**Задание: Разработка простого онлайн-магазина электроники**

**Цель**

Создать простой онлайн-магазин электроники с клиент-серверной архитектурой. Система должна включать сервер, обрабатывающий HTTP-запросы, базу данных для хранения информации и клиентское приложение с графическим интерфейсом на Qt с использованием C++. Приложение поддерживает два уровня доступа: администратор и пользователь, с разным функционалом для просмотра, покупки и управления товарами.

**Требования к проекту**

**Общая архитектура**

* **Сервер**:Реализовать HTTP-сервер с использованием Qt C++ для обработки запросов от клиента.
* **База данных**: использовать реляционную базу данных PostgreSQL для хранения информации о категориях, товарах, пользователях и заказах.
* **Клиент**: Разработать графическое приложение на Qt с C++ для взаимодействия с сервером, отображения данных и управления функционалом.

**Функционал**

**Аутентификация**

* При запуске приложения пользователь вводит логин и пароль.
* Система проверяет данные через сервер и определяет роль: администратор или обычный пользователь.
* Если логин/пароль неверные, отображается сообщение об ошибке.

**Роль: Пользователь**

* **Просмотр категорий**: Пользователь видит список категорий товаров (например, "Смартфоны", "Ноутбуки").
* **Просмотр товаров**: В каждой категории отображаются товары с названием, ценой, картинкой и описанием.
* **Корзина**:
  + Пользователь может добавить товар в корзину.
  + Пользователь может удалить товар из корзины.
  + Отображается список товаров в корзине с общей суммой.
* **Покупка**:
  + Кнопка "Заказать" или "Купить" очищает корзину и выводит сообщение "Товар куплен".
  + При покупке товар удаляется из магазина (каждый товар в единственном экземпляре).

**Роль: Администратор**

* **Управление категориями**:
  + Добавление новых категорий (указывается название).
  + Просмотр списка существующих категорий.
* **Управление товарами**:
  + Добавление товара: указывается название, цена, описание, категория (или несколько категорий), путь к картинке.
  + Привязка товара к одной или нескольким категориям.
  + Удаление товаров (опционально, если требуется).
* Все изменения сохраняются в базе данных.

**Технические требования**

* **Сервер**:
  + Реализовать REST API для обработки запросов (GET для получения данных, POST для добавления/обновления, DELETE для удаления).
  + Примеры эндпоинтов:
    - GET /categories — получить список категорий.
    - GET /products?category\_id=<id> — получить товары в категории.
    - POST /products — добавить товар (для админа).
    - POST /cart — добавить товар в корзину.
    - POST /order — оформить заказ.
  + Без авторизации эндпоинтов
* **База данных**:
  + Таблицы:
    - users (id, login, password\_hash, role).
    - categories (id, name).
    - products (id, name, price, description, image\_path).
    - product\_categories (product\_id, category\_id) — для связи товаров и категорий.
    - cart (user\_id, product\_id) — для корзины.
  + Использовать SQL-запросы для работы с данными.
* **Клиент (Qt C++)**:
  + Создать интерфейс с использованием Qt Widgets.
  + Реализовать окна/экраны:
    - Окно входа (логин, пароль).
    - Главное окно: список категорий, список товаров, корзина, кнопка покупки (для пользователя).
    - Окно админа: формы для добавления категорий и товаров.
  + Для отображения картинок использовать QImage или аналогичные классы Qt.
  + Отправлять HTTP-запросы к серверу (например, с помощью QNetworkAccessManager).

**Упрощения**

* Каждый товар существует в единственном экземпляре (не нужно учитывать количество).
* Не требуется сложная система оплаты — покупка просто удаляет товары из магазина и корзины.
* Пароли можно хранить в хэшированном виде (например, с использованием SHA-256) или в упрощённом виде (для учебных целей).
* Картинки товаров можно хранить как пути к файлам на сервере или локально (для простоты).

**Рекомендации**

* Используйте базу данных PostgreSQL
* Для HTTP-запросов на клиенте применяйте QNetworkAccessManager из Qt.
* Для парсинга JSON-ответов от сервера используйте QJsonDocument.
* Структурируйте код: разделите клиентскую и серверную логику, используйте модули/классы.
* Тестируйте функционал постепенно: начните с авторизации, затем добавьте просмотр категорий/товаров, корзину и админ-функции.

**Результат**

* Рабочий прототип магазина с клиентским приложением на Qt (C++) и сервером.