توضیحات امنیتی پروژه Django

مقدمه

این پروژه با تمرکز بر امنیت طراحی شده و هدف آن پیاده سازی تکنیک‌هایی برای مقابله با آسیب‌پذیری‌های رایج در برنامه های تحت وب است. فناوری اصلی استفاده‌شده در این پروژه فریمورک Django می‌باشد.

1. جلوگیری از SQL Injection

Django به‌صورت پیش‌فرض از ORM استفاده می‌کند که از تزریق کدهای SQL جلوگیری می‌کند. هیچ‌گونه Query خام در پروژه استفاده نشده و تمامی دستورات دیتابیس با استفاده از Model ها و QuerySet های امن نوشته شده اند.

مثال امن:

User.objects.filter(username=username)

در این روش، ورودی کاربر مستقیماً به SQL تبدیل نمی‌شود و امکان تزریق از بین می‌رود.

2. جلوگیری از XSS (Cross-site Scripting)

برای جلوگیری ازXSS ، در قالب‌های HTML از قابلیت auto-escaping جنگو استفاده شده است. این قابلیت به صورت پیش‌فرض فعال است.

ورودی‌هایی که در صفحات نمایش داده می‌شوند، از طریق توابع قالب به‌شکل امن رندر شده‌اند و هیچ‌کدام به‌صورت خام داخل HTML قرار نگرفته‌اند.

نکته: از استفاده از safe| بدون دلیل امنیتی اجتناب شده است.

3. جلوگیری از CSRF (Cross-site Request Forgery)

تمام فرم‌های پروژه از تگ CSRF استفاده می‌کنند. در Django با اضافه کردن {% csrf_token %} در هر فرم، توکن CSRF به‌صورت خودکار تولید و اعتبارسنجی می‌شود.

همچنین برای APIهایی که نیاز به حفاظت دارند، هدر X-CSRFToken استفاده شده است.

4. محدودسازی نرخ درخواست (Rate Limiting)

برای مقابله با حملات Brute Force و سوءاستفاده از API، سیستم Rate Limiting پیاده‌سازی شده است. این سیستم با استفاده از Redis، آدرس IP کاربران را دنبال کرده و در صورت ارسال درخواست بیش از حد مجاز، درخواست‌ها را مسدود می‌کند.

پارامترهایی که پیکربندی شده‌اند:

محدودیت: ۵ درخواست در دقیقه

پاسخ در صورت نقض محدودیت: کد وضعیت 429 (Too Many Requests)

5. سیستم لاگین امن با محافظت در برابر Brute Force

در فرآیند لاگین:

اگر یک کاربر چند بار پشت سر هم رمز اشتباه وارد کند، سیستم موقتاً اکانت یا IP او را برای مدتی مسدود می‌کند.

از الگوریتم رمزنگاری استاندارد Django برای هش کردن رمز عبور استفاده شده است.

پیغام خطا به‌گونه‌ای طراحی شده که مشخص نمی‌کند کدام قسمت (نام کاربری یا رمز عبور) اشتباه است تا از افشای اطلاعات جلوگیری شود.

6. احراز هویت دو مرحله‌ای (Two-Factor Authentication - 2FA)

در این پروژه، پس از ورود موفق به حساب، یک مرحله تأیید دوم به‌وسیله کد ارسالی از طریق ایمیل یا OTP اضافه شده است.

مراحل:

کاربر با نام کاربری و رمز وارد می‌شود.

سیستم یک کد یکبار مصرف ۶ رقمی تولید کرده و برای کاربر ایمیل می‌کند.

کاربر باید این کد را وارد کند تا به داشبورد وارد شود.

در صورت ورود کد اشتباه چند مرتبه پشت سر هم، کاربر از سیستم خارج می‌شود و نیاز به ورود مجدد دارد.

7. تنظیمات امنیتی در Django

فعال‌سازی SECURE_BROWSER_XSS_FILTER

استفاده از X-Content-Type-Options: nosniff

تنظیم SECURE_HSTS_SECONDS برای جلوگیری از Downgrade Attack

فعال بودن CSRF_COOKIE_SECURE و SESSION_COOKIE_SECURE برای انتقال امن در HTTPS

استفاده از HttpOnly برای کوکی‌ها جهت جلوگیری از سرقت کوکی توسط JS