7. Добавление обработки данных реального времени с использованием Web-Socket канала

В результате выполнения лабораторной работы на разрабатываемом веб-сайте на страницах приложения будет добавлен виджет, отображающий последние изменения данных модели по выбору студента.

Для выполнения данной лабораторной работы вам необходимо добавить в приложение канал для работы с соединениями по протоколу WebSocket используя средства библиотеки Socket.IO

Ожидается, что в доменной модели могут быть данные, которые потенциально могут часто обновляться, такие данные следует обрабатывать в режиме реального времени (например: отображение курса валют в виде графика). В случаях, когда такие данные отсутствуют предлагается создать виджет для взаимодействия авторизованных пользователей между собой.

Для того чтобы подключить к Nest приложению возможность работы с веб-сокетами, необходимо установить следующие библиотеки:

```
@nestjs/websockets
@nestjs/platform-socket.io
@types/socket.io
```

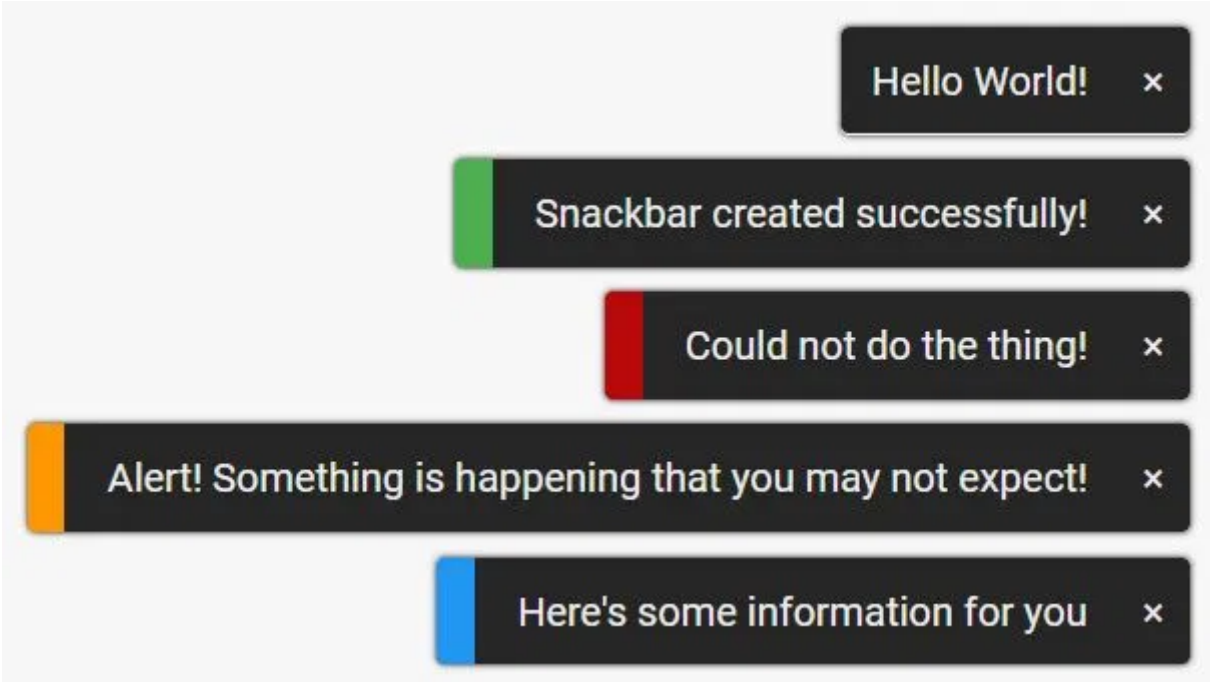
Согласно лучшим практикам из прошлых лабораторных работ, первым делом необходимо создать отдельный модуль под ваш Websocket'ный Gateway.

Добавьте класс Gateway, который будет обслуживать WebSocket канал как в примере из лекции. Для того чтобы подключится к вашему Gateway'ю можно было только с доверенного ресурса, настройке CORS политику на тот же порт и домен, где хостится ваше приложение (желательно вынести этот параметр в Environment Variable)

После создания Gateway'я, выделите в вашей доменной модели одну-две сущности, в обновлении которых может быть заинтересован клиент. Например, если вы делаете интернет магазин, в WebSocket канал можно передать информацию об обновлении товаров и/или цен на них.

Для того чтобы оповещать пользователей об изменениях рекомендуется использовать широковещательную рассылку сообщений по конкретному namespace'у. Для удобства можно создать собственные пространства имен, в которые бы приходила информация об обновлении конкретной группы данных. Чтобы настроить собственное пространство имен, мы можем вызвать функцию 'of' на стороне сервера.

После добавления самого модуля, имплементируйте клиентскую часть. Т.е. необходимо отобразить вариант использования вашего WebSocket канала в приложении. Как вариант, можно использовать библиотеку toster.js для оповещения пользователя о возникновении тех или иных событий, на которые вы создали подписку



.

В случаях, когда вы ожидаете, что к вашему Gateway’ю должны подключаться только авторизованные пользователи, необходимо подключить ваш AuthGuard из предыдущей лабораторной работы с маленькими модификациями. Необходимо переключать контекст не в HTTP, а в WS. Также в случае возникновения ошибок, будут генерироваться WsException’ы вместо HttpException’ов. Такие исключения можно обрабатывать в Gateway’ях отдельно.