**Команда getProduct\_Categories**

Сервер має повернути таблицю **Product\_Categories** яка описана в схемі бази даних (для Inventory)

**Команда getProducts**

Сервер має повернути таблицю **Product** яка описана в схемі бази даних (для Inventory), але не в чистому, а в трансформованому вигляді.

В файлі fakeProducts.txt - формат даних який ми запрограмували.

Поля, що відсутні в схемі, але необхідні нам: nameTranslations, descriptionTranslations та availableQty

--

**Команда getCategories**

Сервер має повернути таблицю **Categories** яка описана в схемі бази даних (для Inventory), але не в чистому, а в трансформованому вигляді.

Див. файл fakeCategories.js - формат даних який ми запрограмували.

Поля, що відсутні в схемі, але необхідні нам: nameTranslations

**getLanguageCodes**

Сервер має повернути на які мови є переклад описів продуктів. Див файл fakeLanguages.js

**Команда changeProductQuantity**

Щоразу коли користувач додає якийсь продукт до кошика або видаляє його наш додаток буде на ваш сервер подавати цю команду, де буде вказувати ID продукта та

Плюсове число (якщо якусь кількість одиниць продукту додали до кошика) або негативне (якщо видалили з кошика). Вам треба буде в базі тимчасово блокувати (або розблокувати) необхідну кількість одиниць продукту. Це необхідно, щоб інший користувач з іншого кіоску який паралельно здійснює покупку не замовив мясо, яке вже інший користувач добавив до кошика (бо необхідної кількості мяса може не вистачити). Задача тут мабуть схожа на купівлю залізничних квитків через інтернет (не допустити купівлю двох користувачів того ж самого місця у вагоні).

Сервер після обробки команди має повернути скільки одиниць продукту на даний момент доступно для продажу (тимчасово заблоковані продукти вважаються не доступними).

**Команда getProductQuantity**

На вхід – ID продукту

На вихід - Сервер має повернути скільки одиниць продукту на даний доступно для продажу.

**Команда getProductsQuantityForCategory**

На вхід – категорія продуктів (beef чи pork…)

На вихід - Сервер має повернути скільки одиниць кожного продукту вказаної категорії на даний доступно для продажу.

**sendMessageReceipt**

Відправка клієнту на емейл чеку.

**sendEmailProductAvailable**

Планування відправки клієнту на емейл чи смс повідомлення про те що продукт, якого не було на момент покупки вже є в наявності.

**sendEmailCookingTips**

Відправка клієнту на емейл рецептів приготування продуктів.

**getDisscountValue**

На вхід ми передаєм або код купона на знижку або про сканований штрих-код картки на знижку, а на вихід бекенд має видати скільки процентів (чи доларів) знижка має бути на цей купон або картку.

Ще будуть команди які будуть передаватись не через HTTP, а через WebSockets (команди для діагностики, команда видати мясо і т.д.) які детальніше описані в документі

<https://docs.google.com/document/d/1sQ_R8ztSqtzyVgwRSqOJNR1ij6GfiUP-R-hwCKHnZj0/edit?ts=5c59b6f3>

Також окрема задача – це аутентифікація нашого фронтенд-додатку на вашому сервері. Для мене ця задача на даний момент взагалі не зрозуміла. Можливо у вас є погляди щодо цієї задачі.