

تطبيقات الويب واساسيات لغة HTML



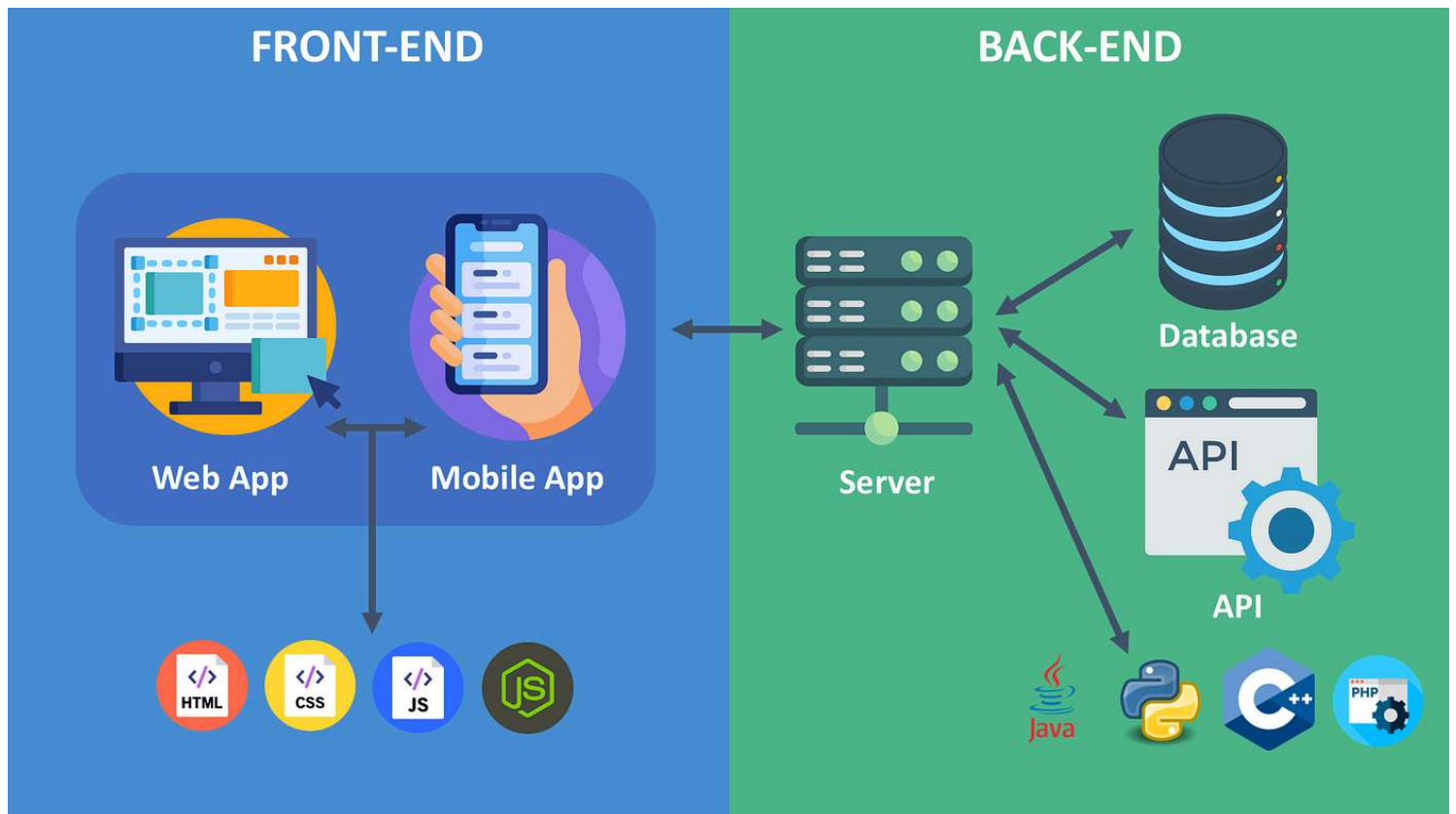
الفرق بين Backend و Frontend

المقدمة

عند بناء أي موقع أو تطبيق ويب، يمكن تقسيمه بشكل عام إلى جزأين رئيسيين:

- **Frontend** (الواجهة الأمامية)
- **Backend** (الواجهة الخلفية)

لكل منهما وظيفة محددة وأدوات وتقنيات مختلفة، ويعملان معًا لتقديم تجربة كاملة للمستخدم.



🌐 ما هو الـ Frontend؟

الـ **Frontend** هو كل ما يراه ويتفاعل معه المستخدم على الشاشة. هو الجزء "المرئي" من الموقع أو التطبيق، ويشمل:

🧱 المكونات الأساسية:

1. **HTML**: الهيكل العام للصفحة (مثل العظام في الجسم).
2. **CSS**: التصميم والمظهر الخارجي (مثل اللباس).
3. **JavaScript**: التفاعل والحركة (مثل الحواس).

🖥️ أمثلة على واجهات **Frontend**:

- الأزرار والنماذج التي تملأها.
- القوائم الجانبية والقوائم المنسدلة.
- الصور والمقالات التي تراها على صفحة الموقع.

⚙️ أشهر أدوات وتقنيات الـ **Frontend**:

- مكتبات: React.js, Vue.js, jQuery
- أطر عمل: Next.js, Angular
- أدوات تنسيق: Tailwind CSS, Bootstrap
- أدوات بناء: Vite, Webpack
- لغات أخرى: TypeScript (امتداد للـ JavaScript)

🖥️ ما هو الـ Backend؟

الـ **Backend** هو الجانب غير المرئي الذي يعمل خلف الكواليس. وهو المسؤول عن معالجة البيانات والمنطق الداخلي.

💡 وظائف الـ **Backend**:

1. إدارة قواعد البيانات (قراءة، كتابة، تعديل).
2. التعامل مع تسجيل الدخول وكلمات المرور.
3. التحكم بالصلاحيات (من يمكنه رؤية ماذا).
4. إرسال واستقبال البيانات عبر الإنترنت.
5. ربط الموقع بأنظمة أخرى (APIs).

أشهر لغات وتقنيات الـ Backend: 📦

- لغات: #Node.js (JavaScript), PHP, Python, Java, Ruby, C
- أطر عمل: Express.js, Laravel, Django, Spring Boot, ASP.NET Core
- قواعد بيانات: MySQL, PostgreSQL, MongoDB
- APIs: RESTful APIs, GraphQL

كيف يتفاعل معًا؟ 🔄

عندما يضغط المستخدم على زر (في الـ Frontend) لإرسال نموذج:

1. يتم إرسال البيانات إلى السيرفر (Backend).
2. يقوم الـ Backend بمعالجة البيانات (مثل حفظها في قاعدة بيانات).
3. يُرسل ردًا (Response) إلى الـ Frontend.
4. يعرض الـ Frontend النتيجة للمستخدم (نجاح أو فشل مثلاً).

مثال:

- المستخدم يكتب بريده وكلمة السر → (Frontend)
- تُرسل البيانات إلى السيرفر للتحقق → (Backend)
- إذا كانت صحيحة → يعود للمستخدم تأكيد الدخول.

تشبيه بسيط: 🌱

العنصر	Frontend	Backend
ما يراه المستخدم	نعم (واجهة مرئية)	لا (يعمل في الخلفية)
الوظيفة	العرض والتفاعل	المعالجة والبيانات
الأدوات	HTML, CSS, JS, React	Node.js, Python, DBs, APIs
مثال	زر "تسجيل دخول"	التحقق من كلمة المرور وتسجيل الجلسة

لماذا من المهم فهم الفرق؟

- لتصبح مطور واجهات أمامية (Frontend Developer) عليك التخصص في التصميم وتجربة المستخدم.
- لتصبح مطور خلفيات (Backend Developer) عليك فهم البيانات والمنطق والتخزين.
- أو يمكنك أن تصبح **Full-Stack Developer** وتتعلم كلا الجانبين.

• مقدمة في تطوير الويب - أكاديمية حسوب

• Frontend vs Backend – FreeCodeCamp

2. كيف يعمل الويب (Request / Response)

مقدمة

عندما تفتح موقعًا مثل google.com في المتصفح، هناك الكثير مما يحدث "خلف الكواليس".
الويب لا يعمل بشكل عشوائي، بل يعتمد على نظام دقيق جدًا يُسمى نموذج الطلب والرد (Request / Response Model) ويُبنى على بروتوكول يُدعى HTTP.

ما هو HTTP

HTTP = HyperText Transfer Protocol

هو البروتوكول الذي يستخدمه الويب لإرسال البيانات بين المتصفح (العميل) و الخادم (السيرفر).

مثال بسيط:

عند كتابة https://example.com في المتصفح:

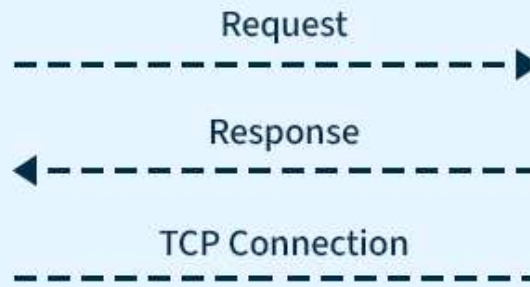
- يقوم المتصفح بإرسال طلب GET للحصول على الصفحة.
- يررد السيرفر بملف HTML (أو رسالة خطأ).



HTTP Connection



Client



Server

1. المستخدم يفتح متصفحًا ويكتب رابط (URL).
2. المتصفح يترجم هذا الرابط إلى عنوان IP عبر DNS.
3. يتم إنشاء طلب (Request) يُرسل إلى السيرفر.
4. السيرفر يستقبل الطلب، يعالجه، ويرسل رد (Response).
5. المتصفح يعرض الرد كمحتوى مرئي.

ما هو الطلب (HTTP Request)؟



طلب يرسله المتصفح للسيرفر، يتكوّن من:

- الطريقة (Method): مثل GET , POST , PUT , DELETE
- الرابط (URL): مثل products/
- الرؤوس (Headers): مثل نوع المتصفح أو ملفات الكوكيز
- الجسم (Body): (اختياري) مثل البيانات المرسلّة في نموذج

مثال:

GET /index.html HTTP/1.1

Host: www.example.com

User-Agent: Mozilla/5.0

ما هو الرد (HTTP Response)؟

هو ما يرجعه السيرفر ردًا على الطلب، ويتكون من:

- رمز الحالة (Status Code): مثل 200 (نجاح)، 404 (غير موجود)
- الرؤوس (Headers): مثل نوع المحتوى أو مدة التخزين
- الجسم (Body): المحتوى مثل HTML أو JSON أو صورة

مثال:

HTTP/1.1 200 OK

Content-Type: text/html

<html> ... </html>

أنواع أشهر الطلبات (Methods)

الطريقة	الاستخدام
GET	جلب بيانات (صفحة، صورة، ملف)
POST	إرسال بيانات (تسجيل الدخول، نموذج)
PUT	تعديل مورد موجود
DELETE	حذف مورد

تشبيه واقعي بسيط

تخيل أنك تطلب بيتزا من مطعم:

- أنت (المتصفح): تطلب بيتزا = Request
- المطعم (السيرفر): يحضر الطلب ويرسله لك = Response
- طريقة الطلب (هاتف؟ تطبيق؟) = HTTP
- نوع الطلب (بيتزا، مشروب) = Method
- العنوان والاسم = Headers

- الطلب نفسه = Body

🔧 أين ترى هذه الطلبات؟

في أي متصفح حديث، يمكنك:

- الضغط على الزر الأيمن < **Inspect** / فحص < التبويب **Network**
- ستري كل الطلبات التي يتم إرسالها لكل ملف (HTML, CSS, JS, صور...)

📌 لماذا يهم أن تفهم هذا؟

- أي تفاعل بين المتصفح والسيرفر يعتمد عليه.
- ضروري لفهم كيف تعمل النماذج، طلبات API، وعمليات تسجيل الدخول.
- مهم لفهم الأمن، الأداء، وأخطاء الشبكة.

🔗 مراجع إضافية:

- شرح شامل لبروتوكول HTTP – MDN
- كيف يعمل الإنترنت؟ – أكاديمية حسوب

✓ 3. هيكل صفحات الويب (HTML, CSS, JS)

- HTML: الهيكل
- CSS: التنسيق
- JS: التفاعل

🔗 مراجع:

- HTML vs CSS vs JS – W3Schools
- دورة برمجة – برمجة.كوم

4. أدوات التطوير (VS Code – Live Server)

VS Code هو محرر شيفرة ممتاز.
Live Server يقوم بعرض الصفحة مباشرةً عند الحفظ.

مراجع: 

- [VS Code Documentation](#)
- [شرح Live Server \(يوتيوب\)](#)

5. عناصر HTML الأساسية

```
<h1/>عنوان<h1>  
<p/>فقرة<p>  
<a/>رابط<"# "=a href>  
</ "وصف" =img src="image.jpg" alt>
```

مراجع: 

- [HTML Elements – MDN](#)
- [عناصر HTML – حسوب](#)

6. القوائم: ul, ol, li

```
<li></ul>عنصر<ul><li>  
<li></ol>عنصر مرتب<ol><li>
```

مراجع: 

- [HTML Lists – W3Schools](#)
- [شرح القوائم – HTML بالعربي](#)

7. الجداول: table, thead, tbody, tr, td, th


```
<table>
<thead><tr><th>عنوان</th></tr></thead>
<tbody><tr><td>معلومة</td></tr></tbody>
</table>
```

مراجع: 

- [HTML Tables – W3Schools](#)
- [الجداول – أكاديمية حسوب](#)

8. النماذج: form, input, textarea, select, button

```
<form>
</ "اسمك"=input type="text" placeholder>
<textarea></textarea>
<select><option>خيار</option></select>
<button>أرسل</button>
</form>
```

مراجع: 

- [HTML Forms – MDN](#)
- [نماذج HTML – HTML.net](#)

9. العناصر الهيكلية: div, section, nav, header, footer, main, article, aside

```
<h1></header><header><h1>عنوان
<a href="#">رابط</a></nav><nav>
<article></section></main><main>محتوى</main>
<footer>نهاية</footer>
```

مراجع: 

- [Semantic Elements – W3Schools](#)
- [العناصر الهيكلية – أكاديمية حسوب](#)

10. الخصائص (Attributes)

```
<a href="url" target="_blank">رابط</a>  
  
<input type="text" placeholder="اكتب هنا" />  
<button disabled>زر</button>
```

مراجع: 

- [HTML Attributes – MDN](#)
- [Attributes – W3Schools](#)

11. المشروع التطبيقي

صفحة شخصية تشمل:

- عنوان
- قائمة مهارات
- جدول
- نموذج
- صورة
- روابط

مراجع: 

- [HTML Project Ideas – FreeCodeCamp](#)
- [مشاريع HTML بسيطة – YouTube](#)

Git و GitHub 12.12

```
git init
. git add
"الرفع الأول" git commit -m
git remote add origin https://github.com/user/repo.git
git push -u origin master
```

مراجع: 

- دورة Git و GitHub – الزيرو
- [GitHub Docs](#)