المشروع الإرشادي

تصميم واجهة برمجة تطبيق بايثون فلاسك: إنشاء واجهة لمدونة

وقت المشروع
60 دقيقة

محمد الانصاري

محاضر في علوم البيانات والذكاء الاصطناعي

عملت كمهندس برمجيات لمدة أربع سنوات ومهندس تعلم اله لمدة سنتين

حاصل على ماجستير الهندسة في الذكاء الاصطناعي وعلوم البيانات من جامعة أوتاوا

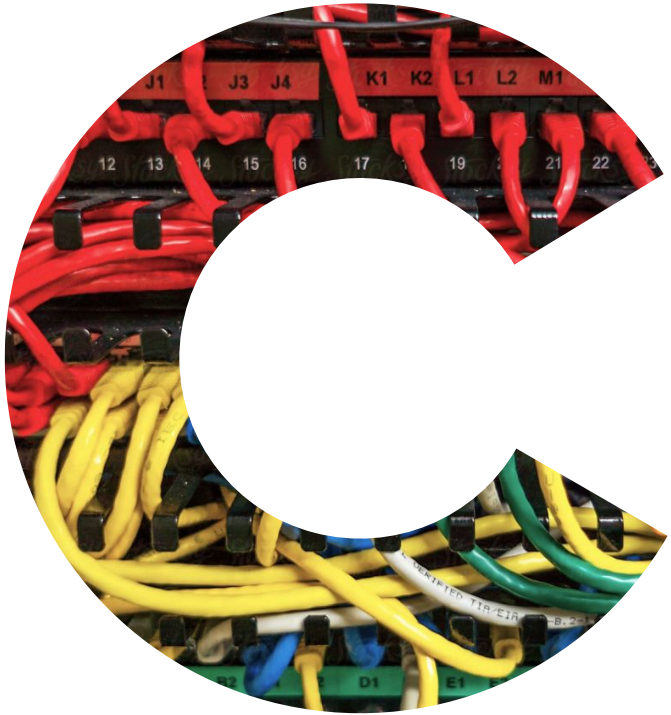
هدف المشروع

إنشاء منصة تدوين متطورة مع إمكانات RESTful API المتكاملة وإنشاء تجربة تدوين تتمحور حول المستخدم و غنية بالميزات. بالإضافة إلى ذلك، فالمنصة توفر واجهة برمجة تطبيق قوية للمطورين الذين يتطلعون إلى التكامل مع النظام الأساسي الخاص بك، مما يفتح إمكانيات الإضافات والتكاملات.

السيناريو

هذا المشروع هو حل لمدون او Blogger يريد نشر مقالاته بدون التقيد بالقيود التي تفرضها مواقع التواصل الاجتماعي ويريد السماح لاى تطبيق اخر ان يقوم بعرض تدويناته وبالتالي في حاجة لنشر نقطه نهايه أو endpoint في واجهة برمجة التطبيق API لكي تقوم باسترداد التدوينات.

مهمتك كمهندس برمجيات هي ان تقوم بتصميم واجهة برمجة التطبيق API وانشاء نقاط النهاية endpoints وتخزين التدوينات بشكل دائم في قاعدة بيانات MySQL وعمل اصدارات لواجهة برمجة التطبيق وهذا ما سنقوم بتنفيذه خلال المشروع.



التعريف ب HTTP و REST وارسال طلب باستخدام cURL و Postman

بنهاية هذه المهمة ستكون على دراية كافية بكيفية عمل HTTP وستعرف ما هو REST وكيف نقوم بإرسال طلب باستخدام cURL و Postman

ملخص المهمة

هدف المهمة

بنهاية المهمة الاولى هتكون علي درايه كافيه بطريقة عمل HTTP وبمفهوم REST وكيف نقوم بإرسال طلب باستخدام Postman و cURL.

النقاط الرئيسية

- طلب HTTP يحتوي على الفعل verb والرؤوس headers والمحتوى content
- استجابة HTTP تحتوي على رمز الاستجابة status code والرؤوس headers والمحتوى content
- أشهر الافعال في HTTP هي GET و POST و PUT و PATCH و DELETE
- أشهر الرؤوس headers التي يتم استخدامها هي نوع المحتوى المرسل Content-Type ونوع المحتوى العائد في الاستجابة Accept والتفويض Authorization وملفات تعريف الارتباط Cookies
- رموز الاستجابة من 100-199 يتم استخدامها في المعلومات ومن 200-299 لنجاح الطلب ومن 300-399 لإعادة التوجيه ومن 400-499 لأخطاء العميل ومن 500-599 لأخطاء الخادم
- مفهوم REST مبني على الفصل بين العميل والخادم Client and Server بحيث ان الطلبات requests تكون stateless أو لا يوجد حاله لها ويتم نسيانها من قبل الخادم بمجرد رجوع الاستجابة ويمكن تخزينها مؤقتا في الـ Cache وتكون موحدة Uniform

تصميم واجهة برمجة تطبيق RESTful احترافية وإعداد المشروع

بنهاية هذه المهمة ستكون على دراية كافية بكيفية تصميم واجهة برمجة تطبيق احترافية وإعداد المشروع للبدء في التنفيذ

ملخص المهمة

هدف المهمة

بنهاية هذه المهمة ستكون على دراية كافية بكيفية تصميم واجهة برمجة تطبيق احترافية

النقاط الرئيسية

- لا يمكننا إصلاح API بعد نشرها للمطورين ولذلك يجب تصميم الـ API بعنايه شديده
- الروابط ما هي إلا مسارات للموارد مثل العملاء والفواتير والمستخدمين وغيرها
- يتم استخدام سلاسل الاستعلام في للمهام التي لا تتعلق بالبيانات مثل التنسيق والترتيب والبحث
- في تصميم ال API من الضروري استخدام الاسماء بدل الافعال مثل /customers بدلا من /getCustomers
- يتم استخدام الفعل GET لاسترداد البيانات والفعل POST لإنشاء بيان جديد والفعل PUT لتحديث بيانات بيان والفعل PATCH لتحديث جزء من بيانات بيان والفعل DELETE لحذف بيان
- الافعال GET و PUT و PATCH و DELETE لها نفس الاستجابة مهما كان عدد مرات الطلب لكن POST لا يمكن أن تعطي نفس الاستجابة كل مرة لنفس الطلب لان من المنطقي وجود بعض الأعمدة التي قد تكون ذات قيم فريدة لا يمكن تكرارها

بناء واجهة برمجة تطبيق RESTful بإستخدام Flask

بناء واجهة برمجة تطبيق RESTful بإستخدام Flask
بناء واجهة برمجة تطبيق RESTful بإستخدام Flask

بنهاية هذه المهمة سنكون قد قمنا بإنشاء اتصال مع قاعدة البيانات وإنشاء أول نهايه في واجهة برمجة التطبيق والمسؤولة عن إنشاء تدوينة جديدة

ملخص المهمة

هدف المهمة

بنهاية هذه المهمة سنكون قد قمنا بإنشاء اتصال مع قاعدة البيانات وإنشاء أول نهايه في واجهة برمجة التطبيق والمسؤولة عن إنشاء تدوينة جديدة

النقاط الرئيسية

- يتم الاحتفاظ بالمكتبات والحزم في ملف requirements.txt ليتم تنصيبها مرة واحدة باستخدام الأمر pip install -r requirements.txt
- اسم مجلد الـ views الأساسي في Flask هو templates ويتم استرجاع ملفات html عن طريق الداله render_template
- يتم استخدام SQLAlchemy للاتصال بقاعدة البيانات
- يجب إنشاء Model لكن جدول في قاعدة البيانات عن طريق إنشاء class يورث من db.Model
- يستخدم الفعل POST لإنشاء بيان (تدوينه) جديد
- يعتبر فحص المحتوى المرسل من المستخدم امرا اساسيا قبل البدء في تخزين البيانات في قاعدة البيانات
- يتم إرجاع البيانات في اغلب الاوقات في صوره JSON
- يتم استخدام الأمر 201 للدلالة على أن البيان تم إنشاؤه بنجاح أما 500 فيستخدم للدلالة على حدوث مشكلة في الخادم و 400 للدلالة على وجود مشكلة في فحص البيانات

مشروع ارشادي

نشاط تطبيقي

هذه المهمة اختيارية وغير مصنفة. الهدف هو التحقق من فهمك.

نشاط تطبيقي

1. أنشئ نقطة نهاية من النوع GET فى واجهة برمجة التطبيق في المورد Posts وسوف تكون مسؤولة عن استرجاع كل التدوينات في المدونة.
2. ستحتاج في البداية لعمل استعلام علي ال model ثم أخذ القيم كلها يمكن تنفيذ ذلك على أي model بصورة عامة عن طريق الكود

```
model.query.all()
```

3. بعد ذلك ستحتاج الي استخدام list comprehension أو المرور على كل القيم الراجعة من الاستعلام حتي يتم تحويل ال PostModel لقاموس dict قبل ما يتم إرجاعه من ال API أو الواجهة ك JSON.
4. سيكون رمز الاستجابة 200 ومعناه أن تم الطلب بنجاح.
5. اخيرا اختبر الطلب والاستجابة في Postman.

(أوقف الفيديو مؤقتًا لإكمال المهمة وقم بإلغاء الإيقاف لرؤية الحل بمجرد اكتمال المهمة)

إنشاء نقاط نهاية في واجهة برمجة التطبيق لاسترجاع بيانات تدوينة وأخرى لتحديث بياناتها وأخرى لحذفها

بنهاية هذه المهمة ستكون قد تعلمت كيفية إنشاء نقاط نهاية في واجهة برمجة التطبيق لاسترجاع بيانات تدوينة وأخرى لتحديث بياناتها وأخرى لحذفها

ملخص المهمة

هدف المهمة

بنهاية هذه المهمة ستكون قد تعلمت كيفية إنشاء نقاط نهاية في واجهة برمجة التطبيق لاسترجاع بيانات تدوينة وأخرى لتحديث بياناتها وأخرى لحذفها

النقاط الرئيسية

- يمكن استخدام الفعل GET إما لاسترجاع بيانات مجموعة من المورد أو استرجاع بيانات بيان محدد من المورد عن طريق معرف فريد كالرقم المعرف ID
- يستخدم رمز الاستجابة 404 للدلالة على أن البيان غير موجود
- يستخدم الفعل PUT لتعديل بيانات بيان محدد
- يستخدم الفعل DELETE لحذف بيان محدد

تعلم استراتيجيات عمل إصدار لواجهة برمجة التطبيق API

بنهاية هذه المهمة سنكون قد تعلمت استراتيجيات عمل إصدار لواجهة برمجة التطبيق (API) لضمان التوافق مع الإصدارات السابقة مع السماح بالتحديثات والتحسينات المستقبلية. وهذا سوف يستوعب كل من عمليات التكامل الحالية والجديدة.

ملخص المهمة

هدف المهمة

بنهاية هذه المهمة سنكون قد تعلمت استراتيجيات عمل إصدار لواجهة برمجة التطبيق (API) لضمان التوافق مع الإصدارات السابقة مع السماح بالتحديثات والتحسينات المستقبلية. وهذا سوف يستوعب كل من عمليات التكامل الحالية والجديدة.

النقاط الرئيسية

- بمجرد نشر واجهة برمجة التطبيق لا يمكن تغييرها إلا بموافقة مستخدمي الواجهة
- لا يجب الربط بين نسخ التطبيق ونسخ الواجهة فمن الممكن نزول تحديث للتطبيق مع استمرار الواجهة بنفس الاصداره
- من الضروري دعم النسخ القديمه والحاليه من الواجهة وإلغاء الوجهات القديمه بطريقه تناسب المستخدمين
- اصدار النسخه في الرابط يجعل الامر واضح جدا للمستخدم بالنسبه للنسخه التي يستخدمها من الواجهة ولكنه يتطلب تغيير كل الروابط مع تغيير اصداره الواجهة
- اصدار النسخة في سلسلة الاستعلام يجعله اختياري مع وجود قيمة أساسية لنسخة الواجهة في حاله عدم ارسال الاصداره في سلسلة الاستعلام لكن من الممكن أن يتسبب في تجاهل المستخدم للإصداره تماما واعتماده على الاصداره الأساسية
- استخدام رؤوس الطلب لتحديد الاصداره يقوم بفصل الاصداره عن باقي أجزاء الواجهة لكن في المقابل يتطلب مستخدم متمرس قادر على التعامل مع هذا النوع من الطلبات
- استخدام Accept Header المسؤول عن تحديد نوع البيانات العائد من الـ API لا يتطلب إنشاء نوع خاص من ال headers ولكنه غير واضح مقارنة باستخدام سلاسل الاستعلام
- استخدام Content-Type Header أو نوع المحتوى المرسل هو أكثرهم مرونة وانسبهم للتطبيقات الكبيرة حيث انه يسمح بتحديد اصداره البيانات المرسلة والبيانات العائده مع الفصل بين اصداراتهم ولكنه يتطلب وعي أكبر من المطور لاستخدامه

مشروع ارشادي

نشاط تطبيقي

هذه المهمة اختيارية وغير مصنفة. الهدف هو التحقق من فهمك.

نشاط تطبيقي

1. مطلوب منك عمل اصدارين بأرقام ١ و ٢ لنقطة النهاية **put** الموجودة بداخل الـ **class Post** والمسؤولة عن تحديث بيانات التدوينة.
2. استخدم **Query String** أو سلسلة الاستعلام لتحقيق هذا الغرض.
3. بإمكانك تسمية ال **parameter v** وتقدر على إحضار قيمة المعامل في سلسلة الاستعلام عن طريق الكود

```
request.args.get('parameter_name_here')
```

4. بالنسبة للطلب سيكون الـ **text** اختياري في الإصدارة الأولي واجباري في الثانية.
5. بالنسبة للاستجابة ستجعل التدوينة ترجع في حالة الإصدارة الأولى وحدها وفي حالة الاصدارة الثانية سترجع في مفتاح يسمى **post**
6. اخيرا لو المستخدم أرسل رقم اصداره خاطئ أو لم يرسل الرقم من الأساس سيكون ال **default** أو الأساسي الإصدار الأولي.

(أوقف الفيديو مؤقتًا لإكمال المهمة وقم بإلغاء الإيقاف لرؤية الحل بمجرد اكتمال المهمة)

مشروع ارشادي

تحدي مجمع

هذا التحدي اختياري و غير مصنف

الهدف هو بناء ثقتك بنفسك

سيناريو/التحدي

أثناء عملك كمهندس برمجيات في شركة تجارة الكترونية عندها ملايين من العملاء والمنتجات وطلبات الشراء التي تتم داخل الفروع التجارية للشركة بشكل يومي. فريق التسويق بالشركة اقترح تنفيذ موقع الكتروني لعرض المنتجات وتحديثها اونلاين وذلك لزيادة حجم المبيعات في الفروع. بعد البحث وجدوا أن أنسب طريقة هي تصميم وبرمجة واجهة برمجة تطبيق RESTful API لتكون مدعومة من اكثر من نظام وبالتالي تمكنهم حاليا من انشاء موقع الكتروني يستخدم هذه الواجهة لعرض البيانات وتمكنهم ايضا من انشاء تطبيق موبايل يستخدم نفس الواجهة في المستقبل هذا بالإضافة لاستخدام نفس الواجهة في النظام الداخلي للشركة لتحديث بيانات المنتجات. مهمتك كمهندس برمجيات هي ان تقوم بتصميم واجهة برمجة التطبيق RESTful API ثم تقوم بتنفيذها باستخدام Python Flask Framework وذلك لتكون متاحة للمطورين الآخرين عند انشاء موقع الكتروني او تطبيق موبايل للشركة أو استخدامها في النظام الداخلي للشركة لتحديث بيانات المنتجات.

- قم إنشاء تطبيق flask جديد
- قم بإنشاء اتصال بقاعدة بيانات SQLite
- قم بإنشاء Product Model لتعريف جدول المنتج وسيكون فيه الأعمدة التاليه:

- id أو المعرف الخاص بالمنتج عبارة عن رقم صحيح integer وسيكون مفتاح رئيسي primary key للجدول
- title او اسم المنتج عبارة عن نص string طوله بحد اقصي 255 حرف وسيكون معرف فريد في الجدول لا يمكن تكراره ولابد من تواجده
- description أو وصف المنتج عبارة عن نص text وسيكون اختياري
- sku أو الكود المستخدم في المخزن للمنتج عبارة عن نص string طوله بحد اقصي 255 حرف وسيكون معرف فريد في الجدول لا يمكن تكراره ولابد من تواجده
- images أو الصور ستكون JSON وهي عبارة عن قائمة روابط الصور للمنتج ويمكن أن تكون فارغة
- video link وهو فيديو توضيحي للمنتج وسيكون string أو نص طوله 255 حرف بحد أقصى ويمكن ان يكون فارغ
- price أو سعر المنتج وسيكون Float أو قيمة عشرية القيمة الافتراضية لها 0.0
- quantity أو الكمية وستكون integer والقيمة الافتراضية لها 0
- وقت الانشاء created_at وسيكون datetime أو تاريخ ووقت إنشاء المنتج وسيكون مطلوب
- اخر وقت للتعديل updated_at وسيكون datetime أو تاريخ ووقت آخر تعديل للمنتج وسيكون اختياري

سيناريو/التحدي

أثناء عملك كمهندس برمجيات في شركة تجارة الكترونية عندها ملايين من العملاء والمنتجات وطلبات الشراء التي تتم داخل الفروع التجارية للشركة بشكل يومي. فريق التسويق بالشركة اقترح تنفيذ موقع الكتروني لعرض المنتجات وتحديثها اونلاين وذلك لزيادة حجم المبيعات في الفروع. بعد البحث وجدوا أن أنسب طريقة هي تصميم وبرمجة واجهة برمجة تطبيق RESTful API لتكون مدعومة من اكثر من نظام وبالتالي تمكنهم حاليا من انشاء موقع الكتروني يستخدم هذه الواجهة لعرض البيانات وتمكنهم ايضا من انشاء تطبيق موبايل يستخدم نفس الواجهة في المستقبل هذا بالإضافة لاستخدام نفس الواجهة في النظام الداخلي للشركة لتحديث بيانات المنتجات. مهمتك كمهندس برمجيات هي ان تقوم بتصميم واجهة برمجة التطبيق RESTful API ثم تقوم بتنفيذها باستخدام Python Flask Framework وذلك لتكون متاحة للمطورين الآخرين عند انشاء موقع الكتروني او تطبيق موبايل للشركة أو استخدامها في النظام الداخلي للشركة لتحديث بيانات المنتجات.

- الان ستقوم بتعريف دالتين في ال Product Model دالة __repr__ لعرض تفاصيل عن المنتج في حالة طباعته ودالة as_dict لتحويل المنتج لقاموس بيانات
- أنشئ قاعدة البيانات لو لم تكن موجودة
- أنشئ ال API أو واجهة برمجة التطبيق
- أنشئ resource أو مورد للمنتجات اسمه Products وقم بتسجيله في الـ API تحت المسار /api/products
- أنشئ دالة post بداخل المورد Products والمسؤولة عن إنشاء منتج جديد
- أنشئ دالة get بداخل المورد Products والمسؤولة عن استرجاع كل المنتجات
- اختبر كل الطلبات باستخدام Postman