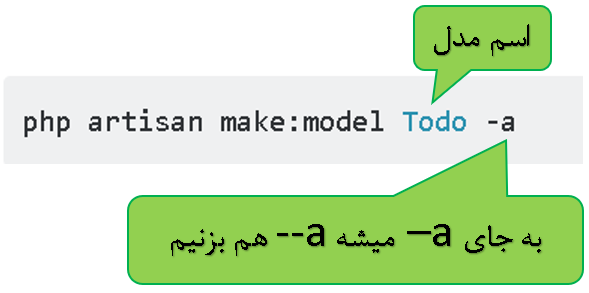
جزوه جلسه 6

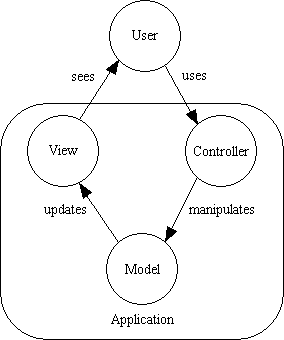
۱۸ مهر

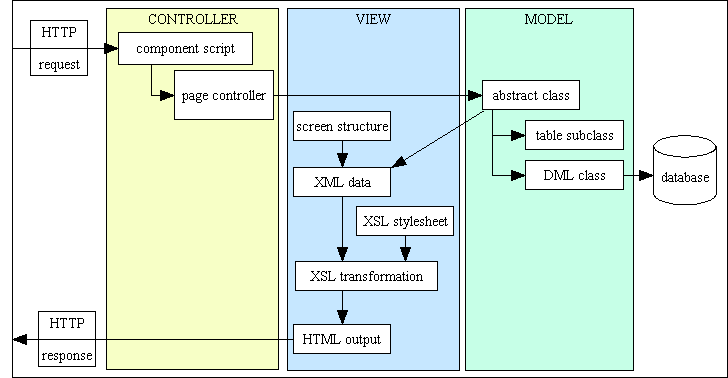
* ویدئو عالی با موضوع migration در یوتیوب: <https://www.youtube.com/watch?v=neSHAWdE44c>
* داکیومنت سایت مرجع در مورد migration: <https://laravel.com/docs/5.1/quickstart>
* داکیومنت سایت مرجع برای get کردن <https://laravel.com/docs/5.7/routing>
* داکیومنت سایت مرجع برای post‌ کردن <https://laravel.com/docs/5.8/eloquent>
* مایگریشن: مفهوم مشترک بین شبکه و نرم افزار که کاملا دو معنی جدا دارند.
  + در شبکه وقتی یک سرور را مثلا ایمیل سرور مایکروسافت که به نام اکسچینج سرور است میخواهید به نسخه ای بالاتر منتقل کنید یا سرور ها را جابجا کنید مایگریشن میگویند. چگونه دیتا ها و دیتابیس ها و کانفیگوریشن ها منتقل شود و نرم افزار هم ارتقا یابد و حتی سرور فیزیکی جابجا شود ولی همه چیز سر جایش باشد.
  + در برنامه نویسی ابتدا باید با مفهوم ORM آشنا شوید. یک مشکلی که همیشه در برنامه نویسی وجود داشت این بود که همه چیز شی گرا بود ولی دیتابیس خیر. کم کم متوجه شدند که خود دیتابیس را نمیتوان شی گرا کرد اما رفتار آن را میتوان شبیه شی گرایی نمود. دیتابیس که شیگرا بشه یعنی شما هیچ کد استاتیک نمیزنید و همون کلاس برنامه نویسی همه ی کارها رو انجام میدهد. حالا وقتی در لاراول فایل مایگریت میسازید چند کار انجام میدهید. اول اینکه میتوانید بفهمید روی هر سرور کدام نسخه نصب شده و تا هر جا که جلو رفتید بتوانید اتوماتیک توسط کامپیوتر آنرا ادامه دهید و بروز رسانی کنید. با خیال راحت و بدون نگرانی . زیرا در هر مرحله از کار برنامه نویسی تست شده و مطمئین هستید که اگر این تغییرات از هر جایی به بعد اتفاق بیافتد دیتابیس و کد همسان و هماهنگ و هم نسخه حرکت میکنند
* درباره ی get و post و انواع روش های صدا زدن وب سرویس ها بهتر است از postman استفاده کنید و کد تولید شده را استفاده کنید.
* دستور زیر که برای ایجاد MVC و migration است را در پروژه لاراولی که در جلسات پیش ساختی بزن:

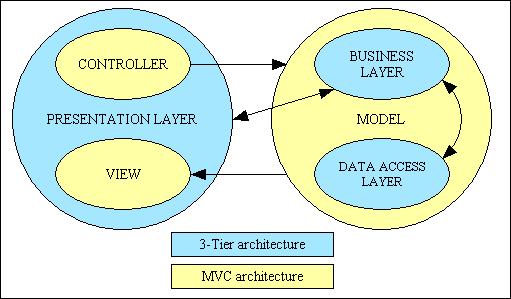


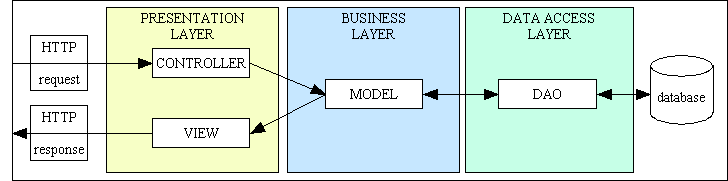
# MVC:

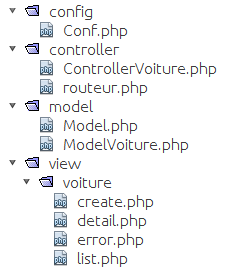
* مدل، رابط کنترلر با جدول است. یعنی مدل، راهنمای کنترلر است که کنترلر بفهمه جدول من چه فیلدهایی داره.
* در زبانهای دیگه باید MVC را بسازی. ولی در لاراول، Eloquent خودش واسطه ی اتصال به بانک و شناسایی فیلدها را برات میسازه.

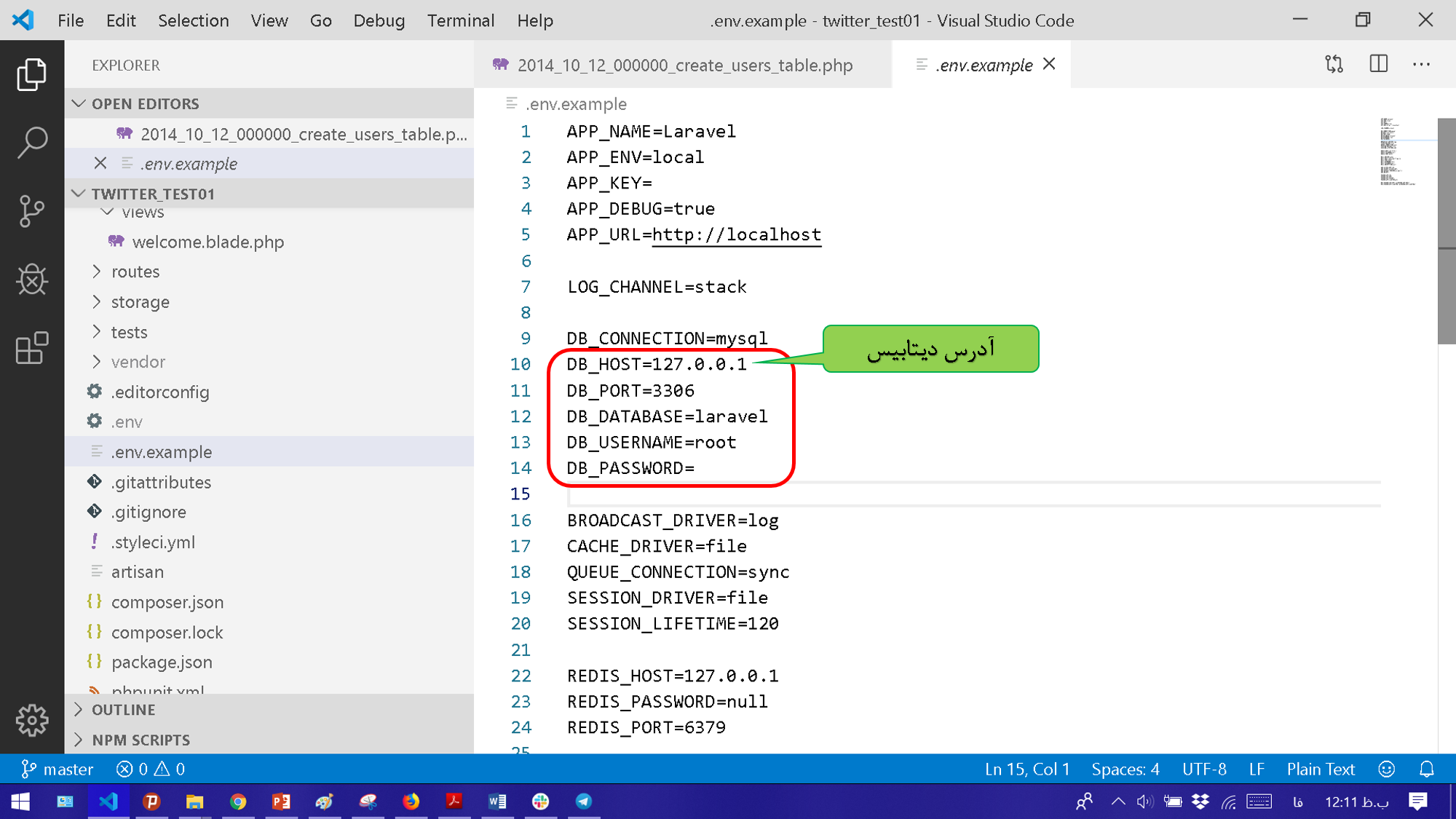








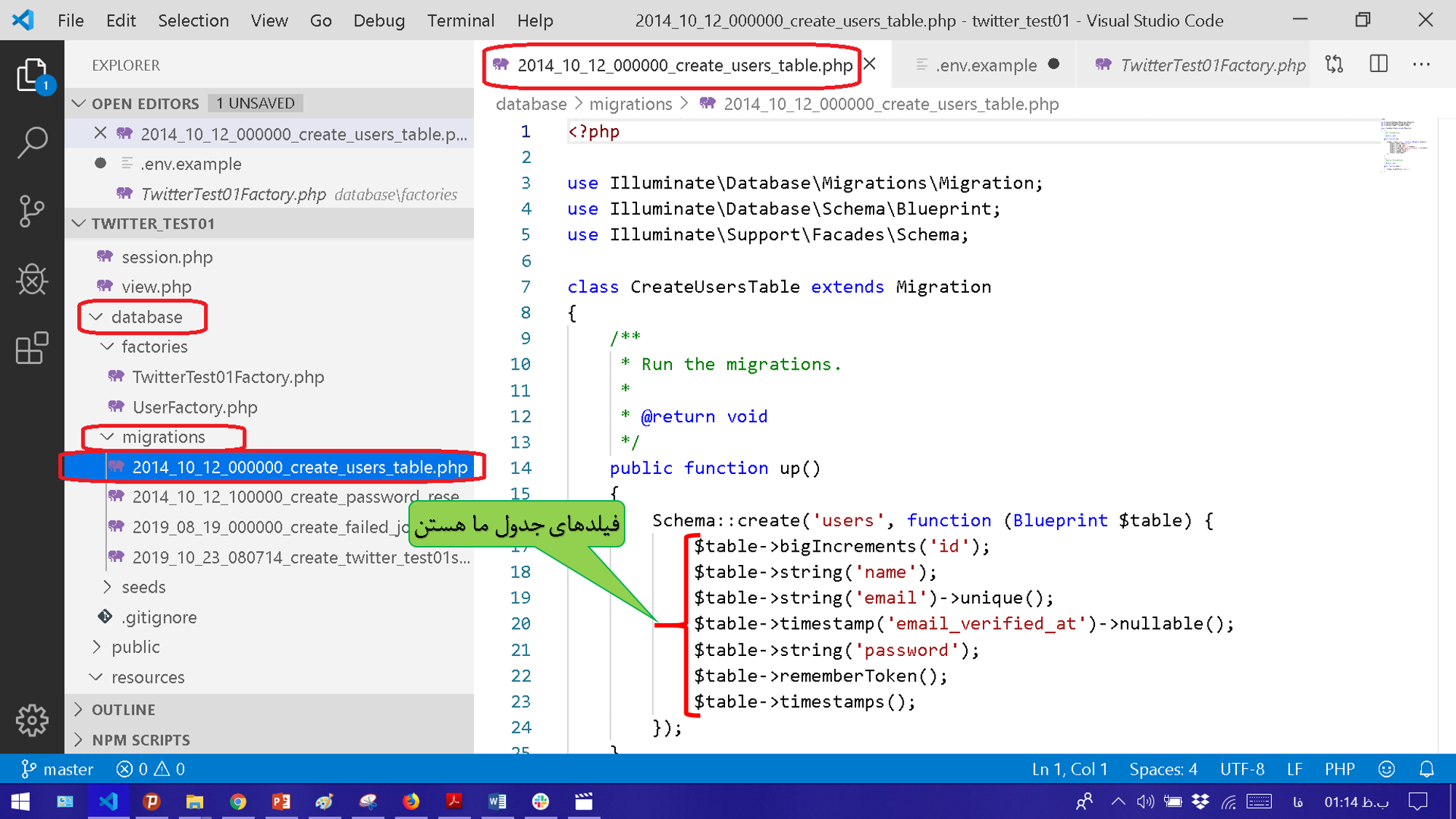




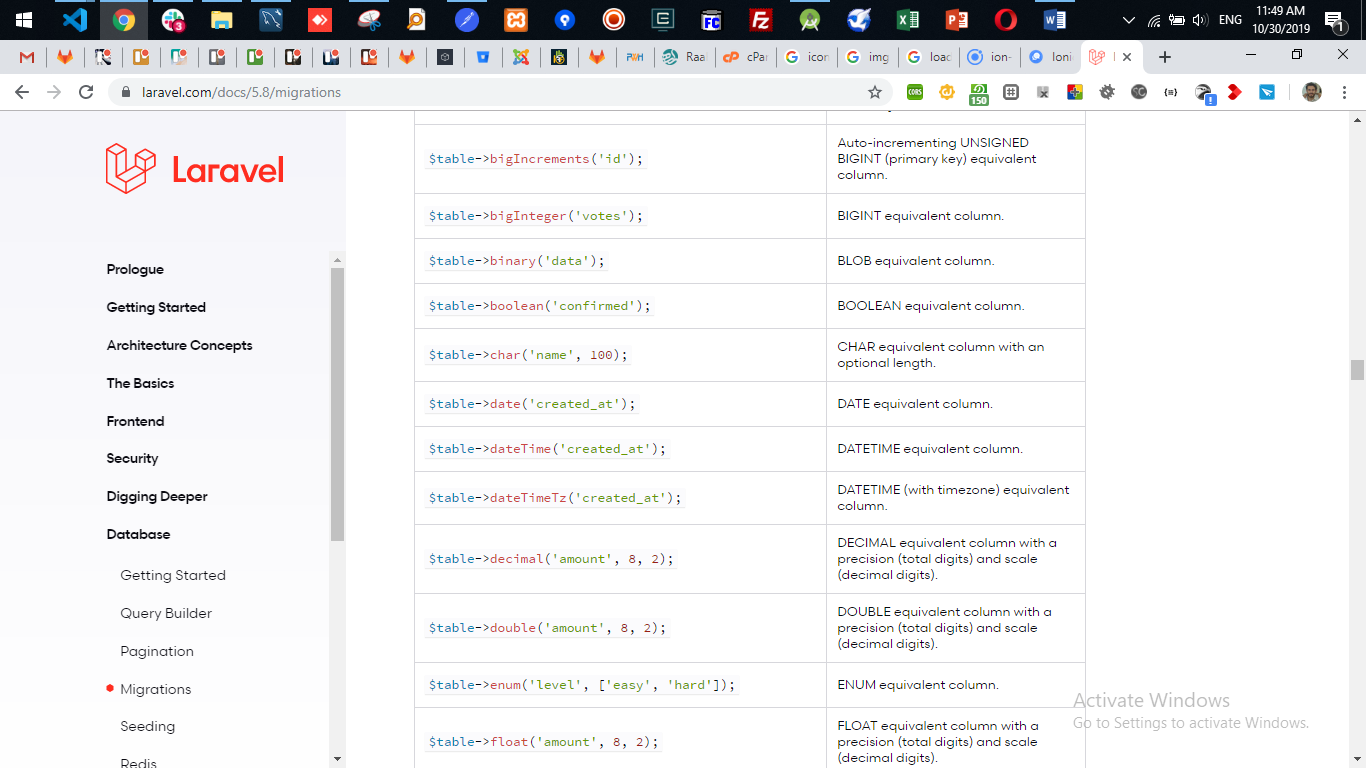
* قابلیتهای کلیدی یک برنامه نویس در تیم حرفه ای:

1. فهم صورت مسئله: چی میخوایم بسازیم ( ابزارها : مستند بک لاگ، جلسات اسکرام هفتگی، جلسات پوکر هفتگی، جلسات هر روز صبح)
2. تقسیم و پیشروی
3. مشورت
4. مطالعه
5. بعد از درک معماری و صورت مساله باید یاد بگیریم چی گوگل کنیم تا روش های مختلف برای پیاده سازی رو پیدا کنیم و تصمیم بگیریم بهترینش کدومه و گوگل کنیم چه اجزایی رو در گیت هاب و اینترنت در اختیار داریم و میخوایم به هم بچسبونیم
6. دیباگ و trace کردن ( یعنی خط به خط کد که نوشتیم رو با یک مقدار و سناریوی مشخص تست میکنیم)
7. Trouble shooting: یعنی مشکلات پیش آمده رو چطوری شناسایی و حل کنم
8. تعامل و کار تیمی
9. نظم

* Migration یه جور version Controller برای دیتابیس هم محسوب میشه.
* دستور migration: <https://laravel.com/docs/5.8/migrations>
  + php artisan migrate



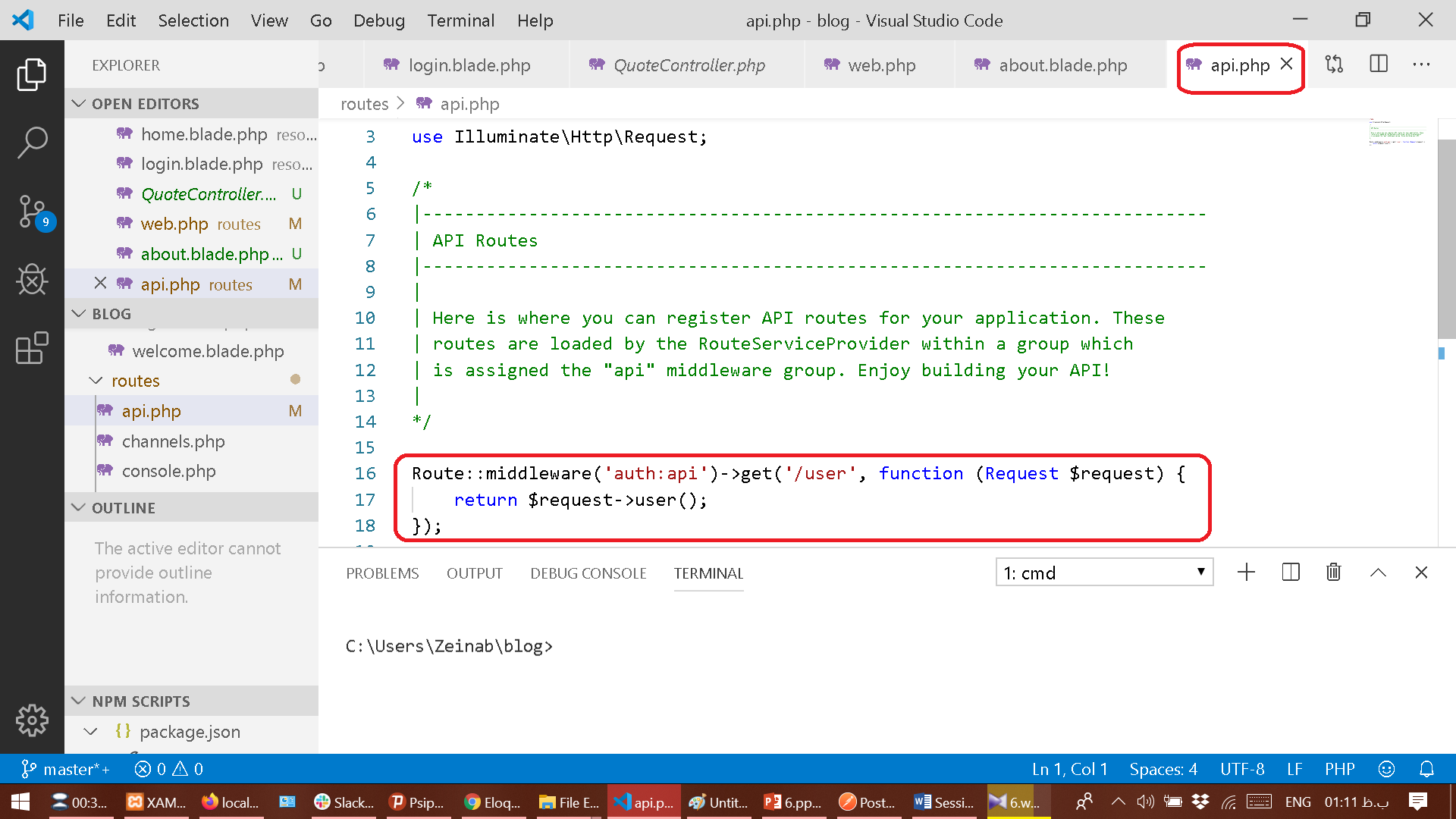
* نوع داده ای bigIncrement کار کلید اصلی رو انجام میده. یعنی خودش یکی یکی اضافه میشه.
* گوگل : **field** **type** in laravel

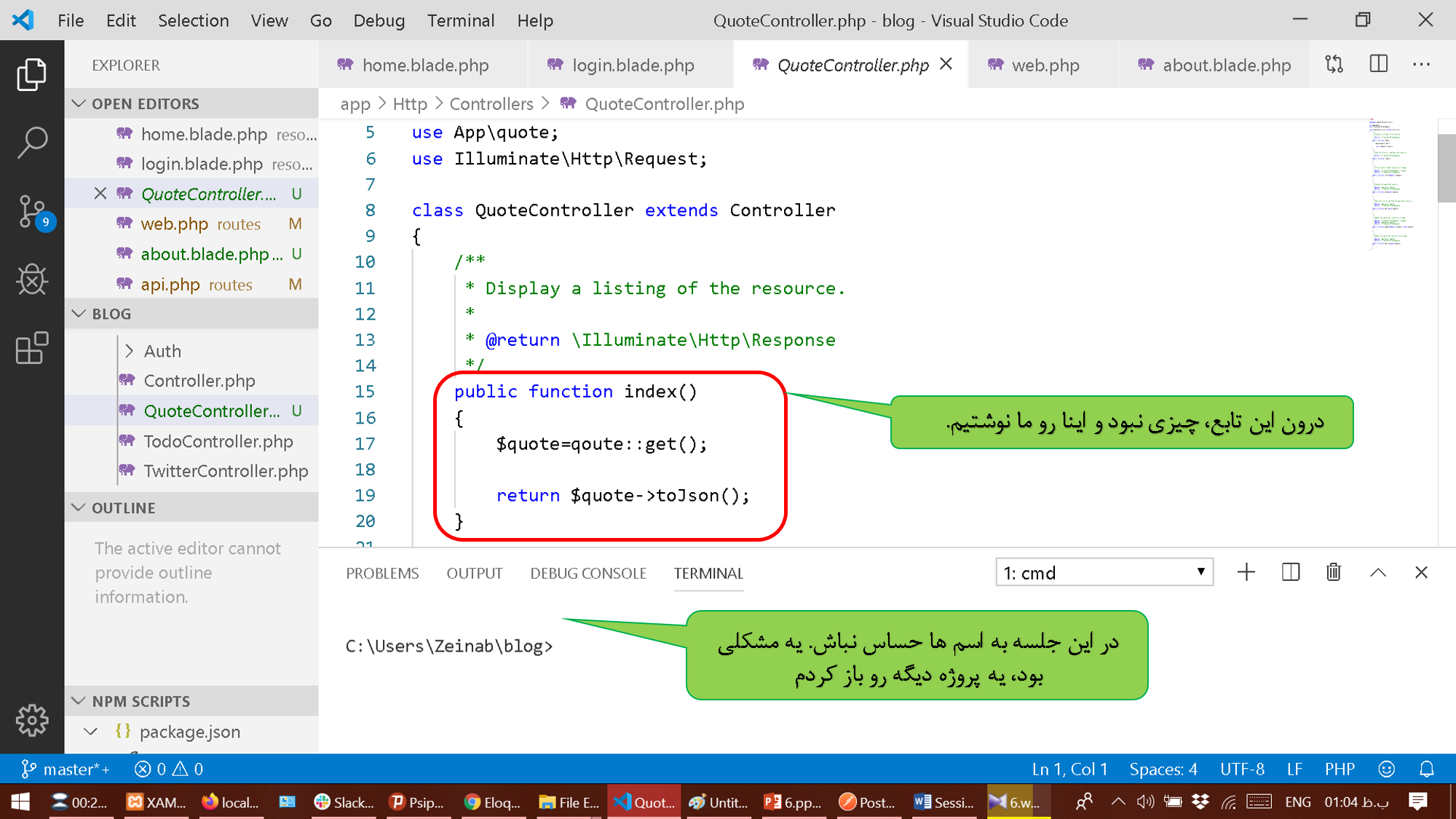


* Schema: یک شاکله از یک جدول در یک پایگاه داده. یعنی مثلا نوع داده های جدول، نام ستونها و...
* الان باید بتونی هر جدولی را با هر نوع فیلدی بسازی
* همه فایلها باینری هستن (عکس، فیلم، صوت و...)
* کد ملی که 10 رقم است رو بهتره در int(10) ذخیره کنیم یا string(10)؟
  + Int(10)<string(10) چون تعداد بیتهای int(10) برای نگهداری هر عدد کمتر از string(10) است.
  + ولی اینکه کدام بهتره بستگی به مورد کاربرد ما داره و جواب قاطعی برای آن وجود نداره.
* گوگل: every unicode size store: <https://stackoverflow.com/questions/5290182/how-many-bytes-does-one-unicode-character-take>
* میخوایم توییتر بسازیم. طبق سناریو جلسه قبلی چی نیاز داریم؟
  + یه برنامه موبایل داری که به کنترلر (وب سرویس) شما وصل میشه و اون وب سرویس وصل میشه به بانک
  + یادتون هست که وقتی وردپرس به عنوان سرور به ما خدمت میداد، خودش براتون وب سرویسها (rest api) رو ایجاد میکرد و بهتون api میداد و شما با پستمن، API های وردپرس را صدا میزدی.
  + یادآوری:
    - وردپرس نصب کردیم
    - با پستمن وب سرویس را صدا زدیم .. در حالت پیشفرض میتونی با وب سرویس posts ها و نوشته ها رو بخوانی.
    - با نصب pods در وردپرس، میتوانی هر entity جدیدی که دلت خواست بسازی مثلا یادت باشد quote را ساختیم و یه فیلدهایی را اضافه کردیم و رکوردهایی در اون ساختیم و خواندیم. (Set get)
  + الان با لاراول بانک را ساختیم و میخوایم یه وب سرویس ایجاد کنیم و با پستمن صداش بزنیم و بگیم این توییت را بخوان یا بذار توی بانک ت:
    - صورت مسئله: میخوایم با پستمن یه api‌ رو صدا بزنیم. توی اون api‌یه تابعی داشته باشیم که بره توی بانکمون یه چیزی رو بنویسه.
      * یعنی الان بانکمون رو ایجاد کردیم و با پستمن میخوایم یه مقادیری رو برای اون api با آدرسی مانند localhost8000/api/login پست کنیم. ولی این آدرس را لاراول هنوز نمیشناسه. راه حلش اینه که بریم در فایل api.php‌ و در آنجا، route اش و مسیرها رو تعریف کنیم.

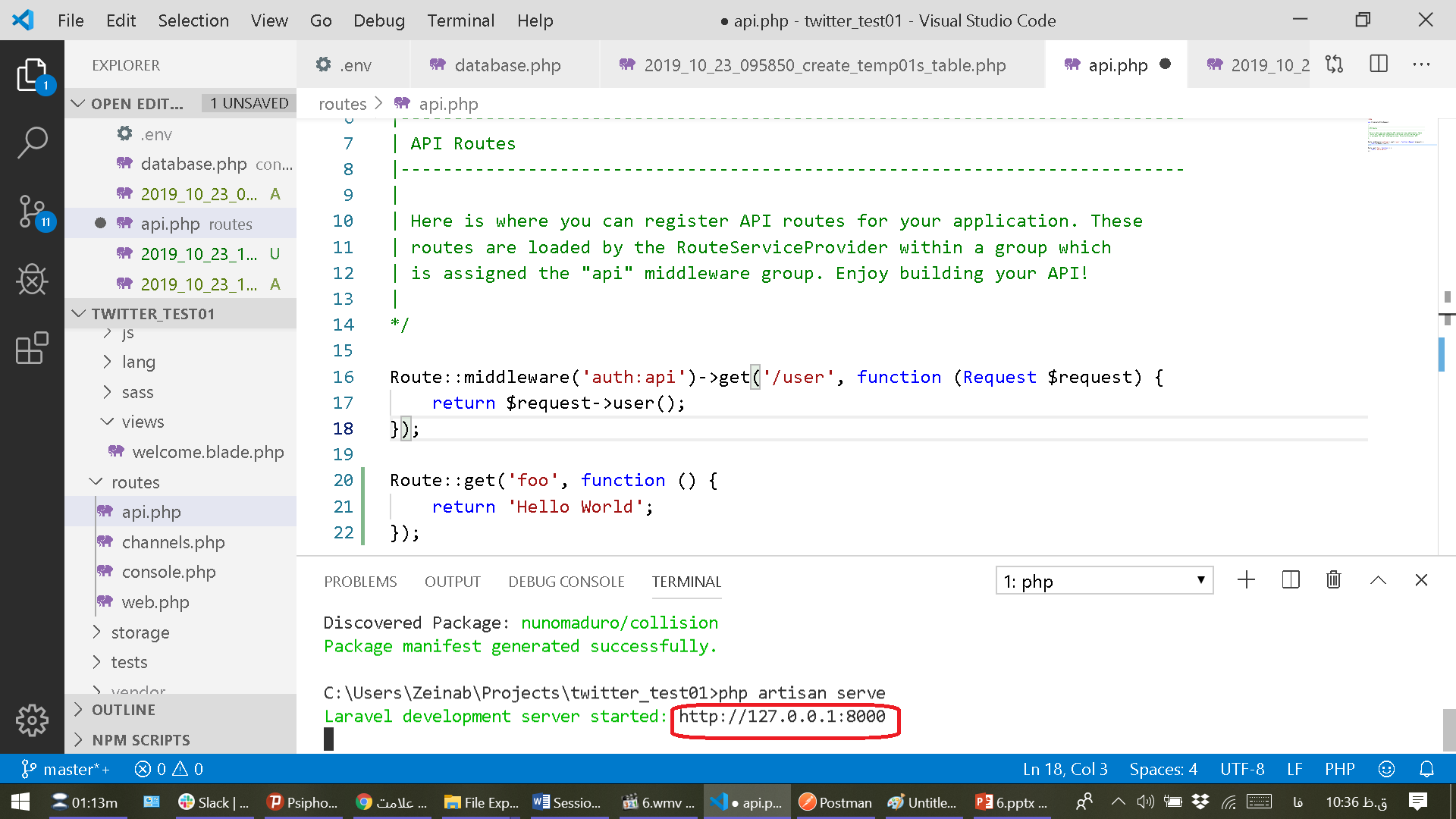
**میخوایم get کنیم**

* + - * گوگل: sample route api.php laravel (یه سمپل ببین و عین اون روانجام بده)
        + نتیجه گوگل: <https://laravel.com/docs/5.7/routing>

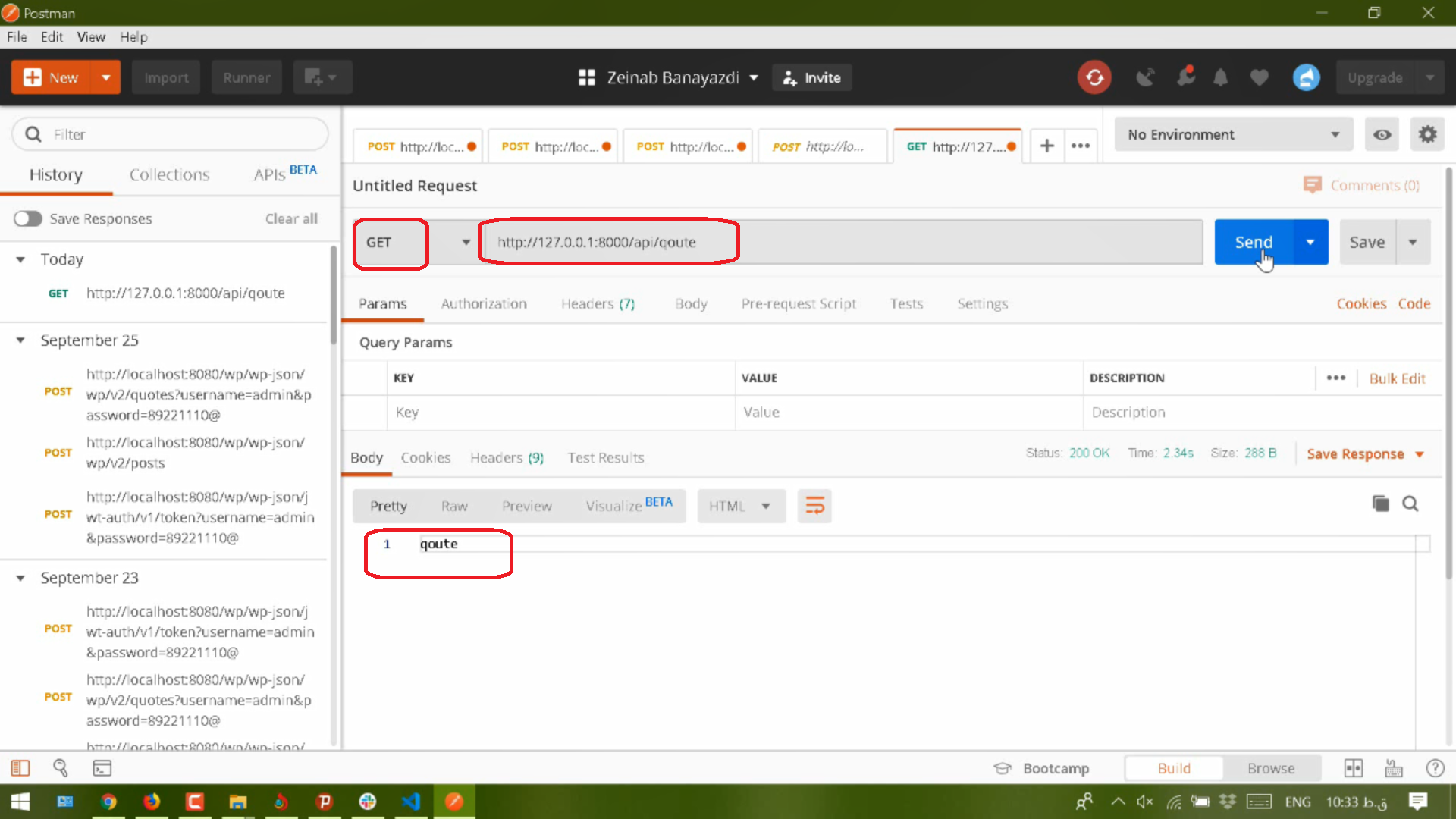




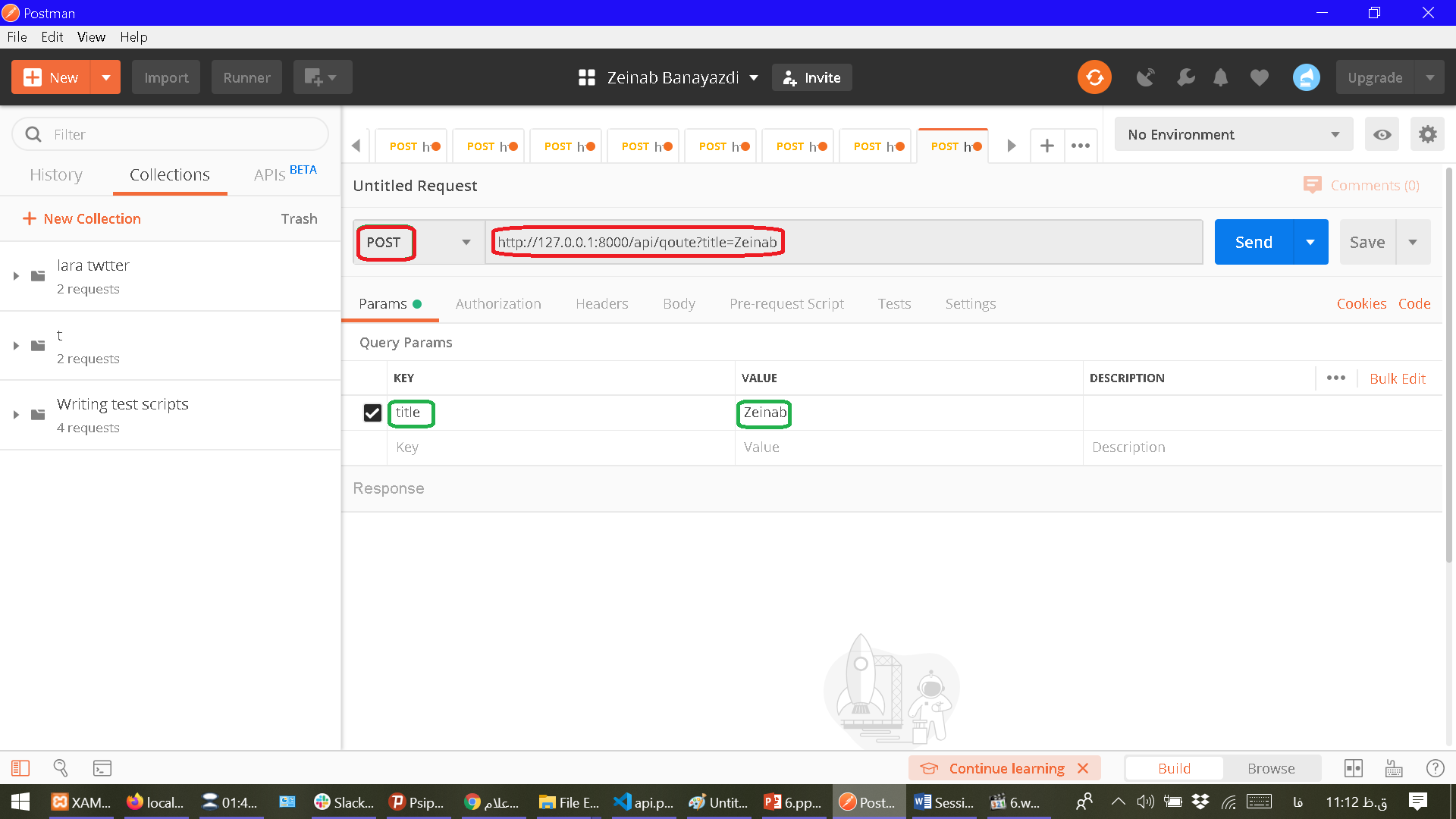
* حالا دستور php artisan serve رو بزن تا URL سایت بیادش:



* URL‌عکس بالا را در پستمن کپی کن و در ادامه اش بنویس: qoute /api/
  + در فایل web.php‌هرچیزی که صدا کیزنی از اون ریشه سایتتون شروع میشه. ولی در فایل api هرچی که صدا میزنی با /api/ شروع میشه. چون دو تا آدرس نمیتونه یه جا باشه.



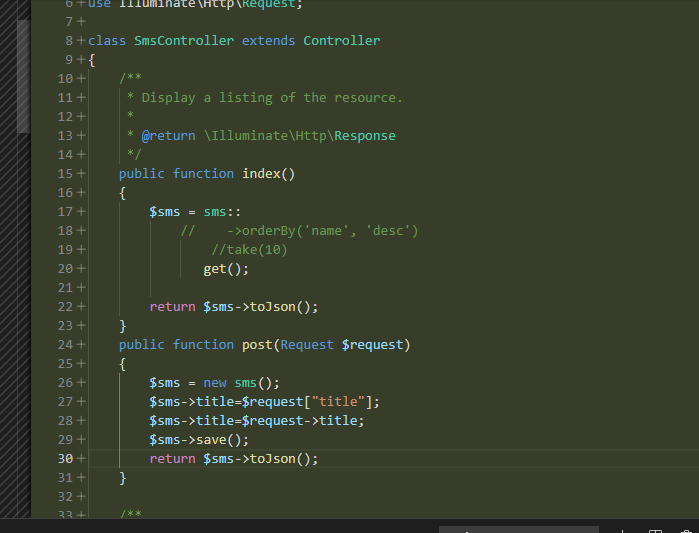
حالا میخوایم جای qoute در عکس بالاُ مقادیری که در بانکمون هست برگرده. ولی الان جدول بانکمون خالیه. پس یه دونه post باید بنویسیم و یه دونه get.  
یعنی برای پست، میخوایم وقتی به صورت عکس زیر پست کردیم، سرور مقادیر سبز در عکس زیر را بگیره و به بانک اضافه کنه:

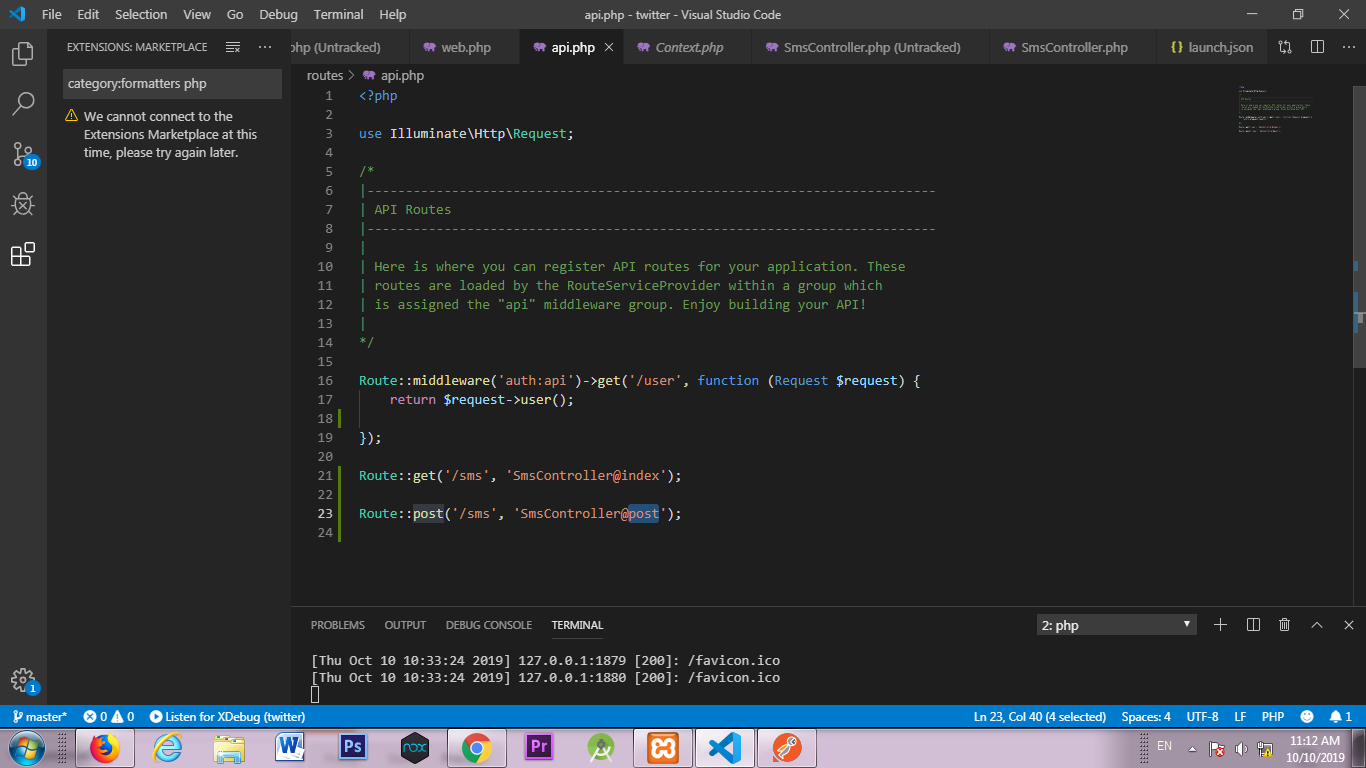


* این،برای get بود.

**حالا میخوایم post کنیم**

* گوگل: insert new record into laravel by model in controller
  + نتیجه گوگل: <https://laravel.com/docs/5.8/eloquent> (این مطالب در این لینک هست)
    - Eloquent یه پیاده سازی ساده برای کار با پایگاه داده است.
    - این نتیجه گوگل میگه:
      * یه مدل بساز. که ما ساختیم (به نام qoute)
      * Migration اش را بساز. که ما ساختیم (برای qoute)
* گوگل: Laravel convert model to json





* Ctrl+Shift+F فرمتر نصب میکنه. همیشه این فرمترها رو داشته باش. کدت رو مرتب میکنه.
* وقتی میری به phpmyadmin، حتما vscode را از حالت debug در بیاری.
* پس در این جلسه:
  + Api ای که در وردپرس بود و قبلا به صورت nocode با pods ایجاد میکردیم را با کدنویسی ایجاد کردیم
  + با صدا زدن این api، توییت مون رو در بانک نوشتیم و خواندیم.
  + حالا جلسه بعد میریم سمت آیونیک و کارهایی که با لاراول انجام دادیم را با آیونیک انجام میدیم:
    - یه صفحه میسازیم:new
    - یه صفحه میسازیم: نمایش همه توییت ها
* تمرین بیشتر:
  + گیت هاب استاد: پروژه های آیونیک ،همشون پست دارن

از نتیجه ای که در گوگل اومد کپی کرد