CRUD

الإنشاء والقراءة والتحديث والحذف (CRUD) هي الوظائف الأربع الأساسية التي يجب أن تكون النماذج قادرة على القيام بها على الأكثر.

إنشاء، قراءة، تحديث، حذف

عندما نقوم ببناء واجهات برمجة التطبيقات، نريد أن توفر نماذجنا أربعة أنواع أساسية من الوظائف. يجب أن يكون النموذج قادرًا على إنشاء الموارد وقراءتها وتحديثها وحذفها. غالبًا ما يشير علماء الكمبيوتر إلى هذه الوظائف بالاختصار CRUD. يجب أن يتمتع النموذج بالقدرة على أداء هذه الوظائف الأربع على الأكثر حتى يكتمل. إذا لم يكن من الممكن وصف الإجراء بإحدى هذه العمليات الأربع، فمن المحتمل أن يكون نموذجًا خاصًا به.

يعد نموذج CRUD شائعًا في إنشاء تطبيقات الويب، لأنه يوفر إطارًا لا يُنسى لتذكير المطورين بكيفية إنشاء نماذج كاملة وقابلة للاستخدام. على سبيل المثال، دعونا نتخيل نظامًا لتتبع كتب المكتبة. في قاعدة بيانات المكتبة الافتراضية هذه، يمكننا أن نتخيل أنه سيكون هناك مصدر للكتب، والذي من شأنه تخزين كائنات الكتاب. لنفترض أن كائن الكتاب يبدو كما يلي:

“book”: {

"id": <Integer>,

“title”: <String>,

“author”: <String>,

“isbn”: <Integer>

}

لجعل نظام المكتبة هذا قابلاً للاستخدام، نود التأكد من وجود آليات واضحة لاستكمال عمليات CRUD:

إنشاء - يتكون هذا من وظيفة يمكن أن نطلق عليها اسم عند إضافة كتاب مكتبة جديد إلى الكتالوج. سيوفر البرنامج الذي يستدعي الوظيفة قيم "العنوان" و"المؤلف" و"isbn". بعد استدعاء هذه الوظيفة، يجب أن يكون هناك إدخال جديد في مورد الكتب المطابق لهذا الكتاب الجديد. بالإضافة إلى ذلك، يتم تعيين معرف فريد للإدخال الجديد، والذي يمكن استخدامه للوصول إلى هذا المورد لاحقًا.

القراءة - سيتكون هذا من وظيفة سيتم استدعاؤها لرؤية جميع الكتب الموجودة حاليًا في الكتالوج. لن يؤدي استدعاء الوظيفة هذا إلى تغيير الكتب الموجودة في الكتالوج - بل سيؤدي ببساطة إلى استرداد المورد وعرض النتائج. سيكون لدينا أيضًا وظيفة لاسترداد كتاب واحد، حيث يمكننا توفير العنوان أو المؤلف أو رقم ISBN له. مرة أخرى، لن يتم تعديل هذا الكتاب، بل سيتم استرجاعه فقط.

التحديث — يجب أن تكون هناك وظيفة يتم الاتصال بها عند الحاجة إلى تغيير معلومات حول كتاب ما. سيوفر البرنامج الذي يستدعي الوظيفة القيم الجديدة لـ "العنوان" و"المؤلف" و"isbn". بعد استدعاء الوظيفة، سيحتوي الإدخال المقابل في مورد الكتب على الحقول الجديدة المتوفرة.

حذف - يجب أن تكون هناك وظيفة يمكن الاتصال بها لإزالة كتاب المكتبة من الكتالوج. سيوفر البرنامج الذي يستدعي الوظيفة قيمة واحدة أو أكثر ("العنوان" و/أو "المؤلف" و/أو "isbn") لتحديد الكتاب، ومن ثم ستتم إزالة هذا الكتاب من مصدر الكتب. بعد استدعاء هذه الوظيفة، يجب أن يحتوي مصدر الكتب على جميع الكتب الموجودة به من قبل، باستثناء الكتاب الذي تم حذفه للتو.

الخام والراحة

في بيئة REST، غالبًا ما يتوافق CRUD مع أساليب HTTP POST وGET وPUT وDELETE، على التوالي. هذه هي العناصر الأساسية لنظام التخزين المستمر.

خلال بقية المقالة، سنوصي بالمعايير وأكواد الاستجابة التي يتبعها المطورون عادةً عند إنشاء تطبيقات RESTful. قد تختلف الاتفاقيات، لذا لا تتردد في تجربة قيم ورموز إرجاع مختلفة عندما تصبح مرتاحًا لنموذج CRUD.

تخيل أننا نعمل مع نظام يتتبع الوجبات وأسعارها المقابلة للمطعم. دعونا نلقي نظرة على كيفية تنفيذ عمليات CRUD.

Create

لإنشاء موارد في بيئة REST، نستخدم بشكل شائع طريقة HTTP POST. يقوم POST بإنشاء مورد جديد من نوع المورد المحدد.

على سبيل المثال، لنتخيل أننا نقوم بإضافة عنصر طعام جديد إلى قائمة الأطباق المخزنة لهذا المطعم، ويتم تخزين كائنات الطبق في مورد الأطباق. إذا أردنا إنشاء عنصر جديد، فسنستخدم طلب POST:

Request:

POST http://www.myrestaurant.com/dishes/

Body -

{

"dish": {

"name": “Avocado Toast”,

"price": 8

}

}

يؤدي هذا إلى إنشاء عنصر جديد بقيمة اسم "Avocado Toast" وقيمة سعر 8. عند الإنشاء الناجح، يجب أن يعرض الخادم رأسًا يحتوي على رابط إلى المورد الذي تم إنشاؤه حديثًا، بالإضافة إلى رمز استجابة HTTP بقيمة 201 ( CREATED).

Response:

Status Code - 201 (CREATED)

Body -

{

"dish": {

"id": 1223,

"name": “Avocado Toast”,

"price": 8

}

}

من هذا الرد، نرى أن الطبق الذي يحمل الاسم "Avocado Toast" والسعر 8 قد تم إنشاؤه بنجاح وإضافته إلى مورد الأطباق.

Read

لقراءة الموارد في بيئة REST، نستخدم طريقة GET. لا ينبغي أن تؤدي قراءة المصدر إلى تغيير أي معلومات، بل يجب فقط استرجاعها. إذا اتصلت بـ GET للحصول على نفس المعلومات 10 مرات متتالية، فيجب أن تحصل على نفس الرد في المكالمة الأولى الذي تلقيته في المكالمة الأخيرة.

يمكن استخدام GET لقراءة قائمة كاملة من العناصر:

Request:

GET http://www.myrestaurant.com/dishes/

Response: Status Code - 200 (OK)

Body -

{

"dishes": [

{

"id": 1,

"name": “Spring Rolls”,

"price": 6

},

{

"id": 2,

"name": “Mozzarella Sticks”,

"price": 7

},

...

{

"id": 1223,

"name": “Avocado Toast”,

"price": 8

},

{

"id": 1224,

"name": “Muesli and Yogurt”,

"price": 5

}

]

}

GET requests can also be used to read a specific item, when its id is specified in the request:

Request:

GET http://www.myrestaurant.com/dishes/1223

Response: Status Code - 200 (OK)

Body -

{

"id": 1223,

"name": “Avocado Toast”,

"price": 8

}

بعد هذا الطلب، لم يتم تغيير أية معلومات في قاعدة البيانات. تم استرداد العنصر ذو المعرف 1223 من مورد الأطباق، ولم يتم تعديله. في حالة عدم وجود أخطاء، سيعيد GET HTML أو JSON للمورد المطلوب، بالإضافة إلى رمز الاستجابة 200 (OK). إذا كان هناك خطأ، فغالبًا ما يُرجع رمز الاستجابة 404 (لم يتم العثور عليه).

Update

PUT هي طريقة HTTP المستخدمة لعملية CRUD، التحديث.

على سبيل المثال، إذا ارتفع سعر توست الأفوكادو، فيجب علينا الدخول إلى قاعدة البيانات وتحديث تلك المعلومات. يمكننا القيام بذلك من خلال طلب PUT.

Request:

PUT http://www.myrestaurant.com/dishes/1223

Body -

{

"dish": {

"name": “Avocado Toast”,

"price": 10

}

}

يجب أن يقوم هذا الطلب بتغيير العنصر بالمعرف 1223 للحصول على السمات المتوفرة في نص الطلب. يجب أن يحمل هذا الطبق الذي يحمل المعرف 1223 الآن اسم "Avocado Toast"، ولكن يجب أن تكون قيمة السعر الآن 10، بينما كانت في السابق 8.

Response: Status Code - 200 (OK)

Body -

{

"dish": {

"name": “Avocado Toast”,

"price": 10

}

}

تتضمن الاستجابة رمز الحالة 200 (موافق) للإشارة إلى نجاح العملية. اختياريًا، يمكن أن تستخدم الاستجابة رمز الحالة 204 (لا يوجد محتوى) ولا تتضمن نص الاستجابة. هذا القرار يعتمد على السياق.

Delete

تتوافق عملية حذف CRUD مع طريقة HTTP DELETE. يتم استخدامه لإزالة مورد من النظام.

لنفترض أن النقص العالمي في الأفوكادو قد وصل إلى نقطة حرجة، ولم يعد بإمكاننا تحمل تكلفة تقديم هذه الأطعمة الشهية الحديثة على الإطلاق. يجب أن نذهب إلى قاعدة البيانات ونحذف العنصر الذي يتوافق مع "Avocado Toast"، والذي نعرف أن رقم تعريفه هو 1223.

طلب:

احذف http://www.myrestaurant.com/dishes/1223

مثل هذه المكالمة، في حالة نجاحها، تقوم بإرجاع رمز استجابة 204 (لا يوجد محتوى)، بدون نص استجابة. يجب ألا يحتوي مورد الأطباق على كائن الطبق بالمعرف 1223.

الرد: رمز الحالة - 204 (لا يوجد محتوى)

الجسم - لا يوجد

سيؤدي استدعاء GET على مورد الأطباق بعد استدعاء DELETE هذا إلى إرجاع القائمة الأصلية للأطباق مع إزالة الإدخال {"id": 1223، "name": "Avocado Toast"، "price": 10. يجب أن تظل كافة كائنات الطبق الأخرى في مورد الأطباق دون تغيير. إذا حاولنا استدعاء GET على العنصر بالمعرف 1223، والذي قمنا بحذفه للتو، فسوف نتلقى رمز الاستجابة 404 (لم يتم العثور عليه) ويجب أن تظل حالة النظام دون تغيير.

لا ينبغي أن يؤدي استدعاء DELETE على مورد غير موجود إلى تغيير حالة النظام. يجب أن تقوم المكالمة بإرجاع رمز الاستجابة 404 (غير موجود) وعدم القيام بأي شيء.