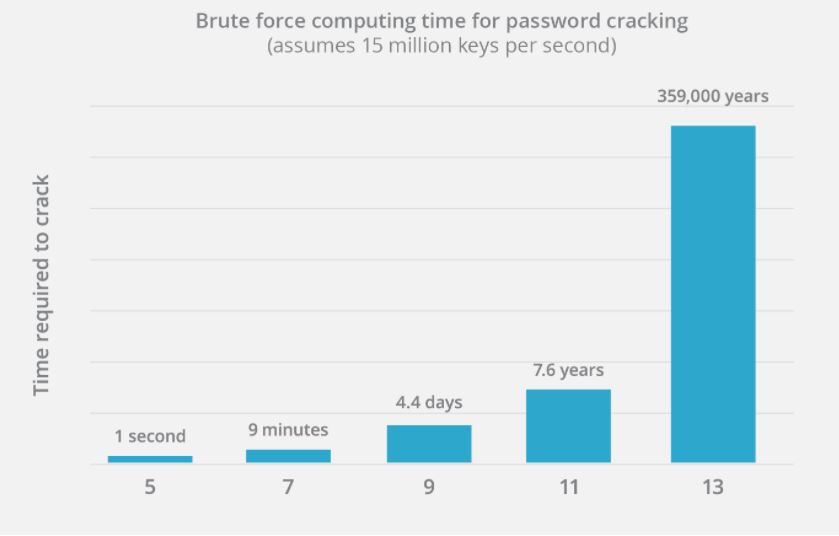
**حمله brute force:**

حمله brute force (حمله جستجوی فراگیر) یک روش آزمون و خطا برای رمزگشایی (decode) داده های مهم و حساس می باشد. متداول ترین برنامه های حملات brute force، شامل کرک کردن رمزهای عبور و کلیدهای رمزگذاری می باشد. اهداف دیگر حملات جستجوی فراگیر کلیدهای API و ورود به سیستم SSH می باشد. حملات brute force password غالباً توسط scriptها و ربات هایی انجام می شود که صفحه ورود یک وب سایت را هدف قرار می دهند.

آنچه که حملات brute force را از سایر روش های کرکینگ (cracking) متمایز می کند این است که از هیچ استراتژی فکری استفاده نمی کند. بلکه سعی دارد کارکترهای مختلف را با هم ترکیب نموده تا به ترکیب صحیح دست پیدا کند.

بزرگ ترین مزیت حملات brute force (حملات جستجوی فراگیر) این است که نسبتا ساده است و به راحتی می توان آن را انجام داد. در ضمن با توجه به فرصت کافی و عدم استراتژی هایی جهت کاهش این حملات، می توان گفت این حمله همیشه کارساز است. با استفاده از حملات بروت فورس، می توان به هر سیستم مبتنی بر پسورد و کلید رمزگذاری دست یافت. در حقیقت، مدت زمانی که لازم است برای کرک کردن یک سیستم در نظر گرفت، معیاری مناسب برای اندازه گیری سطح امنیت آن می باشد.

از طرفی دیگر، سرعت حملات brute force بسیار پایین است، زیرا مهاجمان می بایست تمامی ترکیبات کارکترهای مختلف را امتحان کنند تا به هدف خود برسند. با افزایش تعداد کارکترها در پسورد هدف، میزان کندی نیز بیشتر می گردد. به عنوان مثال، دست یابی به یک پسورد چهار کارکتری به طور قابل توجهی بیشتر از پسورد سه کارکتری طول می کشد و به همین ترتیب رمز عبور پنج کارکتری خیلی بیشتر از رمز عبور چهار کارکتری به زمان نیاز دارد.



در صورتی که تعداد کارکتری های یک پسورد به اندازه کافی طولانی باشد، رمزگشایی آن توسط مهاجم brute force ممکن است روزها، ماه ها و یا حتی سال ها طول بکشد. در نتیجه با پسوردهای طولانی کار مهاجم بسیار دشوار می شود. زمانی که از یک پسورد و رمزگذاری مناسب استفاده شود، معمولا مهاجمان از سایر روش های رمزگشایی (decode) مانند حملات مهندسی اجتماعی و حملات On path استفاده می کنند.

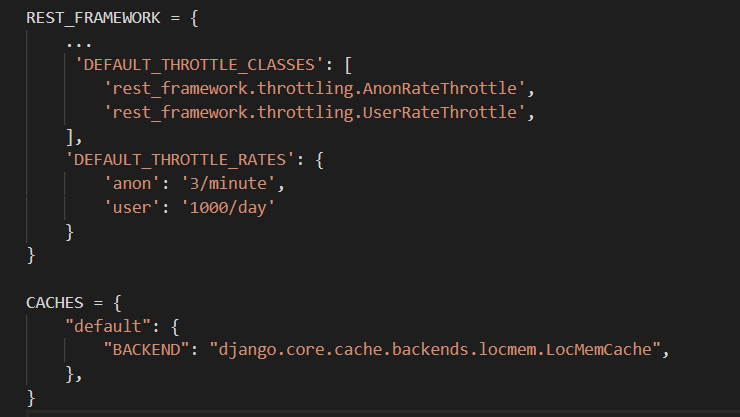
نحوه محافظت در برابر حملات brute force (جستجوی فراگیر)

راهکار ارائه شده در پروژه استفاده از throttling درdjango rest framework است. throttling هم مثل permissions اقدام به ایجاد محدودیت برای درخواست های کاربران می کند. با این تفاوت که throttling تعداد درخواست هایی که کاربران می توانند به یک API ارسال کنند ، را کنترل می کند. به طور مثال throttling می تواند به کاربران معمولی اجازه ارسال تعداد 1000 درخواست در روز را بدهد. همچنین می تواند این محدودیت را در روز, ساعت, دقیقه و ثانیه اعمال کند.

همانند مجوزها ( permissions )، ممکن است از چند throttle استفاده شود. دو سناریو برای استفاده از چند throttle به صورت همزمان وجود دارد که به صورت زیر است:

1. استفاده از یک throttle محدود کننده برای درخواست های احراز هویت نشده و یک throttle محدودتر برای درخواست های احراز هویت شده
2. استفاده از چند throttle به دلیل نیاز به اعمال محدودیت‌های مختلف بر روی بخش‌های مختلف API

throttling در چارچوب django rest framework همیشه به عنوان لیستی از کلاس ها تعریف می شود. و تنظیمات پیش فرض آن همانطور که در شکل زیر نمایش داده شده ، در مسیر settings.py ست شده است.



این تنظیمات به گونه ای است که از حمله ی brute force در برابر کلیه ی APIها محافظت می کند .در این تنظیمات تعداد تلاش های نا موفق برای کاربر احراز هویت نشده و یا به اصطلاح anonymous سه تا در دقیقه ست شده است و همچنین تعداد درخواست های کاربر احراز هویت شده (user) هزار تا در بیست و چهار ساعت می باشد.

چرا از throttle استفاده شد و از axes نه؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟؟

احراز هویت :

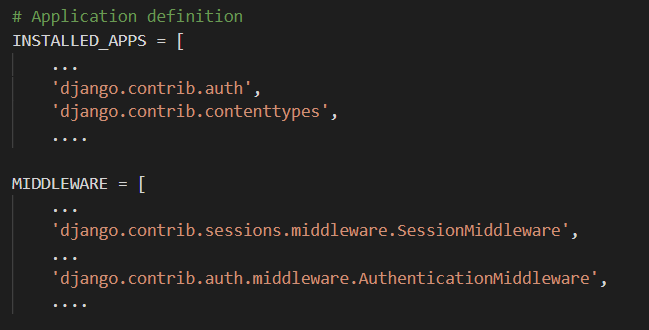
احراز هویت (Authentication) به معنای تعیین صحت و درستی یک ویژگی یا داده است.در شبکه‌های رایانه ای اصالت سنجی بدین معناست که یک سرویس دهنده بتواند تشخیص دهد کسی که تقاضایی را روی آن سیستم دارد شخص حقیقی و یا یک بد افزار است تا بدین ترتیب به گیرنده پیام اطمینان داده شود که پیام از همان مبدا ادعا شده می‌باشد. هر مکانیزمی‌که بتواند هویت واقعی یک فرد را بدون هیچ ابهامی، تایید یا رد کند سرویسی جهت احراز هویت است.

سیستم احراز هویت Django هم موضوع احراز هویت و هم موضوع اعتبارسنجی را بررسی می‌کند.این سیستم احراز هویت به صورت کلی شامل موارد زیر است:

* کاربران
* دسترسی‌ها: از طریق باینری True/False مدیریت می‌شود.
* گروه‌ها: راهی بسیار مناسب برای دادن دسترسی به حجمی بالا از کاربران.
* سیستم هشینگ پسورد
* فرم‌هایی برای مدیریت اعتبارسنجی و احرازهویت کاربران
* ابزارهای View در مدل MVT برای مدیریت ورود کاربران
* سیستم قابل توسعه در back-end

فعال کردن سیستم احراز هویت:

با ساخت skeleton website ، authentication به شکل خود کار فعال می شود. همانطور که در شکل زیر نمایش داده شده است، تنظیمات در بخش INSTALLED\_APPS و MIDDLEWAAREفایل پروژه (locallibrary/locallibrary/settings.py) قرار دارد:



نحوه محافظت در برابر حملات جستجوی فراگیر (brute force)

توسعه دهندگانی که سیستم های تایید مجوز را مدیریت می کنند، می توانند اقداماتی نظیر قفل کردن آدرس های آیپی که تعداد زیادی لاگین ناموفق ایجاد کرده اند و  استفاده از delay در نرم افزار password-checking را انجام دهند. تاخیر زمانی حتی اگر در حد چند ثانیه باشد، می تواند تاثیر حمله جستجوی فراگیر را تا حد قابل ملاحظه ای ضعیف کند.

کاربران وب می توانند آسیب پذیری خود را در برابر حملات brute force با انتخاب پسوردهای پیچیده تر و طولانی تر کاهش دهند. در ضمن احراز هویت دو عاملی و همچنین استفاده از پسورد اختصاصی برای هر دستگاه، نیز به این امر کمک شایانی می نماید. چنانچه یک مهاجم بتواند از طریق حمله brute force به یک رمز عبور دست پیدا کند، ممکن است آن پسورد را برای دستگاه های دیگر نیز امتحان کند. این فرایند تحت عنوان[credential stuffing](https://setakit.com/credential-stuffing-attack/) معروف است.

یکی دیگر از اقداماتی که کاربران باید رعایت کنند این است که پسوردهای اطلاعات شخصی خود مانند شماره کارت اعتباری یا اطلاعات بانکی خود را در وب هایی که از کلیدهای رمزگذاری قوی استفاده نمی کنند، و سطح محافظت آن ها پایین است، وارد نکنند.