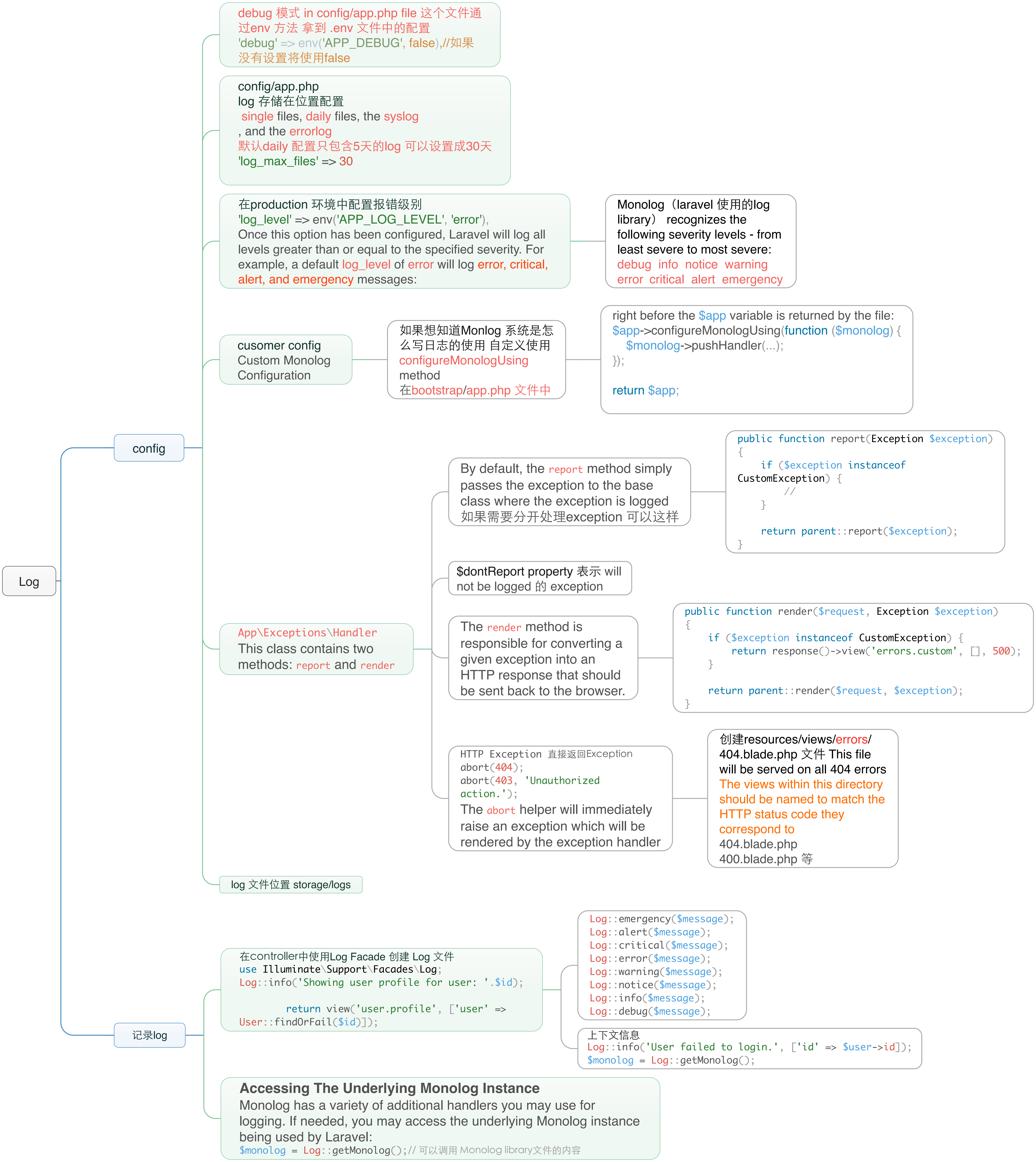

Log

- config
  - debug 模式 in config/app.php file 这个文件通过env 方法 拿到 .env 文件中的配置
    'debug' => env('APP_DEBUG', false),//如果没有设置将使用false
  - config/app.php
    log 存储在位置配置
    single files, daily files, the syslog
    , and the errorlog
    默认daily 配置只包含5天的log 可以设置成30天
    'log_max_files' => 30
  - 在production 环境中配置报错级别
    'log_level' => env('APP_LOG_LEVEL', 'error'),
    Once this option has been configured, Laravel will log all levels greater than or equal to the specified severity. For example, a default log_level of error will log error, critical, alert, and emergency messages:
    - Monolog（laravel 使用的log library） recognizes the following severity levels - from least severe to most severe:
      debug info notice warning
      error critical alert emergency
  - cusomer config
    Custom Monolog Configuration
    - 如果想知道Monlog 系统是怎么写日志的使用 自定义使用 configureMonologUsing method
      在bootstrap/app.php 文件中
      - right before the $app variable is returned by the file:

```
$app->configureMonologUsing(function ($monolog) {
    $monolog->pushHandler(...);
});

return $app;
```

  - `App\Exceptions\Handler`
    This class contains two methods: `report` and `render`
    - By default, the `report` method simply passes the exception to the base class where the exception is logged
      如果需要分开处理exception 可以这样

```
public function report(Exception $exception)
{
    if ($exception instanceof
CustomException) {
        //
    }

    return parent::report($exception);
}
```

    - $dontReport property 表示 will not be logged 的 exception
    - The `render` method is responsible for converting a given exception into an HTTP response that should be sent back to the browser.

```
public function render($request, Exception $exception)
{
    if ($exception instanceof CustomException) {
        return response()->view('errors.custom', [], 500);
    }

    return parent::render($request, $exception);
}
```

    - HTTP Exception 直接返回Exception

```
abort(404);
abort(403, 'Unauthorized
action.');
```

      The `abort` helper will immediately raise an exception which will be rendered by the exception handler
      - 创建resources/views/errors/404.blade.php 文件 This file will be served on all 404 errors
        The views within this directory should be named to match the HTTP status code they correspond to
        404.blade.php
        400.blade.php 等
  - log 文件位置 storage/logs
- 记录log
  - 在controller中使用Log Facade 创建 Log 文件

```
use Illuminate\Support\Facades\Log;
Log::info('Showing user profile for user: '.$id);

        return view('user.profile', ['user' =>
User::findOrFail($id)]);
```

    -
      ```
      Log::emergency($message);
      Log::alert($message);
      Log::critical($message);
      Log::error($message);
      Log::warning($message);
      Log::notice($message);
      Log::info($message);
      Log::debug($message);
      ```
    - 上下文信息

```
Log::info('User failed to login.', ['id' => $user->id]);
$monolog = Log::getMonolog();
```

  - **Accessing The Underlying Monolog Instance**
    Monolog has a variety of additional handlers you may use for logging. If needed, you may access the underlying Monolog instance being used by Laravel:

```
$monolog = Log::getMonolog();// 可以调用 Monolog library文件的内容
```