các ứng dụng Java web.

**Spring Data**

Cung cấp một cách tiếp cận đúng đắn để truy cập dữ liệu từ cơ sở dữ liệu quan hệ, phi quan hệ, map-reduce và thậm chí còn hơn thế nữa.

**Spring Security**

Dự án này cung cấp các cơ chế xác thực (authentication) và phân quyền (authorization) cho ứng dụng của bạn.

**Spring Boot**

Spring Boot là một framework giúp chúng ta phát triển cũng như chạy ứng dụng một cách nhanh chóng.

**Spring Batch**

Dự án này giúp chúng ta dễ dàng tạo các lịch trình (scheduling) và tiến trình (processing) cho các công việc xử lý theo mẻ (batch job).

**Spring Integration**

Spring Integration là một implementation của Enterprise Integration Patterns (EIP). Dự án này thiết kế một kiến trúc hướng thông điệp hỗ trợ việc tích hợp các hệ thống bên ngoài.

**Spring Social**

Dự án này sẽ kết nối ứng dụng của bạn với các API bên thứ ba của Facebook, Twitter, Linkedin,...

**1.3.5. Tại sao phải sử dụng Spring Framework?**

**Xây dựng Web Application với Spring:**



Với nhiều Java developer, phát triển web application là mục tiêu chính của họ. Nếu bạn đã và đang hướng đến mục tiêu này, bạn sẽ nhận thấy rõ 3 thử thách cần vượt qua:  state management, workflow, validation. Một điều khác nữa, HTTP lại là stateless design, vì vậy, vượt qua 3 thử thách kể trên chưa bao giờ đơn giản.

Spring MVC là một framework được xây dựng trên nền của Spring. Nó vẫn bảo tồn được các tinh hoa của Spring và là một công cụ đắc lực hỗ trợ bạn trong vấn đề này. Dựa trên mô hình MVC kinh điển, Spring MVC sẽ giúp bạn xây dựng các web application linh hoạt và mạnh mẽ.

Nếu bạn cần một web app kiểu stateful, hãy thử Spring Web Flow. Spring Web Flow giúp bạn xây dựng một wizard - một tập hợp các cửa sổ mà người dùng phải duyệt qua tuần tự trước khi lấy được dữ liệu. Spring Web Flow giúp bạn tránh được việc lưu trữ quá nhiều dữ liệu tạm.

Với Spring Web Flow, bạn có thể thử sức với một project đặt vé máy bay, vé tàu hỏa.

**Tiếp cận với RESTful Service:**

Với phần này, hãy làm quen với Spring HATEOAS. HATEOAS là một ràng buộc thuộc REST application architecture.

Một web service developer sẽ cung cấp một "hợp đồng" với consumer của service đó trước khi consumer gọi đến service.

Trong thế giới của SOAP, điều này gần giống với việc tạo một interface, nhờ interface đó mà client biết được method nào có thể sử dụng được.

Trong thế giới của REST, chúng ta xây dựng các URL kèm theo dữ liệu phù hợp, chúng ta vẫn cần biết server đang "mong đợi" thông tin gì.

Trong thế giới của Hypermedia, bộ phận resource identifier (bộ nhận dạng tài nguyên) sẽ đưa ra các yếu tố liên quan đến những thông tin mà nó cần. Những yếu tố này có thể coi như các quy ước để tương tác giữa 2 bên trong tương lai.

[*Jersey*](https://jersey.java.net/)*cũng là một open source web service xây dựng trên nền tảng Java. Việc xây dựng web service của Jersey tuân theo đặc tả của JSR.   
Việc convert các API từ Spring sang Jersey và ngược lại tương đối dễ dàng.*

Jersey chỉ tập trung vào REST, Jersey không thể hỗ trợ một JSP webpage trong khi Spring MVC mang lại cho bạn cả 2 thứ một cách xuất sắc. Nếu client muốn dùng jQuery trong JSP webpage, Spring MVC hoàn toàn đáp ứng được. Kể cả khi bạn muốn sử dụng một framework JavaScript MVC bất kỳ, Spring MVC vẫn hỗ trợ bạn. Theo tôi, khi bạn phân vân giữa Spring MVC và các giải pháp khác, hãy theo Spring MVC.

Spring MVC mang đến sự thanh thoát cũng như đảm bảo cấu trúc mạch lạc cho mô hình servlet programming. Nó tách riêng model với view (Thực ra vấn đề này được Struts MVC giải quyết từ cách đây rất lâu. Spring MVC chỉ theo sau một chút)

Các framework Javascript MVC có vài trò gần giống với Spring MVC nhưng chúng được áp dụng cho tầng Javascript. Với các framework Javascript MVC, bạn có thể dựng các trang web độc lập với logic của server.

Hai tầng được dựng từ Java và Javascript sẽ liên lạc và trao đổi dữ liệu với server thông qua các đường RESTful (RESTful URL).

## ****Spring Security và vấn đề bảo mật:****

Spring Security sẽ interceipt các URL theo một mẫu (pattern) nào đó, chẳng hạn, các URL có dạng "/user" thì mọi người dùng đều truy cập được, còn dạng "admin" thì chỉ khả dụng với một vài cá nhân nào đó.

Nhờ Spring Security, bạn có thể dán một annotation lên các method để phục vụ công tác bảo mật. Tôi sẽ dành một bài viết về vấn đề này.

Spring Security cung cấp cả 2 cơ chế **authentication** (xác thực) và **authorization** (phân quyền):

Xác thực - **authentication**: xác định bạn là ai

Phân quyền - **authorization**: giới hạn những hành động của bạn

Khi bắt đầu một project mới, tôi tích hợp Spring Security và sử dụng cơ chế**in-memory authentication**.

Với cơ chế này, bạn sẽ tạo ra sẵn một đống username, password và các phân quyền, nhét chúng vào một file xml hoặc java... Spring Security sẽ cho phép bạn truy cập với một username và password tương ứng, sau khi truy cập, bạn sẽ được phân quyền phù hợp. Sau này, bạn có thể thay đổi cơ chế xác thực, phân quyền mà không làm ảnh hưởng tới business logic của hệ thống.

## ****Thao tác trong CSDL với Spring****

Để làm việc với cơ sở dữ liệu, ta sẽ dùng Spring ORM và hibernate.

ORM = Object Relational Mapping - Một cơ chế cho phép biểu diễn các bảng dữ liệu dưới dạng Java object và ngược lại. **Hibernate**và **myBatis**là hai ORM framework phổ biến nhất. Một sự thật thú vị, trước khi iPhone và iPad trở nên phổ biến, myBatis từng được biết đến dưới cái tên **iBatis**.

Nếu bạn phải xử lý một lượng lớn các đoạn code JDBC cồng kềnh, hãy chuyển qua Spring JDBC. Spring JDBC sẽ giúp thu gọn đáng kể khối code JDBC truyền thống thông qua các Template.

Nếu bạn đang làm project mới, hãy thử Spring Data.... Với Spring Data, bạn không cần viết các CRUD operation. Chúng được tích hợp sẵn, tôi sẽ giải thích điều này trong một bài viết khác. Tôi đã kết hợp Spring Data với Oracle và MongoDB, mọi thứ hoạt động rất trơn tru. Spring Data cũng hộ trợ các CSDL NoSQL và CSDL dạng đồ thị như Redis,Neo4j, Cassandra, Couchbase....

## ****Spring Batch****

Spring Batch cung cấp một tập hợp các reusable function (các hàm có thể tái sử dụng). Các hàm này giữ vai trò trọng yếu khi xử lý một lượng lớn các bản ghi trong CSDL.

Khi đó, mỗi request phát sinh sẽ kích hoạt một Spring Batch Job và ngay lập tức một Job ID sẽ được trả về. Tôi có thể dùng Job ID đó để kiểm soát trạng thái của tiến trình xử lý và thông báo cho người dùng biết tiến độ công việc. Các thông báo cho người dùng có thể xuất hiện dưới dạng email notification hoặc bạn có thể dùng web sockets để bắn trực tiếp notification tới người dùng khi họ đang online.

Spring Batch cũng cung cấp cho bạn một giải pháp để tự động sinh các thông báo khi trạng thái của hệ thống thay đổi. ...

Spring Batch sẽ đảm bảo tính toàn vẹn và nhất quán dữ liệu.

## ****Spring Integration và các hệ thống ngoại vi:****

Nếu đối tác gửi các report qua web service với định dạng XML. Trung bình, chúng tôi nhận 100.000 file XML trong 6 tháng. Các file XML này không quá lớn nhưng vấn đề ở đây là không thể xác định được thời điểm mà các file sẽ được gửi đến. Mục tiêu là phải có một bộ phận tự động "hứng" được các file này ngay khi chúng được gửi dến, sau đó tiến hành tiền xử lý rồi chuyển tiếp vào hệ thống.

Đó là một kịch bản phổ biến khi bạn xây dựng một service phụ thuộc vào hệ thống bên ngoài.

Với trường hợp này, ta nên chọn Spring Integration. Hãy cài đặt một file poller - lấy nơi đó để lưu trữ các file mà bên khách hàng gửi đến. Khi có bất cứ file nào "bay" đến, một event sẽ được tự động kích hoạt, thực hiện các nghiệp vụ xử lý rồi chuyển vào hệ thống.

Tất nhiên, phần code xử lý các file này độc lập hoàn toàn so với phần còn lại của hệ thống. Nhờ vậy, hệ thống của chúng tôi không phụ thuộc hoàn toàn vào hệ thống của đối tác. Bên cạnh đó, việc test các mô-đun nhận file này cũng dễ dàng hơn rất nhiều.

## ****Spring Test****

Spring không hề đứng ngoài xu thế TDD. Nó cung cấp một module phục vụ cho test các application - Spring Test

Trong module này, với Unit Test, bạn sẽ có một tập các mock object implementation để viết unit test tương thích với JNDI, servlet, portlets. Còn với Integration Test, module này hỗ trợ việc load các bean theo nhiều context khác nhau.

## ****Áp dụng Spring cho các Java project đơn lẻ****

Nếu sản phẩm có các business rules engine dựa trên nền Java và rất nặng về XML parsing. Khi hệ thống có vấn đề, rất khó để test xem rắc rối đang xảy ra ở đâu.

Giải pháp đề xuất lúc đó là áp dụng Spring Framework vào quy trình test. Hãy dùng các mô-đun Spring Context, Spring Test và JUnit, sau đó tạo một test package trong project đó. Tester sẽ có một file XML và các unit test để chuẩn bị dữ liệu trước mỗi lượt test.

## ****Chuyển đổi application thành dạng executable với Spring boot:****

Spring Boot là một project mới dựa trên ý tưởng sử dụng chính Spring để đơn giản hóa Spring.

Spring Boot chủ yếu sử dụng automatic configuration để loại bỏ Spring configuration truyền thống. Dù ta sử dụng Gradle hay Maven, Spring Boot đều cung cấp một số starter project được tối ưu sẵn để giảm kích thước build files.

**Spring Social và quá trình tích hợp các phương tiện truyền thông**

Spring Social để cho phép người dùng đăng nhập vào application của mình. Spring Socical sử dụng cơ chế OAuth để việc login từ các trang mạng xã hội trở nên suôn sẻ.

Spring Mobile cho phép bạn tinh chỉnh application sao cho phù hợp với các thiết bị di động. Dù vậy, thực tế cho thấy Spring Mobile không phải là lựa chọn đúng đắn nhất cho các thiết bị di động. Thay vào đó, hãy cân nhắc Bootstrap và AngularJS.

## ****11. Bắt đầu một project với Spring****

Spring sẽ giúp bạn bắt đầu một project bất kì phù hợp với nhu cầu của chính bạn.

Trong vòng vài phút, Spring Boot sẽ giúp bạn tạo một web app hoặc standalone app, rất đơn giản nhưng không kém phần mạnh mẽ. Bạn có thể dùng Grails, kết quả thu được tương tự!

Spring Cloud là nền tảng để xây dựng micro service một cách nhanh chóng. Micro service xây dựng bởi Spring Cloud đã được tự động config phục vụ cho resgistration và discovery.

Spring XD sẽ là trợ thủ đắc lực cho các hệ thống phân tán và big data.

**1.4. Cài đặt**

Các phần mềm cần cài đặt khi thực hiện đồ án này: Visual Studio Code, IntelliJ, Xamp, SQLyog

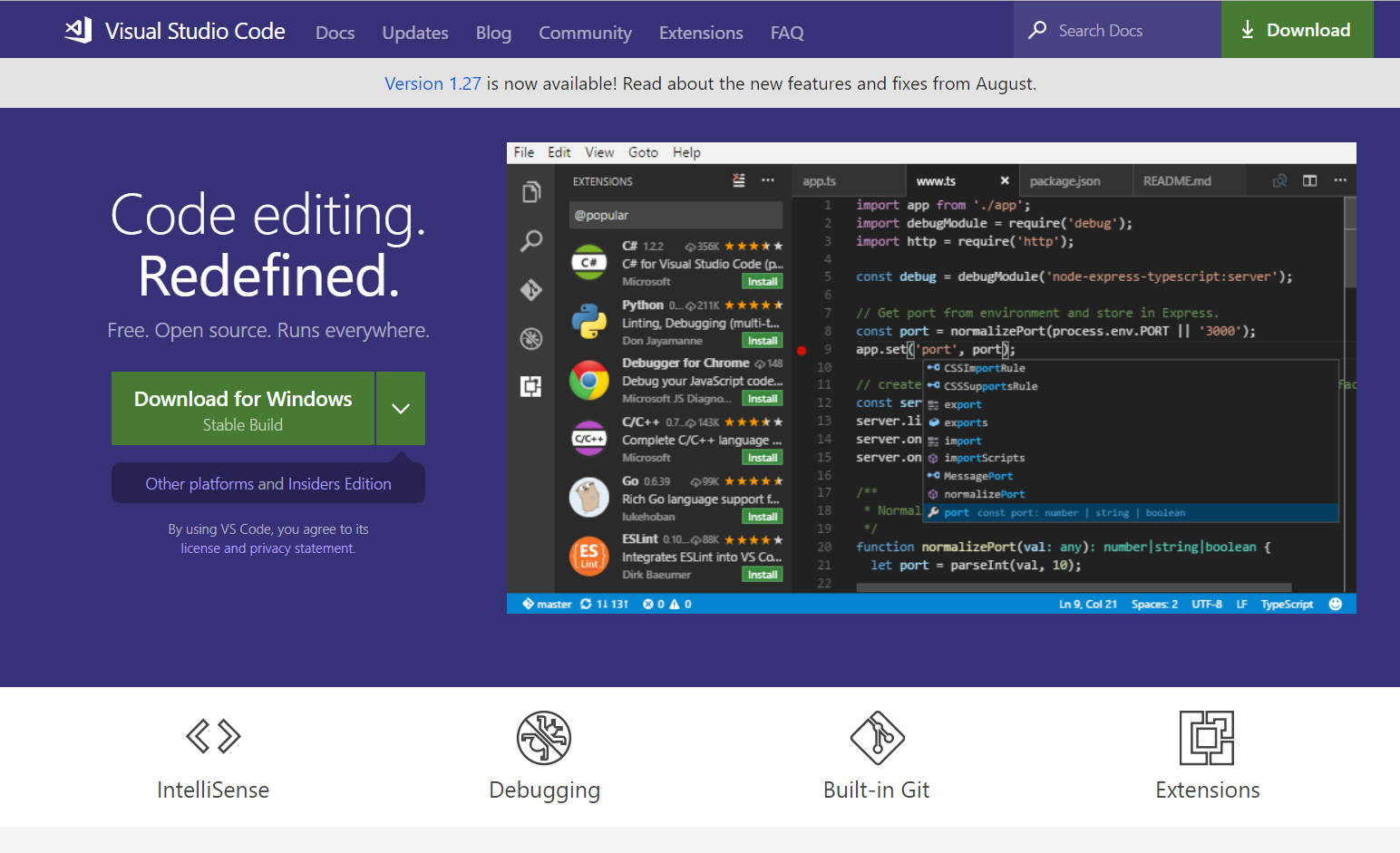
**1.4.1. Visual Studio Code**

**Yêu cầu phần cứng, phần mềm**

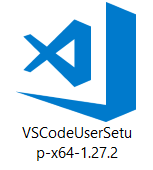
Visual Studio Code yêu cầu máy tính phải được cài đặt sẵn **.NET Framework 4.5.2**. Ta có thể kiểm tra bằng cách vào **Control Panel > Programs and Features**.

**Cách cài đặt**

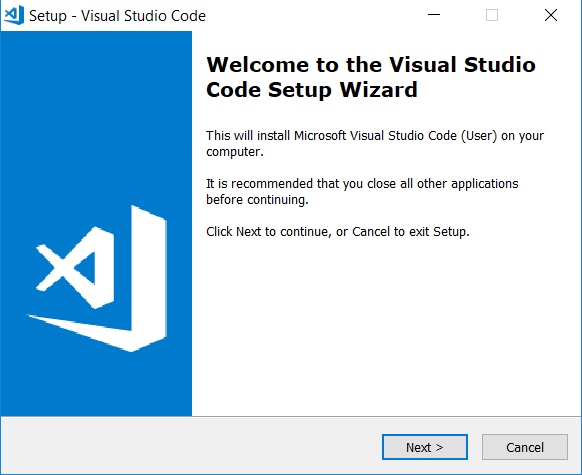
Tải Visual Studio Code theo đường dẫn: <https://code.visualstudio.com/>



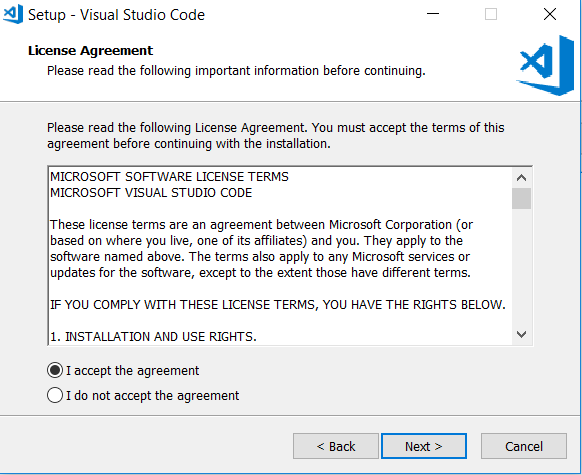
Sau khi quá trình tải hoàn tất, chạy file cài đặt (**VSCodeSetup-1.27.2.exe**).



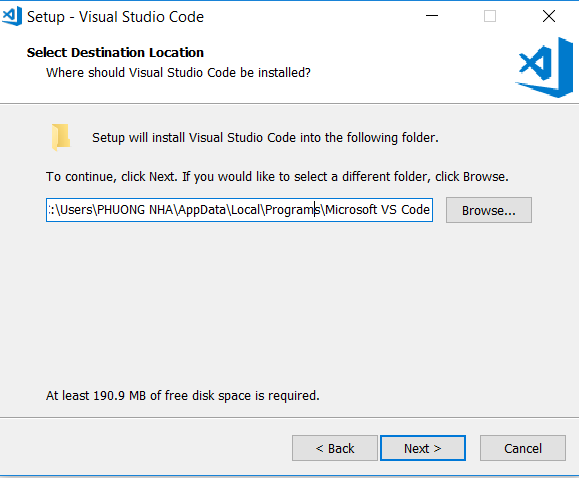
Trình cài đặt Visual Studio Code xuất hiện, chọn **Next**để bắt đầu quá trình cài đặt.



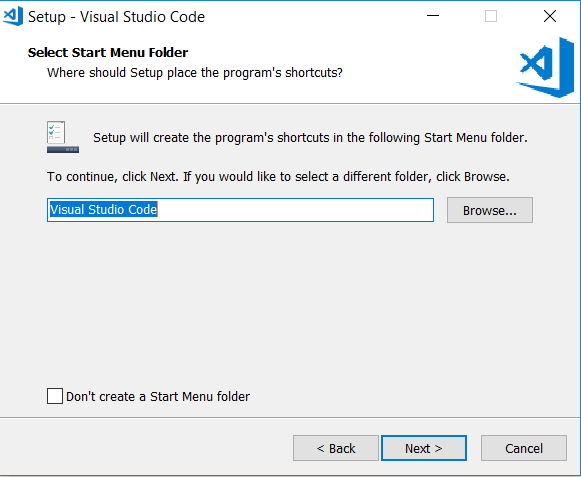
Chọn đồng ý với các điều khoản của phần mềm và chọn **Next**để tiếp tục.



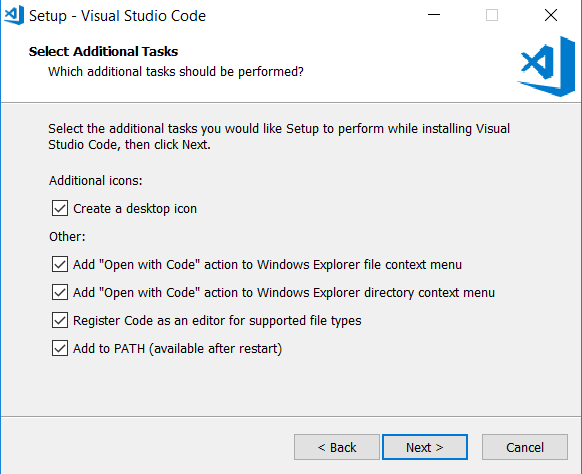
Tại bước này ta sẽ lựa chọn vị trí cài đặt của Visual Studio Code sau đó chọn **Next**.



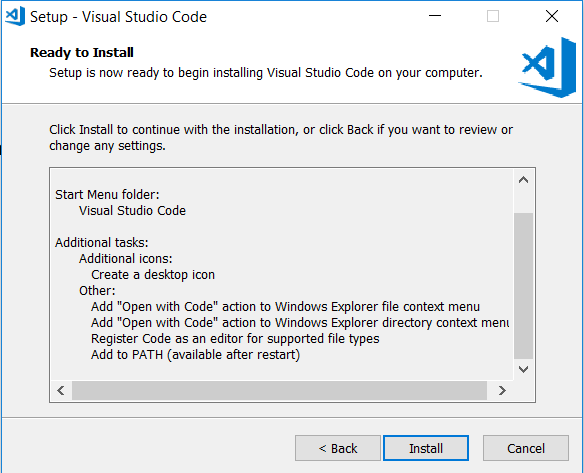
Cho phép Visual Studio Code tạo Menu Folder và chọn **Next**để tiếp tục.



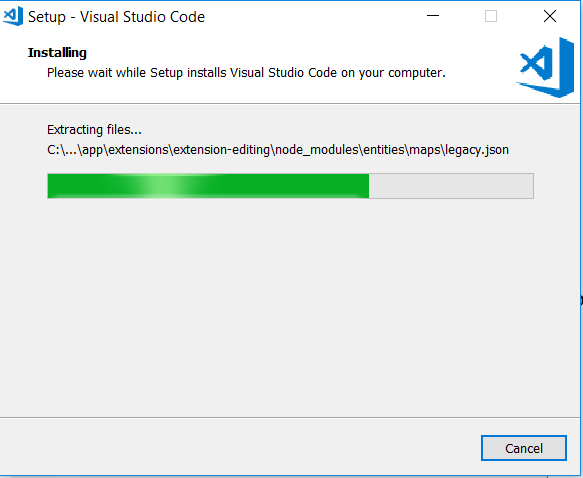
Các lựa chọn cài đặt thêm, chọn **Next**để tiếp tục.



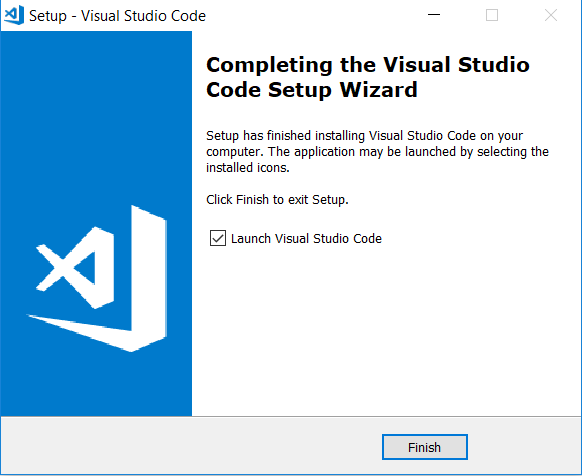
Xác nhận lần cuối các lựa chọn trước khi bắt đầu quá trình cài đặt. Chọn **Install**để bắt đầu quá trình cài đặt.



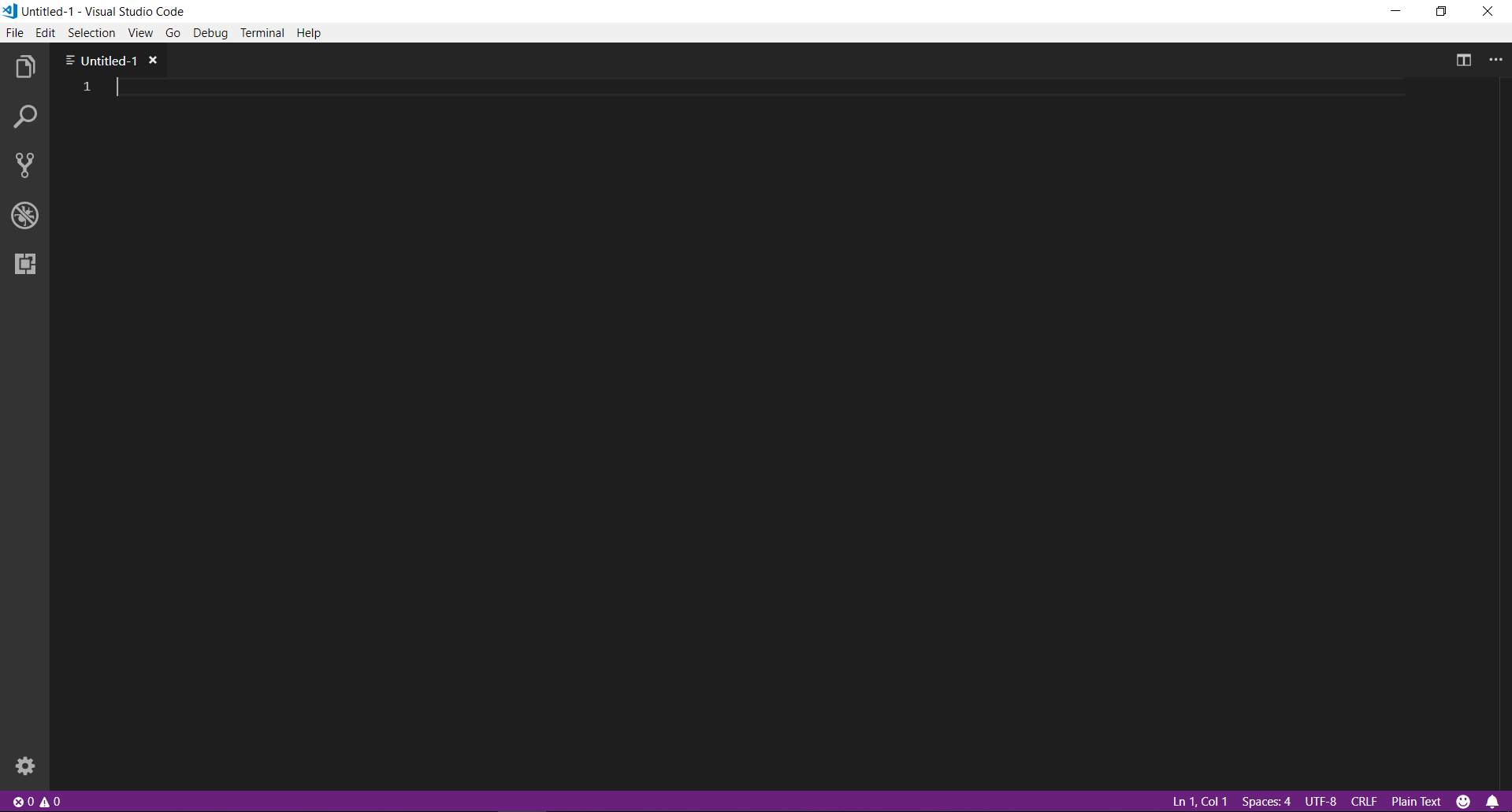
Quá trình cài đặt đang diễn ra:



Chọn **Finish**để hoàn tất quá trình cài đặt, ta cho phép Visual Studio Code chạy sau khi hộp thoại này được đóng lại.



Giao diện làm việc của Visual Studio Code.

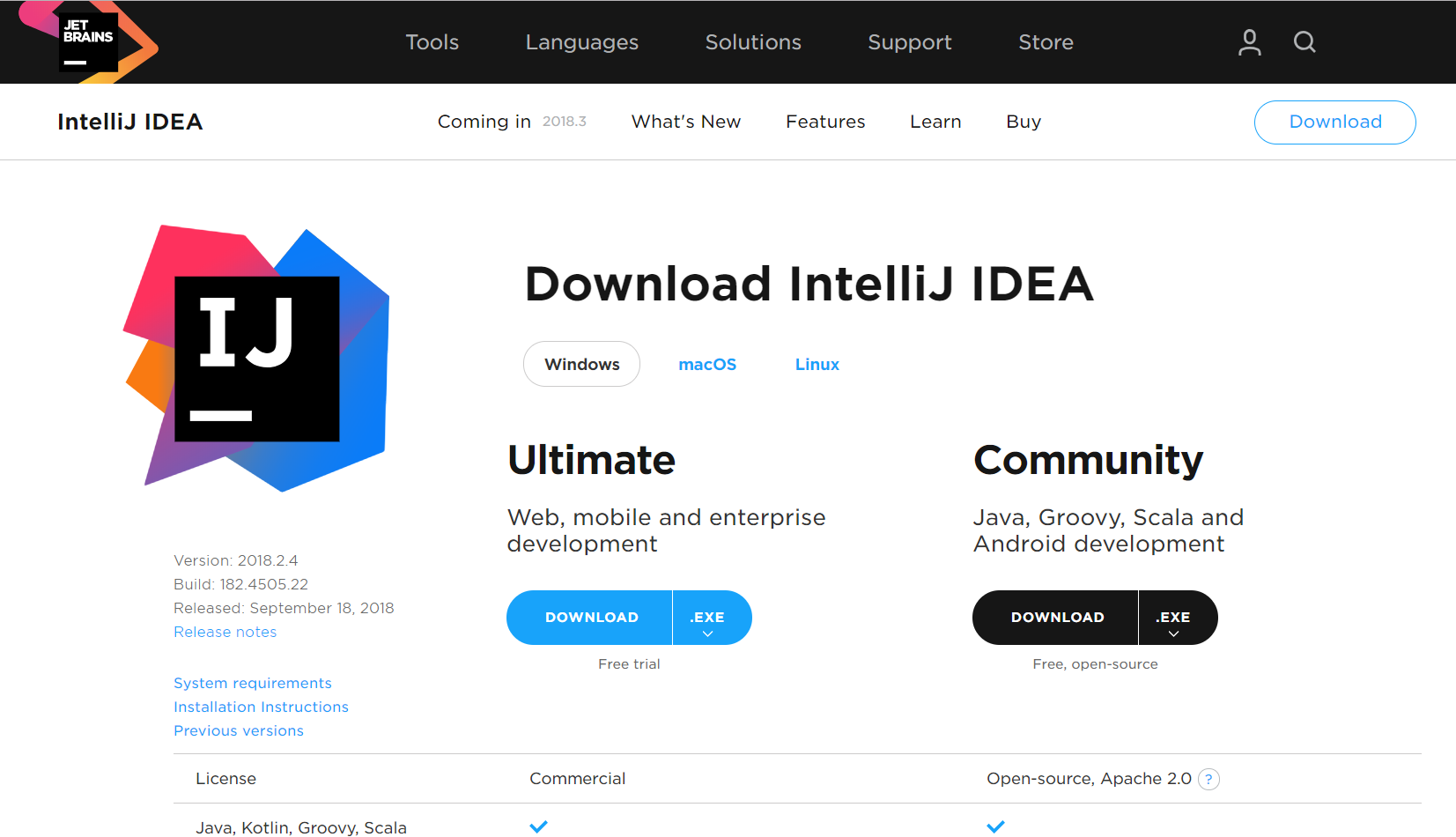


**1.4.2. IntelliJ**

**Hướng dẫn cài đặt**

Download file cài của IntelliJ IDEA tại trang chính thức của Jetbrains. Địa chỉ download: <https://www.jetbrains.com/idea/download/>

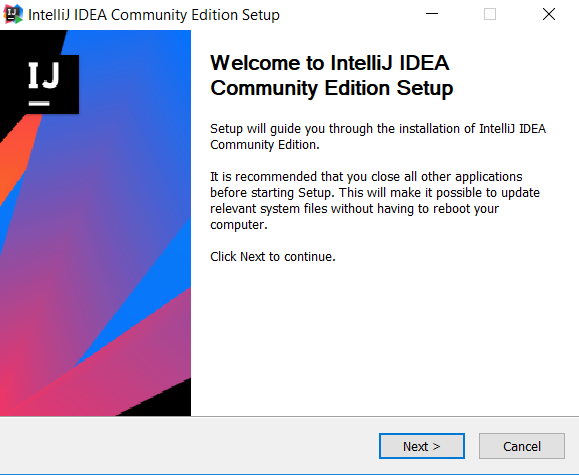
Ở mục Community bấm nút download



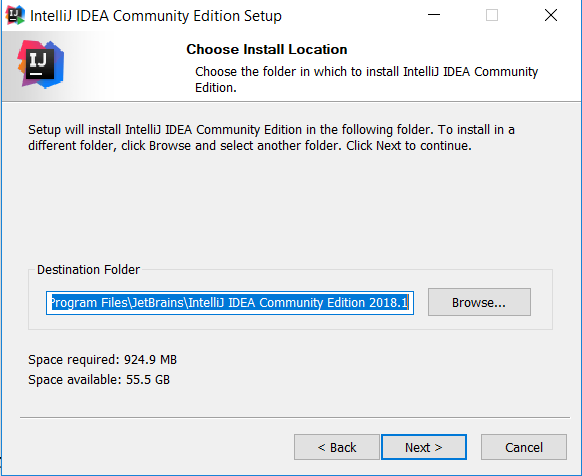
Sau khi quá trình tải hoàn tất, chạy file cài đặt.



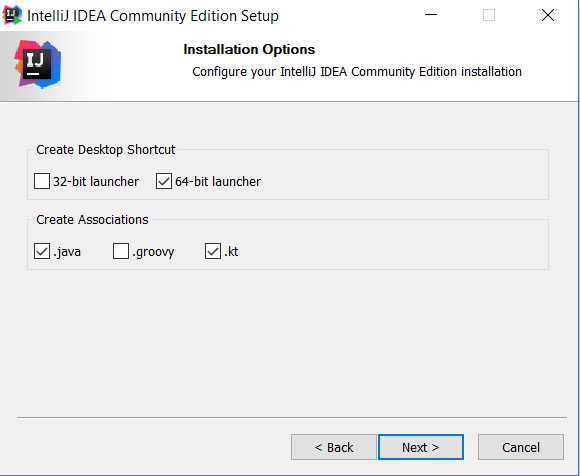
Trình cài đặt IntelliJ xuất hiện, chọn **Next**để bắt đầu quá trình cài đặt.



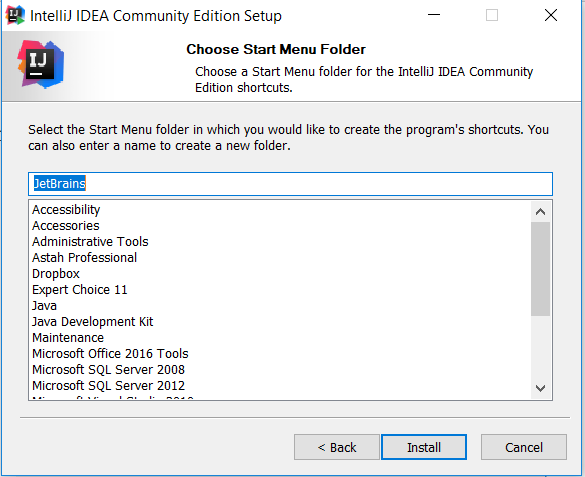
Tại bước này ta sẽ lựa chọn vị trí cài đặt sau đó chọn **Next**.



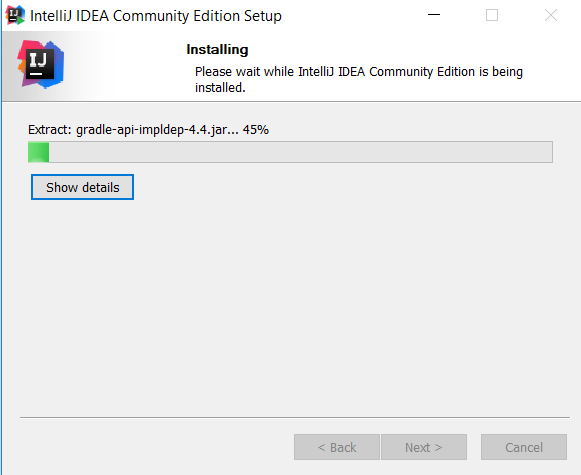
Chọn cấu hình để cài đặt



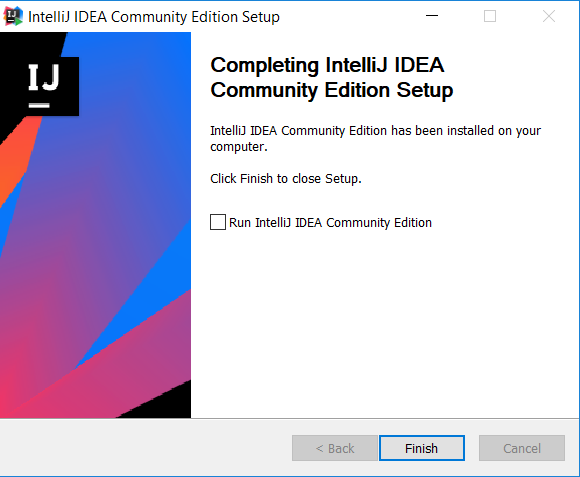
Cho phép tạo Menu Folder và chọn **Install**để tiếp tục.



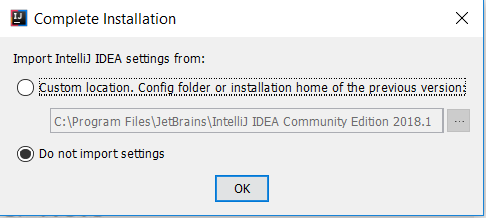
Quá trình Cài đặt đang diễn ra



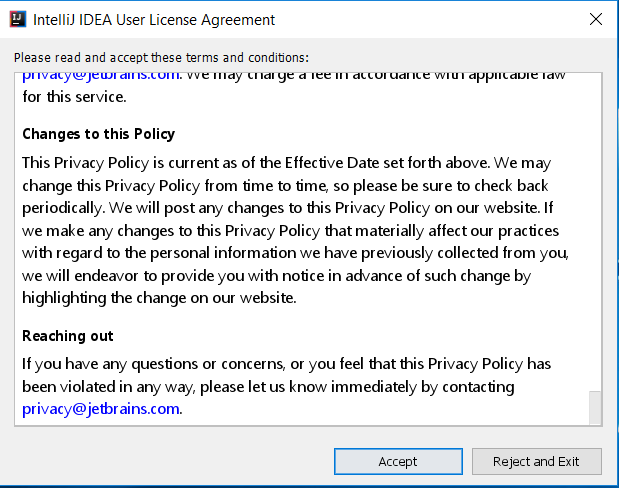
Chọn **Finish**để hoàn tất quá trình cài đặt, ta cho phép Visual Studio Code chạy sau khi hộp thoại này được đóng lại.



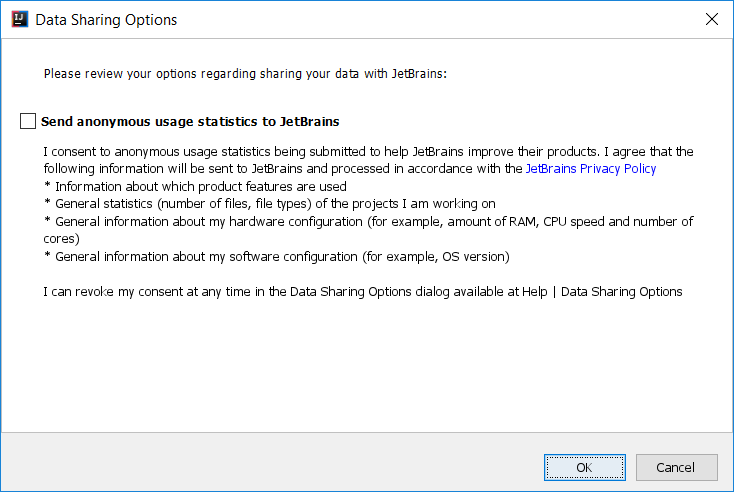
Cài xong chạy IDE lên ta sẽ có thế này, tick chọn **Do not import settings** để sử dụng cấu hình mặc định của IntelliJ IDEA.



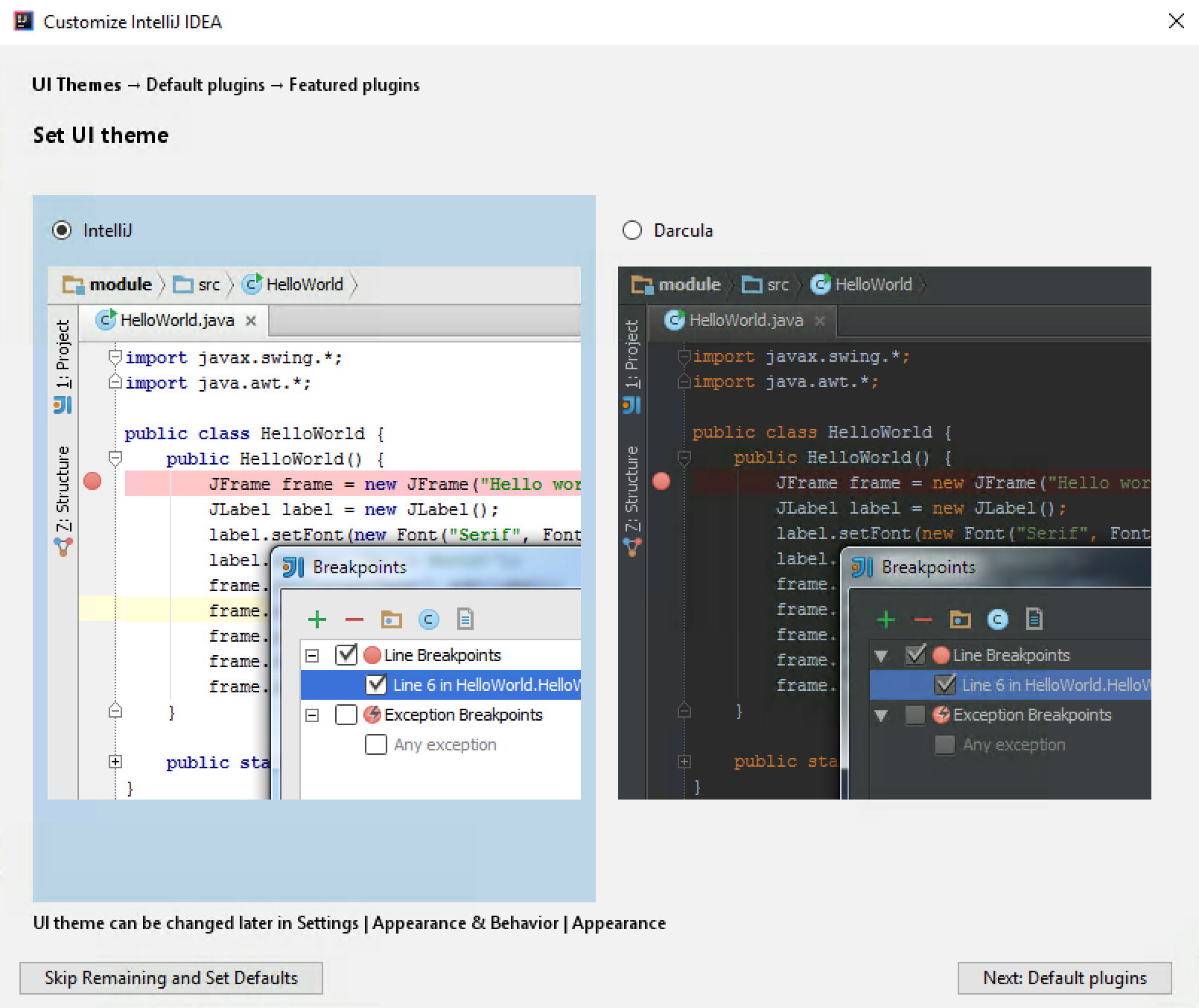
Kéo xuống hết, Chọn **Accept** với các điều khoản của phần mềm để tiếp tục.



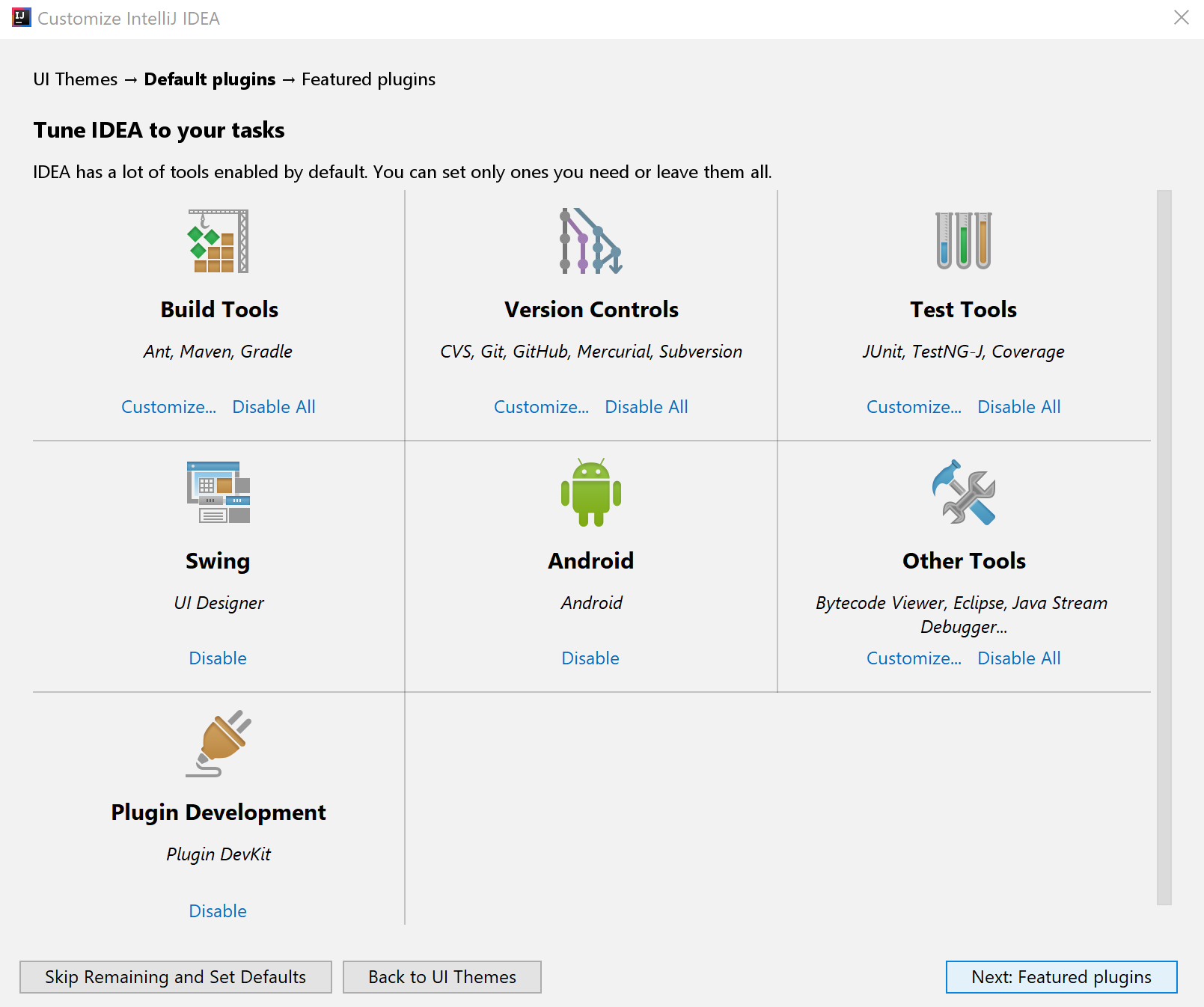
Chọn **OK** để tiếp tục



Chọn theme sáng hoặc tối tùy sở thích, sau đó bấm **Next: Default plugins** để qua bước tiếp theo



Đây là bước cho phép ta disable 1 số plugins (nếu muốn). Sau đó, chon Next: Featured plugins



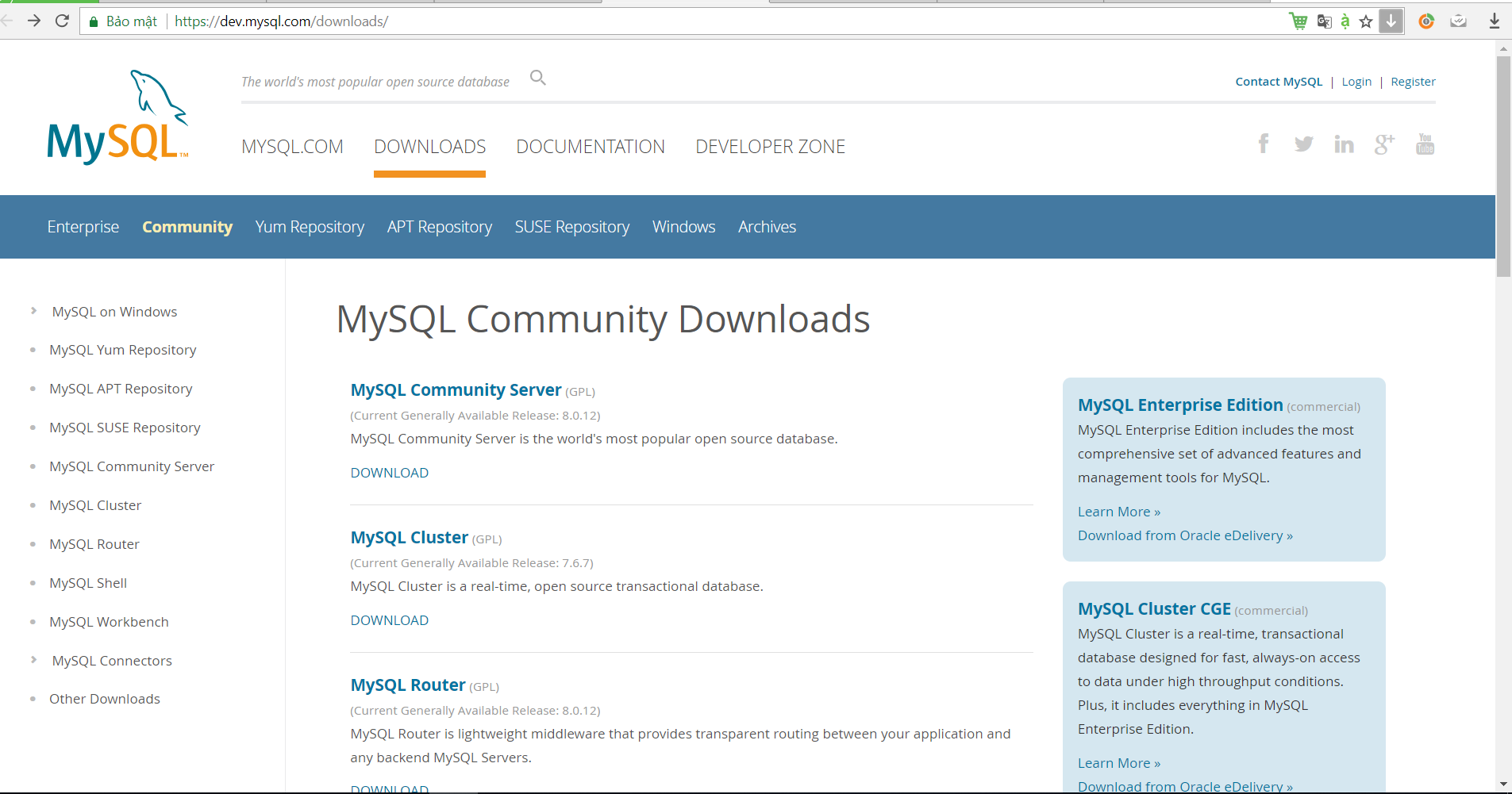
Đây là giao diện đầu tiên khi chạy IDE lên:



**1.4.3. MySQL**

**Hướng dẫn cài đặt**

Để download MySQL, ta vào địa chỉ này: <https://dev.mysql.com/downloads/>



**CHƯƠNG 2**

**PHÂN TÍCH HỆ THỐNG WEBSITE HỌC TIẾNG ANH ONLINE**

**2.1. Giới thiệu sơ lược về nội dung trang web**

Website học tiếng anh được xây dựng nhằm giúp người học tối ưu hoá lại các kĩ năng tiếng anh từ cơ bản đến nâng cao. Mục tiêu của website là giúp lấy lại cho các bạn trình độ tiếng anh cơ bản, đặc biệt là những người mất gốc, lâu ngày không học, khó khăn trong việc không biết bắt đầu từ đâu. Website xây dựng trên nền tảng những từ vựng, những câu nói tiếng anh, ngữ pháp thông dụng dành cho giao tiếp, cho những kì thi cụ thể. Các bài tập sẽ được tính điểm và lặp lại dựa trên số lần đúng sai trong quá khứ, giúp người học tiết kiệm thời gian học những từ thuộc rồi.

Website gồm có 4 tính năng: học từ vựng, luyện nói, luyện nghe và học ngữ pháp.

**Học từ vựng tiếng anh:**

Các bài học từ vựng trên website được tích hợp theo từng chuyên mục cụ thể. Nổi bật là 5000 từ vựng thông dụng được website lọc từ những từ vựng thông dụng nhất dùng trong giao tiếp.

Ngoài ra còn vô số chủ đề từ vựng riêng biệt, từ vựng cho các kì thi (TOEIC, IELTS, TOEFL, SAT).

**Luyện nói tiếng anh:**

Hơn 50.000 cặp câu tiếng anh được xây dựng trên website theo từng bài học. Đây là những câu nói được xây dựng tên nền tảng tiếng anh giao tiếp từ những câu cơ bản cho đến nâng cao.

**Luyện nghe tiếng anh:**

Phần luyện nghe trên website được tích hợp các bài nghe thông dụng theo từng chủ đề cụ thể.

**Học ngữ pháp tiếng anh:**

Phần ngữ pháp hiện tại bao gồm hơn 50 chủ điểm ngữ pháp thông dụng, được xây dựng dưới dạng slide. Phần bài tập sẽ được cập nhật trong thời gian sớm nhất.

Ngoài ra, sẽ còn rất nhiều tính năng được website phát triển trong quá trình làm.

**2.2. Đặc tả**

Website học tiếng anh này nhằm giúp người học tối ưu hoá lại các kĩ năng tiếng anh từ cơ bản đến nâng cao. Website gồm có 4 tính năng: học từ vựng, luyện nói, luyện nghe và học ngữ pháp. Mục tiêu của website là giúp lấy lại cho các bạn trình độ tiếng anh cơ bản, đặc biệt là những người mất gốc, lâu ngày không học, khó khăn trong việc không biết bắt đầu từ đâu.

**(1) Nhiệm vụ cơ bản**

Admin làm nhiệm vụ quản lý người dùng, quản lý các bài học cũng như thực hiện được tất cả các chức năng trong website.

Người dùng đăng nhập (hoặc không) vào hệ thống để học các bài học, làm bài test kiểm tra, cập nhật tài khoản hay đặt câu hỏi để các admin trả lời.

**(2) Cơ cấu tổ chức và sự phân công trách nhiệm**

Website được đăng nhập bởi hai vai trò là admin và người dùng, hoạt động độc lập với nhau.

**Admin** quản lý việc đăng nhập cũng như việc đăng kí tài khoản hay lấy lại mật khẩu của người dùng, đảm nhận việc thêm, sửa hoặc xóa bài học để phù hợp để đáp ứng nhu cầu học người dùng. Admin cũng quản lý luôn những thông tin cá nhân của người dùng, quản lý điểm thi, lịch sử học cũng như lịch sử làm bài test của học viên.

**Người dùng** đăng nhập (hoặc không) để học hay làm bài test, cập nhật profile, hỏi đáp với admin hay bình luận, đánh giá về bài học.

**(3) Quy trình xử lý và các dữ liệu xử lý**

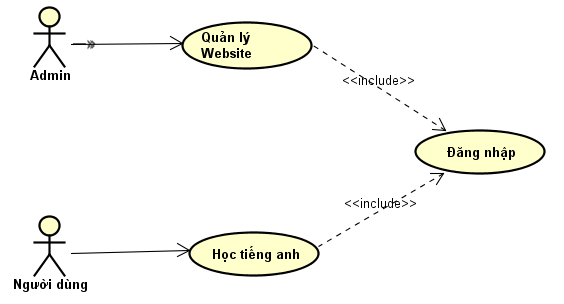
Để đăng bài học, bài test hay tin tức, Admin phải vào trang web quản lý và đăng nhập vào đó bằng tài khoản của mình. Sau khi đăng nhập vào trang quản lý, Admin có thể quản lý được tất cả các tài khoản người dùng đã đăng nhập vào, có thể xóa những người dùng gây spam cho trang web của mình. Admin hiện việc đăng bài học hay các tin tức liên quan, quản lý việc truy cập cũng như lịch sử học của người dùng.

Người dùng có thể xem các tin tức, giới thiệu về tang web mà không cần đăng nhập. Nếu muốn học các bài học, học viên phải đăng nhập bằng tài khoản google để có thể xem bài, làm các bài test, bình luận cũng như đánh giá về bài học.

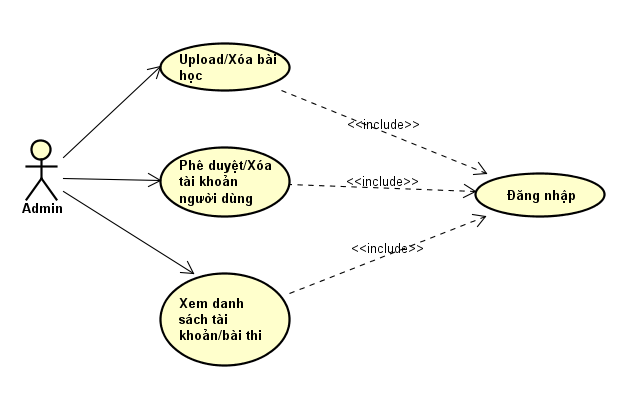
Khi Admin hay người dùng không có nhu cầu sử dụng nữa thì có thể đăng xuất ra để tránh tình trạng người khác truy cập vào gây mất mác thông tin của mình.

**3. Use Case Diagram**

Use Case học tiếng anh online:



Use Case Quản lý Website:



Use Case Học tiếng anh:

