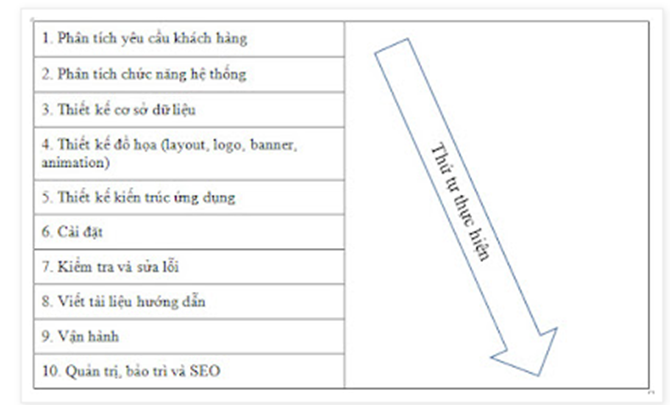
**1.1 Làm web là làm gì?**

**làm web là làm ra sản phẩm mà nó cung cấp các tài nguyên web cho người dùng. Sản phẩm này chính là các trang web được hiển thị trong các trình duyệt web (hay gọi tắt là trình duyệt).**

**1.1.1 Các công đoạn căn bản để làm web**

**Để làm được một sản phẩm trong lĩnh vực web (cũng như một sản phầm phần mềm nói chung) sẽ có nhiều cách làm khác nhau, ví dụ, bạn có thể thực hiện theo các công đoạn căn bản sau:**



1.1.1 Một số vị trí nấc thang nghề nghiệp

| **Fresher/Junior developer**  Là các bạn sinh viên thực tập, sinh viên mới ra trường, sẽ được công ty đào tạo thêm về quy trình làm việc, văn hóa công ty, các kĩ năng, kiến thức, công nghệ, để có thể làm việc cùng mọi người trong công ty. Công việc thường là tìm hiểu dự án hiện tại, làm một số chức năng nhỏ, sửa một số lỗi, làm việc dưới sự trợ giúp của những người có kinh nghiệm. | |
| --- | --- |
| **Developer**  Là người đã có kinh nghiệm làm việc từ một tới ba năm, đã làm qua một số dự án. Sau vị trí developer, có thể lựa chọn để đi theo một trong hai hướng sau: | |
| **Hướng quản lý** | **Hướng kĩ thuật** |
| **Team leader**  Là người quản lý một nhóm nhỏ (3-6 người), cùng lập trình với các thành viên trong nhóm, họp với cấp trên, báo cáo với khách hàng, quản lý các thành viên. | **Senior developer**  Là người hiểu sâu về công nghệ, quy trình, đưa ra được các giải pháp, hướng dẫn cho người mới, xem xét và đánh giá mã nguồn của các thành viên. |
| **Project manager**  Là người làm công việc quản lý, lãnh đạo, báo cáo, phỏng vấn tuyển người cho dự án. Chịu trách nhiệm về sự thành công hay thất bại của dự án. | **Technical lead**  Là người có hiểu biết sâu và rộng về công nghệ, quy trình; lựa chọn công nghệ, thiết kế, quy trình cho dự án. |
| **Manager**  Là người quản lý cấp cao, công việc hàng ngày là: giao việc, phỏng vấn, họp, trao đổi công việc với các bộ phận, phòng ban. | **Software architect**  Là người có kinh nghiệm từ mười đến hai mươi năm trong nghề, có khả năng đánh giá giải pháp, lập nhóm, đưa ra lựa chọn về kiến trúc hệ thống. |
| **Một số công việc khác**  **Sales:** Người bán hàng  **BrSE:** Kĩ sư cầu nối  **Business Analyst:** chuyên viên phân tích nghiệp vụ  **CTO (Chief Technical Officer):** Giám đốc kĩ thuật, chịu trách nhiệm chính về toàn bộ các vấn đề kĩ thuật, công nghệ, xu hướng, đào tạo, tuyển dụng của công ty. | |

Trong lập trình web, thường có ba hướng đi sau:

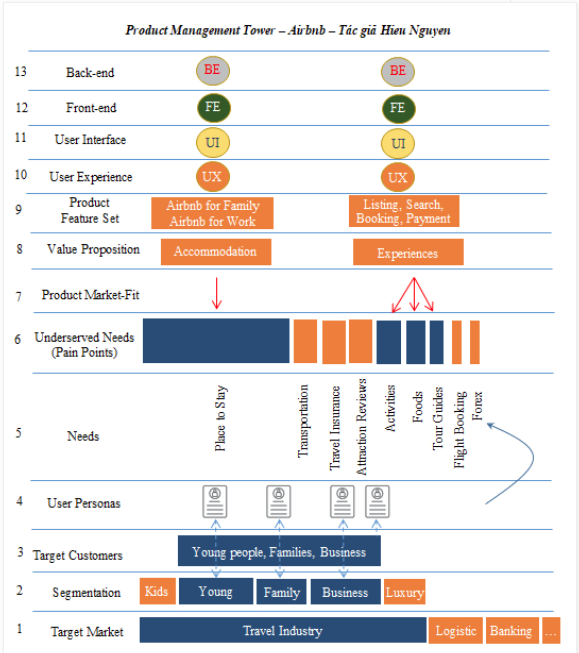
| **Tên công việc** | **Mô tả** |
| --- | --- |
| Lập trình front-end | Lập trình các phần liên quan đến giao diện người dùng. Nếu coi trang web như một ngôi nhà, thì front-end chính là lớp sơn, là mặt tiền của ngôi nhà. |
| Lập trình back-end | Lập trình các phần liên quan đến xử lý logic, xử lý dữ liệu. Nếu coi trang web như một ngôi nhà, thì back-end chính là hệ thống cốt thép, bê tông, điện, nước của ngôi nhà. |
| Full-stack | Làm việc với cả hai vai trò: lập trình front-end và lập trình back-end. Ngoài ra, còn làm việc với vai trò của một người thiết kế (designer), người phát triển và triển khai (DevOps là viết tắt của Development & Operations) |

Khi đi làm, việc lựa chọn theo front-end, back-end hay full-stack phụ thuộc vào tính người, có người thích tìm hiểu rộng, có người thích tìm hiểu sâu, có người rất giỏi: vừa có khả năng hiểu rộng lại vừa có khả năng hiểu sâu. Vậy, tùy sở thích và khả năng để lựa chọn hướng đi, quan trọng là phải hiểu và làm thật chuyên nghiệp trong công việc của mình

### **1.1.3** **Tham khảo quy trình để làm ra một sản phẩm phần mềm**

Phần này lược thuật lại từ bài nói chuyện về Product Management for Managers, của tác giả Hieu Nguyen (link bài nói chuyện ở phần Xem và đọc thêm).

Theo tác giả, để làm ra một sản phẩm hiệu quả (web cũng là một sản phẩm), có xác suất thành công cao thì cần làm theo các bước trong Tháp quản lý sản phẩm (Product Management Tower). Tác giả minh họa bằng sản phẩm cho Airbnb (giả định).



– Target Market: thị trường mục tiêu mà chúng ta muốn hướng đến, nó dựa trên hai yếu tố là thị trường mà chúng ta am hiểu và độ lớn của nó. Vì hiểu thị trường nên biết nó đang cần gì? Biết mình phải làm gì để đáp ứng nhu cầu của thị trường. Chúng ta cũng cần biết được độ lớn của thị trường để đảm bảo khả năng sinh lời của sản phẩm.

– Segmentation: lựa chọn phân khúc khách hàng. Chia thị trường thành các phân khúc khách hàng khác nhau. Ví dụ: trẻ em, giới trẻ, gia đình, thương nhân, nhà giàu. Sau đó loại đi các phân khúc mà chúng ta không muốn phục vụ hoặc không thể phục vụ (loại đi những nhóm khách hàng không có tiềm năng).

– Target Customers: sau khi lựa chọn được phân khúc khách hàng, chúng ta sẽ có Tập khách hàng tiềm năng.

– User Personas: từ Tập khách hàng tiềm năng, chúng ta sẽ đưa ra được bản mô tả về khách hàng hay Chân dung khách hàng (profile). Đây là bản mô tả chung về một nhóm người dùng mục tiêu của một sản phẩm. Tập trung vào hành vi và mục tiêu của khách hàng khi sử dụng sản phẩm của chúng ta.

– Needs: thực hiện phỏng vấn, khảo sát, để biết được các nhu cầu của khách hàng mục tiêu (ví dụ người đi du lịch sẽ cần một nơi để ở, phương tiện đi lại, bảo hiểm, các đánh giá về địa điểm họ sẽ tới, các hoạt động, ăn, hướng dẫn viên du lịch, đặt vé máy bay, đổi tiền).

– Underserved Needs (Pain Points): sắp xếp các nhu cầu của khách hàng mục tiêu, từ quan trọng nhất đến ít quan trọng hơn, tìm ra được những nhu cầu khách hàng cần, nhưng chưa được phục vụ tốt bởi các dịch vụ đang có. Cách làm: loại đi các dịch vụ đã được phục vụ tốt hoặc các dịch vụ mà không phải là thế mạnh của chúng ta.

– Product Market-Fit: sản phẩm thị trường đang cần, và người dùng cảm thấy hài lòng khi sử dụng sản phẩm đó.

– Value Proposition: đưa ra các dịch vụ mà khách hàng cần, nhằm giúp họ được sử dụng các dịch vụ tốt hơn những cái đang có.

– Product Feature Set: cần sản phầm gì? sản phầm gồm các chức năng gì? để đáp ứng nhu cầu của khách hàng.

– User Experience: không chỉ là trải nghiệm người dùng, mà quan trọng hơn là đáp ứng được nhu cầu của người dùng.

– User Interface: giao diện ứng dụng

– Front-end: làm phần front-end của ứng dụng

– Back-end: làm phần back-end của ứng dụng

Như vậy, qua mô hình Tháp quản lý sản phẩm, bạn đã biết được công việc cụ thể của một người làm về UX, UI, front-end, back-end. Họ sẽ làm việc ở giai đoạn nào trong quá trình tạo ra một sản phẩm phần mềm nói chung, và ứng dụng web nói riêng.

## **1.1** **Muốn làm web cần học những gì**

Muốn làm web cần học tiếng Anh chuyên ngành và chuyên môn.

Sau đây là một cách để có thể học tiếng Anh chuyên ngành:

– Chọn một chủ đề mà bạn quan tâm (về HTML chẳng hạn)

– Đọc sách/tài liệu tiếng Việt về HTML trước, chọn cuốn sách/tài liệu nào mà tác giả có ghi các thuật ngữ tiếng Anh đi kèm là tốt nhất

– Đọc hiểu và làm theo sách/tài liệu tiếng Anh về HTML

– Nếu được, nên tập dịch tài liệu từ tiếng Anh ra tiếng Việt

| **Kiến thức** | **Mô tả chi tiết** |
| --- | --- |
| Kiến thức chung (dù theo front-end hay back-end) | – Kiến thức nền về Hệ điều hành, Kĩ thuật lập trình, Cấu trúc dữ liệu và thuật toán, Lập trình hướng đối tượng, Cơ sở dữ liệu  – Kiến thức nền về WWW, Mạng máy tính  – Kĩ năng và thái độ về khía cạnh kĩ thuật (cách viết mã, lựa chọn giải thuật, tổ chức mã nguồn chuyên nghiệp), cũng như khía cạnh nghề nghiệp (cách làm việc chuyên nghiệp, hiệu quả và hạnh phúc với nghề)  – Phần mềm Quản lý phiên bản/quản lý mã nguồn: Git, SVN, TFS  – AJAX / Web API  – RESTful API / HTTP method  – Cơ bản về lập trình front-end và back-end |
| Front-end | – HTML, CSS, JavaScript  – Các thư viện và framework phổ biến (ví dụ: jQuery, Bootstrap, AngularJS, ReactJS, VueJS)  – Responsive design, UX/UI  – Các kiến thức nâng cao (ví dụ: mô hình MVC, design pattern, lập trình bất đồng bộ trong JavaScript) |
| Back-end | – Một trong các ngôn ngữ phổ biến: Java, C#, PHP, Ruby, Python, JavaScript, Go  – Cơ sở dữ liệu (SQL, mySQL, MongoDB)  – Mô hình MVC, Design pattern  – Các framework của mỗi ngôn ngữ  – Các kiến thức nâng cao (ví dụ Dependency Injection, Inversion of Control, SOLID)  – Các kiến thức về triển khai, bảo mật |

Phương pháp học là:

– Học từ dễ đến khó, học chắc từng khái niệm cơ bản, khi có cái nền tốt rồi, cứ từng bước bồi đắp thêm các kiến thức mới, mọi thứ rất logic và nhẹ nhàng

– Bắt đầu là HTML, CSS, JavaScript, một vài thư viện/framework phổ biến (ví dụ: jQuery, Bootstrap, AngularJS, VueJS, ReactJS)

– Sau đó là một ngôn ngữ back-end nào đó cùng với cơ sở dữ liệu tương ứng (Java, JavaScript, C#, PHP, Python, Ruby, Go, mySQL, SQL), một vài framework phổ biến

– Mục tiêu là hiểu được cách thức để làm được một sản phẩm web hoàn chỉnh (đây chính là học front-end và back-end cơ bản)

– Trong quá trình học sẽ gặp một số khái niệm, mô hình, cố gắng tìm hiểu để nắm bắt

– Làm thực hành thật nhiều, tự làm các dự án từ dễ đến khó

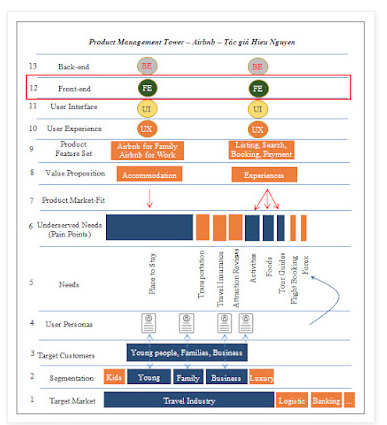
– Học từ các trang dạy làm web tại Facebook/Github/Youtube/Website/Blog/Tik tok hoặc học từ chia sẻ của các anh chị trong nghề

– Cuối cùng, quan trọng nhất là phải kiên trì, chăm chỉ, “cứ đi ắt sẽ đến!”

1 LỘ TRÌNH HỌC FRONT END

Hãy học vì làm nghề dựa vào sở trường vì nó có vẻ lâu bền và hợp lý hơn dựa vào đam mê

ngòi ra nhớ thực hành và ứng dụng một cách có hiệu quả.



Theo quy trình trên, sau bước 10, và 11, chúng ta đã có bản thiết kế (UX/UI) của ứng dụng web. Công việc tiếp theo là làm phần front-end và back-end cho ứng dụng.

Làm UI/UX và back-end là các nhánh công việc khác, bạn có thể tự tìm hiểu thêm. Phần này sẽ giúp bạn học để làm phần front-end của một ứng dụng web (bước 12).

Hiểu một cách đơn giản, làm web front-end là việc chuyển đổi từ bản thiết kế (UI/UX) sang HTML, CSS và JavaScript.

Một cách bài bản, để làm phần front-end, các bạn phải học hết Kiến thức đại cương, và Kiến thức cơ sở ngành. Một số chủ đề cần học như:

| Tiếng Anh căn bản & chuyên ngành CNTT | Nhập môn CNTT | Nhập môn lập trình |
| --- | --- | --- |
| Kỹ thuật lập trình | Phương pháp lập trình hướng đối tượng | Kỹ năng mềm |
| Cấu trúc dữ liệu và thuật giải | Cơ sở dữ liệu | Hệ điều hành |
| Kiến trúc máy tính và hợp ngữ | Mạng máy tính | Phần mềm Quản lý phiên bản/quản lý mã nguồn: Git, hoặc SVN, hoặc TFS |
| Kĩ năng và thái độ liên quan đến kĩ thuật: cách viết mã, lựa chọn giải thuật, tổ chức mã nguồn chuyên nghiệp. Kĩ năng và thái độ liên quan đến môi trường làm nghề: cách làm việc chuyên nghiệp, hiệu quả và hạnh phúc với nghề. | | |

Nếu vì lý do gì đó, bạn không thể học tuần tự các chủ đề có tính nền tảng như trên, thì bạn vẫn có thể bắt đầu ngay vào việc học để làm web front-end. Trong quá trình học, bạn thấy thiếu kiến thức gì thì học bổ sung.

| Web căn bản | – Hoạt động của Internet  – Giao thức HTTP  – Trình duyệt web  – DNS  – Tên miền (domain name)  – Lưu trữ website (hosting) |
| --- | --- |
| HTML | – HTML căn bản  – Forms và validation  – Quy ước đặt tên và kinh nghiệm viết mã |
| CSS | – CSS căn bản  – Các kĩ thuật tạo bố cục trang (layout): float, positioning, display, box model, CSS grid, flex box  – Responsive và media queries |
| JavaScript | – JavaScript căn bản  – DOM  – Fetch API, Ajax  – ES6+  – Hoisting, event bubbling, scope, prototype, shadow DOM, strict |
| Quản lý phiên bản | – Git căn bản  – Github, hoặc Gitlab, hoặc Bitbucket |
| Chương trình quản lý gói | – Npm, hoặc yarn, hoặc pnpm |
| CSS architecture | – BEM |
| CSS preprocessor | – SASS  – PostCSS |
| Build tools | – Linters and formatters  – Task runners (npm scripts)  – Module bundlers (vite, hoặc esbuild) |
| Framework | – React, hoặc Angular, hoặc Vue |
| Modern CSS | – Styled components  – CSS modules |
| CSS framework | – Bootstrap  – Tailwind |
| Kiểm thử phần mềm (Testing your Apps) |  |
| *… (còn nữa)* |  |

Các bạn nên học tuần tự các chủ đề ở bảng trên, theo thứ tự từ trên xuống dưới. Nếu không, sẽ rất khó hiểu, dễ dẫn tới chán nản và bỏ cuộc.

Bạn cũng nên để ý, nếu đọc nhiều trên mạng sẽ thấy có nhiều lộ trình học. Mỗi lộ trình có sự khác biệt nhau một chút. Điều này cũng dễ hiểu, bởi để đi tới đích thì luôn có nhiều hơn một con đường. Vì vậy, hãy cứ chọn một lộ trình mà bạn thấy thích, và học theo nó. Trong quá trình học sẽ linh hoạt học thêm các chủ đề mà bạn thấy cần, thấy hữu ích.