**COMP50016**

**Server-Side Programming - II**

**Security Documentation**

Name – Mudeen Mohammed

CB No – CB016713

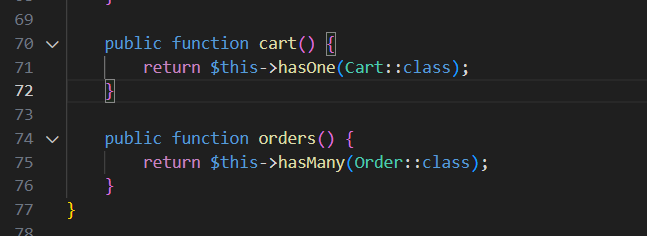
Table of Content

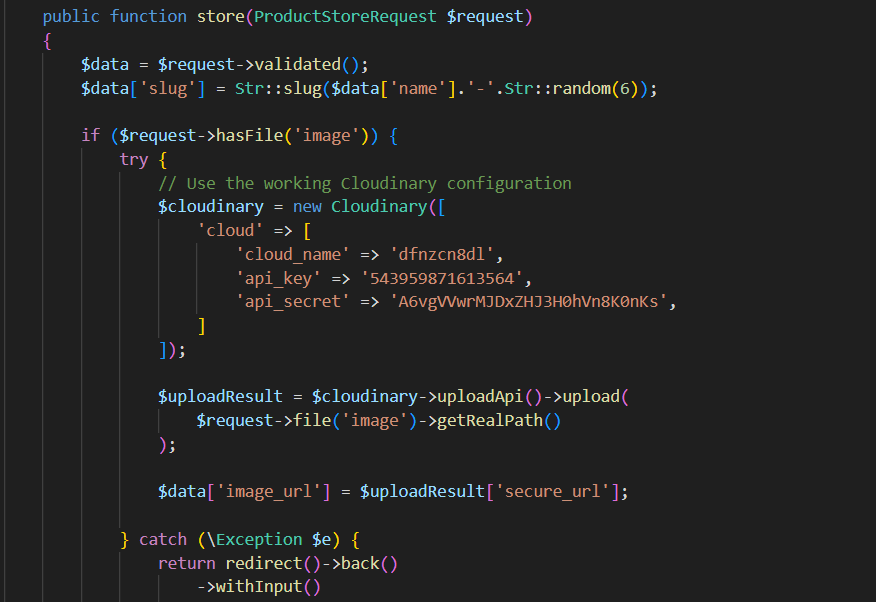
1. Security Risks and Mitigations
   1. Injection (SQL, Command, etc.)
   2. Broken Authentication
   3. Sensitive Data Exposure
   4. XML External Entities (XXE)
   5. Broken Access Control
   6. Security Misconfiguration
   7. Cross-Site Scripting (XSS)
2. Laravel-Specific Security Implementations
   1. CSRF Protection
   2. HTTPS Enforcement
   3. Input Validation
3. Security Risks and Mitigations
   1. Injection (SQL, Command, etc.)

**Threat**: Unauthorized user input could lead to SQL Injection or command injection.

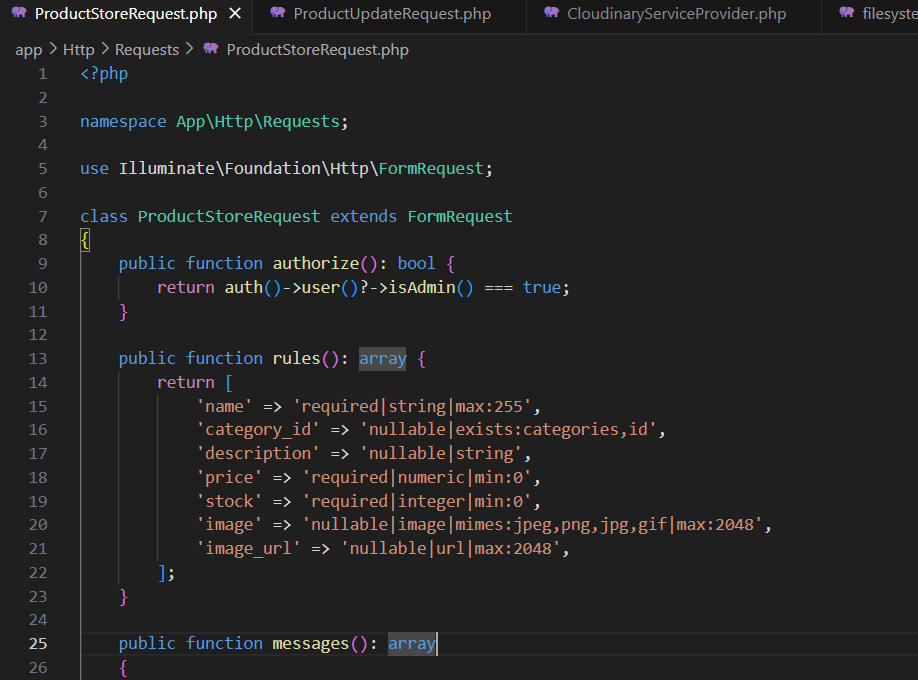
**Mitigations**

* Use Laravel Eloquent ORM and query builder which rely on prepared statements.





* Validate all inputs with Laravel’s Form Request Validation.

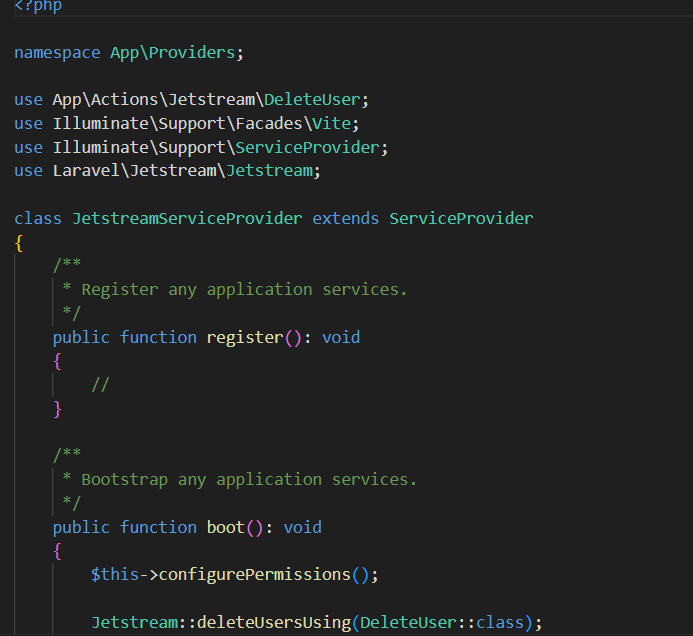


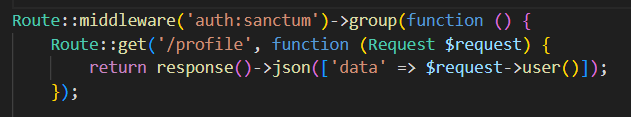
* 1. Broken Authentication

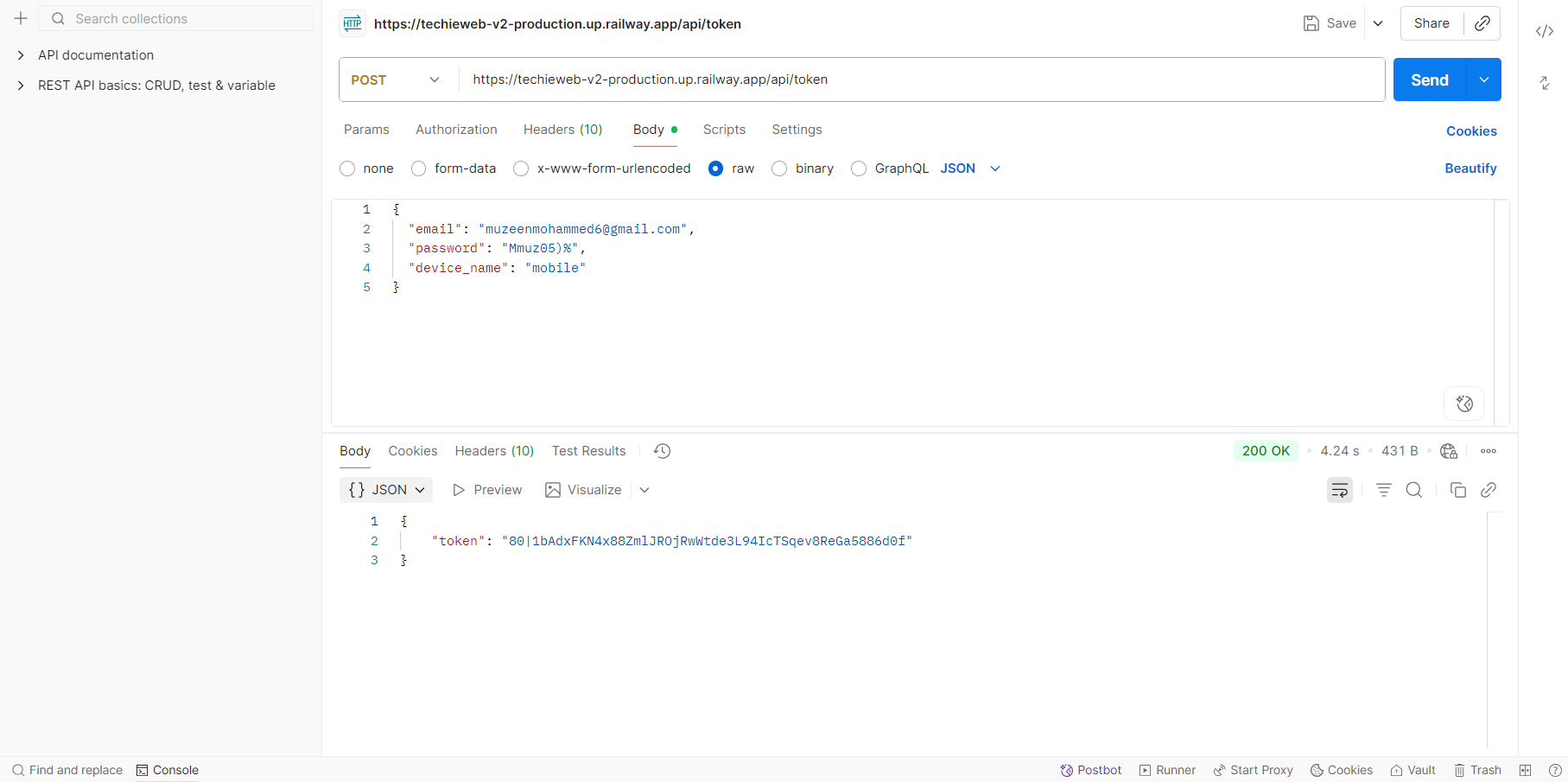
**Threat**: Attackers may compromise authentication mechanisms.

**Mitigation**

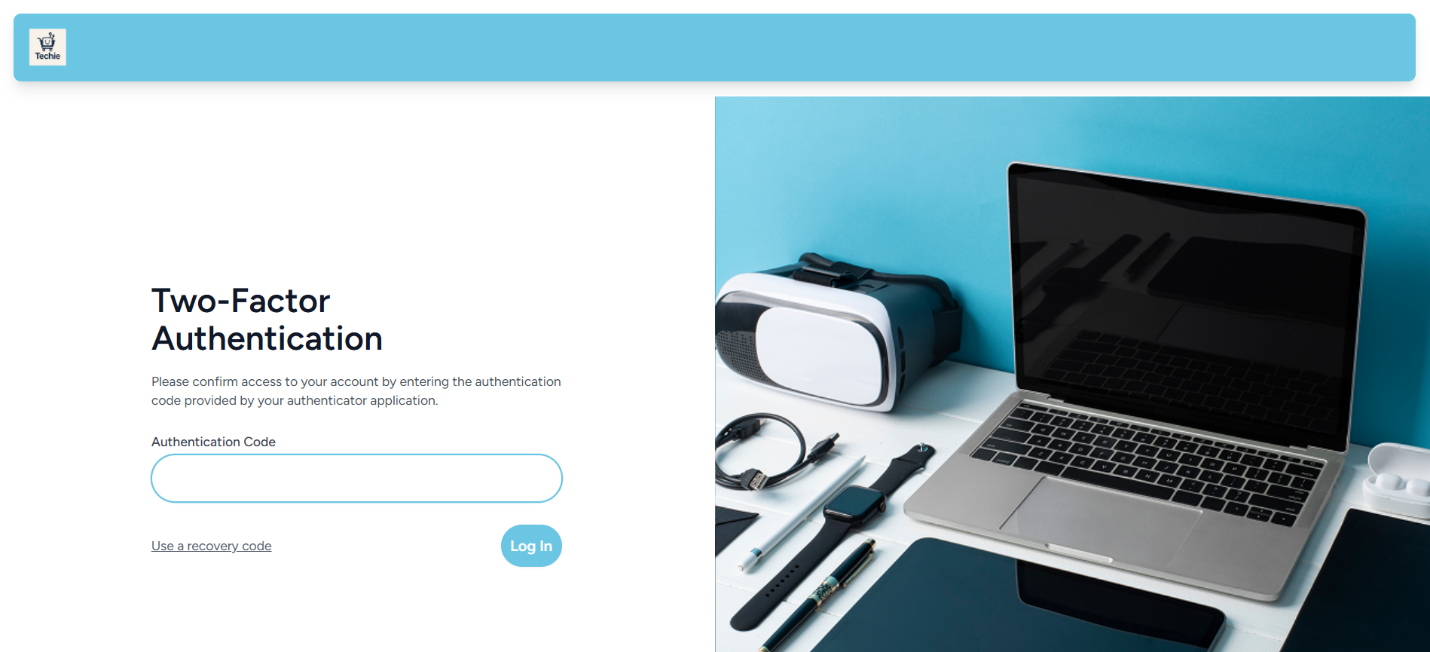
* Laravel Jetstream + Sanctum for secure authentication.

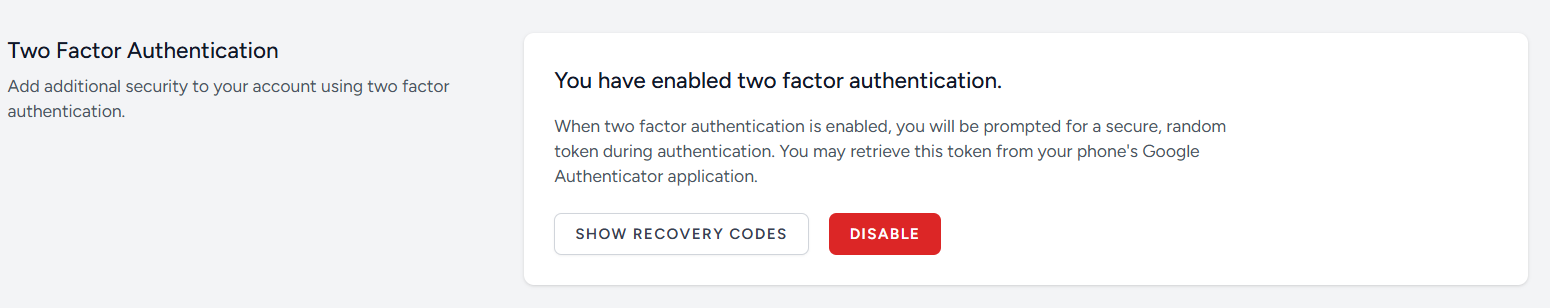




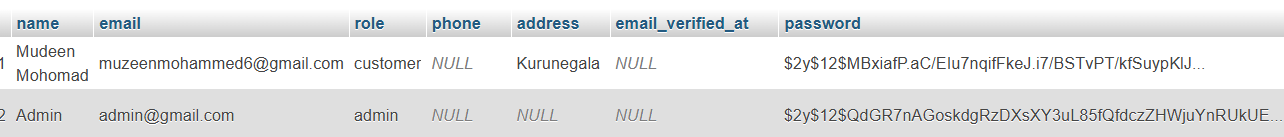


* Two-factor authentication





* Passwords hashed

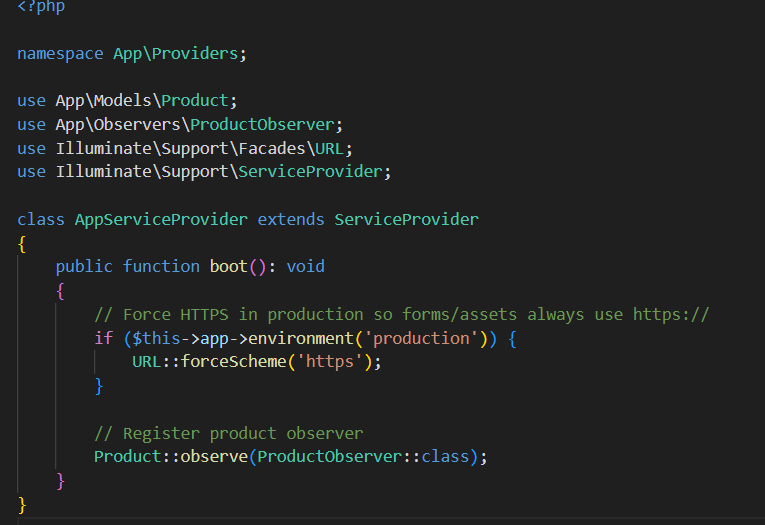


* 1. Sensitive Data Exposure

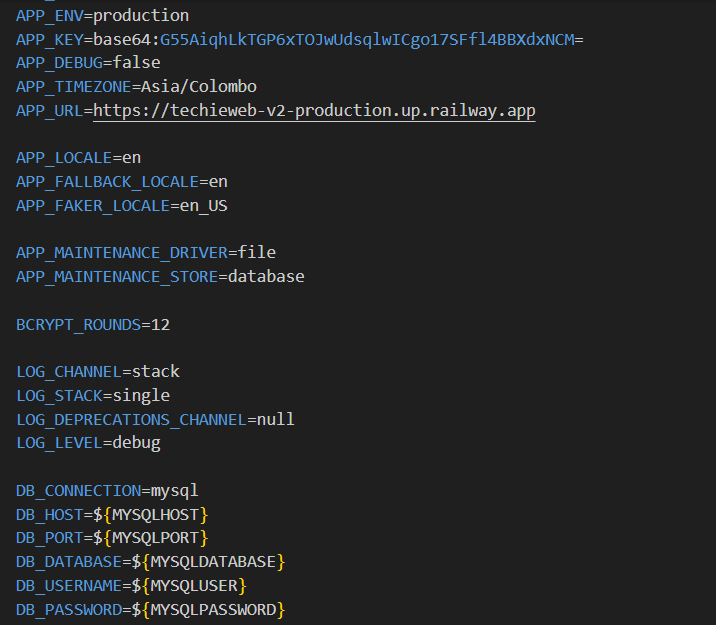
**Threat**: Sensitive data like passwords, tokens, users’ personal info

**Mitigation**

* Enforce HTTPS/TLS site-wide with HSTS headers



* Store secrets in .env file with proper server permissions.

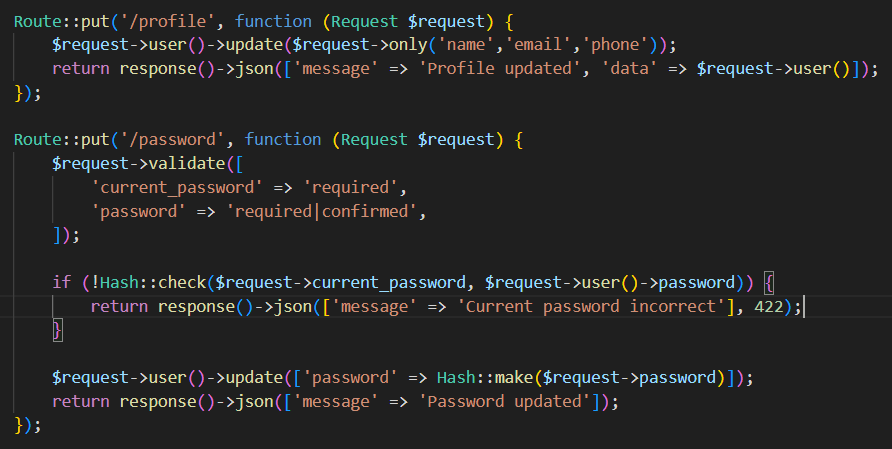


* 1. XML External Entities (XXE)

**Threat**: XXE attacks via malicious XML inputs

**Mitigation**

* Avoid XML parsing where possible; use JSON APIs.

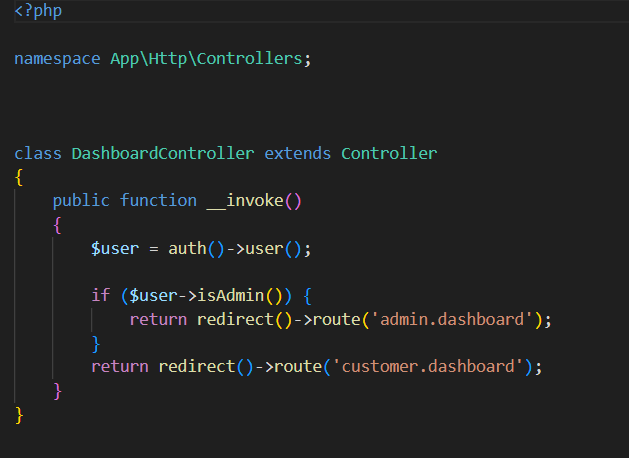


* 1. Broken Access Control

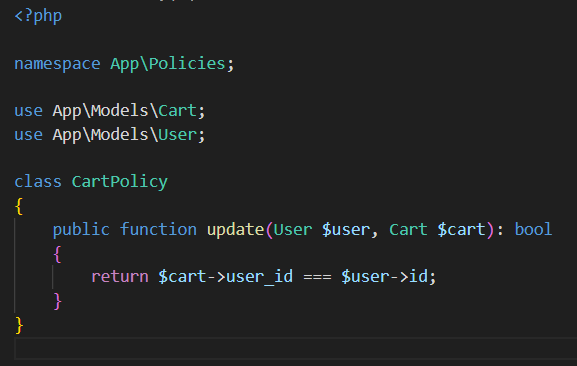
**Threat**: Unauthorized users gaining access to restricted resources.

**Mitigation**

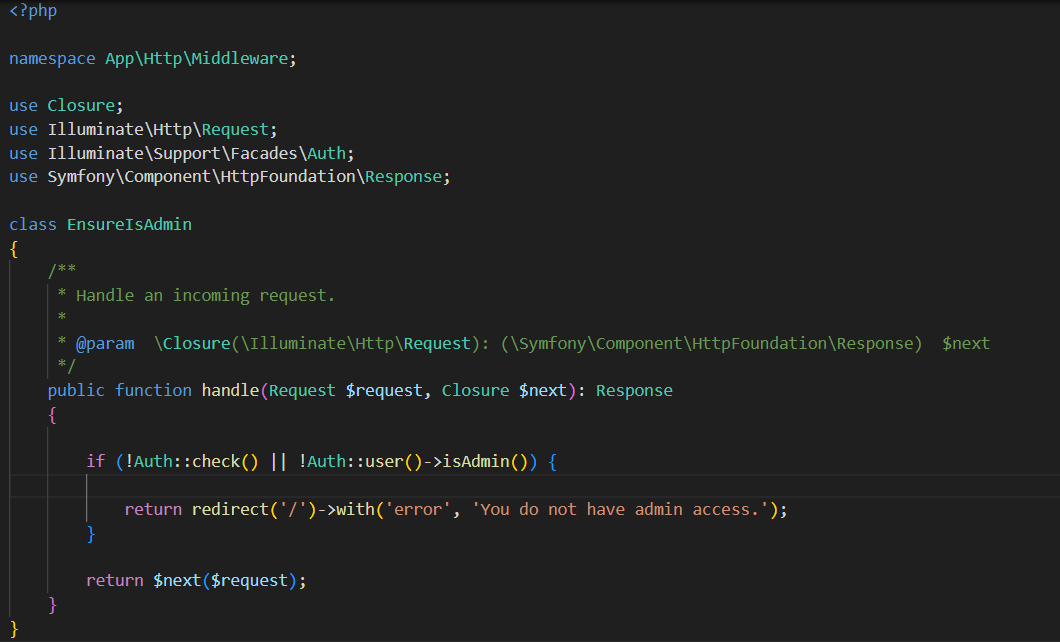
* Enforce RBAC (Role-Based Access Control).



* Use Laravel Gates & Policies for fine-grained permissions



* Admin-only routes protected with middleware

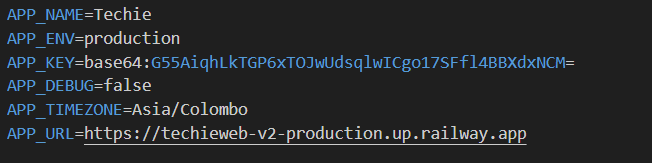


* 1. Security Misconfiguration

**Threat**: Weak server configuration exposes system to attacks.

**Mitigation**

* Disable debugging in production (APP\_DEBUG=false).

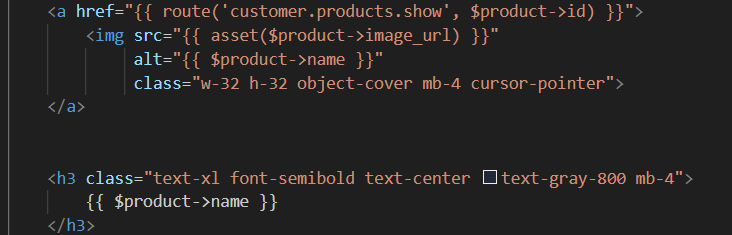


* 1. Cross-Site Scripting (XSS)

**Threat**: Malicious JavaScript injected into user browsers.

**Mitigation**

* Use Blade’s auto-escaping {{ }}.



1. Laravel-Specific Security Implementations
   1. CSRF Protection



* 1. HTTPS Enforcement
* Force HTTPS using middleware (\App\Http\Middleware\ForceHttps).



* 1. Input Validation

