Курсовой проект. Создание интернет магазина в Telegram Bot

Группа: 30ПР31
Студент: Пойденко А. А.

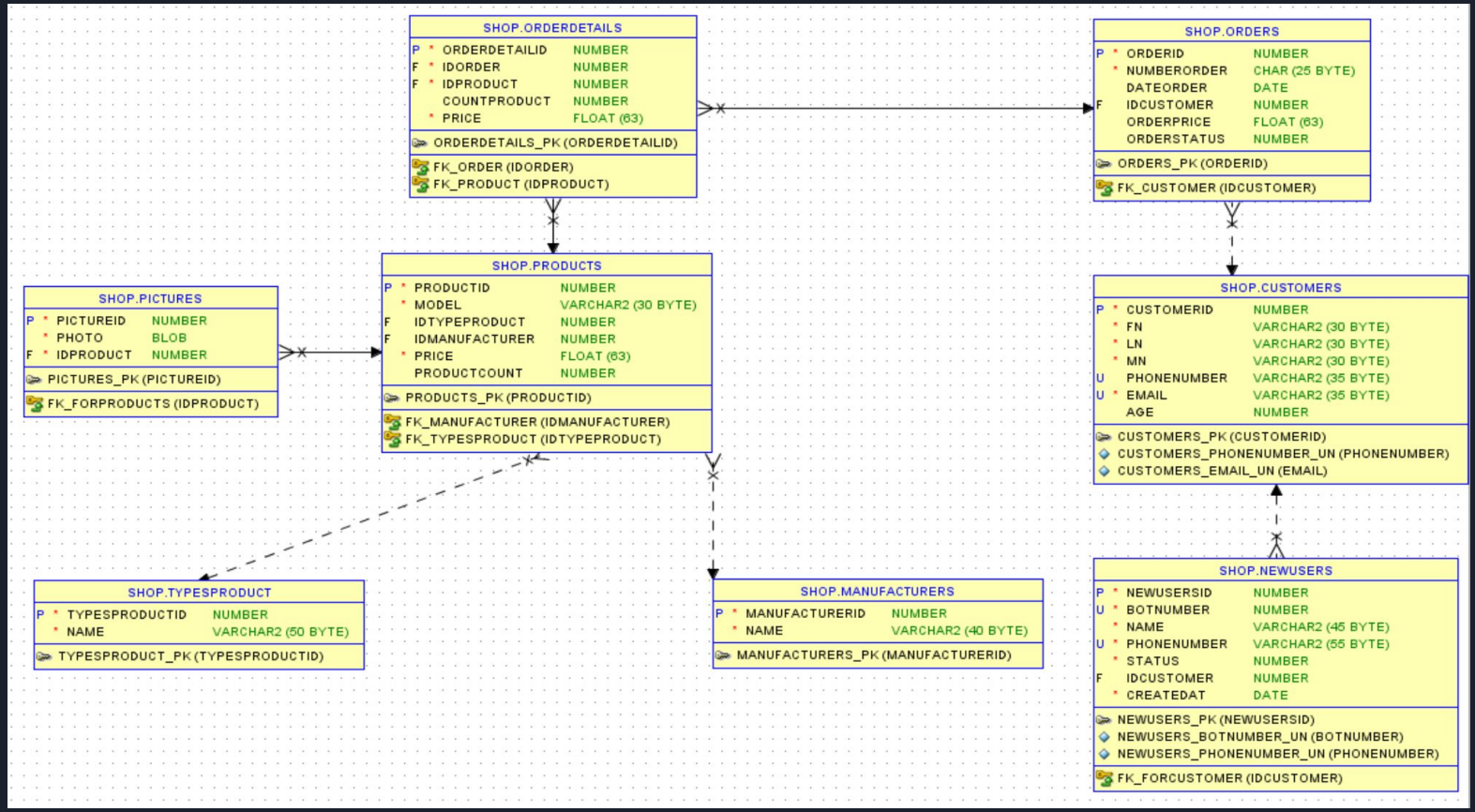
Применяемые технологии:

- JavaEE (EnterpriseEdition)
- JDBC
- Oracle

Применяемые framework и библиотеки:

- Hibernate
- JavaMail API
- Telegram Bots

Схема Базы Данных:



Описание БД

ТАБЛИЦЫ:

- CUSTOMERS - информация о заказчике. Эта таблица связана с таблицей NEWUSERS (один ко многим)
- NEWUSERS - информация о пользователях Телеграм Бота. При обращении нового пользователя к боту (на этапе передачи своего контакта) идёт сопоставление с информацией в таблице CUSTOMERS по номеру мобильного телефона

ТАБЛИЦЫ:

- MANUFACTURERS - информация о производителях. Эта таблица связана с таблицей PRODUCTS (один ко многим)
- TYPESPRODUCT информация о категориях товара. Эта таблица связана с таблицей PRODUCTS (один ко многим)
- PICTURES - фото товара. Эта таблица связана с таблицей PRODUCTS (многие к одному)

Описание БД

ТАБЛИЦЫ:

- PRODUCTS - информация о товаре. Также в этой таблице в поле PRODUCTCOUNT указано количество доступного в данный момент товара
- ORDERS - информация о заказах. Эта таблица связана с таблицей ORDERDETAILS (один ко многим).

  В поле ORDERSTATUS указан статус заказа. Заказ может быть открытым (значение поля 0) или закрытым

ТАБЛИЦЫ:

(значение поля 1). Если заказ открыт, то все позиции, которые заказывает пользователь относятся к данному открытому заказу.

В поле ORDERPRICE указана стоимость всех позиций по данному заказу

- ORDERDETAILS - информация о заказанном товаре по конкретному заказу.

Описание БД

ФУНКЦИИ:

- CALCORDERPRICE - функция, которая считает общую стоимость всех позиций по заказу (таблица ORDERS) в таблице позиций в заказе (ORDERDETAILS). Расчёт происходит в цикле путем перебора всех позиций по конкретному заказу и умножения количества заказанной позиции на ее стоимость. Входящий параметр функции - Id заказа. Функция возвращает итоговую стоимость по заказу.

ТРИГГЕРЫ:

- ORDERDETAILS_TRIGGER - триггер, срабатывающий при добавлении данных в таблицу ORDERDETAILS (заказ товара пользователем).

Действия в триггере:

1. Корректируется остаток по товару, который добавляется в таблицу ORDERDETAILS. Добавляемое количество по позиции отнимается от значения в поле PRODUCTCOUNT

Описание БД

ТРИГГЕРЫ:

таблицы PRODUCTS по данной позиции.

2. Корректируется итоговая стоимость по заказу (таблица ORDERS, поле ORDERPRICE). Стоимость увеличивается на общую стоимость товара по позиции в таблице ORDERDETAILS

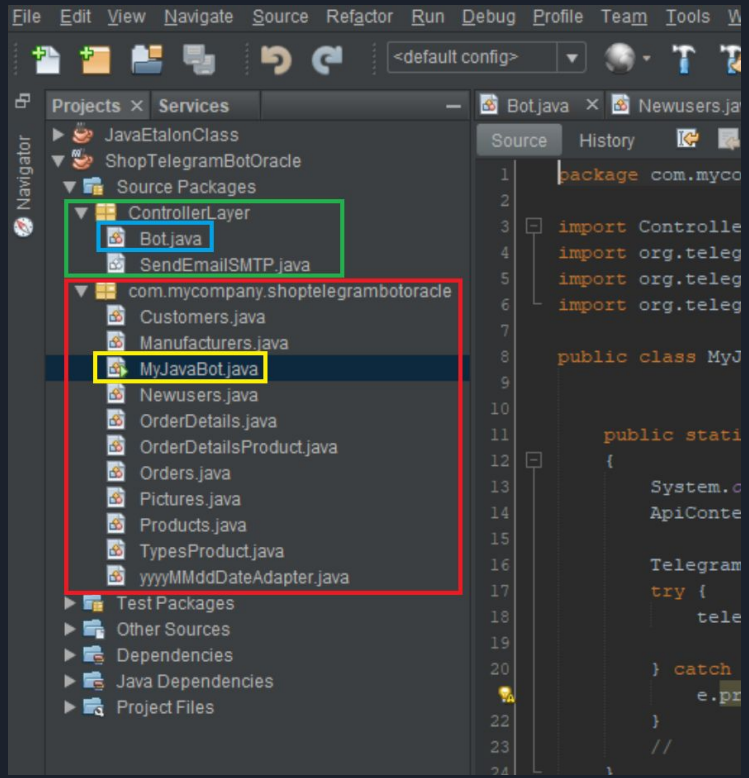
- OrderDetails_DELETE_TRIGGER - триггер, срабатывающий при удалении данных из таблицы ORDERDETAILS (удаление товара из корзины).

Действия в триггере:

1. Корректируется остаток по товару, который удаляется из таблицы ORDERDETAILS. Удаляемое количество по позиции прибавляется к значению в поле PRODUCTCOUNT

2. Корректируется итоговая стоимость по заказу (таблица ORDERS, поле ORDERPRICE). Стоимость уменьшается на общую стоимость товара по позиции в таблице ORDERDETAILS

Пример программного кода:



- В package com.mycompany.shoptele - grambotoracle находятся классы, описывающие таблицы Базы Данных. Они необходимы для работы framework hibernate (выделено красным). Также в этом пакете находится основной класс, из которого запускается программа MyJavaBot (выделено желтым).

- В package ControllerLayer находится класс Bot и класс SendEmailSMTP. Класс Bot - это класс, в котором описаны все методы взаимодействия с пользователем через интерфейс программы Telegram.
Класс SendEmailSMTP - класс, в котором описан функционал отправки сообщений по электронной почте

Пример программного кода:

Класс Bot (часть основного метода класса onUpdateReceived. В данном методе описаны действия программы при вызове пользователем команд в боте)

```
@Override
public void onUpdateReceived(Update update) {
    if (update.hasMessage() || update.hasCallbackQuery()) {
        long chat_id;

        if (update.hasMessage()) {
            chat_id = update.getMessage().getChatId();
        } else {
            CallbackQuery callbackQuery = update.getCallbackQuery();
            chat_id = callbackQuery.getMessage().getChatId();
        }

        try {
            List<Newusers> newUsers = getNewusers();

            if (newUsers.stream().anyMatch(x -> x.getBotNumber() == chat_id)) {
                Newusers newusers = newUsers.stream().filter(x -> x.getBotNumber() == chat_id).findFirst().get();

                String message_text = null;

                if (newusers.getStatus() < 5) {
                    message_text = update.getMessage().getText();
                } else {
                    message_text = update.getCallbackQuery().getData();
                }
                switch (newusers.getStatus()) {
                    case 1: {
                        if (isFullname(message_text)) {
                            newusers.setName(message_text);
                            newusers.setStatus(2);
                            editNewUser(newusers);
                            sendContact(Long.toString(chat_id), "Теперь нажмите на появившуюся кнопку, что бы передать боту
                        } else {
                            sendMsg(Long.toString(chat_id), "Имя введено не верно");
                        }
                    }break;
                    case 2: {
                        if (update.getMessage().getContact() != null) {
```

Демо приложения:

- Работа самого приложения с точки зрения пользователя показана в видео TelegramShopVideo.avi

- В презентации Слайды работы программы.pdf показаны слайды работы программы с точки зрения пользователя с кратким описание событий.

Спасибо за внимание!