الأكاديمية العربية الدولية
Arab International Academy

برمجة الويب

Web programming

م. خليل المحمد

كلية العلوم – ماجستير علم البيانات

المحاور

1. مقدمة إلى برمجة الويب

1.1 تعريف برمجة الويب وأهميتها.

1.2 العلاقة بين برمجة الويب والأمن السيبراني.

1.3 دور برمجة الويب في علم البيانات.

2. هيكلية تطبيقات الويب

2.1 مكونات تطبيقات الويب (العميل والخادم).

2.2 نظرة عامة على بروتوكول HTTP و HTTPS.

2.3 دور الـ API في تطبيقات الويب.

3. تطوير الخادم وقواعد البيانات:

3.1Node.js و Express.js لتطوير الخادم.

3.2 ربط التطبيقات مع قواعد البيانات باستخدام MongoDB أو MySQL.

3.3 تصميم الـ RESTful APIs.

4. أمن تطبيقات الويب

4.1 التهديدات الشائعة في تطبيقات الويب (XSS ، SQL Injection، CSRF).

4.2 أفضل الممارسات لتأمين تطبيقات الويب.

المخرجات المتوقعة من المحاضرة

1. **فهم أساسيات برمجة الويب**: استيعاب المفاهيم الأساسية في **برمجة الويب**، بما في ذلك كيفية عمل **العميل والخادم** ودور الـ **HTML**، **CSS**، **JavaScript** في تطوير صفحات الويب.
2. **الوعي بالأمن السيبراني في تطبيقات الويب**: التعرف على **التهديدات الأمنية الشائعة** مثل **XSS** و **SQL Injection** وكيفية استخدام **أفضل الممارسات** لحماية التطبيقات.
3. **دمج برمجة الويب مع علم البيانات**: القدرة على جمع وتحليل البيانات من تطبيقات الويب وربطها مع **نماذج التعلم الآلي** لتقديم تحليلات متقدمة.
4. **التعرف على أطر عمل وأدوات الويب**: فهم كيفية استخدام **Node.js** و **Express.js** لتطوير الخادم.
5. **استيعاب التقنيات المتقدمة**: التعرف على كيفية استخدام **WebSockets** لتطوير تطبيقات تفاعلية.
    1. استكشاف كيفية تحسين استعلامات البيانات باستخدام **GraphQL**.
    2. فهم **تطبيقات الويب التقدمية (PWA)** وكيفية تحسين تجربة المستخدم.
6. **القدرة على تطبيق المعرفة في مشاريع عملية**: الاستعداد لتطبيق ما تم تعلمه في مشاريع **ويب متكاملة** تجمع بين **الأمن السيبراني** وعلم **البيانات**.

1. مقدمة إلى برمجة الويب

1.1 تعريف برمجة الويب وأهميتها

برمجة الويب هي فن استخدام التقنيات المختلفة لإنشاء صفحات وتطبيقات ويب تتفاعل مع المستخدمين وتوفر تجربة استخدام غنية. تشمل التقنيات المستخدمة في برمجة الويب لغات البرمجة مثلHTML ، CSS، و .JavaScriptأهمية برمجة الويب تنبع من قدرتها على جعل المعلومات والخدمات متاحة للجميع في أي وقت ومن أي مكان، مما يعزز من التواصل العالمي ويساهم في النمو الاقتصادي والابتكار.

تتضمن هذه العملية كتابة الشفرات والتعليمات التي تُنشئ صفحات وتطبيقات ويب يمكن للمستخدمين التفاعل معها من خلال متصفحات الويب. برمجة الويب تشمل جميع الجوانب التقنية لإنشاء موقع أو تطبيق ويب، من تصميم الواجهة الأمامية التي يراها المستخدم إلى إعداد الخوادم وقواعد البيانات التي تعالج الطلبات وتخزن البيانات.

1. مقدمة إلى برمجة الويب

1. **تمكين الوصول العالمي :**برمجة الويب تتيح للأفراد والشركات الوصول إلى جمهور عالمي من خلال إنشاء مواقع وتطبيقات يمكن استخدامها من أي مكان في العالم، مما يعزز التواصل والتفاعل على نطاق واسع.
2. **التكامل مع الأعمال :**في العصر الرقمي، أصبحت مواقع وتطبيقات الويب جزءًا أساسيًا من استراتيجيات الأعمال، حيث تساعد في تحسين تجربة العملاء وزيادة الكفاءة التشغيلية من خلال توفير خدمات مثل التسوق عبر الإنترنت، الحجز الإلكتروني، والتفاعل مع العملاء.
3. **تعزيز الابتكار :**توفر برمجة الويب بيئة خصبة للابتكار والإبداع. يمكن للمطورين إنشاء تطبيقات وخدمات جديدة تلبي احتياجات الأسواق الناشئة والمستخدمين المتنوعين، مما يسهم في تطوير الاقتصاد الرقمي.
4. **التعليم والتعلم :**توفر مواقع الويب وتطبيقات التعلم الإلكتروني موارد تعليمية متنوعة يمكن الوصول إليها بسهولة، مما يعزز من فرص التعليم والتدريب المستمر.
5. **أمان المعلومات :**برمجة الويب تسهم في تأمين البيانات وحمايتها من التهديدات السيبرانية من خلال تنفيذ تدابير الأمان المناسبة، مما يضمن حماية المعلومات الحساسة للمستخدمين والشركات.

1. مقدمة إلى برمجة الويب

1.2 العلاقة بين برمجة الويب والأمن السيبراني :

1. الاختبارات الأمنية: تطبيقات الويب تخضع لاختبارات أمان مكثفة، مثل اختبارات الاختراق ((Penetration Testing لتحديد الثغرات الأمنية قبل إطلاقها للعامة.

2. التوعية والتدريب: الأمن السيبراني جزء لا يتجزأ من تعليم المطورين كيفية بناء تطبيقات آمنة. من خلال ورش العمل والدورات التدريبية، يتعلم المطورون كيفية التعامل مع التهديدات السيبرانية بشكل فعال.

3. الاستجابة للحوادث: في حالة وقوع حادث أمني، تتعاون فرق الأمن السيبراني مع فرق تطوير الويب لتحديد مصدر المشكلة وحلها بسرعة، وتقليل الأضرار.

1. مقدمة إلى برمجة الويب

1.3 دور برمجة الويب في علم البيانات :

1. جمع البيانات من مصادر متعددة:

تطبيقات الويب تُستخدم لجمع البيانات من المستخدمين، سواء كان ذلك من خلال استبيانات، أو تسجيلات الدخول، أو أنشطة المستخدمين عبر الإنترنت. هذه البيانات تُخزن في قواعد بيانات يمكن استخدامها لاحقًا للتحليل.

2. تكامل البيانات من خلال الـAPIs:

برمجة الويب تُسهل تكامل البيانات من مصادر متعددة عبر استخدام واجهات برمجة التطبيقات .(APIs) هذه الواجهات تسمح للتطبيقات بالتفاعل مع خدمات أخرى.

3. معالجة البيانات وتحليلها:

تطبيقات الويب يمكنها القيام بعمليات معالجة البيانات الأولية قبل تحليلها. على سبيل المثال، يمكن لتطبيقات الويب تنظيف البيانات من المدخلات غير الصحيحة، أو تنظيمها بشكل معين يجعلها جاهزة للتحليل.

1. مقدمة إلى برمجة الويب

4. عرض البيانات بشكل تفاعلي: برمجة الويب تسهم بشكل كبير في عرض البيانات بطريقة مرئية وتفاعلية. باستخدام مكتبات مثل D3.jsوChart.js، يمكن إنشاء لوحات معلومات (dashboards) تفاعلية تعرض البيانات بطريقة يمكن للمستخدمين فهمها بسهولة.

5. تمكين النماذج التنبؤية: علم البيانات يعتمد على بناء نماذج تنبؤية تساعد في التوقعات واتخاذ القرارات. برمجة الويب تلعب دورًا في دمج هذه النماذج داخل التطبيقات بحيث يتم تحديث التنبؤات في الوقت الحقيقي بناءً على البيانات الجديدة. هذا يجعل التطبيقات أكثر ذكاءً وفعالية.

6. الأتمتة والذكاء الاصطناعي: برمجة الويب تُمكن من أتمتة العمليات المختلفة من خلال بناء تطبيقات تعتمد على الذكاء الاصطناعي. يمكن لتطبيقات الويب جمع البيانات، تحليلها، وتقديم توصيات أو حلول تلقائيًا بناءً على النتائج، مما يوفر الوقت والجهد على المستخدمين وعلماء البيانات.

2. هيكلية تطبيقات الويب

2.1 مكونات تطبيقات الويب (العميل والخادم):

أولاً: العميل:(Client) : العميل هو جهاز المستخدم النهائي الذي يتفاعل مع التطبيق عبر الإنترنت باستخدام متصفح الويب. يمكن أن يكون العميل جهاز حاسوب، هاتف ذكي، أو جهاز لوحي، وكل هذه الأجهزة تستخدم متصفحًا للوصول إلى تطبيقات الويب.

المهام الرئيسية للعميل:

1. **إرسال الطلبات:(Requests) :** يقوم العميل بإرسال طلبات إلى الخادم للحصول على البيانات أو تنفيذ بعض العمليات.
2. **عرض البيانات:(Rendering Data) :** بعد استلام البيانات من الخادم، يقوم العميل (المتصفح) بعرض هذه البيانات للمستخدم.
   - يتم تحويل البيانات النصية أو الجداول إلى تنسيق مرئي باستخدام لغات مثل) HTML الهيكل(، CSS(التنسيق)، و) JavaScriptالتفاعل(
3. **التفاعل مع المستخدم:** يتيح العميل للمستخدم التفاعل مع التطبيق عن طريق النقر على الأزرار، إدخال البيانات في النماذج، وتصفح المحتوى.

2. هيكلية تطبيقات الويب

4. **إدارة الجلسات:(Session Management) :** العميل يُدير الجلسات عبر تخزين البيانات في الكوكيز أو الجلسات المحلية (localStorage) لتتبع حالة المستخدم بين الصفحات المختلفة.

5. **أدوات العميل:**

- **المتصفح :**مثل Google Chrome ، Firefox، Safari، والذي يترجم الأكواد البرمجية مثل HTML و CSS و JavaScriptإلى محتوى مرئي تفاعلي.
- **التقنيات المساعدة :**مثل) JavaScript libraries مثل React أو (Angular و) CSS frameworksمثل Bootstrap)لتحسين التفاعل والواجهة.

2. هيكلية تطبيقات الويب

ثانياً: الخادم:(Server) : الخادم هو جهاز كمبيوتر أو مجموعة من الأجهزة تعمل على معالجة البيانات وتخزينها، ومن ثم إرسالها إلى العميل عند الحاجة.

المهام الرئيسية للخادم:

1. **معالجة الطلبات:(Request Processing)**
   - عند تلقي الخادم طلبًا من العميل، يقوم بتحديد نوع الطلب (مثل الحصول على صفحة، إرسال بيانات نموذج، إلخ).
   - الخادم يتعامل مع الطلبات ويُعيد الإجابة المناسبة، مثل إرسال HTML أو JSON أو ملفات أخرى إلى العميل.
2. **إدارة البيانات:(Data Management)**
   - الخادم يتعامل مع قواعد البيانات لتخزين واسترجاع البيانات عند الحاجة. على سبيل المثال، إذا طلب العميل صفحة تحتوي على معلومات المستخدم، يقوم الخادم بالاستعلام من قاعدة البيانات وإرسال البيانات.
   - الخادم يمكن أن يكون مسؤولًا أيضًا عن معالجة البيانات قبل إرسالها إلى العميل.

2. هيكلية تطبيقات الويب

3. **تنفيذ المنطق التجاري:(Business Logic) :** الخادم يتعامل مع جميع العمليات الحسابية أو المنطقية المعقدة. على سبيل المثال، إذا كان العميل يطلب حسابًا معينًا أو عملية تحقق من البيانات، يقوم الخادم بتنفيذ المنطق اللازم.

4. **الرد على الطلبات:(Response Generation) :** بعد معالجة الطلب، يقوم الخادم بتوليد استجابة تُرسل إلى العميل. قد تتضمن هذه الاستجابة صفحة HTML كاملة، بيانات JSON أو) XML في حالة استخدام(APIs ، أو ملفات ميديا مثل الصور أو المستندات.

5. **أدوات الخادم: الخوادم البرمجية :**مثلApache ، Nginx، أو خوادم مدمجة مثل Node.js التي تعمل على تنفيذ الكود البرمجي المرسل من العميل.

   - **لغات البرمجة الخلفية :**مثلPHP ، Python، Ruby، Java، Node.js التي تُستخدم لكتابة الكود البرمجي في الخادم.
   - **إطارات العمل الخلفية :**مثلDjango ، Express.js، أوRuby on Rails ، التي تُسهل إنشاء تطبيقات ويب مع خوادم سريعة وآمنة.

2. هيكلية تطبيقات الويب

2.2 نظرة عامة على بروتوكول HTTP و HTTPS

مميزات بروتوكول:HTTP (HyperText Transfer Protocol) :

1. **بروتوكول بدون حالة:(Stateless)**
   - HTTPهو بروتوكول "بدون حالة"، مما يعني أن كل طلب يُعالج بشكل منفصل. الخادم لا يحتفظ بأي معلومات حول الطلبات السابقة، وكل طلب يُعتبر جديدًا.
2. **اتصال غير مشفر:(Unencrypted Connection)**
   - البيانات التي تنتقل عبر HTTP تكون غير مشفرة، مما يجعلها عرضة للتنصت أو التعديل من قبل أطراف ثالثة. يمكن لأي شخص على الشبكة (مثل مستخدمين آخرين على نفس شبكة الواي فاي) التقاط البيانات التي تُرسل بين العميل والخادم.
3. **يستخدم المنفذ 80:**
   - بروتوكول HTTP يعمل على المنفذ 80 في الشبكة، وهو المنفذ الافتراضي الذي يستخدمه المتصفح عند إرسال طلبات غير مشفرة.

2. هيكلية تطبيقات الويب

مميزات بروتوكول:HTTPS (HyperText Transfer Protocol Secure)

1. **اتصال مشفر:(Encrypted Connection)**
   - باستخدامSSL/TLS ، يتم تشفير البيانات بين العميل والخادم، مما يحميها من التنصت أو التلاعب من قبل أطراف ثالثة. يُشفر المحتوى بشكل لا يمكن قراءته إلا من قبل العميل والخادم المعنيين.
2. **توثيق الخادم:(Server Authentication)**
   - فيHTTPS ، يتم توثيق الخادم من خلال شهادة SSL/TLS التي يصدرها مزود الشهادات المعتمد. هذه الشهادات تضمن أن العميل يتصل بالخادم الصحيح وليس خادمًا مزيفًا.
3. **حماية ضد هجمات:"Man-in-the-Middle" (MitM)**
   - HTTPSيحمي من الهجمات التي قد يتنصت فيها مهاجم على الاتصال بين العميل والخادم، مما يسمح له بالتعديل على البيانات أو سرقة المعلومات.

2. هيكلية تطبيقات الويب

2.3 دور الـ API في تطبيقات الويب:

واجهة برمجة التطبيقات: هي هو مجموعة من الأدوات والبروتوكولات التي تُتيح للتطبيقات المختلفة التفاعل مع بعضها البعض.

أنواع الـ APIs المستخدمة في تطبيقات الويب:

1. **APIsالويب:(Web APIs) :** هذه هي APIs التي تتيح للتطبيقات التفاعل مع بعضها عبر الإنترنت باستخدام بروتوكولات HTTP أو HTTPS.
2. **APIsلقاعدة البيانات:(Database APIs) :** هذه APIs تتيح لتطبيقات الويب التفاعل مع قواعد البيانات مثل MySQL أو MongoDB
3. **RESTful APIs:** هو نمط معماري يُستخدم في بناء APIs التي تتيح (Representational State Transfer REST) للتطبيقات التفاعل مع بعضها.
4. **GraphQL APIs:** هو إطار عمل آخر لبناء APIs ولكنه يختلف عنGraphQL REST
5. **SOAP APIs:** هو بروتوكول قديم يُستخدم لتبادل البيانات بين الأنظمة باستخدام (Simple Object Access Protocol SOAP) .XML

2. هيكلية تطبيقات الويب

دور الـ API في تطبيقات الويب:

1. **التواصل بين العميل والخادم:** الـ API تُسهل عملية التواصل بين العميل (مثل المتصفح) والخادم. عندما يرسل العميل طلبًا للحصول على بيانات أو لتنفيذ عملية معينة، يتم توجيه هذا الطلب إلى الـ API الخاصة بالخادم.
2. **التكامل مع الخدمات الخارجية:** تُستخدم APIs لتمكين تطبيقات الويب من التفاعل مع خدمات أخرى.
3. **إدارة البيانات في الوقت الفعلي:** APIsتُسهل معالجة البيانات في الوقت الفعلي. على سبيل المثال، في تطبيقات المحادثات) مثل WhatsAppأو(Slack ، تُستخدم APIs لنقل الرسائل بشكل فوري بين المستخدمين.
4. **تحسين الأداء والتوسع:** استخدام الـ APIs يساعد في فصل المهام المختلفة بين أجزاء تطبيق الويب، مما يُسهل تحسين الأداء والتوسع.
5. **أتمتة العمليات:** APIsتُستخدم لأتمتة العديد من العمليات في تطبيقات الويب.
6. **الأمان والتحكم في الوصول:** الـ API تُسهم في تعزيز الأمان في تطبيقات الويب عبر توفير طرق للتحكم في الوصول.

2. هيكلية تطبيقات الويب

فوائد استخدام الـ API في تطبيقات الويب:

1. **التفاعل مع الأنظمة المختلفة:** يُتيح الـ API للتطبيقات التفاعل مع أنظمة متعددة. على سبيل المثال، تطبيق ويب قد يحتاج إلى التفاعل مع خدمات خارجية مثل أنظمة الدفع، أو مع بيانات من مصادر أخرى.
2. **سهولة التوسع والتحديث:** عبر استخدامAPIs ، يمكن لتطبيقات الويب توسيع وظائفها بسهولة أو تحديثها دون التأثير على باقي الأجزاء.
3. **تبسيط التطوير:** باستخدام APIs جاهزة مثل تلك المتاحة من Google أو Facebook أوTwitter
4. **دعم التطبيقات المتعددة:** يمكن استخدام نفس الـ API لدعم تطبيقات متعددة تعمل على منصات مختلفة مثل الويب، الأجهزة المحمولة، أو الأجهزة الذكية.

اسئلة

1. ما هو تعريف برمجة الويب وما أهميتها؟
2. كيف ترتبط برمجة الويب بالأمن السيبراني؟
3. ما هو دور برمجة الويب في علم البيانات؟
4. ما هي المكونات الأساسية لتطبيقات الويب؟
5. ما الفرق بين بروتوكول HTTPوHTTPS؟
6. ما هو دور الـ APIفي تطبيقات الويب؟

الاجوبة

1. برمجة الويب هي عملية إنشاء وتصميم المواقع والتطبيقات التي تعمل على الإنترنت باستخدام تقنيات ولغات مثل HTML، CSS، JavaScript، و.PHP
أهميتها تكمن في:

- تسهيل التواصل عبر الإنترنت من خلال إنشاء مواقع ديناميكية وتفاعلية.
- دعم العمليات التجارية عبر التطبيقات مثل التجارة الإلكترونية.
- تقديم حلول مبتكرة للعديد من القطاعات كالتعليم والرعاية الصحية.

2. برمجة الويب والأمن السيبراني مترابطان بشكل كبير، حيث تعتمد أمان تطبيقات الويب على تطبيق أفضل الممارسات البرمجية لتجنب الثغرات الأمنية مثل SQL Injection وCross-Site Scripting (XSS).
كما تشمل العلاقة:

- ضمان أمان البيانات المنقولة بين العميل والخادم باستخدام بروتوكولات مثل HTTPS.
- تقليل المخاطر عبر تطوير تطبيقات ويب مقاومة للاختراق.

الاجوبة

3. تلعب برمجة الويب دورًا حيويًا في علم البيانات من خلال:

- إنشاء تطبيقات واجهات عرض البيانات Data Visualizationباستخدام مكتبات مثل D3.js.
- تطوير لوحات التحكم التفاعلية التي تساعد في تحليل البيانات.
- تسهيل جمع البيانات عبر تطبيقات الويب مثل النماذج والاستبيانات.

4. المكونات الأساسية لتطبيقات الويب هي:

1.العميل Client: يمثل واجهة المستخدم التي تتعامل مع التطبيق، مثل المتصفح.
2.الخادم Server: يعمل على معالجة الطلبات القادمة من العميل وإرجاع البيانات المطلوبة.
3.قاعدة البيانات Database: تُخزن فيها البيانات التي يعتمد عليها التطبيق.

الاجوبة

.5

- **HTTP (Hypertext Transfer Protocol):** بروتوكول لنقل البيانات بين العميل والخادم، لكنه غير مشفر.
- **HTTPS (HTTP Secure):** إصدار مشفر من HTTP يستخدم طبقة SSL/TLS لتأمين البيانات وحمايتها أثناء النقل، مما يوفر مزيدًا من الأمان.

6. الـ API واجهة برمجة التطبيقات: هي وسيلة لتمكين تطبيقات الويب من التفاعل مع أنظمة أخرى.

دورها يشمل:

- ربط العميل بالخادم وتبادل البيانات بشكل فعال.
- تسهيل دمج خدمات خارجية مثل خدمات الدفع أو الخرائط.
- تحسين تجربة المستخدم من خلال تبسيط عمليات التفاعل بين التطبيقات.

3. تطوير الخادم وقواعد البيانات

1. دور الخادم في تطبيقات الويب:

الخادم هو الجهاز أو النظام الذي يستقبل الطلبات من العميل (عادة عبر الإنترنت) ويرسل له الاستجابات المناسبة. يُعد الخادم عنصرًا أساسيًا في تطبيقات الويب الحديثة، حيث يتعامل مع الطلبات المرسلة من المتصفح.

- **معالجة الطلبات :**يتلقى الخادم طلبات HTTP من العميل، ويقوم بمعالجتها بناءً على نوع الطلب مثل GET لجلب البيانات، POSTلإرسال البيانات.
- **الخوادم المختلفة :**الخوادم قد تكون متنوعة حسب نوع الخدمة التي تقدمها، مثل:
  - **خوادم الويب :**مسؤولة عن تقديم الصفحات والمحتويات للعميل مثل.Apache, Nginx
  - **خوادم التطبيقات :**تتعامل مع منطق العمل لتطبيق الويب مثل.Node.js, Django, Ruby on Rails
  - **خوادم قواعد البيانات :**مسؤولة عن التعامل مع استفسارات البيانات والرد عليها مثل.MySQL, PostgreSQL
- **إدارة الاتصال :**يقوم الخادم أيضًا بإدارة الاتصال بين العميل وقاعدة البيانات أو أي مكونات أخرى في النظام، مثل خدمة البريد الإلكتروني أو أنظمة الدفع.

3. تطوير الخادم وقواعد البيانات

2. تقنيات تطوير الخادم:

تتعدد تقنيات تطوير الخوادم وفقًا للغة البرمجة التي يتم استخدامها:

- **Node.js:** هو بيئة تشغيل مفتوحة المصدر تستند إلى JavaScript لبناء خوادم عالية الأداء. يتيح Node.js للمطورين كتابة كود الخادم باستخدامJavaScript ، ويتميز بسرعته في معالجة الطلبات غير المتزامنة.
- **Django (Python):** هو إطار عمل لبناء تطبيقات الويب بلغةPython ، يقدم بنية قوية لبناء تطبيقات الويب مع ميزات مثل التوجيه، التحقق من الصحة، والاتصال بقاعدة البيانات.
- **Ruby on Rails (Ruby):** إطار عمل يُستخدم مع لغةRuby ، ويساعد في تطوير تطبيقات الويب من خلال توفير أدوات لتسريع عملية التطوير وتسهيل التعامل مع قواعد البيانات.
- **Spring Boot (Java):** هو إطار عمل متكامل لبناء تطبيقات الخوادم باستخدامJava ، ويستهدف توفير إعدادات تلقائية ومكونات مرنة لبناء تطبيقات قابلة للتوسع.

3. تطوير الخادم وقواعد البيانات

3. تقنيات التفاعل بين الخادم والعميل:

- **REST (Representational State Transfer):** هو نمط معماري يُستخدم لبناء واجهات برمجة التطبيقات (APIs) التي تسمح للتطبيقات بالتفاعل عبر HTTP. يتبع REST مبدأ الموارد، حيث تتم المعاملات عبر استدعاء أنواع معينة من الطلبات مثل GET, POST, PUT, DELETE.
- **WebSockets:** هو بروتوكول يسمح بتبادل البيانات بين الخادم والعميل في الوقت الفعلي باستخدام قناة اتصال ثابتة ومفتوحة، مما يجعله مثاليًا لتطبيقات الدردشة أو الألعاب عبر الإنترنت.
- **GraphQL:** هو استعلام بيانات مرن يسمح للعملاء بالطلب الدقيق للبيانات التي يحتاجون إليها فقط، مما يقلل من حجم البيانات المنقولة عبر الشبكة.

3. تطوير الخادم وقواعد البيانات

4. أفضل الممارسات في تطوير الخوادم وقواعد البيانات:

- **الأمان :**يجب تأمين الخوادم وقواعد البيانات باستخدام تقنيات مثل **التشفير**، **التوثيق**، و **التحقق من الصلاحيات** لحماية البيانات من الهجمات مثل SQL Injection و.XSS
- **الأداء :**لتحسين أداء الخوادم، يمكن استخدام تقنيات مثل **التخزين المؤقت** (Caching)لتخزين البيانات المسترجعة بشكل مؤقت، و **تقنيات التحميل المتوازن** (Load Balancing)لتوزيع الحمل على الخوادم المتعددة.
- **التوسع :**من المهم تصميم الخوادم وقواعد البيانات بشكل يمكن توسيعه بسهولة، حيث يمكن تحسين الأداء عند الحاجة من خلال استخدام **الخوادم الموزعة** أو **قواعد البيانات الموزعة**.
- **النسخ الاحتياطي :**من الضروري اتخاذ تدابير للنسخ الاحتياطي المنتظم لقواعد البيانات لحمايتها من فقدان البيانات في حالة حدوث خطأ أو عطل في النظام.

3. تطوير الخادم وقواعد البيانات

3.1 Node.js و Express.js لتطوير الخادم:

1. ما هو Node.js ؟

Node.js هو بيئة تشغيل مفتوحة المصدر تتيح للمطورين تشغيل **JavaScript** على الخادم، بعيدًا عن المتصفح. بمعنى آخر، يمكن استخدام JavaScript لكتابة الكود الخاص بالخوادم، بدلاً من الاقتصار على كتابة الكود في المتصفح فقط.

- **المزايا الرئيسية لـ Node.js:**
  - **الأداء العالي :** بفضل محرك V8 غير المتزامن، يمكن لـ Node.js التعامل مع عدد كبير من الطلبات في نفس الوقت دون التأثير الكبير على الأداء.
  - **البنية الواحدة (Single-threaded):** يعتمد Node.js على خيط واحد فقط للتعامل مع الطلبات، مما يقلل من تعقيد إدارة الموارد مقارنةً بالخوادم التي تستخدم عدة خيوط.
  - **مجتمع دعم قوي :** يتوفر مجموعة كبيرة من المكتبات والموارد عبر **npm** (مدير الحزم الخاص بـ Node.js) ، مما يسمح للمطورين بإضافة وظائف متعددة بسهولة.

3. تطوير الخادم وقواعد البيانات

2. ما هو Express.js ؟

Express.js هو إطار عمل (Framework) يُستخدم مع **Node.js**، وهو يوفر مجموعة من الأدوات والمكتبات لبناء تطبيقات الخادم بشكل أسرع وأسهل.

- **التوجيه (Routing):** أحد المفاهيم الأساسية في Express.js هو **التوجيه**، الذي يسمح للمطورين بتحديد المسارات المختلفة للطلبات، مثل GET و POST و PUT وDELETE ، وربط هذه المسارات بوظائف أو معالجات معينة.
- **إدارة الطلبات والاستجابات** Express: يتيح لك التعامل مع الطلبات (requests) والاستجابات (responses) بشكل فعال، بما في ذلك التعامل مع بيانات النموذج، ملفاتJSON ، وملفات الوسائط المتعددة.
- **المرونة والسهولة في الاستخدام** Express.js: مُصمم ليكون بسيطًا وقابلًا للتوسعة، مما يعني أنه يمكن للمطورين البدء بسرعة دون الحاجة إلى كتابة الكثير من الكود. كما يمكن توسيعه بسهولة من خلال إضافات (middleware) لتوسيع وظائفه.

3. تطوير الخادم وقواعد البيانات

3. كيفية عمل Node.js و Express.js معًا؟

Node.js و **Express.js** يعملان معًا لتطوير خوادم الويب من خلال توفير بيئة تشغيل (Node.js) وواجهة موجهة للبناء السريع لتطبيقات الويب .(Express.js) إليك كيفية دمجهما:

- **الخادم في Node.js:** عند بناء خادم باستخدامNode.js ، يمكن استخدام **HTTP module** المدمج لإنشاء الخادم. لكن في حال استخدام **Express.js**، يمكن الحصول على خادم أكثر تنظيمًا وسهولة في التعامل مع الطلبات والاستجابات.
- **التوجيه مع Express.js:** باستخدامExpress.js ، يتم تحديد المسارات عبر التوجيه، مما يسمح لك بتحديد كيفية التعامل مع كل طلب على حدة. مثلاً، يمكنك تحديد المسار /users للإجابة على طلبات GET للحصول على قائمة المستخدمين، بينما قد تحدد مسار /users مع POST لإنشاء مستخدم جديد.
- **المعالجة غير المتزامنة:** يمكن لـ **Node.js** و **Express.js** التعامل مع الطلبات غير المتزامنة بشكل فعال. حيث يقوم الخادم بمعالجة طلبات متعددة في الوقت ذاته باستخدام **callbacks** أو **Promises**.

3. تطوير الخادم وقواعد البيانات

3.2 ربط التطبيقات مع قواعد البيانات باستخدام MongoDB أو MySQL

- **مزايا:MongoDB**
  - **مرونة في هيكلة البيانات** MongoDB :لا يتطلب تعريفات صارمة للبيانات مثل قواعد البيانات العلائقية، مما يجعلها مثالية للتطبيقات التي تحتوي على بيانات غير مهيكلة أو متغيرة.
  - **قابلية التوسع** :يسمح MongoDB بالتوسع الأفقي بسهولة من خلال تقسيم البيانات عبر الخوادم.(Sharding)
- **كيفية ربط MongoDB بتطبيق:Express.js**
  - **تنصيب مكتبة Mongoose:** Mongoose هو مكتبة JavaScript تُستخدم للعمل مع MongoDB بسهولة في تطبيقات Node.js.
  - **إعداد اتصال MongoDB:** يمكنك إعداد الاتصال بين تطبيق Express.js و MongoDB باستخدام عنوان URL للقاعدة.
  - **إجراء عمليات CRUD) إنشاء، قراءة، تحديث، حذف** :(باستخدامMongoose ، يمكن إجراء العمليات الأساسية مثل إضافة مستندات جديدة(Insert) ، استرجاع مستندات(Find) ، وتحديثها أو حذفها.

3. تطوير الخادم وقواعد البيانات

- **مزايا:MySQL**
  - **هيكل البيانات الثابت :**تعتمد MySQL على هيكل ثابت للجداول والعلاقات بين البيانات، مما يجعلها مناسبة للتطبيقات التي تتطلب تناسقًا قويًا في البيانات.
  - **الاستعلامات المعقدة :**يتيح لك MySQL إجراء استعلامات معقدة باستخدام SQL مثل الربط بين الجداول (JOIN) وتحليل البيانات.
- **كيفية ربط MySQL بتطبيق:Express.js**
  - **تنصيب مكتبة :MySQL** في تطبيقاتNode.js ، يمكن استخدام مكتبة **mysql2** أو (**sequelize**وهو-ORM - Object Relational Mapping)للتفاعل مع قاعدة بيانات.MySQL
  - **إعداد الاتصال :**لتوصيل التطبيق بـMySQL ، يجب تحديد معلومات الاتصال مثل **المضيف، اسم المستخدم، كلمة المرور**، واسم **قاعدة البيانات**.
  - **إجراء عمليات :CRUD** بعد الاتصال بقاعدة البيانات، يمكن إجراء العمليات الأساسية باستخدام SQL مثل **SELECT**لاسترجاع البيانات و **INSERT**لإضافة بيانات جديدة.

3. تطوير الخادم وقواعد البيانات

3. مقارنة بين MongoDB و MySQL:

الميزة	MongoDB	MySQL
نوع قاعدة البيانات	NoSQL (قائمة على المستندات)	SQL (قائمة على الجداول)
هيكل البيانات	مرن (مستندات JSON/BSON)	ثابت (جداول مترابطة)
التوسع	أفقي (Sharding)	رأسي (Vertical Scaling)
التكامل مع البيانات	غير علائقي (لا توجد علاقات بين الجداول)	علائقي (علاقات بين الجداول)
الاستعلامات	استخدام BSON واستخدام موجه الاستعلامات	استخدام SQL لاستعلامات معقدة
المناسب للتطبيقات	التطبيقات التي تحتوي على بيانات غير مهيكلة أو التي تتطلب مرونة في هيكلة البيانات	التطبيقات التي تحتوي على بيانات علائقية ومتطلبات هيكلية صارمة

3. تطوير الخادم وقواعد البيانات

3.3 تصميم الـ RESTful APIs

ما هو REST ؟ : اختصار لـ **Representational State Transfer** هو نمط معماري يُستخدم لتصميم واجهات برمجة التطبيقات (APIs)التي تتيح لتطبيقات مختلفة التواصل مع بعضها البعض عبر الإنترنت باستخدام بروتوكول **HTTP.**

المبادئ الأساسية لتصميم:RESTful APIs

1 . **استخدام:HTTP Methods**

- **GET:** لاسترجاع البيانات من الخادم.
- **POST:** لإنشاء مورد جديد على الخادم.
- **PUT:** لتحديث أو استبدال البيانات.
- **DELETE:** لحذف البيانات.

2. الاستفادة من الموارد:(Resources) : يتم تمثيل كل جزء من البيانات التي يتعامل معها الـ API كـ **مورد .(Resource)** يمكن أن يكون هذا المورد عبارة عن كائنات أو سجلات معينة، مثل المستخدمين أو المقالات أو المنتجات. يتم تحديد هذه الموارد باستخدام **مسارات.URL**

3. تطوير الخادم وقواعد البيانات

تحديد المسارات:(URLs)

- بعد تحديد الموارد والعمليات، يتم تحديد المسارات (URLs) التي يمكن الوصول إليها لتنفيذ هذه العمليات. من المهم أن تكون هذه المسارات واضحة ومنطقية. على سبيل المثال:
  - products/للحصول على قائمة بالمنتجات.
  - products/{id}/للحصول على منتج معين باستخدام معرّف المنتج.
  - products/{id}/reviews/لعرض مراجعات منتج معين.

تحديد استجابات الأخطاء:

- يجب أن يكون API قادرًا على إرسال رسائل خطأ واضحة عند حدوث مشاكل. على سبيل المثال، إذا حاول المستخدم الوصول إلى مورد غير موجود، يجب أن يرسل الـ API استجابة HTTP مع رمز حالة مناسب مثل 404.(Not Found)

4. أمن تطبيقات الويب

4.1 التهديدات الشائعة في تطبيقات الويب:

تطبيقات الويب معرضة لعدة أنواع من الهجمات، بعضها يمكن أن يؤدي إلى فقدان البيانات أو الوصول غير المصرح به إلى النظام. إليك أبرز التهديدات:

- **الحقن(Injection Attacks) :** هجمات الحقن مثل **حقن SQL (SQL Injection)** و **حقن شيل (Shell Injection)**
- **التخزين عبر المواقع:(Cross-Site Scripting - XSS) :** يحدث هذا الهجوم عندما يُسمح للمهاجم بحقن **سكربتات JavaScript**ضارة في صفحات الويب، التي يتم تحميلها لاحقًا على متصفح المستخدم.
- **التزوير عبر المواقع:(Cross-Site Request Forgery - CSRF) :** في هذا الهجوم، يقوم المهاجم بتزوير طلب من المستخدم إلى خادم موثوق، بحيث يتم تنفيذ عمليات غير مصرح بها
- **التسلل إلى الجلسات: :(Session Hijacking):** في هذا الهجوم، يحاول المهاجم سرقة **معرف الجلسة (Session ID)**
- **هجمات القوة العمياء:(Brute Force Attacks)** يحاول المهاجم تخمين كلمات المرور من خلال تجربة مجموعة ضخمة من التركيبات حتى يجد الكلمة الصحيحة.

4. أمن تطبيقات الويب

4.2 أفضل الممارسات لتأمين تطبيقات الويب:

1. الحماية ضد حقن:SQL
   - استخدام الاستعلامات المعدة مسبقًا (Prepared Statements):
   - التحقق من المدخلات (Input Validation):
2. الحماية ضد:XSS
   - ترميز المدخلات (Input Encoding):
   - استخدام Content Security Policy (CSP):
3. الحماية ضد:CSRF
   - استخدام الرموز المميزة (Tokens):
   - التأكد من المراجع (Referer and Origin headers):

4. أمن تطبيقات الويب

4. تأمين الجلسات:

- **توليد معرفات جلسة قوية وفريدة.**
- **استخدام HTTPOnly و Secure Cookies:**

5. حماية كلمة المرور:

- **استخدام تشفير قوي لكلمات المرور :**تخزين كلمات المرور بشكل مشفر باستخدام تقنيات مثل **bcrypt** أو **PBKDF2**
- **فرض سياسة كلمات مرور قوية**

6. التحقق من الهوية والتفويض:(Authentication and Authorization)

- **المصادقة متعددة العوامل :(MFA)**
- **التحقق من الصلاحيات :(Authorization)**

7. تحديثات أمنية دورية:يجب على فرق التطوير تطبيق تحديثات الأمان بشكل منتظم لتطبيقات الويب، وتثبيت تصحيحات الأمان.

اسئلة

س1: ما هي Node.jsو Express.jsوما أهميتهما في تطوير الخادم؟

س2: كيف يتم ربط التطبيقات مع قواعد البيانات باستخدام MongoDBأو MySQL؟

س3: ما هي RESTful APIsوكيف يتم تصميمها؟

س4: ما هي التهديدات الشائعة التي تواجه تطبيقات الويب؟

س5: ما هي أفضل الممارسات لتأمين تطبيقات الويب؟

س6: كيف يمكن تأمين الـ RESTful APIs؟

الاجوبة

ج1:

- **Node.js:** منصة تشغيل مفتوحة المصدر تعتمد على محرك V8 من Google لتشغيل JavaScript على الخادم، وتتميز بسرعة الأداء ودعم العمليات غير المتزامنة Asynchronous.
- **Express.js:** إطار عمل لتطوير تطبيقات الويب باستخدام Node.js، يتيح بناء الخوادم بسهولة ويوفر ميزات مثل التوجيه (Routing) وإدارة الطلبات والاستجابات.

أهميتهما تكمن في:

- تبسيط عملية تطوير خوادم ويب فعالة وسريعة.
- دعم بناء تطبيقات قابلة للتوسع ومناسبة للتطبيقات في الوقت الحقيقي.

ج2:

- **MongoDB:** قاعدة بيانات NoSQL تُستخدم مع Node.js باستخدام مكتبة Mongoose، وتخزن البيانات بصيغة JSON، مما يجعلها مناسبة للتطبيقات الديناميكية.

MySQL: قاعدة بيانات علائقية تُستخدم مع Node.js باستخدام مكتبة mysql2 وتخزن البيانات في جداول.

الاجوبة

ج3:

RESTful APIsهي واجهات برمجة تطبيقات تتبع نمط REST، وهو أسلوب معماري يستخدم بروتوكول HTTP. لتصميمها:

- استخدام طرق HTTPالقياسية (GET، POST، PUT، DELETE)لتحديد العمليات.
- تنظيم الموارد باستخدام عناوين URLواضحة
- إرسال البيانات بصيغ مثل JSONأو XML.
- توفير استجابات متسقة تحتوي على أكواد حالة) HTTPمثل 200 للنجاح و404 للموارد غير الموجودة)

ج4:

1. **XSS (Cross-Site Scripting):**هجوم يتم فيه إدخال أكواد ضارة في صفحات الويب التي يراها المستخدم.
2. **SQL Injection:**هجوم يتم فيه إدخال أوامر SQLضارة للتلاعب بقاعدة البيانات.
3. **CSRF (Cross-Site Request Forgery):**هجوم يتم فيه إجبار المستخدم على تنفيذ طلبات غير مرغوب فيها على موقع موثوق به.

الاجوبة

ج5:

- استخدام بروتوكولات HTTPSلتأمين البيانات أثناء النقل.
- التحقق من صحة إدخال المستخدم ((Input Validationلتجنب الهجمات مثل XSSو SQL Injection.
- تخزين كلمات المرور باستخدام خوارزميات التشفير مثل bcrypt.
- استخدام رموز CSRFلحماية التطبيقات من هجمات CSRF.
- تحديث المكتبات وإصلاح الثغرات الأمنية بانتظام.
- تحديد أذونات المستخدمين وفقًا لمبدأ أقل الامتيازات (.Principle of Least Privilege)

ج6:

- استخدام الرموز المميزة ((Tokensمثل JSON Web Tokens (JWT)للمصادقة.
- فرض التشفير باستخدام HTTPS.التحقق من صلاحيات المستخدمين قبل السماح بتنفيذ العمليات.
- الحد من الطلبات ((Rate Limitingلتجنب هجمات القوة الغاشمة.
- تقديم استجابات شاملة للأخطاء دون الكشف عن تفاصيل النظام الداخلية.

4. أمن تطبيقات الويب

4.3 استخدام الأدوات والتقنيات مثل OAuth وJWT

1. Oauth : هو بروتوكول تفويض مفتوح يسمح لتطبيقات الويب بمنح الوصول إلى بيانات المستخدمين من خلال تطبيقات أخرى، دون الحاجة لمشاركة كلمات المرور.

كيفية عمل:OAuth

- **تفويض المستخدم :**أولاً، يقوم المستخدم بتسجيل الدخول إلى التطبيق الذي يريد منحه صلاحية الوصول إلى بياناته.
- **إصدار رمز التفويض :**بعد تسجيل الدخول، يتم إرسال **رمز تفويض** (Authorization Code)إلى تطبيق الطرف الثالث.
- **إصدار الرمز المميز :(Access Token)** بعد استلام الرمز التفويضي، يقوم تطبيق الطرف الثالث بإرسال هذا الرمز إلى **خادم OAuth**للحصول على **رمز الوصول** (Access Token)الذي يتيح له الوصول إلى بيانات المستخدم.
- **الوصول إلى البيانات :**يُستخدم **رمز الوصول** لاسترجاع البيانات المطلوبة من الخادم.

4. أمن تطبيقات الويب

2. JSON Web Token (JWT)

JWTهو معيار مفتوح يستخدم بشكل رئيسي لنقل البيانات بشكل آمن بين العميل والخادم كـ **رمز مميز** يحتوي على معلومات محددة يمكن التحقق منها. يعتمد JWT على هيكل مكون من ثلاثة أجزاء :**الرأس(Header)** ، **الحمولة(Payload)** ، و **التوقيع(Signature)**

كيفية عمل:JWT

- **الرأس :(Header)** يحتوي على نوع الرمز) عادة (JWT وآلية التشفير المستخدمة) مثل HMAC SHA256 أو.(RSA
- **الحمولة :(Payload)** تحتوي على البيانات التي سيتم نقلها (مثل المعلومات الشخصية للمستخدم). قد تتضمن الحمولة بيانات مؤقتة.
- **التوقيع :(Signature)** يتم إنشاء التوقيع باستخدام **المفتاح السري** الذي يضمن أن البيانات لم تُعدّل أثناء النقل.

فوائد:JWT

- **مصداقية وأمان :**التوقيع الرقمي يضمن أن البيانات لم تُعدّل من قبل أطراف غير موثوقة.
- **محمول وسهل النقل :**يُمكن إرسال JWT عبر (**HTTP Headers**عادة في رأس (Authorization أو .**Cookies**
- **التحقق من الهوية والتفويض :**يمكن استخدام JWT لتوثيق الهوية والتحقق من الصلاحيات داخل تطبيقات الويب.

روابط خارجية

عنوان الفيديو	الرابط
تعلم برمجة الويب من الصفر	https://www.youtube.com/watch?v=E4veOfnLu0E&list=PL7VOgFQ42C7e-lWJiV4QeqRg3GqCOgoab
html css js php sql + projects	https://www.youtube.com/watch?v=WWn8tOHjZbw&list=PLSiLeKadTQ7kF7p-kd3gkHr6BAwQKYYSv
تعلم html	https://www.youtube.com/watch?v=q3yFo-t1ykw

المراجع

- برمجة تطبيقات الويب
- Programing PHP
- تصميم تطبيقات الويب

شكرا لكم