### **Заметки к занятию 1. 04.11.2024**

#### **1. Глобальная установка Laravel**

Чтобы установить Laravel глобально и использовать его для создания новых проектов, выполните следующую команду:

composer global require laravel/installer

Это добавит Laravel в список глобальных Composer-зависимостей, и его можно будет вызывать напрямую через терминал.

#### **2. Установка нового проекта Laravel**

После установки можно создать новый проект, используя команду:

laravel new example-app

Здесь example-app — имя папки проекта. Laravel автоматически создаст новую структуру папок и установит все необходимые зависимости.

#### **3. Создание контроллера**

Laravel предоставляет команду для быстрого создания контроллеров. Для создания контроллера SiteController выполните:

php artisan make:controller SiteController

Это создаст файл SiteController.php в папке app/Http/Controllers, где будут располагаться методы, связанные с логикой обработки запросов.

#### **4. Установка API в Laravel**

Команда для установки базовой структуры API:

php artisan install:api

(Примечание: убедитесь, что используемая версия Laravel поддерживает эту команду. В последних версиях Laravel API уже встроен, и эта команда может быть неактуальна.)

#### **5. Миграции в Laravel**

Миграции позволяют создавать, изменять и управлять структурой базы данных с помощью кода. Основные команды:

**Запуск миграций**: Выполните все доступные миграции (например, для создания таблиц) с помощью:  
php artisan migrate

**Откат последней миграции**: Чтобы отменить последнюю операцию миграции, выполните:  
php artisan migrate:rollback

Эти команды помогают легко контролировать изменения в структуре базы данных в процессе разработки.

#### **6. Создание кастомного запроса**

Кастомные запросы полезны для валидации данных, передаваемых в контроллеры. Для создания нового запроса RegistrationRequest используйте:

php artisan make:request RegistrationRequest

Запрос создается в папке app/Http/Requests и может использоваться для добавления правил валидации данных перед их передачей в контроллер.

#### **7. Пример создания кастомного атрибута**

Laravel позволяет создавать «виртуальные» атрибуты, которые не хранятся в базе данных, но доступны в модели. Пример с использованием метода getFullNameAttribute:

**Метод getFullNameAttribute()**:  
// Создаем виртуальный атрибут "full\_name" для объединения данных

public function getFullNameAttribute()

{

return $this->last\_name . ' ' . $this->first\_name . ' ' . $this->patronymic;

}

**С использованием класса Attribute** (доступно в Laravel 9 и выше):  
public function fullName(): Attribute

{

return Attribute::get(fn () => $this->last\_name . ' ' . $this->first\_name . ' ' . $this->patronymic);

}

Эти методы позволяют получить full\_name в любом месте, где используется экземпляр модели, например, $user->full\_name.

Этот план и заметки помогут вам последовательно представить материалы на занятии и поддержать студентов в понимании ключевых команд и функций Laravel.