# Онлайн-магазин

## 

[Онлайн-магазин](#_j5671zetm34w)

[0. Изменения.](#_vk9m777m8j5e)

[1. Введение](#_e6xlzab33lp4)

[2. Описание системы.](#_4y2tcc1qjc3)

[2.1. Администраторы и клиенты.](#_fi8rksfwpnyv)

[2.2. Категории и товары](#_vavevkxiztr7)

[2.3. Покупки](#_i5uvz7dick9u)

[3. REST-интерфейс](#_6jvau9gx1251)

[3.1. Общие правила](#_wy0g9gcjf5g1)

[3.2. Регистрация администратора.](#_do68cxn1jmgc)

[3.3. Регистрация клиента.](#_n2yba9rxn4oj)

[3.4. Login.](#_2ywm16oqmg2t)

[3.5. Logout.](#_h2ytru4qs5ac)

[3.6. Получение информации о текущем пользователе](#_k8vgnop2evz5)

[3.7. Администратор - получение информации о клиентах](#_r506hjulte0w)

[3.8. Редактирование профиля администратора](#_8q5a2fkf856u)

[3.9. Редактирование профиля клиента](#_z6q62l393i1n)

[3.10. Администратор - добавление категории или подкатегории.](#_oukkwgn8w7f)

[3.11. Администратор - получение категории или подкатегории.](#_ieuzboyfbb8a)

[3.12. Администратор - редактирование категории или подкатегории.](#_rea9paq12jmp)

[3.13. Администратор - удаление категории или подкатегории.](#_ftbp0ibirnsh)

[3.14. Администратор - получение списка категорий и подкатегорий.](#_iak84c3tj807)

[3.15. Администратор - добавление товара.](#_a6uiiw7c8hy5)

[3.16. Администратор - изменение данных товара.](#_dc28hm5a106x)

[3.17. Администратор - удаление товара.](#_9p60qp6dukq0)

[3.18. Администратор и клиент - получение данных товара.](#_ry802lqpjie)

[3.19. Администратор и клиент - получение списка товаров.](#_ct19c319pnyk)

[3.20. Клиент - внесение денег на счет](#_9n4dpp7fqhoi)

[3.21. Клиент - получение суммы денег, имеющейся на счете](#_450m3y4phzhv)

[3.22. Клиент - покупка товара.](#_bkz50oksi6w8)

[3.23. Клиент - добавление товара в корзину.](#_icx43rjcd2s)

[3.24. Клиент - удаление товара из корзины.](#_npcpu7j9z307)

[3.25. Клиент - изменение количества товара в корзине.](#_gj34aqui97pi)

[3.26. Клиент - получение состава корзины.](#_yyv6tjrtltn4)

[3.27. Клиент - покупка товаров из корзины.](#_v0hgamli4o05)

[3.28. Администратор - получение сводной ведомости.](#_1aiwaiqjv6i0)

[3.29. Получение настроек сервера](#_yrr7vn3c6fkr)

[3.30. Очистка базы данных.](#_kb147ss7gy5k)

[Приложение 1. Конфигурационный файл сервера.](#_nitfmae1ixwi)

[Таблица 1. Параметры конфигурационного файла](#_mf19ltjfmpsn)

[Приложение 2 . Требования к оформлению.](#_qnd3a0q5ydtp)

## 

## 0. Изменения.

## 

|  |  |  |
| --- | --- | --- |
| № | Дата | Содержание |
| 1 | 11.3 | Из п.3.3 удален формат адреса клиента. Адрес - просто некоторая текстовая строка произвольного вида. |
| 2 | 14.3 | Убраны “” в описании ID. ID - всегда целое число, а не строка |
| 3 | 15.3 | Уточнено, что для всех текстовых данных при регистрации максимальная длина не более ***max\_name\_length*** |
| 4 | 20.3 | Добавлен эндпойнт “Получение состава корзины” (3.26) |
| 5 | 21.3, 25.3 | В эндпойнте получения сводной ведомости (3.28), внесено уточнение. Выдачу информации по отдельным категориям или списку категорий нужно проводить в соответствии с тем, к каким категориям принадлежат товары сейчас, а не в момент покупки.  Это же относится к названию продукта, цене и т.д. - при выдаче отчета по продукту используются его текущие реквизиты без учета того, что в прошлом он мог иметь иные. |
| 5 | 4.4 | В п.3.12 внесено уточнение. При редактировании категория не может стать подкатегорией, и наоборот, подкатегория не может стать категорией. |
| 6 | 13.4 | В п. 3.27 уточнено условие пропуска товара. Теперь оно звучит так  Если в покупку включен товар из числа удаленных, с несуществующим ID или отсутствующий в корзине, запрос для этого товара не выполняется. |
| 7 | 15.4 | Уточнены требования в п 3.3. Теперь они звучат так : Требования к логину, паролю, имени, фамилии и отчеству клиента - те же, что и для администратора. Для всех текстовых строк максимальная длина не может превышать ***max\_name\_length.*** |
| 8 | 17.4 | Допускается хранение параметров конфигурационного файла в файле application.properties. В этом случае отдельный конфигурационный файл не требуется. Формат параметров тот же самый. |

## 1. Введение

В компании X, занимающейся продажей товаров, решено создать онлайн-магазин, в котором клиенты могли бы делать свои заказы и оплачивать их.

Для товара определено понятие “категории”. Любой товар может относиться к одной или нескольким категориям.

Администраторы магазина должны иметь возможность добавлять и удалять категории и товары. Клиенты должны иметь возможность формировать корзину клиента и покупать товары.

Вам предстоит написать серверную часть приложения.

Руководство компании предполагает, что высказанные им требования к сайту являются логичными и непротиворечивыми. Так ли это в действительности или нет - должно выясниться в процессе разработки и пробной эксплуатации сайта. Если в ходе разработки или пробной эксплуатации появится необходимость корректировки тех или иных пунктов задания, то этот вопрос должен обсуждаться совместно представителями заказчика и разработчика с внесением в случае необходимости изменений в техническое задание.

## 2. Описание системы.

### 2.1. Администраторы и клиенты.

Администраторы и клиенты должны зарегистрироваться на сайте. При регистрации они указывают

Администратор :

* фамилию, имя, отчество (последнее - если имеется)
* должность
* логин и пароль для входа на сайт.

Клиент :

* фамилию, имя, отчество (последнее - если имеется)
* e-mail
* почтовый адрес
* номер сотового телефона
* логин и пароль для входа на сайт.

Администраторы и клиенты могут изменить свои регистрационные данные, но не могут изменять логин.

### 2.2. Категории и товары

Любой товар может (но не обязан) относиться к одной или нескольким категориям. Имена всех категорий уникальны. Примеры категорий : “галантерея”, “одежда”, “обувь” и т.д.

Для категорий могут быть созданы подкатегории. Например, для категории “одежда” могут быть созданы подкатегории “мужская одежда”, “женская одежда”, “детская одежда”. В настоящее время подкатегории создаются только для категорий, для подкатегорий подкатегории не создаются, хотя не исключено, что в дальнейшем это может быть изменено. Изначально список категорий пуст. Администратор может добавить новую категорию или подкатегорию к ранее добавленной категории.

Администратор может добавить новый товар, при этом он может указать список категорий и подкатегорий, к которым этот товар относится. Если такой список категорий при добавлении товара не указывается, то товар считается не относящимся ни к какой категории. Товар может относиться к любому набору категорий и подкатегорий, в частности, возможна ситуация, когда товар относится одновременно к некоторой категории и любому набору ее подкатегорий. В дальнейшем список категорий и подкатегорий для товара может быть изменен. Стоимость товара указывается в рублях, всегда целое число, копейки не используются.

При поступлении в магазин новой партии товара администратор добавляет количество единиц для этого товара. Администратор имеет право удалить некоторое количество единиц товара (например, если экземпляры оказались бракованными) или вообще удалить товар (например, при истечении срока годности).

### 

### 2.3. Покупки

Для того, чтобы купить товар, зарегистрированный клиент должен сначала положить некоторую сумму денег (рублей) на свой счет. Счет создается для каждого клиента автоматически при регистрации. Зарегистрированный клиент может купить любой товар в любом количестве, если он имеется в наличии. Если клиент принимает решение купить товар, то он условно оплачивает его. Оплата товара производится только со счета клиента.

Альтернативно, клиент может добавить товар в свою Корзину клиента. В дальнейшем он может удалить этот товар из корзины или купить все или некоторые товары, находящиеся в Корзине.

Товары, находящиеся в Корзине, не считаются купленными клиентом, поэтому не исключена ситуация, что товар имелся в наличии в момент его добавления в Корзину, а в момент покупки этого товара уже нет либо в нужном количестве, либо вообще.

## 3. REST-интерфейс

### 3.1. Общие правила

Для взаимодействия клиентской и серверной частей приложения используется REST-интерфейс. Клиент посылает REST - запрос (Request), передавая в нем строку в формате JSON. Некоторые параметры передаются через заголовок HTTP-запроса. В ответ (Response) сервер возвращает свою строку, также в формате JSON, которая содержит информацию о результате выполненной операции.

**Для идентификации администратора или клиента используются cookie. При выполнении операции “Login” сервер возвращает cookie с именем “JAVASESSIONID”. Во всех остальных запросах предполагается, что клиент передает эту cookie.**

Далее в документе принята следующая система обозначений.

Текст, содержащий латинские буквы и цифры, должен присутствовать в Request или Response в точности так, как написано. Текст, содержащий русские буквы, должен быть заменен на некоторое допустимое значение.

Например, "name":"имя пользователя" означает, что имя параметра должно быть “name”, а вместо “имя пользователя” должно быть указано некоторое допустимое имя.

Если значение параметра приведено в кавычках, параметр является текстовым, если без кавычек - целый типа int или long.

Порядок следования полей в Request и Response может быть произвольным.

При успешном выполнении всегда возвращается код HTTP 200.

В случае, если запрос не был выполнен из-за какой-то ошибки в нем, возвращается код HTTP 400 (для запроса по несуществующему URL - 404), и json следующего вида

{

"errors" : [  
 {  
 **"errorCode"**: "код ошибки",   
 **"field"**: "поле запроса, являющееся причиной ошибки",   
 **"message"**: "причина ошибки"  
 }

...  
]

}

Например

{

"errors" : [  
 {  
 **"errorCode"**: "LOGIN\_ALREADY\_EXISTS" ,  
 **"field"**: "login",   
 **"message"**: "User Ivanov already exists"   
 }  
]

}

### 3.2. Регистрация администратора.

POST /api/admins

Request

|  |  |
| --- | --- |
| json | **{**  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"position"**: "должность",  **"login"**: "логин",   **"password"**: "пароль"  **}** |

Логин может содержать только латинские и русские буквы и цифры и не может быть пустым. Пароль может содержать любые символы и тоже не может быть пустым. Максимальная длина логина, пароля, фамилии , имени и отчества и должности не более ***max\_name\_length[[1]](#footnote-1)*** символов. Минимальная длина пароля ***min\_password\_length*** символов. Логин должен храниться так, как он задан, но не является case-sensitive при дальнейшей работе. Например, если администратор регистрировался с логином “Иванов”, он может впоследствии заходить на сервер, используя логины “Иванов”, “иванов”, “иВаНоВ” и т.д. Пароль является case-sensitive.

Имя, фамилия и отчество администратора могут содержать только русские буквы , пробелы и знак “минус” (используемый как тире).

Для зарегистрировавшегося администратора автоматически выполняется операция “Login” (п.3.4)

Response

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| json | **{**  **"id"**: идентификационный номер,  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"position"**: "должность",  **}** |

### 

### 3.3. Регистрация клиента.

POST /api/clients

Request

|  |  |
| --- | --- |
| json | **{**   **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"email"**: "email",  **"address"**: "почтовый адрес",  **"phone"**: "номер телефона",   **"login"**: "логин",  **"password"**: "пароль",  **}** |

Требования к логину, паролю, имени, фамилии и отчеству клиента - те же, что и для администратора. Для всех текстовых строк максимальная длина не может превышать ***max\_name\_length.***

E-mail должен соответствовать требованиям, предъявляемым к формату e-mail.

Почтовый адрес записывается в произвольной форме. Он не может быть пустым, другие требования к нему не предъявляются.

Допустимые телефонные номера - сотовые номера любых операторов России. Номер может начинаться как с “8”, так и с “+7”. Наличие в номере знаков “-” (дефис) ошибкой не является, но перед записью в БД все знаки “-” удаляются. Номера телефонов стационарной связи указывать не разрешается.

Для зарегистрировавшегося клиента автоматически выполняется операция “Login” (п.3.4)

Response

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| json | **{**  **"id"**: идентификационный номер,  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"email"**: "email",  **"address"**: "почтовый адрес",  **"phone"**: "номер телефона",  **"deposit"**: "сумма денег на счете",  **}** |

Для этого запроса значение “deposit” всегда 0.

### 

### 

### 3.4. Login.

POST /api/sessions

Request

|  |  |
| --- | --- |
| json | **{**  **"login"**: "логин",   **"password"**: "пароль" **}** |

Response

То же, что и для 3.2 или 3.3 соответственно.

### 3.5. Logout.

DELETE /api/sessions

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

|  |  |
| --- | --- |
| json | **{}** |

### 

### 3.6. Получение информации о текущем пользователе

GET /api/accounts

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

То же, что и для 3.2 или 3.3 соответственно.

### 

### 

### 3.7. Администратор - получение информации о клиентах

GET /api/clients

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

|  |  |
| --- | --- |
| json | **[**  **{**  **"id"**: идентификационный номер,  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"email"**: "email",  **"address"**: "почтовый адрес",  **"phone"**: "номер телефона",   **"userType"**: "client"  **}**  **]** |

Величина депозита в этом запросе не возвращается !

### 3.8. Редактирование профиля администратора

PUT /api/admins

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

|  |  |
| --- | --- |
| json | **{**  **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество",   **"position"**: "должность",   **"oldPassword"**: "прежний пароль",  **"newPassword"**: "новый пароль"  **}** |

Response

То же, что и для 3.2

### 

### 3.9. Редактирование профиля клиента

PUT /api/clients

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

|  |  |
| --- | --- |
| json | **{**   **"firstName"**: "имя",   **"lastName"**: "фамилия",   **"patronymic"**: "отчество", // необязателен  **"email"**: "email",  **"address"**: "почтовый адрес",  **"phone"**: "номер телефона",   **"oldPassword"**: "прежний пароль",  **"newPassword"**: "новый пароль"  **}** |

Response

То же, что и для 3.3

### 

### 3.10. Администратор - добавление категории или подкатегории.

POST /api/categories

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"name"**: "название категории или подкатегории",   **"parentId"**: идентификатор родительской категории // необязателен **}** |

Имена всех категорий (в том числе подкатегорий) уникальны. Имя категории не может быть пустым.

Если поле “parentId” в запросе не указано или равно 0, добавляется категория, в противном случае добавляется подкатегория к категории “parentId”, которая должна существовать.

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификатор категории или подкатегории,   **"name"**: "название категории или подкатегории",   **"parentId"**: "идентификатор родительской категории", // только для подкатегорий   **"parentName"**: "название родительской категории" // только для подкатегорий  **}** |

### 3.11. Администратор - получение категории или подкатегории.

GET /api/categories/номер\_категории

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификатор категории или подкатегории,   **"name"**: "название категории или подкатегории",   **"parentId"**: "идентификатор родительской категории", // только для подкатегорий   **"parentName"**: "название родительской категории" // только для подкатегорий  **}** |

### 

### 

### 3.12. Администратор - редактирование категории или подкатегории.

PUT /api/categories/номер\_категории

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"name"**: "новое название категории или подкатегории",   **"parentId"**: идентификатор родительской категории **}** |

При редактировании категория не может стать подкатегорией, и наоборот, подкатегория не может стать категорией.

Хотя бы одно из полей запроса не должно быть пустым.

Если в запросе присутствует поле “name” - изменяется название категории или подкатегории.

Если в запросе присутствует поле “parentId” - подкатегория перемещается в другую категорию. Для категорий такой запрос недопустим.

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификатор категории или подкатегории,   **"name"**: "название категории или подкатегории",   **"parentId"**: идентификатор родительской категории, // только для подкатегорий   **"parentName"**: "название родительской категории" // только для подкатегорий |

Редактирование категории или подкатегории не влияет на находящиеся в ней товары - они по-прежнему принадлежат этой категории или подкатегории

### 3.13. Администратор - удаление категории или подкатегории.

DELETE /api/categories/идентификатор\_категории

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

|  |  |
| --- | --- |
| json | **{}** |

Удаление подкатегории приводит к тому, что все находившиеся в нем товары больше к этой подкатегории не принадлежат. Сами товары не удаляются.

Удаление категории приводит к удалению всех ее подкатегорий. Все находившиеся в подкатегориях этой категории товары больше не принадлежат к этим подкатегориям.

Если после удаления категории или подкатегории список категорий для некоторого товара оказывается пустым, считается, что этот товар теперь не принадлежит ни к одной категории или подкатегории.

### 3.14. Администратор - получение списка категорий и подкатегорий.

GET /api/categories

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

|  |  |
| --- | --- |
| json | **[**  **{**  **"id"**: идентификатор категории или подкатегории,   **"name"**: "название категории или подкатегории",   **"parentId"**: идентификатор родительской категории, // только для подкатегорий   **"parentName"**: "название родительской категории" // только для подкатегорий   **}**  **]** |

Список выдается, отсортированный по именам категорий, за каждой из которых следуют ее подкатегории, также отсортированные по имени

### 3.15. Администратор - добавление товара.

POST /api/products

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"name"**: "название товара",   **"price"**: стоимость за единицу товара,   **"count"**: количество единиц товара, // необязателен  **"categories"**: // необязателен  **[**  идентификатор категории1, ...идентификатор категорииN  **] }** |

### 

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара,   **"count"**: количество единиц товара,   **"categories"**:  **[**  идентификатор категории1, идентификатор категории2,...идентификатор категорииN  **] }** |

### 

Названия товаров могут совпадать у разных товаров.

Если количество единиц товара равно 0 или отсутствует - товар создается с количеством единиц, равным 0 (иными словами, товар вносится в базу, но в наличии его пока нет).

Цена товара не может быть <=0.

Список категорий может быть пустым или отсутствовать вообще, в этом случае товару устанавливается пустой список категорий.

### 3.16. Администратор - изменение данных товара.

PUT /api/products/идентификатор\_товара

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"name"**: "название товара",   **"price"**: стоимость за единицу товара,   **"count"**: количество единиц товара,  **"categories"**:  **[**  идентификатор категории1,...идентификатор категорииN  **] }** |

Все параметры необязательны. При отсутствии параметра соответствующий элемент не изменяется.

Если поле “categories” отсутствует, список категорий продукта не изменяется. Если поле “categories” представляет собой пустой список, продукту устанавливается пустой список категорий .

### 

Response

Тот же, что и для 3.15

### 3.17. Администратор - удаление товара.

DELETE /api/products/идентификатор\_товара

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Response

|  |  |
| --- | --- |
| json | **{}** |

### 

Удаление товара разрешается даже если количество его единиц не равно 0.

Удаление товара разрешено даже если удаляемый товар находится в какой-то корзине покупателя. В случае удаления такого товара он из корзины не удаляется, но покупка его из корзины становится невозможной.

### 

### 3.18. Администратор и клиент - получение данных товара.

GET /api/products/идентификатор\_товара

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"count"**: количество единиц товара,   **"price"**: стоимость за единицу товара,  **"categories"**: [“название категории1”, … название категорииN”]  **}** |

### 

### 3.19. Администратор и клиент - получение списка товаров.

GET /api/products

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |
| RequestParam | **category =** [идентификатор категории1, ...идентификатор категорииN] // необязателен |
| RequestParam | **order =** “product” | “category” // необязателен |

Response

|  |  |
| --- | --- |
| json | **[**  **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"count"**: количество единиц товара,   **"price"**: стоимость за единицу товара,  **"categories"**: [“название категории1”, … название категорииN”]  **}**  **]** |

Возвращается список товаров, принадлежащих хотя бы одной из указанных категорий. Если список категорий не указан в запросе, возвращается полный список товаров, включая товары, не относящиеся ни к одной категории. Если передается пустой список категорий, выдается список товаров, не относящихся ни к одной категории.

Если order = “product”, список выдается, отсортированный по именам товаров, и в этом случае в поле “categories” приводится список категорий, к которым он относится. Каждый товар приводится в списке только один раз.

Если order = “category”, список выдается, отсортированный по именам категорий, а внутри категории - по именам товаров. В этих случаях каждый товар указывается для каждой своей категории, а в поле categories возвращается лишь одна категория. Товары, не относящиеся ни к одной категории, выдаются в начале списка.

Если order не присутствует в запросе,то считается, что order = “product”.

### 

### 3.20. Клиент - внесение денег на счет

PUT /api/deposits

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"deposit"**: "сумма денег, которая кладется на счет” **}** |

Клиент может вносить деньги только на свой счет.

Response

То же, что и 3.3

### 

### 3.21. Клиент - получение суммы денег, имеющейся на счете

GET /api/deposits

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Response

То же, что и 3.3

### 

### 3.22. Клиент - покупка товара.

POST /api/purchases

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"Id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара // необязателен **}** |

### 

Response

|  |  |
| --- | --- |
| json | **{**  **"id"**: "идентификатор товара",   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}** |

### 

Если количество единиц в запросе не указано, то оно принимается равным 1.

Запрос отвергается, если

* не имеется требуемое количество единиц товара,
* суммарная стоимость всех единиц товара превышает количество денег на счете клиента,
* указанные в запросе название товара или стоимость за единицу отличаются от текущих значений для этого продукта

### 3.23. Клиент - добавление товара в корзину.

POST /api/baskets

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара // необязателен **}** |

### 

Response

|  |  |
| --- | --- |
| json | **[**  **{**  **"id"**: "идентификатор товара",   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}**  **]** |

### 

Если количество единиц в запросе не указано, то оно принимается равным 1.

Запрос отвергается, если указанные в запросе название товара или стоимость за единицу отличаются от текущих значений для этого продукта

Допускается добавление в корзину любого количества единиц товара независимо от того, сколько единиц такого товара имеется в продаже.

Клиент может добавить в корзину любое количество единиц любого товара, независимо от того, сколько денег у него имеется на счете.

В Response возвращается полный состав корзины.

### 

### 3.24. Клиент - удаление товара из корзины.

DELETE /api/baskets/идентификатор\_товара

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Response

|  |  |
| --- | --- |
| json | **{}** |

### 

### 

### 3.25. Клиент - изменение количества товара в корзине.

PUT /api/baskets

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара **}** |

### 

Response

Тот же, что и для 3.23

Запрос отвергается, если указанные в запросе название товара или стоимость за единицу отличаются от текущих значений для этого продукта

Допускается изменение количества единиц товара независимо от того, сколько единиц такого товара имеется в продаже , более того, разрешается изменение количества единиц для удаленного товара.

Данный запрос не может добавлять новые товары в корзину.

### 

### 3.26. Клиент - получение состава корзины.

GET /api/baskets

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

### 

Response

|  |  |
| --- | --- |
| json | **[**  **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}**  **…**  **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}**  **]** |

### 3.27. Клиент - покупка товаров из корзины.

POST /api/purchases/baskets

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

|  |  |
| --- | --- |
| json | **[**  **{**  **"Id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара // необязателен  **}**  **...**  **{**  **"id"**: "идентификатор товара",   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара // необязателен  **}**  **]** |

### 

Response

|  |  |
| --- | --- |
| json | **{**  **“bought” : [**  **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}**  **…**  **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}**  **]**  **“remaining” : [**  **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}**  **...**  **{**  **"id"**: идентификатор товара,   **"name"**: "название товара",   **"price"**: стоимость за единицу товара  **"count"**: количество единиц товара,  **}**  **]**  **}** |

### 

Если для какого-то товара количество единиц в запросе не указано, то покупается то количество единиц, которое есть сейчас в корзине.

Если для какого-то товара указанные в запросе название товара или стоимость за единицу отличаются от текущих значений для этого продукта, запрос для этого продукта не выполняется, товар остается в корзине.

Не допускается покупка большего числа единиц, чем имеется в корзине. Если такое имеет место для какого-то товара, то покупается то количество единиц, которое есть сейчас в корзине

При покупке товара должно проверяться наличие в продаже требуемого количества единиц по каждому товару. Для тех товаров, количество единиц которых, имеющееся в продаже, меньше количества единиц в корзине, запрос не выполняется, товары остаются в корзине.

Если в покупку включен товар из числа удаленных, с несуществующим ID или отсутствующий в корзине, запрос для этого товара не выполняется.

Если суммарная стоимость всей покупки (с учетом вышесказанного) превышает количество денег на счете клиента, запрос отвергается целиком.

В Response возвращаются список купленных товаров и список товаров, оставшихся в корзине.

### 3.28. Администратор - получение сводной ведомости.

GET /api/purchases/параметры

Этот endpoint Вам предстоит разработать самим. Необходимо предусмотреть как выдачу всей информации, так и выдачу информации по отдельным категориям или списку категорий (в соответствии с тем, к каким категориям принадлежат товары сейчас, а не в момент покупки), по товару или списку товаров, по клиентам и т.д. Желательно предусмотреть критерии упорядочения результирующей выборки. Ответ должен также содержать итоговые значения по выборке. Например, если возвращается список покупок некоторого клиента, в ответ надо включить их суммарную стоимость. Также необходимо предусмотреть вариант, когда выдаются только итоговые значения, без подробностей - в тех случаях, когда это имеет смысл.

Ввиду того, что данный запрос может возвращать очень много данных, следует предусмотреть пагинацию результатов, введя параметры запроса “offset” (номер строки результата, с которой начать выдачу) и “limit” (количество строк). Итоговые значения при этом приводятся для возвращаемой выборки, а не для всего списка.

### 3.29. Получение настроек сервера

GET /api/settings

Request

|  |  |
| --- | --- |
| cookie | JAVASESSIONID |

Параметр “cookie” для этого запроса не является обязательным. Если он передается, то для администратора выдаются доступные ему настройки, а для клиента - доступные ему. Если cookie не передается в запросе, возвращается список настроек, доступных до выполнения операции “Login”

Response

В настоящее время для всех 3 случаев выдается один и тот же результат. Это поведение может быть в дальнейшем изменено.

|  |  |
| --- | --- |
| json | **{**  **"maxNameLength"**: значение ***max\_name\_length,***  **"minPasswordLength"**: значение ***min\_password\_length,*** **}** |

### 3.30. Очистка базы данных.

POST /api/debug/clear

Request

Отсутствует

Response

|  |  |
| --- | --- |
| json | {} |

### 

Удаляет все записи в БД. Метод предназначен для отладки, в production должен быть отключен.

### 

## Приложение 1. Конфигурационный файл сервера.

Конфигурационный файл используется для задания параметров работы сервера и содержит элементы вида “параметр = значение”. Каждый элемент находится в отдельной строке файла (таблица 1). Порядок следования элементов в файле произвольный. Все параметры обязательные. Формат данных в конфигурационном файле всегда правильный и не проверяется, поэтому установка недопустимых по типу или величине значений может привести к неопределенному поведению.

Во время работы сервера изменение параметров конфигурационного файла на работу сервера не влияет. Если значения параметров были изменены, необходимо перезапустить сервер для того, чтобы новые значения вступили в действие,.

Имя конфигурационного файла передается через командную строку при запуске сервера. В случае, если командная строка не содержит имени конфигурационного файла, или файл не найден, или не содержит хотя бы одного параметра, сервер выводит сообщение в лог и завершает свою работу.

Вместо отдельного конфигурационного файла допускается помещение этих параметров в файл application.properies.

### Таблица 1. Параметры конфигурационного файла

|  |  |  |
| --- | --- | --- |
| Параметр | Тип | Назначение |
| rest\_http\_port | int | Порт, на котором работает REST-сервер. |
| max\_name\_length | int | Максимальная длина имени, логина и пароля. |
| min\_password\_length | int | Минимальная длина пароля |

Пример конфигурационного файла

rest\_http\_port = 8888

max\_name\_length = 50

min\_password\_length = 8

## Приложение 2 . Требования к оформлению.

В pom.xml groupId и artifactId должны быть заданы следующим образом :

<groupId>net.thumbtack</groupId>

<artifactId>onlineshop</artifactId>

Класс, содержащий метод main, должен находиться в пакете net.thumbtack.onlineshop и иметь имя OnlineShopServer

Имена остальных пакетов и классов - на ваше усмотрение, но рекомендуется следовать общепринятому способу именования, аналогично Заданию 11.

Имя БД - onlineshop. Имена таблиц - на Ваше усмотрение

## Приложение 3. Этапы разработки

1. Неделя 1 - Архитектура проекта, структура БД, IOC, DAO, интерфейсы сервисов, unit-тестирование функционала
2. Неделя 2 - REST функционал администратора, интеграционные е2е тесты
3. Неделя 3 - REST функционал клиента, интеграционные е2е тесты
4. Неделя 4 - REST функционал сводная ведомость, интеграционные е2е тесты

1. Здесь и далее слова, выделенные жирным курсивом, обозначают параметры конфигурационного файла (Приложение 1) [↑](#footnote-ref-1)