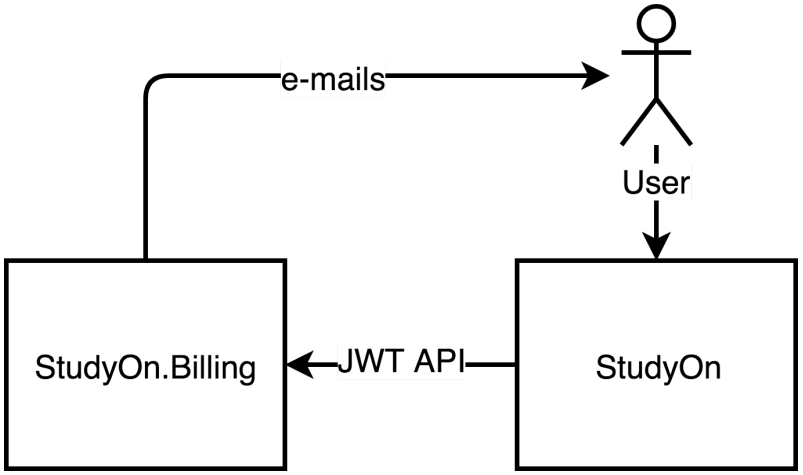
Практические занятия

Реализуемый проект

В рамках практических занятий требуется разработать сервис онлайн-обучения *StudyOn* и биллинг-систему *StudyOn.Billing* для него. Пользователи могут регистрироваться в сервисе, оплачивать и проходить курсы, получая сертификаты по итогу прохождения курсов.



Функциональность сервиса *StudyOn*

1. Предоставление платных курсов и уроков
2. Личный кабинет с персональной информацией и состоянии баланса и оплаченных курсах
3. Статистика по курсу
4. Выдача и отправка сертификата о прохождении курса на почту

Функциональность сервиса *StudyOn.Billing*

1. Сервис биллинга не виден пользователю и предоставляет только API с JWT аутентификацией
2. Регистрация нового пользователя
3. Авторизация пользователя в сервисе StudyOn
4. Оплата курса (курс можно покупать или арендовать на неделю)
5. Напоминание пользователю о скором окончании аренды
6. Отправка на служебную почту отчета по состоянию и статистике оплат за последний месяц

Возможности Symfony, которые изучаются в рамках программы:

1. Роутинг, контроллеры, шаблоны
2. Doctrine, ORM, фикстуры, миграции
3. Формы, валидация

4. API, сериализация
5. Безопасность, JWT
6. Сервисы
7. Комманды
8. Фронтенд в контексте Symfony
9. Тестирование
10. Деплоймент, CI
11. Email

Занятие 1. Создание проектов, настройка окружения

Данное занятие посвящено развертыванию скелетов будущих сервисов StudyOn и StudyOn.Billing.

Охватываемые темы: *Symfony skeleton, PostgreSQL, docker, Composer, Git, Make, PHPStorm*

У нас будет 2 symfony-приложения. Каждый из них будет работать на стеке nginx + php-fpm (php7) + PostgreSQL. Весь стек каждого сервиса будет работать в Docker.

Глобальная установка Composer

Рекомендуется не держать исполняемый файл композера в каждом проекте, а завести один глобальный. Это позволит использовать его для команд вне контекста проекта (например, нужно создать новый проект). Кроме того, можно будет его централизованно обновлять.

И еще важный момент. Composer кеширует все устанавливаемые пакеты в своей служебной директории. Поэтому установка пакетов через глобальный композер позволит сильно ускорить их установку, если они ранее устанавливались в текущем или другом проекте.

https://getcomposer.org/doc/00-intro.md#globally

Создание репозиториев под проект

Условимся, что личный аккаунт на гитхабе будет называться `sfuser`.

Создайте в личном аккаунте на github.com 2 пустых репозитория под сервисы:
https://github.com/sfuser/study-on
https://github.com/sfuser/study-on.billing

Создание скелетонов проектов

Проект StudyOn будем создавать из website-skeleton, а проект StudyOn.Billing — из обычного skeleton.

В обычном скелетоне не ставятся компоненты Form, Validator, Translation, Doctrine, Twig, веб-профайлер и ряд других. Его размер в 6 раз меньше, чем размер полного скелетона, за счет чего проект будет работать быстрее. Большинство из библиотек не понадобятся в биллинге, а те, что понадобятся, мы поставим при необходимости.

https://github.com/intaro/symfony-course/tree/master/practice/lesson-01#создание-скелетонов-проектов

Подробнее про создание нового symfony-проекта
https://symfony.com/doc/current/setup.html

Настройка .env

В файле .env не должно быть секретных данных и локальных настроек.

Подробнее про работу с .env читайте в статьях:
https://symfony.com/doc/current/configuration.html#the-env-file-environment-variables
https://symfony.com/doc/current/configuration/external_parameters.html
https://symfony.com/doc/current/configuration/environments.html

Создайте файл .env.local, перенесите в него APP_SECRET из .env, а также перенесите и задайте TRUSTED_HOSTS в соответствии с доменом сервиса

```
# в study-on/.env.local
TRUSTED_HOSTS='^study-on\.local$'
# в study-on.billing/.env.local
TRUSTED_HOSTS='^billing\.study-on\.local$'
```

Настройка проекта в PHPStorm и тюнинг .gitignore

Создайте 2 проекта в PHPStorm под каждый из сервисов, включите поддержку Symfony. Также добавьте в .gitignore исключение служебной папки PHPStorm-а — .idea.

Пример полного .gitignore
https://github.com/intaro/symfony-course/blob/master/practice/lesson-01/gitignore.template

Установите в PHPStorm плагины:

- PHP Annotations
- PHP Inspections
- Symfony plugin
- Makefile Support
- .env files support

Развертывание проектов в Docker

1. Скопируйте файл `docker-compose.yml` и папку `docker/` из https://github.com/intaro/symfony-course/tree/master/practice/lesson-01 в корень каждого проекта
2. Изучите содержимое папки и файла. В нем располагается сборка из nginx + php-fpm + postgresql.
3. Переименуйте файл `docker/hosts/app.conf.dist` в `docker/hosts/app.conf` и задайте домены для проектов: `study-on.local` и `billing.study-on.local`

4. Задайте в `.env`-файле NGINX_PORT порт, на котором будет слушать Nginx https://github.com/intaro/symfony-course/tree/master/practice/lesson-01#указание-порта-на-котором-будет-работать-nginx. Порты проектов должны быть разными, чтобы была возможность открывать проекты одновременно. `.env`-файл читает docker-compose, поэтому порт нужно задать именно в нем, а не в `.env.local`.
5. Зарегистрируйте домены сервисов в локальном hosts-файле https://github.com/intaro/symfony-course/tree/master/practice/lesson-01#регистрация-доменов-сервисов-в-локальных-хостах

Запустите оба сервиса
https://github.com/intaro/symfony-course/tree/master/practice/lesson-01#работа-с-docker-compose

После этого можно попробовать открыть проекте в браузере:

http://study-on.local:81
http://billing.study-on.local:82

где 81 и 82 – порты, которые вы указали в `NGINX_PORT`.

Также вы можете попробовать выполнить консольную команду в symfony-проекте
https://github.com/intaro/symfony-course/tree/master/practice/lesson-01#работа-с-docker-compose

Подробнее про docker-compose:
https://docs.docker.com/compose/overview/
https://docs.docker.com/compose/env-file/
https://docs.docker.com/compose/environment-variables/
https://docs.docker.com/compose/compose-file/

Настройка make

Скопируйте Makefile в корневую папку своих проектов
https://github.com/intaro/symfony-course/blob/master/practice/lesson-01/Makefile

Он предоставляет более удобную работу с распространенными командами
https://github.com/intaro/symfony-course/tree/master/practice/lesson-01#работа-с-make

Попробуйте запустить некоторые из них.

Сохранение проекта в Git

Закоммитьте файлы проекта и запушьте в репозиторий.

Подробнее про работу с Git
https://git-scm.com/book/ru/v2/Основы-Git-Запись-изменений-в-репозиторий
https://git-scm.com/book/ru/v2/Основы-Git-Просмотр-истории-коммитов
https://git-scm.com/book/ru/v2/Основы-Git-Операции-отмены
https://git-scm.com/book/ru/v2/Основы-Git-Работа-с-удалёнными-репозиториями