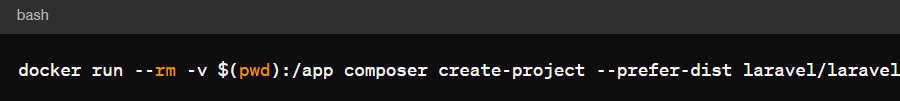
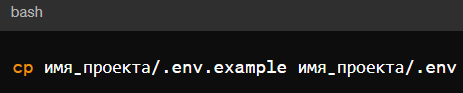
**Шаг 1: Создание проекта Laravel через Composer**



**Шаг 2: Настройка файла .env**



Далее, открываем файл .env — меняем значения APP\_NAME и APP\_DEBUG:

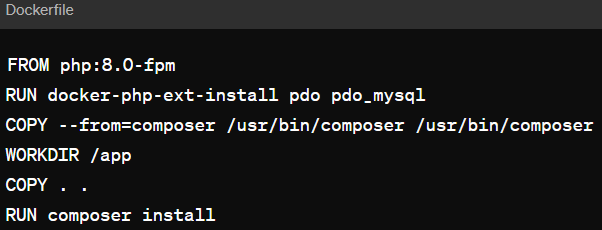
APP\_NAME=MyLaravelProject

APP\_DEBUG=true

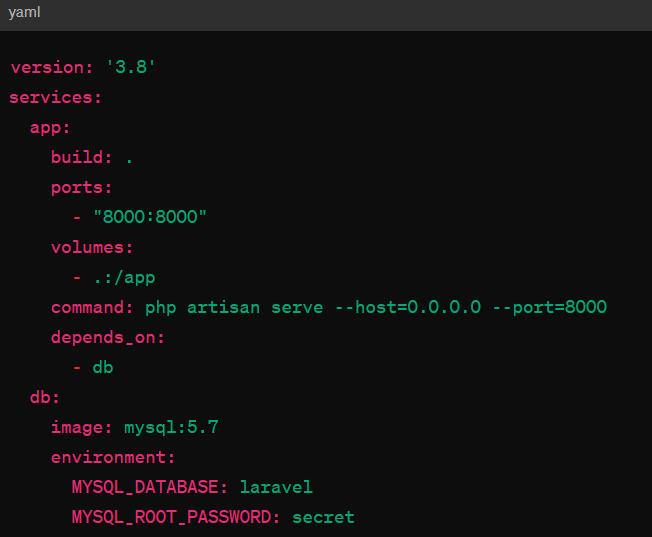
**Шаг 3: Запуск проекта с Docker**

Для запуска проекта Laravel с Docker, нам нужно создать Dockerfile и docker-compose.yml в корне проекта.

**Dockerfile**



**docker-compose.yml**

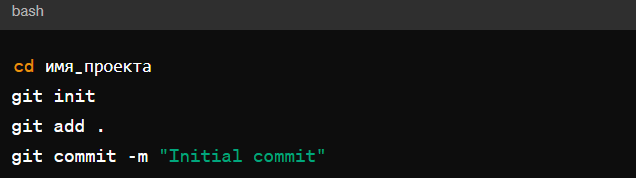


**Запуск проект:**

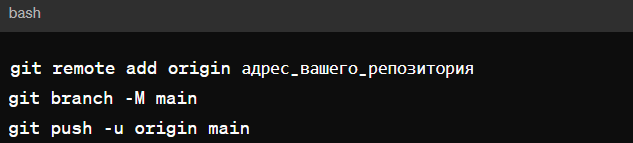


**Шаг 4: Результат**

Инициализируем Git репозиторий в проекте:



Создаем новый репозиторий на GitHub и связываем его с локальным репозиторием.



**Ответьте на вопросы**

1. **Перечислите список composer-пакетов, которые использует фреймворк Laravel после установки.**

После установки Laravel, фреймворк автоматически устанавливает ряд пакетов через Composer. Эти пакеты включают в себя как зависимости Laravel, так и пакеты, необходимые для его работы. Примерный список (не полный и может отличаться в зависимости от версии Laravel):

* laravel/framework: основной пакет фреймворка Laravel.
* fideloper/proxy: обеспечивает поддержку прокси для Laravel.
* fruitcake/laravel-cors: обрабатывает CORS (Cross-Origin Resource Sharing) в приложениях Laravel.
* guzzlehttp/guzzle: HTTP-клиент для отправки запросов и работу с API.
* laravel/tinker: позволяет взаимодействовать с приложением Laravel через командную строку.
* monolog/monolog: мощный логгер, используемый Laravel для ведения журналов.
* nesbot/carbon: расширение для работы с датами в PHP, интегрированное в Laravel.
* phpunit/phpunit: фреймворк для тестирования, используемый в Laravel для разработки и выполнения тестов.

Это лишь часть пакетов, которые может использовать Laravel. Для просмотра полного списка зависимостей можно открыть файл composer.json в корне установленного Laravel-проекта или воспользоваться командой composer show в терминале.

1. **Изучите директорию config и опишите какие файлы хранятся в этой директории.**

В директории config Laravel хранятся файлы конфигурации приложения. Эти файлы позволяют настраивать различные аспекты приложения, такие как:

* app.php: общие настройки приложения, такие как часовой пояс, локаль, ключ приложения и т.д.
* auth.php: настройки аутентификации и авторизации, включая настройки охранителей и провайдеров пользователей.
* broadcasting.php: настройки вещания событий через различные драйвера, например, Redis, Pusher.
* cache.php: настройки кэширования.
* database.php: настройки подключения к базам данных.
* filesystems.php: настройки файловой системы и облачного хранения.
* logging.php: конфигурация логирования.
* mail.php: настройки отправки почты.
* queue.php: настройки очередей задач.
* services.php: настройки сторонних сервисов, таких как Stripe, Mailgun.
* session.php: настройки сессии.
* view.php: настройки шаблонизатора.

Это основные файлы конфигурации, которые Laravel предоставляет из коробки для управления настройками приложения.

1. **В какой директории хранятся основные файлы (классы) с бизнес-логикой приложения?**

Основные файлы с бизнес-логикой приложения в Laravel обычно хранятся в директории app/. В этой директории располагаются модели, контроллеры и другие классы, необходимые для работы приложения. В частности:

* app/Models: модели, представляющие таблицы базы данных и их взаимосвязи.
* app/Http/Controllers: контроллеры, управляющие взаимодействием между моделями и представлениями, обрабатывают пользовательский ввод.
* app/Http/Middleware: промежуточное программное обеспечение (middleware) для фильтрации HTTP-запросов перед их обработкой контроллерами.

Эта структура позволяет организовать код приложения логично и эффективно.