

2. Шаблонизация веб-страниц приложения

Целью данной лабораторной работы является получение навыков работы с шаблонизаторами пользовательских представлений.

Для выполнения данной лабораторной работы вам необходимо подключить шаблонизатор ([любой из списка](#)) и выделить повторяющиеся блоки в отдельные представления в вашем проекте.

Чтобы зарегистрировать выбранный вами шаблонизатор в приложение NestJS, необходимо выполнить его явно определить при инициализации приложения согласно [инструкции в документации](#).

Части (partials) веб страниц обязательных к выделению в отдельные представления:

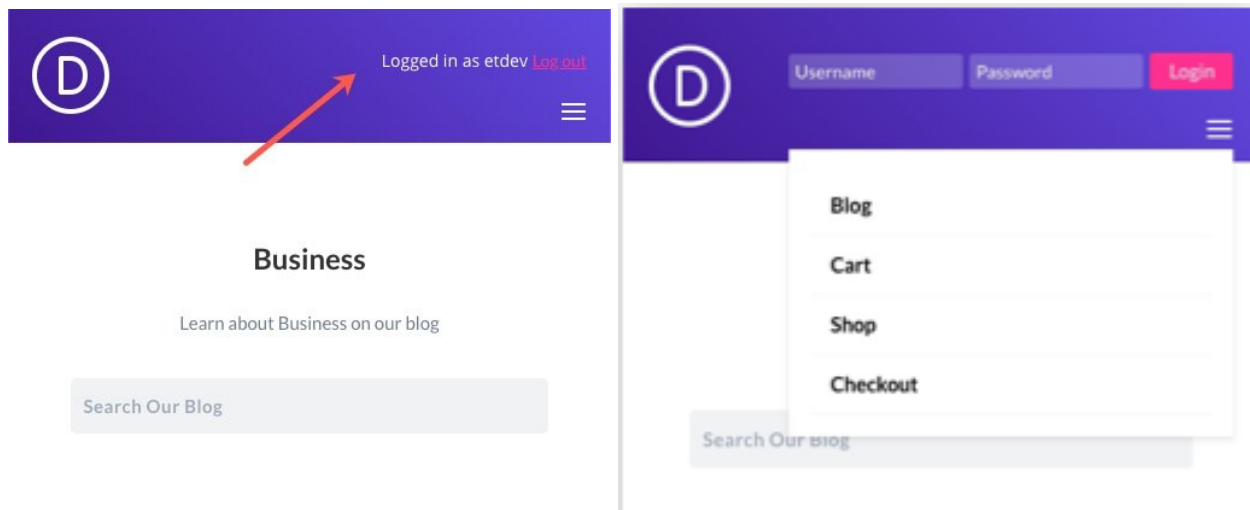
- Shared Script / Style bundles (общие стили и скрипты используемые на всех страницах повсеместно)
- Header (Заголовок)
 - Пункты меню
 - Информация о сессии (Вы вошли как ... / Кнопка регистрация и т.д.)
- Content (Контейнер под основное содержание страницы)
- Footer (Подвал)
 - Кол-во времени затраченного на выдачу страницы сервером

Ввиду отсутствия, на данный момент, контроллеров и маршрутов для ваших страниц, необходимо будет их добавить для обеспечения возможности рендеринга ваших представлений, например:

```
@Controller()
export class AppController {
  @Get()
  @Render('index') // <= Название вашего представления
  getIndexPage() {
    return { user: 'Hello world!' }; // Модель представления
  }
}
```

Подготовьте как минимум два состояния для представления информации

о текущей сессии пользователя (Авторизован и неавторизован).



После того как все представления будут готовы, необходимо обновить поле отвечающее за то, сколько времени потребовалось для отрисовки страницы, но, с включением того времени, которое понадобилось серверу для его обработки.

Для решения данной задачи рекомендуется имплементировать свой класс типа [Interceptor](#) и зарегистрировать его для выполнения перед каждым запросом внутри NestJS приложения. По сути ваш перехватчик запросов будет выполнять роль [логгирующего заместителя](#) с сохранением времени потраченного на обработку запроса непосредственно в HTTP ответе.

Т.к. Interceptor'ы используют концепции реактивного программирования, для того чтобы трансформировать ответ отправляемый сервером, необходимо будет создать подписку на объект наблюдения средствами RxJS. Рекомендуется ознакомиться с [документацией по RxJS](#) прежде чем применять [готовый рецепт](#) для модификации ответов.

После того как вы добавите метаданные (например в Header'ы ответа) о том, сколько времени понадобилось серверу на формирование страницы, добавьте это время к тому сколько браузер генерировал DOM и отобразите эту информацию на ваших страницах в подвале:

```
Total load time: 46 ms (client) + 1523 ms (server)
```