دورة تطوير الويب

الملف الأول

اعداد: م. محمد أسامة بيطار

ما هو الويب؟

الويب هو نظام عالمي للمعلومات المتصلة عبر شبكة الإنترنت. يعتبر الويب بمثابة مجموعة من الصفحات والمواقع والتطبيقات التي يمكن الوصول إليها عبر المتصفحات الإلكترونية.

كيف تعمل صفحات الويب؟

صفحات الويب تعمل بناءً على نموذج يسمى نموذج العميل-الخادم (Client-Server Model). وهذا يعني أن هناك اتصال وتفاعل بين متصفح الويب الذي يعمل على جهاز العميل (مثل الكمبيوتر أو الهاتف الذكي) والخادم الذي يحتوي على الموقع الويب.

عندما يقوم المستخدم بفتح صفحة ويب في متصفحه، يتم إرسال طلب من المتصفح إلى الخادم للحصول على المحتوى المطلوب (مثل صفحة HTML). يستخدم المتصفح بروتوكول (Hypertext Transfer Protocol: HTTP) لإرسال الطلب إلى الخادم.

عند استلام الخادم للطلب، يبحث عن الملف المطلوب ويقوم بإرساله إلى المتصفح كجزء من الاستجابة. يستخدم الخادم لغات الويب مثل HTML وCSS وJavaScript لبناء صفحات الويب وتنسيقها وإضافة وظائف تفاعلية.

عندما يتلقى المتصفح الملفات، يقوم بتفسيرها وعرض المحتوى للمستخدم. يترجم المتصفح العناصر في صفحة HTML إلى عرض مرئي يمكن للمستخدم رؤيته والتفاعل معه، مع تطبيق التنسيقات المحددة في CSS. كما يستخدم المتصفح لغة JavaScript لتنفيذ البرامج النصية التفاعلية والتحكم في سلوك الصفحة.

بالإضافة إلى ذلك، قد يتم استرداد بيانات إضافية من الخادم بواسطة المتصفح بواسطة طلبات إضافية، مثل الصور أو مقاطع الفيديو أو البيانات المرتبطة.

عملية تطوير الويب:

تطوير الويب يشير إلى عملية بناء وتطوير مواقع الويب وتطبيقاتها. يشمل هذا العملية العديد من الأنشطة والمهارات التقنية التي تهدف إلى إنشاء وتحسين وصيانة المواقع الويب. فيما يلي نظرة عامة على بعض جوانب تطوير الويب:

1. تصميم الواجهة الأمامية (Front-end Development): يتعلق بتطوير جزء المستخدم الظاهري للموقع الذي يتفاعل المستخدم معه مباشرة. يشمل ذلك استخدام

لغات برمجة مثل HTML وCSS وJavaScript لبناء صفحات الويب وتنسيقها وتوفير تجربة مستخدم مريحة وجذابة.

2. تطوير الجانب الخلفي (Back-end Development): يتعلق ببناء الأجزاء التي تدير عمليات الموقع وتتعامل مع قواعد البيانات والمنطق البرمجي. يشمل ذلك استخدام لغات برمجة مثل PHP، Python، Ruby، Java أو .NET وإطارات العمل المرتبطة بها لتطوير وتشغيل الخوادم والمنطق الخلفي.

3. قواعد البيانات (Database): يشمل تصميم وإنشاء وإدارة قواعد البيانات التي تخزن وتنظم المعلومات المتعلقة بالموقع، مثل المستخدمين والمحتوى والتفاصيل الأخرى. يمكن استخدام نظم إدارة قواعد البيانات مثل MySQL، PostgreSQL، MongoDB وغيرها.

4. تجربة المستخدم (User Experience - UX): يتعلق بتحسين تفاعل المستخدم مع الموقع أو التطبيق، بحيث يكون التصميم سهل الاستخدام ومريح للمستخدمين. يشمل ذلك تحليل استجابة المستخدمين وتصميم واجهات مستخدم فعالة وعملية.

5. الأمان والأداء: يشمل تطوير الويب أيضًا الاهتمام بأمان الموقع وحماية البيانات الحساسة، بالإضافة إلى تحسين أداء الموقع وسرعته في التحميل والاستجابة.

ملاحظة: يجب أن نشير إلى أن كل جانب من الجوانب المذكورة في الفقرة السابقة يشكل بحد ذاته عالمًا متعلقًا بالعلوم المختلفة. على سبيل المثال، تطوير الواجهة الأمامية يتطلب معرفة عميقة بتقنيات HTML وCSS وJavaScript، بينما تطوير الجانب الخلفي يستدعي فهماً متقدماً للغات البرمجة وإطارات العمل. علاوة على ذلك، تصميم قواعد البيانات يتضمن معرفة بنماذج البيانات والاستعلامات، وتحسين تجربة المستخدم يستلزم دراسة عميقة في علم النفس وتصميم الواجهات.

لذا، يمكن اعتبار تطوير الويب بأكملها كبحر من العلوم المترابطة، حيث يجب على المطورين أن يكتسبوا معرفة متعددة الجوانب للعمل بكفاءة في هذا المجال. لذلك، قد يحتاج المطورون إلى الاستفادة من مختلف المصادر التعليمية والتدريبية المتاحة لتعلم وتطوير المهارات المطلوبة في كل جانب من جوانب تطوير الويب.

كيف نحصل على رباط بين الجوانب السابقة؟

عندما يتعلق الأمر بترابط عملية برمجة الواجهة الأمامية (Front-end) والجانب الخلفي (Backend) في تطوير الويب، هناك بعض الأساليب التي يمكن اتباعها للتحقيق بنجاح تلك الربط. إليك خطوات عامة لتوضيح العملية:

1. تحديد وتوثيق واجهة البرمجة (Application Programming Interface: API): الهدف منها هو عملية تقوم بالربط بين الواجهة الأمامية front-end والجانب الخلفي back-end. يجب تحديد النقاط النهائية (Endpoints) والبيانات المتوقعة للتبادل بين الجانبين.

2. بناء الواجهة الأمامية: يبدأ الفريق في بناء الواجهة الأمامية باستخدام لغات الويب مثل HTML وCSS وJavaScript. يتم استخدام هذه اللغات لتصميم وتنسيق صفحات الويب وتحقيق وظائف تفاعلية. في هذه المرحلة، ال API من الج الاتصال بالنقاط النهاية المحددة والحصول على البيانات المطلوبة من الخادم.

3. تطوير الجانب الخلفي: يتم بناء الجانب الخلفي من التطبيق باستخدام لغات البرمجة وإطارات العمل المناسبة. يتم استخدام وثائق واجهة البرمجة لتعريف نقاط النهاية وتطبيق الوظائف المطلوبة للتفاعل مع قواعد البيانات وتنفيذ العمليات المنطقية.

4. الاتصال بين الجانبين: بمجرد الانتهاء من بناء الواجهة الأمامية والجانب الخلفي، يتم استخدام طرق الاتصال المتاحة مثل استدعاءات API (API calls) أو AJAX للتواصل بين الجانبين.

5. اختبار وتحسين: يتم إجراء اختبار شامل للتأكد من أن الواجهة الأمامية والجانب الخلفي يعملان بشكل صحيح ويتواصلان بشكل صحيح. يتم تحسين الأداء وإصلاح أي أخطاء محتملة في هذه المرحلة.

6. النشر والصيانة: بعد اجتياز مرحلة الاختبار والتحسين، يتم نشر التطبيق بشكل عام على الخوادم المناسبة. وبعد النشر، يتم تتبع وصيانة التطبيق لضمان استمراريته وتحسين أدائه وحل المشكلات التي تنشأ.

بعض الأسئلة المهمة:

هل يجب تعلم كل الجوانب السابقة لكي أستطيع ان أقوم بتطوير موقع الكتروني؟

لا، لا يجب عليك تعلم كل الجوانب السابقة لتطوير موقع إلكتروني.

ماهي المهارات الأساسية التي يجب تعلمها؟

لتطوير موقع إلكتروني، قد تحتاج إلى مهارات برمجية وتقنية أساسية مثل:

1. لغات الويب الاساسية: مثل HTML وCSS لتصميم وتنسيق الصفحات، وJavaScript لإضافة الوظائف التفاعلية والديناميكية.

2. قواعد البيانات: مثل SQL للتعامل مع قواعد البيانات، وتخزين واسترجاع البيانات المرتبطة بالموقع.

3. تطوير الواجهة الأمامية (Front-end Development): يتطلب فهمًا لتصميم المستخدم وتجربة المستخدم، واستخدام أدوات وإطارات عمل مثل React.js أو Angular.js او Vue.js

4. تطوير الواجهة الخلفية (Back-end Development): يتطلب إتقان لغات البرمجة مثل Python أو PHP أو Ruby أو Node.js، وإطارات العمل مثل Django أو Laravel أو Ruby on Rails.

5. أمان الموقع وحماية البيانات: يجب أن تكون على دراية بمفاهيم أمان الويب وحماية البيانات وتنفيذ إجراءات أمان مناسبة.

هل يمكن ان أصبح خبيرا في كل الجوانب الخاصة بعملية تطوير الويب؟

يمكن أن تصبح خبيرًا في مجالات مختلفة من عملية تطوير الويب، ولكن من الصعب أن تصبح خبيرًا في كل الجوانب الخاصة بها. عملية تطوير الويب تشمل نطاقًا واسعًا من المهارات والتقنيات، وكل مجال يتطلب تعلم وتجربة مستقلة.

من أين يجب أن أبدأ في تعلم تطوير الويب؟

بعض الخطوات التي يمكنك اتخاذها:

1. تعلم اللغات الأساسية: يمكنك البدء بتعلم لغات الويب الأساسية مثل HTML تستخدم لبناء هيكل وتنسيق المحتوى، CSS تستخدم لتصميم وتنسيق الواجهة الأمامية، وJavaScript يتيح لك إضافة التفاعل والوظائف الديناميكية لموقعك.

2. تجربة الأدوات والإطارات: قم بتجربة الأدوات والإطارات الشائعة في تطوير الويب مثل Bootstrap Reactg Angularg Vue.jsg. هذه الأدوات تساعدك في بناء وتطوير المشاريع بشكل أكثر فعالية وسرعة.

3. تعلم الواجهة الخلفية: بمجرد أن تتقن الواجهة الأمامية، يمكنك التعمق في تطوير الواجهة الخلفية باستخدام لغات البرمجة مثل Python أو Node.js أو PHP. هذه اللغات تستخدم لبناء الخوادم والتفاعل مع قواعد البيانات.

4. بناء مشاريع تطبيقية: قم ببناء مشاريع تطبيقية صغيرة لتطبيق المفاهيم والمهارات التي تعلمتها. يمكنك بدء المشاريع البسيطة مثل موقع شخصي أو مدونة ومن ثم التحرك إلى مشاريع أكثر تعقيدًا تتطلب تفاعلات أكثر.

5.من اهم الخطوات هي استكشاف المصادر التعليمية عبر الإنترنت: هناك العديد من المصادر التعليمية المجانية والمدفوعة عبر الإنترنت التي تقدم دروسًا ومواد تعليمية حول تطوير الويب. يمكنك الاستفادة من الدروس المباشرة والفيديوهات والمقالات والمنتديات النقاشية.

6.الخطوة الأهم هي الممارسة والتطبيق: لا تنسى أهمية الممارسة والتطبيق العملي. قم ببناء مشاريع وحاول حل التحديات التي تواجهك. كلما قمت بمزيد من التطبيق العملي، ستتعلم وتتقن المزيد.

تذكر أن تعلم تطوير الويب يستغرق الوقت والجهد، فالممارسة المنتظمة والاستمرارية في التعلم هي المفتاح لتحقيق التقدم وتطوير مهاراتك.

المسار المقترح لهذا الكورس:

تقسم الجلسات الى:

جلسة 1: مقدمة

- مفهوم تطوير الويب وأهميته.

- لمحة سريعة عن محتويات الدورة.

جلسة 2-4: HTML

- مفاهيم أساسية في HTML

- تنسيق النصوص والعناوين

- إضافة الصور والروابط

- الجداول والنماذج

جلسة 5-8: CSS

- أساسيات CSS وتنسيق العناصر
- التحكم في الألوان والخطوط
- تخطيط الصفحات والعناصر المرنة

جلسة 9-12 (16 ساعة): JavaScript

- أساسيات البرمجة بلغة JavaScript
- المتغيرات والعمليات الحسابية
- الشروط والحلقات
- إنشاء واستخدام الدوال

جلسة 13-16 (16 ساعة): Node.js

- مقدمة في Node.js ومفهوم السيرفر
- إنشاء سيرفر بسيط باستخدام Node.js و Express.js
- التعامل مع قواعد البيانات باستخدام MongoDB
- إنشاء API والتفاعل معها باستخدام Node.js

جلسة 17-20 (16 ساعة): المشاريع

- تطوير مشروع تطبيقي كامل باستخدام HTML وCSS وJavaScript وNode.js.

المهارات التي سوف يحصل عليها الطالب مع انتهاء الدورة:

عند انتهاء الدورة واستكمال المسار الذي تم ذكره، سيحصل الطلاب على مجموعة من المهارات في تطوير الويب. ستشمل المهارات التي يكتسبها الطلاب ما يلي:

1. إنشاء صفحات ويب قائمة على HTML: ستكون لدى الطلاب القدرة على إنشاء صفحات ويب بناءً على لغة HTML، والتي تمكنهم من هيكلة المحتوى وإضافة الروابط والصور والجداول والنماذج.

2. تنسيق وتصميم المواقع باستخدام CSS: سيتعلم الطلاب كيفية استخدام لغة CSS لتنسيق وتصميم المواقع الإلكترونية، بما في ذلك تحكمهم في الألوان

والخطوط والتخطيطات والانتقالات بالإضافة الى تعلم التعامل مع بعض الأدوات مثل: Bootstrapأو Tailwind.

3. إضافة التفاعل والديناميكية باستخدام JavaScript: سيكتسب الطلاب مهارات في استخدام لغة JavaScript لإضافة التفاعل والوظائف الديناميكية إلى صفحات الويب، مثل التحقق من صحة البيانات المدخلة وتغيير المحتوى بناءً على إجراءات المستخدم.

4. بناء الخوادم والتفاعل مع قواعد البيانات باستخدام Node.js: سيتعلم الطلاب كيفية استخدام Node.js لبناء الخوادم وإنشاء API والتفاعل مع قواعد البيانات، مما يمكنهم من إنشاء تطبيقات ويب قابلة للتوسع والتفاعلية.

5. تطوير مشروع موقع لإدارة المكتبات: سيعمل الطلاب على مشروع موقع لإدارة المكتبات كجزء من الدورة، وهذا سيسمح لهم بتطبيق المفاهيم والمهارات التي تعلموها في المحاضرات السابقة وتحويلها إلى تطبيق عملي قابل للعرض.

باستكمال هذا المسار والمشروع، ستكون لدى الطلاب قاعدة قوية في تطوير الويب وإدارة المشاريع، ويمكنهم استخدام هذه المهارات لبناء مواقع وتطبيقات ويب متقدمة واحترافية.

انشاء البيئة المناسبة للتطوير:

أولا: **تحميل برنامج** VS Code

https://code.visualstudio.com/

ثانيا: بعض الإضافات التي تسهل العمل:

Live server

Auto Close Tag

Prettier - Code formatter

Comment Divider

Bootstrap IntelliSense

node-snippets

ثالثا: تحميل Node JS

https://nodejs.org/en

Hypertext Markup Language

HTML هي اختصار لـ Hypertext Markup Language، وهي لغة توصيف تستخدم لبناء وتنسيق صفحات الويب. تعتبر HTML لغة أساسية في تطوير الويب، حيث يتم استخدامها لتحديد بنية وتنظيم المحتوى على صفحة الويب وتعريف العناصر والعناوين والفقرات والصور والروابط والجداول وغيرها من العناصر التفاعلية.
تستخدم HTML عادة بالاشتراك مع لغات أخرى مثل CSS وJavaScriptg لإضفاء الشكل والتفاعل على صفحات الويب. تستخدم العناصر في HTML علامات معينة تحيط بالنص لتحديد نوع العنصر وتنسيقه وترتيبه في الصفحة.

العناصر الاساسية للصفحة HTML:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Document</title>
</head>
<body>

</body>
</html>
```

صفحة HTML تتكون من مجموعة من العناصر التي تحدد بنية ومحتوى الصفحة.
هنا بعض العناصر الأساسية في صفحة HTML:

القطعة الأولى في الشيفرة هي <DOCTYPE html!> وتستخدم لتحديد نوع المستند وهنا يتم تحديده كمستند HTML.

القطعة الثانية في الشيفرة هي <"html lang="en> وتحدد بداية العنصر الجذر لصفحة HTML. يتم استخدام الخاصية lang لتحديد لغة الصفحة، وفي هذا المثال تم تعيينها للغة الإنجليزية (en).

القطعة الثالثة في الشيفرة هي <head> وتحتوي على المعلومات الاساسية. في هذا المثال تحتوي على عنصر <meta> الذي يعرف ترميز الحروف (charset) وعرض الصفحة على أجهزة مختلفة (viewport)، وعنصر <title> الذي يحدد عنوان الصفحة.

القطعة الرابعة في الشيفرة هي <body> وتحتوي على محتوى الصفحة القابل للعرض. يتم وضع النص والصور والروابط والجداول وغيرها من العناصر التفاعلية داخل هذا العنصر.

القطعة الأخيرة في الشيفرة هي </html> وتحدد نهاية العنصر الجذر لصفحة HTML.

بشكل عام، هذه الشيفرة تشكل هيكل أساسي لصفحة HTML وتتضمن العناصر الأساسية مثل DOCTYPE والعنصر الجذر <html> وعنصر <head> وعنصر <body>. يمكنك وضع المحتوى المطلوب بين عناصر <body> و </body> لعرضه على الصفحة.

العناصر والوسوم tags في HTML:

في سياق لغة HTML، الوسم (Tag) هو عنصر تمييز يستخدم لتحديد وتعريف أجزاء مختلفة في صفحة الويب. الوسوم تشكل جزءًا أساسيًا من بنية وتنسيق صفحة HTML.

الوسوم تكون عبارة عن علامات تفتح وتغلق وتحيط بالعناصر المختلفة في صفحة HTML. وتكون بالصيغة التالية:

<Open tag> <Close tag/>

كل ما يتم كتابته بين وسمي البداية والنهاية يعتبر محتوى العنصر.

يتم تمرير خصائص العنصر في وسم البداية وتدعى بال Attribute.

ملاحظة: ليست كل العناصر تحوي على محتوى وعندها يندمج وسم البداية مع وسم النهاية ويصبح عنصرا عديم المحتوى.

عناصر العناوين Heading:

عناصر العنوان (Headings) في HTML تستخدم لتحديد المستوى الهرمي للعناوين في صفحة الويب. توفر عناصر العنوان ستة مستويات مختلفة من العناوين، تبدأ من <h1> كأكبر عنوان وتنتهي بـ <h6> كأصغر عنوان.

يمكن استخدام عناصر العنوان لتحديد تسلسل وتنظيم العناوين في صفحة الويب. عند استخدام العناصر بشكل صحيح، فإنها تساعد في هيكلة المحتوى وتعطي للقراء فهمًا أفضل للموضوع وتساعدهم في التنقل في الصفحة.

يجب استخدام العناوين بشكل مناسب وفقًا للتسلسل الهرمي، حيث يجب أن يكون <h1> العنوان الرئيسي للصفحة ويمكن استخدام العناوين الفرعية بشكل تسلسلي لتوضيح تفاصيل أدق.

مثال على استخدام عناصر العنوان في ترتيب هرمي صحيح:

```
<h1>Title</h1>
<h2>heading 1</h2>
<h3>heading 2</h3>
<h4>heading 3</h4>
<H5>heading 4</H5>
<H6>heading 5</H6>
```

بعض الخصائص المشتركة للعناصر في HTML

Name	يستخدم فقط لتميز العنصر برمجيا
Id	هو الاسم الفريد الذي يميز هذا العنصر عن باقي العناصر المشابهة له
Class	وهو اسم لاحد الفئات في css
Style	تستخدم لتعين بعض التنسيقات للعنصر

عناصر النصوص Paragraphs

عناصر النصوص في HTML تستخدم لتنسيق وتقسيم المحتوى النصي في صفحة الويب. واحدة من أهم عناصر النص هي عنصر الفقرة (<p>)، والذي يستخدم لعرض فقرات من النص.

هنا هو مثال بسيط يوضح كيفية استخدام عنصر الفقرة في HTML:

```
<p>this is a paragraph</p>
```

في هذا المثال، تم استخدام عنصر الفقرة (<p>) لتحديد كل فقرة من النص. سيتم عرض كل فقرة في سطر منفصل وستتم إضافة مساحة فارغة (فراغ) قبل وبعد كل فقرة.

بالإضافة إلى عنصر الفقرة (<p>)، هناك أيضًا عناصر أخرى للنصوص في HTML مثل:

- عنصر الفقرة المختصرة (<abbr>) الذي يستخدم لتوضيح اختصارات النص.

- عنصر الاقتباس (<blockquote>) الذي يستخدم لتعليق النص المقتبس من مصدر آخر.

- عنصر العنوان الفرعي (<sub>) الذي يستخدم لعرض النص المنخفض في السطر الأسفل.

مثال لاستخدام ما سبق:

```
  <!-- paragraph -->
 <p>
  this is a paragraph
  <!-- breack line -->
  <br>
  <abbr title="abbr">abbr</abbr>
  <!-- quote -->
  <blockquote>this is a qoute</blockquote>
  <!-- supplementary text -->
  <sub>this is a sub text </sub>
</p>
```

المحارف والنصوص الخاصة في html

<	<
>	>
≠	≠
™	™
©	©
المحرف الفارغ	

عناصر الروابط hyper links

عناصر الروابط في HTML تستخدم لإنشاء روابط قابلة للنقر (hyperlinks) تسمح للمستخدمين بالانتقال إلى صفحات ويب أخرى أو مواقع أو موارد أخرى عبر الإنترنت. هنا هي العناصر الأساسية للروابط في HTML:

يستخدم لإنشاء رابط قابل للنقر. يتم استخدام العنصر <a> مع الخاصية "href" والخاصية target,لتحديد عنوان URL للصفحة المستهدفة. هنا مثال بسيط:

```
<a href="https://github.com/bittarwork/BookStore.git"
target="_blank">source code</a>
```

خاصية "target":

تستخدم لتحديد كيفية فتح الصفحة المستهدفة عند النقر على الرابط. هنا قائمة بالقيم الشائعة التي يمكن استخدامها للخاصية "target":

- "_blank": تفتح الصفحة المستهدفة في نافذة أو تبويب جديدة.

- "_self": تفتح الصفحة المستهدفة في نفس النافذة أو التبويب الحالي.

- "_parent": تفتح الصفحة المستهدفة في النافذة الأم العلوية إذا كانت الصفحة الحالية مدمجة في إطارات (frameset).

- "_top": تفتح الصفحة المستهدفة في النافذة العلوية بإغلاق أي إطارات مضمنة.

الفرق بين الروابط المطلقة والروابط النسبية:

يتعلق بكيفية تحديد موقع الصفحة المستهدفة.

1. الروابط المطلقة (Absolute Links):

الروابط المطلقة تحدد موقع الصفحة المستهدفة بشكل كامل باستخدام عنوان URL كامل.

```
<a href="https://github.com/bittarwork/BookStore.git"
target="_blank">source code</a>
```

2. الروابط النسبية (Relative Links):

الروابط النسبية تحدد موقع الصفحة المستهدفة بالنسبة إلى موقع الصفحة الحالية. بدلاً من استخدام عنوان URL كامل، يتم استخدام مسار نسبي للوصول إلى الصفحة المستهدفة. هنا مثال للروابط النسبية:

```
<a href="./pages/index2.html">click here to go to the seconde page</a>
```

في هذا المثال، المسار النسبي "page/" يشير إلى صفحة موجودة في نفس النطاق الفرعي للصفحة الحالية.

مثال اخر لاستخدام الروابط النسبية:

```
<a href="#first">click here</a>
```

وفي هذا المثال تم استخدام id وهو معرف خاص بعنصر اخر ضمن الصفحة.

عناصر الصور images

عنصر الصور في HTML هو عنصر <img>. يستخدم هذا العنصر لإدراج صورة في صفحة الويب. إليك بعض الخصائص الشائعة لعنصر الصور:

1. src: هذه الخاصية تحدد مسار المصدر (URL) للصورة. يجب توفير قيمة لهذه الخاصية لتحديد موقع الصورة.

2. alt: هذه الخاصية توفر نصًا بديلًا للصورة في حالة عدم تمكن المستخدم من رؤية الصورة. يجب توفير وصف موجز للصورة باستخدام هذه الخاصية لتحسين تجربة المستخدمين ذوي الإعاقات أو عندما لا يتم تحميل الصورة بشكل صحيح.

3. height و width: تستخدم هذه الخصائص لتحديد عرض وارتفاع الصورة بالبكسل. يمكن استخدامها لتعيين الحجم الظاهري للصورة في صفحة الويب.

4. title: تستخدم هذه الخاصية لتوفير عنوان توضيحي للصورة. عندما يمر المستخدم بالمؤشر فوق الصورة، قد يتم عرض هذا العنوان كتلميح أدوات.

مثال:

```
<img src="./images/HTML5_logo_and_wordmark.svg.png" alt="html logo"
title="html" width="300px" height="250px">
```

تمرين:

قم بإنشاء صفحة HTML تعرض المعلومات التالية:

الصفحة الأولى:

أولا يتم عرض الصورة الخاصة بشعار HTML

HTML:

HTML stands for HyperText Markup Language.

It is the standard markup language used for creating web pages and applications.

HTML uses a set of tags and attributes to structure and present content on the internet.

HTML documents consist of a series of elements or tags that define the structure, formatting, and functionality of web pages.

These elements are enclosed in angle brackets (< >) and are usually paired with opening and closing tags.

The content of the web page is placed between these tags.

HTML provides a wide range of elements and attributes to structure content,

create headings, paragraphs, lists, tables, forms, media, and more.

It also supports the use of CSS (Cascading Style Sheets) for styling and JavaScript for adding interactivity to web pages

وبعد كتابة المعلومات السابقة يوجد لدينا رابط يقوم بأخذنا الى الصفحة الثانية التي سوف نقوم بإنشائها.

الصفحة الثانية:

some of HTML tags:

some commonly used HTML tags:

1` .html`: Defines the root element of an HTML document.

2` .head`: Contains metadata and other non-visible elements for the document, such as the title.

3` .title`: Specifies the title of the document, which is displayed in the browser's title bar or tab.

4` .body`: Represents the main content of the HTML document.

5` .h1` to `h6`: Headings of different levels, with `h1` being the highest and `h6` being the lowest.

6` .p`: Defines a paragraph of text.

7` .a`: Creates a hyperlink to another web page or resource.

8` .img`: Inserts an image into the document.

9` .ul`: Defines an unordered (bulleted) list.

10` .ol`: Defines an ordered (numbered) list.

11` .li`: Represents a list item within `ul` or `ol.`

12` .div`: Defines a division or a container for grouping other elements.

13` .span`: Inline element used for styling or grouping text.

14` .table`: Creates a table for organizing tabular data.

15` .tr`: Defines a row within a table.

16` .td`: Represents a cell within a table row.

17` .th`: Defines a header cell within a table row.

18` .form`: Creates a form for user input.

19` .input`: Represents an input field within a form.

20` .button`: Creates a clickable button.

to read more:

you can go www.w3schools.com and see some HTML Tutorial

ومن ثم يوجد رابط يقوم بأخذنا الى الموقع التالي:

/https://www.w3schools.com/html

ومن ثم رابط اخر يقوم بالعودة الى الصفحة الأولى.

بحيث يكون الشكل الخاص بهذه الصفحة كما يلي: