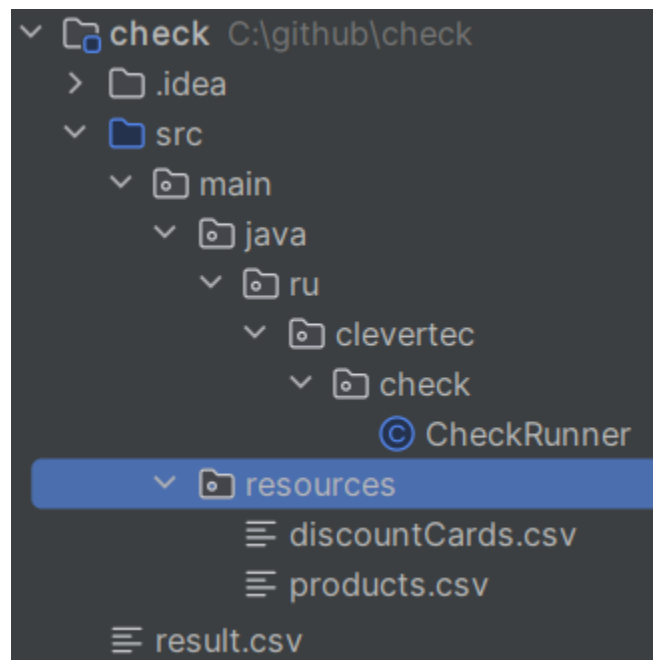


# Консольное приложение, реализующее функционал формирования чека в магазине

## Тестовое задание 1

### Требования к приложению:

1. Стек: **Java 21**
2. В данном задании важно показать понимание ООП, умение строить модели (выделять классы, интерфейсы, их связи), разделять функционал между ними, а также знать синтаксис самого языка. Обязательно применение паттернов проектирования (например, Factory, Builder, FactoryMethod). Обратить внимание на устойчивость к изменениям (последующая поддержка этой программы) в логике и избегать копипаста
3. Все классы должны быть в пакете **src/main/java/ru/clevertec/check**



4. main-класс: `./src/main/java/ru/clevertec/check/CheckRunner.java`
5. Место хранения файлов `products.csv` и `discountCards.csv`: `./src/main/resources`
  - а. При указании пути к вычитываемым файлам использовать строгий формат

```
"./src/main/resources/products.csv"
"./src/main/resources/discountCards.csv"
```

6. Приложение должно запускаться с помощью консольной команды:

```
java -cp src ./src/main/java/ru/clevertec/check/CheckRunner.java id-quantity discountCard=xxxx  
balanceDebitCard=xxxx
```

где:

<i>id</i>	- идентификатор товара (см. Таблицу 1)
<i>quantity</i>	- количество товара
<i>discountCard=xxxx</i>	- название и номер дисконтной карты (см. Таблицу 2)
<i>balanceDebitCard=xxxx</i>	- баланс на дебетовой карте

Примечание:

- Название и путь CSV-файла: **result.csv** в корне проекта
- всё указываем через пробел
- *id-quantity* - добавляем префикс *id*-(количество товара). В наборе параметров должна быть минимум одна такая связка "*id-quantity*"
- *discountCard=xxxx* - добавляем префикс *discountCard*=(любые четыре цифры)
- *balanceDebitCard=xxxx* - указываем префикс *balanceDebitCard*=(любая сумма на счете). Обязательно должна присутствовать. Баланс может быть как и с двумя знаками после запятой, так и отрицательный

**ВАЖНО!** *id* могут повторяться: 1-3 2-5 1-1 тоже, что и 1-4 2-5

**ВАЖНО!** *xxxx* - для числа с плавающей точкой разделитель точка

Пример: 1.12

Пример:

```
java -cp src ./src/main/java/ru/clevertec/check/CheckRunner.java 3-1 2-5 5-1 discountCard=1111  
balanceDebitCard=100
```

По команде выше должен сформироваться CSV-файл (**result.csv**) в корне проекта, содержащий в себе список товаров и их количество с ценой, а также рассчитанную сумму с учетом скидки по предъявленной карте, если она есть.

**Расшифровка команды:** (3-1) 3 - это товар с *id* = 3, 1 - количество (1шт); тоже самое с (2-5) *id*=2 в количестве 5 штук, (5-1) *id*=5 - одна штука и т. д.; *discountCard=1111* - означает, что была предъявлена скидочная карта с номером 1111. Необходимо продублировать чек в консоль (рисунок 1)

7. Среди товаров, предусмотрены оптовые (**Схема 10**. Таблица 1, колонка wholesale product). Если в чеке, в оптовой позиции пять и более товаров, то сделать скидку 10% по этой позиции (если применяется оптовая скидка 10% на позицию, то на данную позицию не действует персональная скидка по карте, например 3%). Данную информацию отразить в чеке
8. Набор товаров и скидочных карт вычитывать из файла. Их количество и номенклатура взять строго из Таблицы 1 и Таблицы 2 (**изменять или добавлять ничего нельзя**)
9. Реализовать обработку исключений (например, нет товара с id или не передан баланс карты и т. д.) смотри **схемы 5, 6, 7**
10. Реализовать вывод чека в CSV-файл (формат приведен на **схеме 1**). Также продублировать расширенно всю необходимую информацию в консоль (на свое усмотрение что и как)
11. Описать README.md файл в свободной форме (инструкцию по запуску)
12. **Исходники разместить в присланном репозитории в ветке `feature/entry-core`**

“\*” - Необязательно, но будет существенным плюсом. К этому заданию лучше приступать после качественного решения базовой задачи с применением принципов SOLID, декларативных подходов, оптимальных алгоритмов.

## Тестовое задание 2\*

13. Реализовать считывание списка товаров из внешнего файла (CSV заданного формата в **схеме 11**) и сохранение данных чека в указанный CSV-файл на вход добавлены параметры *pathToFile=xxxx* и *saveToFile=xxxx*

- *pathToFile* - включает относительный (от корневой директории проекта) путь + название файла с расширением (всегда присутствует в заданном формате)
- *saveToFile* - включает относительный (от корневой директории проекта) путь + название файла с расширением

**ВАЖНО!** если не передан аргумент - *pathToFile*- ошибку сохранить в **result.csv** (если передан *saveToFile*, тогда сохранить по пути из *saveToFile*)

**ВАЖНО!** если не передан аргумент - *saveToFile*- ошибку сохранить в **result.csv**

**ВАЖНО!** смотрите **схему 7** (примеры заполнения сообщений с ошибками)

Приложение должно запускаться с помощью консольной команды:

```
java -cp src ./src/main/java/ru/clevertec/check/CheckRunner.java id-quantity discountCard=xxxx  
balanceDebitCard=xxxx pathToFile=XXXX saveToFile=xxxx
```

Пример с ошибкой 1: должен создаваться файл **result.csv** в корне проекта с ошибкой **BAD REQUEST**:

```
java -cp src ./src/main/java/ru/clevertec/check/CheckRunner.java 1-1 discountCard=1111  
balanceDebitCard=100 pathToFile=./products.csv
```

Пример с ошибкой 2: должен создаваться файл **error\_result.csv** в корне проекта с ошибкой **BAD REQUEST**:

```
java -cp src ./src/main/java/ru/clevertec/check/CheckRunner.java 1-1 discountCard=1111  
balanceDebitCard=12.1 saveToFile=./error_result.csv
```

14. Исходники разместить в присланном репозитории в ветке **feature/entry-file**

“\*” - Необязательно, но будет существенным плюсом. К этому заданию лучше приступать после качественного решения базовой задачи с применением принципов SOLID, декларативных подходов, оптимальных алгоритмов.

## Тестовое задание 3\*

15.Стек: **Gradle 8.5**

16.В качестве DB использовать **PostgreSQL**

17.Разрешено использовать только JDBC (org.postgresql.Driver)

18.Замена хранения исходных данных (**схема 8, 9**) в файлах на хранение в таблицах PostgreSQL: product и discount\_card

19.Убран параметр *pathToFile*

20.Настройки подключения к БД передавать через аргументы командной строки на вход добавлены параметры: (*Обязательные*)

`datasource.url=xxx datasource.username=xxx datasource.password=xxx`

### Пример:

```
java -jar clevertec-check.jar 3-1 2-5 5-1 discountCard=1111 balanceDebitCard=100 saveToFile=./result.csv  
datasource.url=jdbc:postgresql://localhost:5432/check datasource.username=postgres datasource.password=postgres
```

21.DDL/DML операции должны храниться в файле src/main/resources/data.sql (но не использовать)

22.Покрыть функционал юнит-тестами (не менее 70 %)

23.Исходники разместить в присланном репозитории в ветке **feature/entry-database**

“\*” - Необязательно, но будет существенным плюсом. К этому заданию лучше приступать после качественного решения базовой задачи с применением принципов SOLID, декларативных подходов, оптимальных алгоритмов.

## Тестовое задание 4\*

24. Реализовать RESTFUL - API (Servlet). Данные вычитывать из БД (п.18).

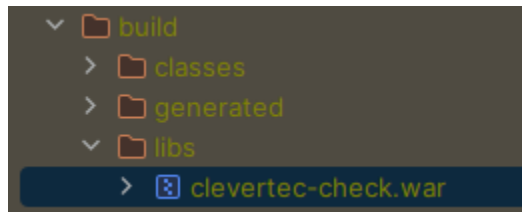
Реализовать добавление товаров и карт по REST (реализовать CRUD на эти две таблицы). При формировании чека товары в БД должны уменьшаться на запрошенное количество (если не было ошибки и хватило денег на счету). UI реализовывать не нужно.

25. Название war:

a. Добавляем в build.gradle

```
8 archivesBaseName = 'clevertec-check'
```

b. При gradle build должен получиться 1 .war архив:



26. **datasource.url** брать из системных переменных

27. **datasource.password** брать из системных переменных

28. **datasource.username** брать из системных переменных

Либо настраиваем системные переменные, либо томкат.

Конфигурация томката:

Открываем ...\\apache-tomcat-10.1.17\\conf\\catalina.properties и добавляем переменные в конец файла:

```
219 # Disable use of some privilege blocks Tomcat doesn't need since calls to the
220 # code in question are always already inside a privilege block
221 org.apache.el.GET_CLASSLOADER_USE_PRIVILEGED=false
222
223 datasource.url=jdbc:postgresql://localhost:5432/check
224 datasource.username=postgres
225 datasource.password=postgres
```

Как это выглядит в коде:

```
dataSource.setUrl(System.getProperty("datasource.url"));  
dataSource.setUser(System.getProperty("datasource.username"));  
dataSource.setPassword(System.getProperty("datasource.password"));
```

**!!!ВАЖНО: Проверьте правильность названия переменных!!!**

- 29. Покрыть функционал юнит-тестами (не менее 70 %)
- 30. Вместо файлов при ошибках клиента\сервера возвращать только статус коды без тела ответа см. **Схема 7**.
- 31. Исходники разместить в присланном репозитории в ветке **feature/entry-rest**

Примеры запросов:

- Получать чек (в ответе должен быть CSV-файл)

POST <http://localhost:8080/check>

```
{  
  "products": [  
    {  
      "id": 1,  
      "quantity": 5  
    },  
    {  
      "id": 1,  
      "quantity": 5  
    }  
  ],  
  "discountCard": 1234,  
  "balanceDebitCard": 100  
}
```

- Вернуть товар из БД  
GET <http://localhost:8080/products?id=1>
- Добавить товар в БД  
POST <http://localhost:8080/products>

```
{  
  "description": "Eat 100g.",  
  "price": 3.25,  
  "quantity": 5,  
  "isWholesale": true  
}
```

- Обновить товар в БД

PUT <http://localhost:8080/products?id=1>

```
{  
  "description": "Chocolate Ritter sport 100g.",  
  "price": 3.25,  
  "quantity": 5,  
  "isWholesale ": true  
}
```

- Удалить товар из БД

DELETE <http://localhost:8080/products?id=1>

- Вернуть дисконтную карту из БД

GET <http://localhost:8080/discountcards?id=1>

- Добавить дисконтную карту в БД

POST <http://localhost:8080/discountcards>

```
{  
  "discountCard": 5265,  
  "discountAmount": 2  
}
```

- Обновить дисконтную карту в БД по id.

PUT <http://localhost:8080/discountcards?id=1>

```
{  
  "discountCard": 6786,  
  "discountAmount": 3  
}
```



- Удалить дисконтную карту из БД

DELETE <http://localhost:8080/discountcards?id=1>

“\*” - Необязательно, но будет существенным плюсом. К этому заданию лучше приступать после качественного решения базовой задачи с применением принципов SOLID, декларативных подходов, оптимальных алгоритмов.

## Тестовое задание 5\*

32.Расширить функционал на свое усмотрение. Исходники разместить в присланном репозитории в ветке `feature/custom`. Обязательно указать то, что сделали в файле README.md

## Описание CSV-файла:

**Date** - Дата генерации файла

**Time** - Время генерации файла

**QTY** - Количество заказанного товара

**DESCRIPTION** - Название заказанного товара

**PRICE** - Цена за 1 шт.

**TOTAL** - Итоговая цена без скидки ( $QTY * PRICE$ )

**DISCOUNT** - Скидка на итоговую цену товара ( $TOTAL * DISCOUNT PERCENTAGE / 100$ )

**DISCOUNT CARD** - Номер скидочной карты

**DISCOUNT PERCENTAGE** - Процент скидки по карте

**TOTAL PRICE** - Итоговая цена за все товары ( $TOTAL 1 + TOTAL 2 + \dots$ )

**TOTAL DISCOUNT** - Итоговая скидка на все товары ( $DISCOUNT 1 + DISCOUNT 2 + \dots$ )

**TOTAL WITH DISCOUNT** - Итоговая стоимость ( $TOTAL PRICE - TOTAL DISCOUNT$ )

Смотри схемы 1, 2, 3, 4

**ВАЖНО!** Значения разделяются: “;”

**ВАЖНО!** Значения с плавающей точкой пишутся через **точку**

**ВАЖНО!** Все значения имеющие \$ в конце пишутся до 2 знаков после запятой

Пример: 1.21\$; 0.00\$; 142199.09\$;

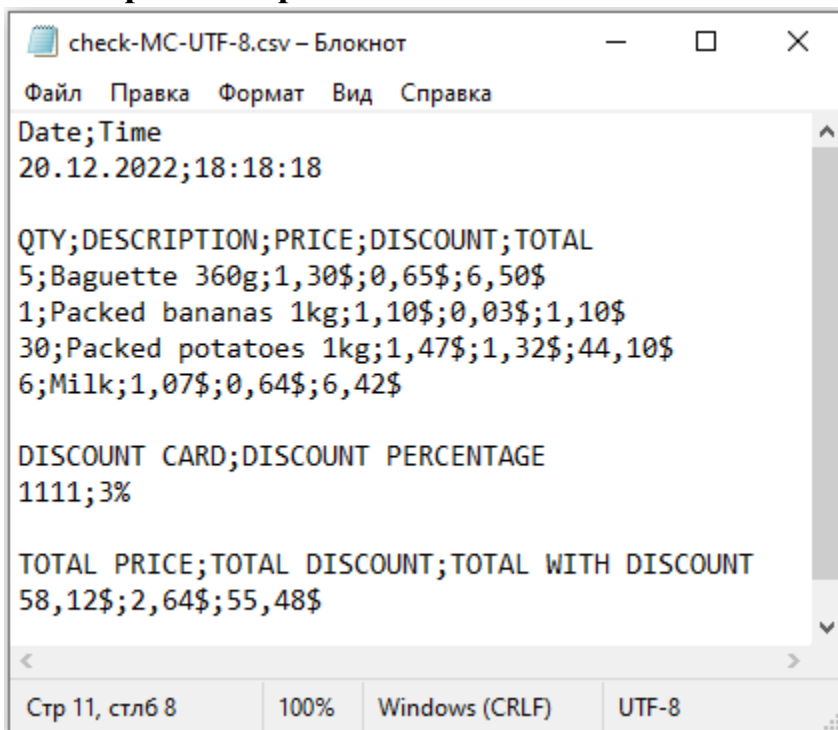
## Описание CSV-файла с исключением:

**ERROR** - при возникновении исключения. С новой строки заполнить сообщение.

Смотри схемы 5, 6, 7

## Примеры CSV-файла с дисконтной картой:

### Схема 1. Открытие через блокнот:



```

check-MC-UTF-8.csv – Блокнот
Файл  Правка  Формат  Вид  Справка
Date;Time
20.12.2022;18:18:18

QTY;DESCRIPTION;PRICE;DISCOUNT;TOTAL
5;Baguette 360g;1,30$;0,65$;6,50$
1;Packed bananas 1kg;1,10$;0,03$;1,10$
30;Packed potatoes 1kg;1,47$;1,32$;44,10$
6;Milk;1,07$;0,64$;6,42$

DISCOUNT CARD;DISCOUNT PERCENTAGE
1111;3%

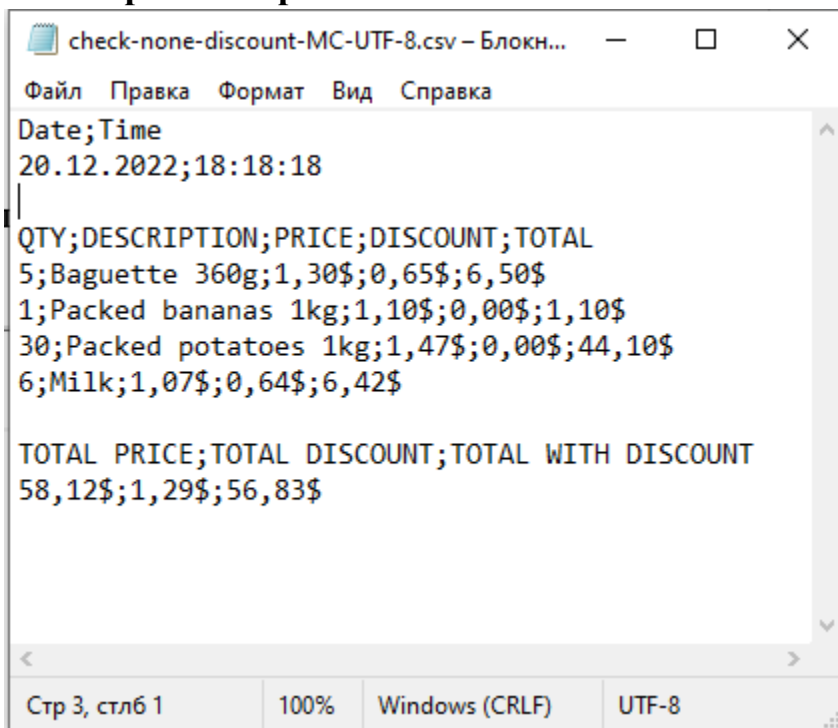
TOTAL PRICE;TOTAL DISCOUNT;TOTAL WITH DISCOUNT
58,12$;2,64$;55,48$
  
```

### Схема 2. Открытие через EXCEL:

	A	B	C	D	E
1	Date	Time			
2	20.12.2022	18:18:18			
3					
4	QTY	DESCRIPTION	PRICE	DISCOUNT	TOTAL
5	5	Baguette 360g	1,30\$	0,65\$	6,50\$
6	1	Packed bananas 1kg	1,10\$	0,03\$	1,10\$
7	30	Packed potatoes 1kg	1,47\$	1,32\$	44,10\$
8	6	Milk	1,07\$	0,64\$	6,42\$
9					
10	DISCOUNT CARD	DISCOUNT PERCENTAGE			
11	1111	3%			
12					
13	TOTAL PRICE	TOTAL DISCOUNT	TOTAL WITH DISCOUNT		
14	58,12\$	2,64\$	55,48\$		

## Примеры CSV-файла без дисконтной карты:

### Схема 3. Открытие через блокнот:



```

check-none-discount-MC-UTF-8.csv - Блокнот...
Файл  Правка  Формат  Вид  Справка
Date;Time
20.12.2022;18:18:18

QTY;DESCRIPTION;PRICE;DISCOUNT;TOTAL
5;Baguette 360g;1,30$;0,65$;6,50$
1;Packed bananas 1kg;1,10$;0,00$;1,10$
30;Packed potatoes 1kg;1,47$;0,00$;44,10$
6;Milk;1,07$;0,64$;6,42$

TOTAL PRICE;TOTAL DISCOUNT;TOTAL WITH DISCOUNT
58,12$;1,29$;56,83$

Стр 3, стлб 1    100%    Windows (CRLF)    UTF-8

```

### Схема 4. Открытие через EXCEL:

	A	B	C	D	E
1	Date	Time			
2	20.12.2022	18:18:18			
3					
4	QTY	DESCRIPTION	PRICE	DISCOUNT	TOTAL
5	5	Baguette 360g	1,30\$	0,65\$	6,50\$
6	1	Packed bananas 1kg	1,10\$	0,00\$	1,10\$
7	30	Packed potatoes 1kg	1,47\$	0,00\$	44,10\$
8	6	Milk	1,07\$	0,64\$	6,42\$
9					
10	TOTAL PRICE	TOTAL DISCOUNT	TOTAL WITH DISCOUNT		
11	58,12\$	1,29\$	56,83\$		

Примеры CSV-файла с исключением:

Схема 5. Открытие через блокнот:

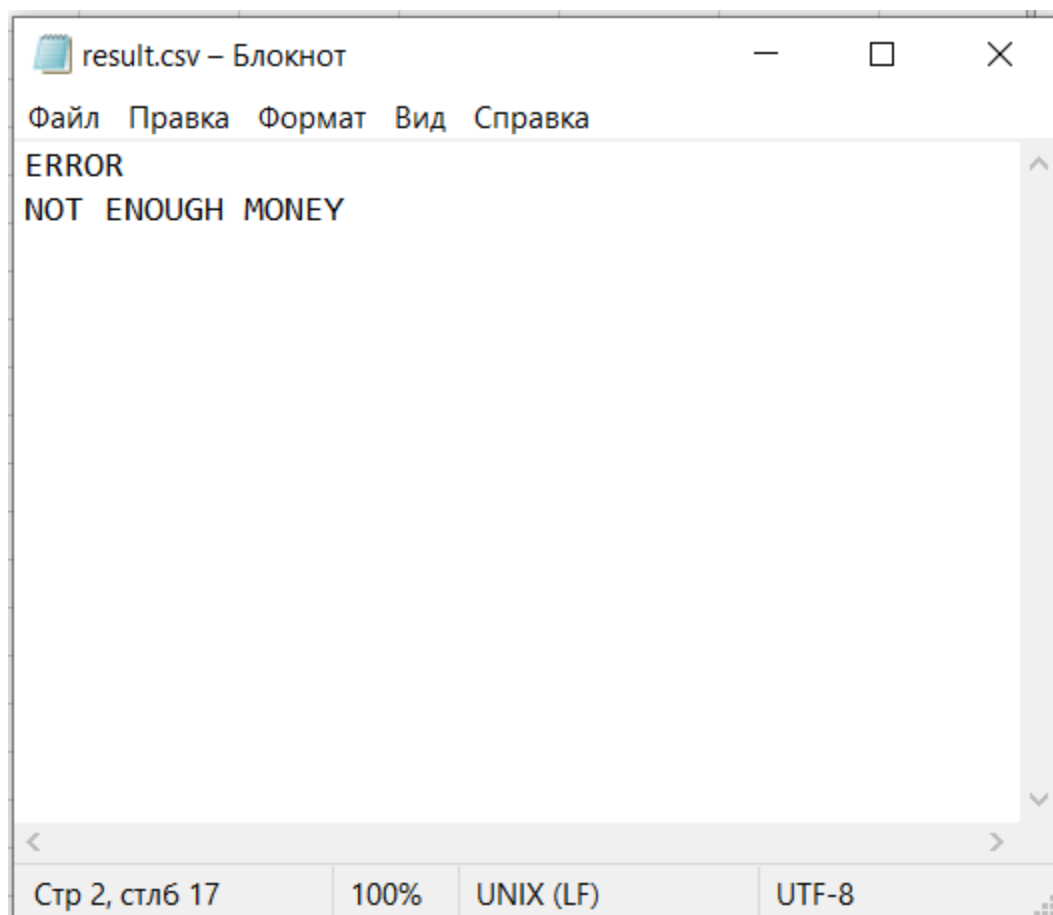


Схема 6. Открытие через EXCEL:

	A
1	ERROR
2	BAD REQUEST

**ВАЖНО!** Смотрите схему 7 заполнения сообщений.

**Схема 7. Примеры заполнения сообщений при возникновении исключительных ситуаций:**

Сообщение	Описание
<b>BAD REQUEST</b>	При неверных входных данных (не верно заполнены аргументы, ошибки количества, отсутствия products)
<b>NOT ENOUGH MONEY</b>	При недостатке средств (баланс меньше, чем сумма в чеке)
<b>INTERNAL SERVER ERROR</b>	При любых других ситуациях
Для задания 2	
<b>BAD REQUEST</b>	Если не переданы аргументы: <i>pathToFile</i> и/или <i>saveToFile</i>
Для задания 4	
Статус	Описание
<b>400</b>	При неверных входных данных (неверно заполнено body запроса, ошибки количества products, их наличия)
<b>400</b>	При недостатке средств (баланс меньше, чем сумма в чеке)
<b>404</b>	Продукт не найден в базе данных
<b>500</b>	При любых других ситуациях

**Схема 8. Таблица в DB **public.product** должна выглядеть следующим образом:**

<b>column:</b>	<b>id</b>	<b>description</b>	<b>price</b>	<b>quantity_in_stock</b>	<b>wholesale_product</b>
<b>type:</b>	<b>BIGSERIAL</b>	<b>VARCHAR(50)</b>	<b>DECIMAL</b>	<b>INTEGER</b>	<b>BOOLEAN</b>

Схема 9. Таблица в DB **public.discount\_card** должна выглядеть следующим образом:

column:	id	number	amount
type:	BIGSERIAL	INTEGER	SMALLINT

**ВАЖНО!** Ограничения выставить самостоятельно

**ВАЖНО!** **price** - должна храниться до 2 знаков, сокращать по правилам математики (например: 1,20)

**ВАЖНО!**  $0 \leq \text{discount amount} \leq 100$



**Схема 10.**
**Таблица 1. Список доступных products**

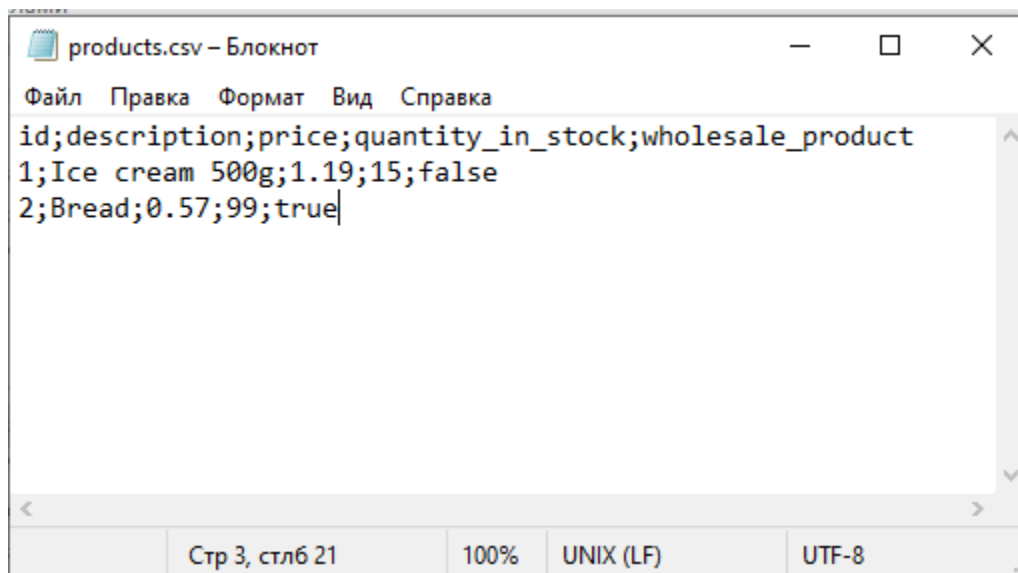
<b>id</b>	<b>description</b>	<b>price, \$</b>	<b>quantity in stock</b>	<b>wholesale product</b>
1	Milk	1,07	10	+
2	Cream 400g	2,71	20	+
3	Yogurt 400g	2,10	7	+
4	Packed potatoes 1kg	1,47	30	-
5	Packed cabbage 1kg	1,19	15	-
6	Packed tomatoes 350g	1,60	50	-
7	Packed apples 1kg	2,78	18	-
8	Packed oranges 1kg	3,20	12	-
9	Packed bananas 1kg	1,10	25	+
10	Packed beef fillet 1kg	12,8	7	-
11	Packed pork fillet 1kg	8,52	14	-
12	Packed chicken breasts 1kg Sour	10,75	18	-
13	Baguette 360g	1,30	10	+
14	Drinking water 1,5l	0,80	100	-
15	Olive oil 500ml	5,30	16	-
16	Sunflower oil 1l	1,20	12	-
17	Chocolate Ritter sport 100g	1,10	50	+
18	Paulaner 0,5l	1,10	100	-
19	Whiskey Jim Beam 1l	13,99	30	-
20	Whiskey Jack Daniels 1l	17,19	20	-

**Таблица 2. Список доступных discount cards**

id	number	discount amount, %
1	1111	3
2	2222	3
3	3333	4
4	4444	5
Любая <a href="#">другая дисконтная карта</a> с вашим номером предоставляет 2% скидки.		

**Схема 11. Пример передаваемого файла с продуктами:**

**1) Открытие файла в блокноте**



```

products.csv – Блокнот
Файл  Правка  Формат  Вид  Справка
id;description;price;quantity_in_stock;wholesale_product
1;Ice cream 500g;1.19;15;false
2;Bread;0.57;99;true
  
```

Стр 3, столб 21    100%    UNIX (LF)    UTF-8

**2) Открытие файла в Excel:**

	A	B	C	D	E
1	id	description	price	quantity_in_stock	wholesale_product
2	1	Ice cream 500g	1.19	15	false
3	2	Bread	0.57	99	true