1. **ЧТО В КОДЕ НЕ ОПТИМАЛЬНО?**

* Избегать хардкода как в `App\Models\City::ONLINE`, `App\Models\City::currentCityId`, `App\Http\Controllers\CatalogController::categoryPage,markers` и прочих. Некоторые данные можно вынести в аргументы и свойства.
* Избегать излишнего применения функциональной парадигмы описания алгоритма для лучшего понимания описания. Некоторые стрелочные и ананомные функции можно вынести в именованные приватные методы или даже методы отдельных классов-помощников.
* Строки кода ограничить длиной в 200-230 символов для лучшей читаемости.
* Оптимизировать алгоритмы вычислений. В том числе, не связанные прямо со сторонними пакетами.
* Избегать излишней вложенности блоков путём вынесения вычислений в отдельные методы или скорейшего выхода из блока (throw, break, continue, return, ...).
* Если свойство URL не является объектом описания всего класса модели, то следует
* перенести эти вычисления в Middleware или Controller. Модель не должна
* ориентироваться на URL при вычислениях. В том числе, инструкция `config`.
* Если описывается API, то не удобно каждый рез указывать формат возвращаемых данных. Можно использовать внутренние механизмы API для форматирования и применять ресурсы.
* Конкатенацию строк преимущественно представлять в виде интерполиемых строк.
* `App\Models\City::getPrefixAttribute,getSlugNameAttribute,appUrl` и прочее прямо не относится к модели. На счёт URL уже писал. Но есть ещё аспект в том что это могут быть некоторые общие методы, не связанные прямо с вычислением данных этой модели. Стоит их вынести в абстрактный или другой класс.
* Имена методов составлять с учётом префиксов “get”|“set”|“create”|“update”|”is”, если это не процедура.
* Документирование кода phpdocumentor и\или swagger.

1. **КАК УЛУЧШИТЬ БЫСТРОДЕЙСТВИЕ?**

* Не доверяться ORM, а применять DDL|DML|DQL, относительно используемой СУБД.
* Выбрать более производительную СУБД.
* Разбить вычисления, связанные с сущностями большой мощности, на набор более простых.
* Следить за запросами и индексами.
* Очереди.
* Кэширование.
* Предвычисления.
* Шардирование.
* Распределённые вычисления (многопроцессность, многопоточность).
* Разные СУБД или только БД для записи и чтения с фоновой синхронизацией.
* В крайнем случае, увеличить мощность аппаратного обеспечения.