Diferença entre Eloquent ORM e Query Builder no Laravel:

- **Eloquent ORM**: É uma implementação de ORM que permite trabalhar com bancos de dados de forma orientada a objetos, representando tabelas como classes Model.
    - **Prós**: Sintaxe simples e intuitiva, facilita o trabalho com relacionamentos entre tabelas, oferece recursos como Mutators, Collections, Factories.
    - **Contras**: Pode ter desempenho inferior em consultas complexas e consumir mais memória, especialmente com grandes volumes de dados.
- **Query Builder**: Permite criar consultas SQL de forma mais direta e programática, sem a camada de abstração do ORM.
    - **Prós**: Melhor desempenho e flexibilidade para criar consultas SQL mais complexas e específicas, baixo consumo de recursos.
    - **Contras**: Menos organizado e mais verboso que o Eloquent, exigindo mais esforço na escrita de código, além de não ser orientado a objetos.

Três práticas de segurança ao lidar com dados sensíveis em Laravel:

- **Validação de Entrada**: Sempre valide os dados recebidos de usuários antes de processá-los. Isso evita que informações maliciosas entrem no sistema. Use `Request` ou `validate()` para isso.
- **Criptografia de Dados Sensíveis**: Senhas devem ser hashadas com `bcrypt()` e outros dados confidenciais podem ser criptografados com `encrypt()` para garantir que informações críticas não fiquem expostas, mesmo em caso de vazamento.
- **Proteção contra CSRF**: Use os tokens CSRF automáticos que o Laravel gera para proteger sua aplicação contra ataques de Cross-Site Request Forgery, garantindo que apenas requisições legítimas sejam processadas.

Outras práticas:

- **Proteção contra Injeção SQL**: Usar Eloquent ou Query Builder ajuda a evitar injeção SQL automaticamente, já que eles sanitizam as entradas do usuário.
- **Autenticação e Autorização**: Use o sistema de autenticação do Laravel para controlar quem pode acessar o quê, e aplique políticas de autorização (`policies` e `gates`) para definir permissões detalhadas.
- **Hashing seguro**: Garanta que senhas sejam armazenadas de forma segura, usando funções de hashing como `bcrypt` ou `argon2`, para proteger credenciais de usuários mesmo em caso de vazamento.

Papel dos Middlewares no Laravel:

- **Middlewares**: São filtros que processam a requisição antes ou depois dela passar pela aplicação. Eles podem ser usados para autenticação, controle de permissões, registro de logs, proteção contra CSRF, entre outros.

Exemplo prático:

- Criar um middleware para verificar se o usuário está ativo.

    - Crie o middleware com `php artisan make:middleware CheckIfUserIsActive`.
    - No método `handle`, verifique se `auth()->user()->is_active`.
    - Aplique o middleware na rota: `Route::middleware('check.active')->group(...)`.

Como o Laravel gerencia migrations:

- **Migrations**: São usadas para criar e modificar tabelas no banco de dados de forma versionada e controlada, facilitando a manutenção do esquema entre diferentes ambientes de desenvolvimento e produção.
- **Melhores práticas**:
    - Use nomes descritivos para as migrations.
    - Faça alterações de forma incremental, evitando alterações destrutivas.
    - Sempre teste migrations em ambiente de desenvolvimento antes de aplicar em produção.

Diferença entre transações e savepoints no SQL Server, e uso de transações no Laravel:

- **Transações**: São blocos de operações SQL que são executadas juntas. Se algo falhar, você pode reverter todas as operações com um `ROLLBACK`, garantindo a integridade dos dados. No Laravel, você pode usar `DB::transaction()` para garantir que um grupo de operações seja tratado de forma atômica.
- **Savepoints**: Permitem reverter parte de uma transação sem desfazer tudo. No SQL Server, você usa `SAVE TRANSACTION` para criar esses pontos.

Observação: Conheço pouco SQL Server, pois trabalho principalmente com bancos como MySQL e SQLite. Mesmo assim, o conceito de transações pode ser aplicado de maneira semelhante em vários SGBDs, e em Laravel o uso de transações com `DB::transaction()` é essencial para garantir a consistência nas operações de banco de dados.