Ссылка на статью:
// Ученые записки УлГУ. Сер. Математика и информационные технологии. УлГУ. Электрон. журн. 2020, № 1, с. 109-117.

Поступила: 27.05.2020
Окончательный вариант: 08.06.2020

УДК 004.94

Разработка мобильного приложения для администрирования интернет-магазина на базе CS-Cart

Самойленко А.А.*[*], *Волков М.А.

[*]andryxa73-samojlenko@yandex.ru
УлГУ, Ульяновск, Россия

В работе представлено описание программного продукта, которое дает возможность администрировать любой интернет-магазин на базе CS-Cart. Мобильное приложение для ОС Android написано на языке Kotlin. Данное приложение позволяет подключить свой интернет-магазин, используя нативный интерфейс API CMS CS-Cart. Используя приложение, можно провести основные действия администратора интернет-магазина: работа с заказами, работа с товарами и работа с пользователями.

Одним из важных преимуществ данного приложение является подключение разных изданий CMS CS-Cart (CS-Cart B2B, CS-Cart B2C, Multi-Vendor), при этом нет необходимости создавать свое приложение для конкретного магазина. Мобильное приложение, написанное на нативном языке для ОС Android, работает в разы быстрее, чем мобильная версия сайта. Разработанный программный продукт может использовать администратор с любой привилегией: менеджер заказов сможет работать только с заказами, а менеджер товаров – только с товарами.

***Ключевые слова**: Мобильное приложение, интернет-магазин, CMS, CS-Cart, Multi-Vendor, Android, Kotlin*

Введение

Сегодня все большая часть бизнеса, связанная с торговлей и предоставлением услуг, переходит в Интернет или подключает поддержку формата онлайн. Такая популярность обуславливается тем, что данный формат услуги выбирают большинство покупателей: приобретение товара через Интернет – это удобнее и безопаснее, к примеру, в период эпидемии.

Владелец интернет-магазина должен в любой момент времени произвести необходимые операции в своем магазине, что не всегда возможно. Решением этой проблемы явля-

ется наличие мобильного приложения, позволяющего выполнить основные задачи интернет-магазина – обработку заказа. Чем быстрее покупатель получит свой товар, тем лучше будет его впечатление от использования данного магазина. Если владелец или администратор магазина быстро обработает заказ, то покупатель может еще раз сделать покупку или посоветовать ресурс своим друзьям или знакомым.

Основная задача заключается в разработке мобильного приложения для администрирования интернет-магазина на базе CS-Cart, при этом код приложения должен быть легко поддерживаемым и простым в понимании.

У мобильного приложения есть дополнительные преимущества перед использованием сайта:

- скорость работы мобильного приложения превышает скорость работы сайта;
- в мобильном приложении есть возможность использования Push-уведомлений;
- удобство использования – мобильное устройство всегда под рукой, в отличие от компьютера;
- малая нагрузка на интернет-трафик – мобильное приложение требует меньше трафика, чем сайт.

В связи с этим разработано мобильное приложение для ОС Android. Приложение дает возможность произвести основные действия администратора интернет-магазина и уменьшить время обработки заказов. Главным преимуществом данного приложения является отсутствие аналогов.

1. Функционал программного продукта

В качестве основы для создания программного продукта выбрано мобильное приложение на базе ОС Android. Приложение использует нативный интерфейс API CMS CS-Cart. Основными преимуществами такого подхода являются:

- Отсутствие дополнительных модулей – зачастую владельцы интернет-магазина предпочитают не устанавливать дополнительные модули, так как боятся утечки конфиденциальной информации и не любят постоянно производить лишние действия с дополнительными модулями (обновлениями).
- Скорость – мобильное приложение быстрее загружает информацию, используя API, так как нет необходимости каждый раз прорисовывать визуальное оформление сайта. Мобильное приложение использует интерфейс, который уже хранится на мобильном устройстве [2].

Приложение разработано на нативном языке ОС Android – Kotlin. Данный язык работает поверх Java Virtual Machine, вследствие чего во время компиляции он конвертируется в язык Java [3].

Архитектура мобильного приложения спроектирована с использованием связки двух архитектурных паттернов MVC и MVP [5]. Такая связка обусловлена легкостью чтения и понимания кода в таком паттерне. Также данная связка не требует изучение сложных и тяжелых библиотек (RxJava в паттерне MVI, MVVM) [5,6].

Визуальная структура мобильного приложения выполнена в стиле Material Design [1] с сохранением цветовой палитры основной панели администратора, которая используется на сайте.

На данный момент приложение имеет исключительно основные элементы панели администратора:

- страница со статистикой;
- страница со списком заказов;
- страница со списком товаров;
- страница со списком пользователей.

При открытии приложения пользователь попадает на страницу входа (см. рис. 1). Войти в приложение можно двумя способами: используя email и пароль (API key) или сканировав QR-код. Сканируя QR-код, приложение получает всю необходимую информацию для входа.

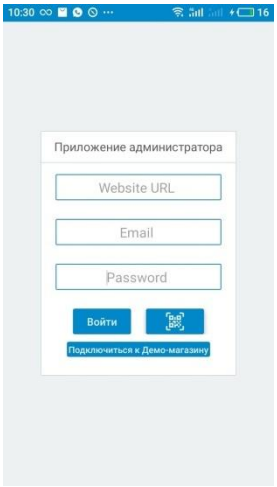

Рис. 1. Страница входа и страница статистики.

Также на странице входа есть кнопка для подключения к Демо-магазину – это дает возможность любому пользователю ознакомиться с функционалом приложения до подключения «живого» магазина.

После успешного входа приложение начинает отображать нижнее меню с основными страницами панели администратора: статистика, заказы, товары, пользователи и кнопка бокового меню.

На странице статистики (см. рис. 2) отображается основная статистика магазина, которая может быть полезна для администратора магазина:

- количество новых (открытых) заказов;
- общее количество заказов;
- количество товаров, отображаемых на витрине (на стороне клиента);
- количество товаров, которые закончились (отсутствуют в наличии);
- количество категорий в магазине;

- количество зарегистрированных покупателей.

Рис.2. Страница статистики интернет-магазина

Страница заказов отображает все заказы, которые есть в магазине. В верхней части экрана есть строка поиска. Она позволяет найти заказы по имени покупателя или по номеру заказа.

Если нажать на заказ, то откроется детальная страница заказа, которая отображает всю информацию о заказе: стоимость заказа, налоги за заказ, список купленных товаров, информацию о способе оплаты и информацию о покупателе.

Также на детальной странице заказа можно сменить статус заказа при необходимости. Пример страницы заказов можно увидеть на рис. 3.

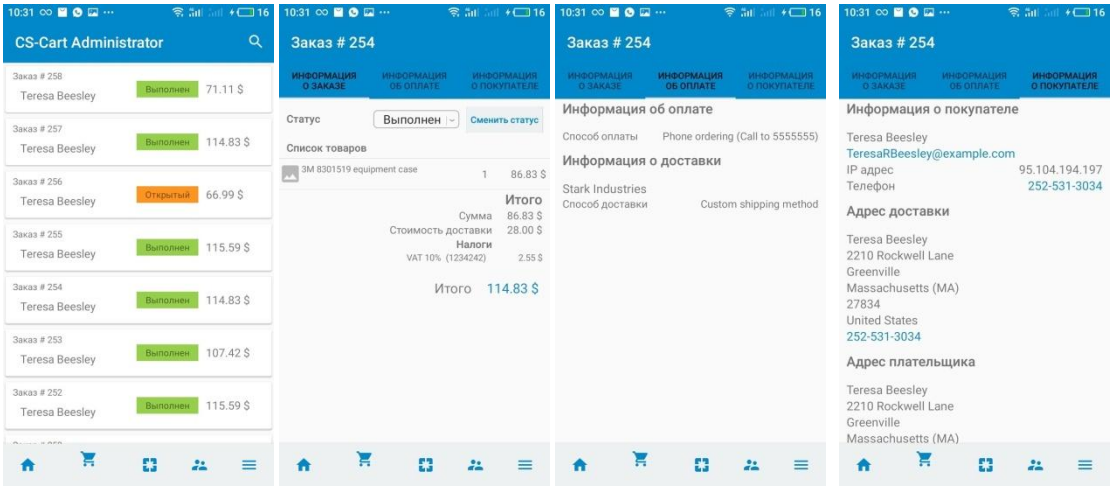

Рис. 3. Страница заказов и детальная страница заказа

Страница списка товаров отображает все товары, которые есть в магазине. В верхней части также есть строка поиска, благодаря которой можно найти необходимый товар по названию.

Нажав на необходимый товар, можно открыть детальную информацию о товаре (см. рис. 4), где также можно отредактировать основные параметры товара (код товара – артикул, название товара, цену товара, количество товаров в наличии) и сменить статус товара.

В верхней части детальной страницы товара есть кнопка сохранения. Нажав на нее, вся информация через API будет отправлена на сервер, и измененные параметры товара будут сохранены.

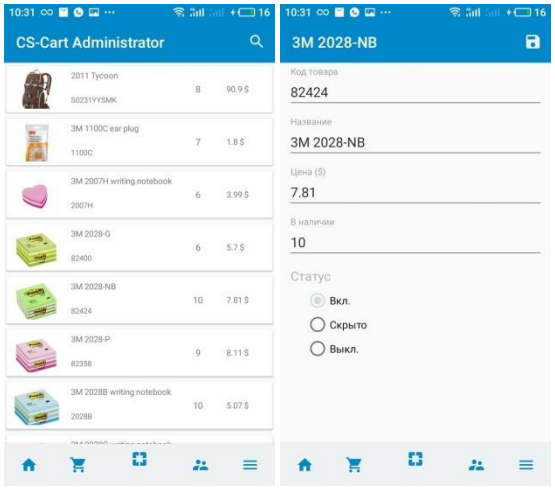

Рис. 4. Страница списка товаров и детальная страница товара

На странице пользователей есть два списка пользователей (или три в случае, если магазин является изданием Multi-Vendor): покупатели, администраторы и продавцы.

В верхней части страницы есть строка поиска, через которую можно найти пользователя по имени/фамилии.

Если кликнуть на пользователя, то откроется детальная страница (см. рис. 5), где можно увидеть и изменить основную информацию о пользователе (имя, фамилию, email, телефон и статус). Вся информация сохраняется при нажатии кнопки сохранения в верхней части страницы.

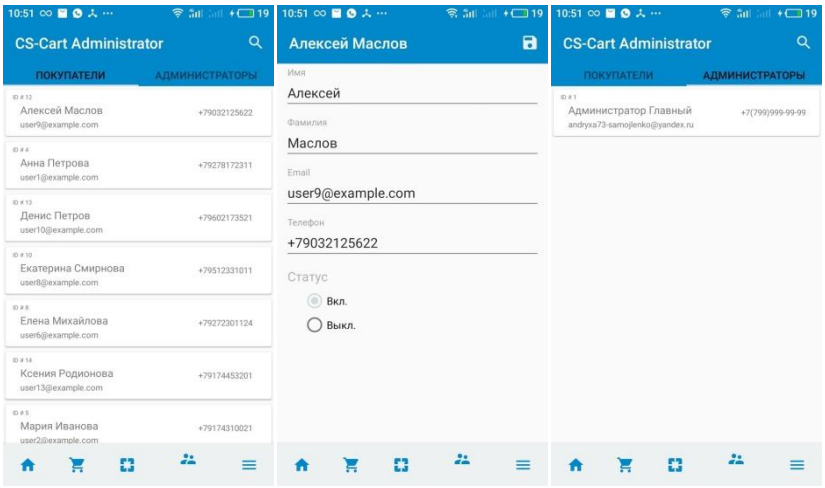

Рис. 5. Страница списка пользователей и детальная страница пользователя

На боковом меню (см. рис. 6) располагается две кнопки: Отключиться (выход из панели администратора магазина) и Выход (выход из приложения). В дальнейшем здесь будут располагаться дополнительные страницы: категории, фильтры, характеристики, опции, способы оплаты, способы доставки и др.

Рис. 6. Боковое меню.

2. Используемые технологии

Разработка программного продукта выполнена в программной среде Android Studio, которая имеет удобный интерфейс, легкую навигацию по коду и подсказки вызываемых функций и параметров при написании кода. Есть возможность быстро собрать и скомпилировать приложение.

Данная программа доступна на любой компьютерной ОС (macOS, Windows, Linux).

Android Studio разработана компанией Google, основана на программном обеспечении IntelliJ IDEA от компании JetBrains.

Программный код хранится в репозитории онлайн-сервиса Bitbucket. Сервис является бесплатным с ограниченным доступом. При использовании сервиса гарантируется сохранность кода. Также использование таких репозиториев дает возможность быстро загрузить код на любой компьютер без каких-либо потерь.

Программный продукт написан на языке Kotlin, который поддерживается Google с 2017 года и используется в Android Studio, как официальный язык программирования для платформы Android в добавление к Java и C++ [4].

Также в дополнение к написанному коду использованы библиотеки, которые помогают ускорить и упростить написание кода:

- Glide – для удобства работы и изображениями;
- Retrofit2 – для упрощения работы с API;
- BarCodeScanner – для упрощения работы со сканнером QR-кода.

При разработке использован нативный интерфейс API, разработанный компаний ООО «Симбирские технологии» в рамках кода CMS CS-Cart.

Схема приложения отображена на рис. 7. Главная активность включает в себя все сопутствующие фрагменты, которые отвечают за основные страницы приложения.

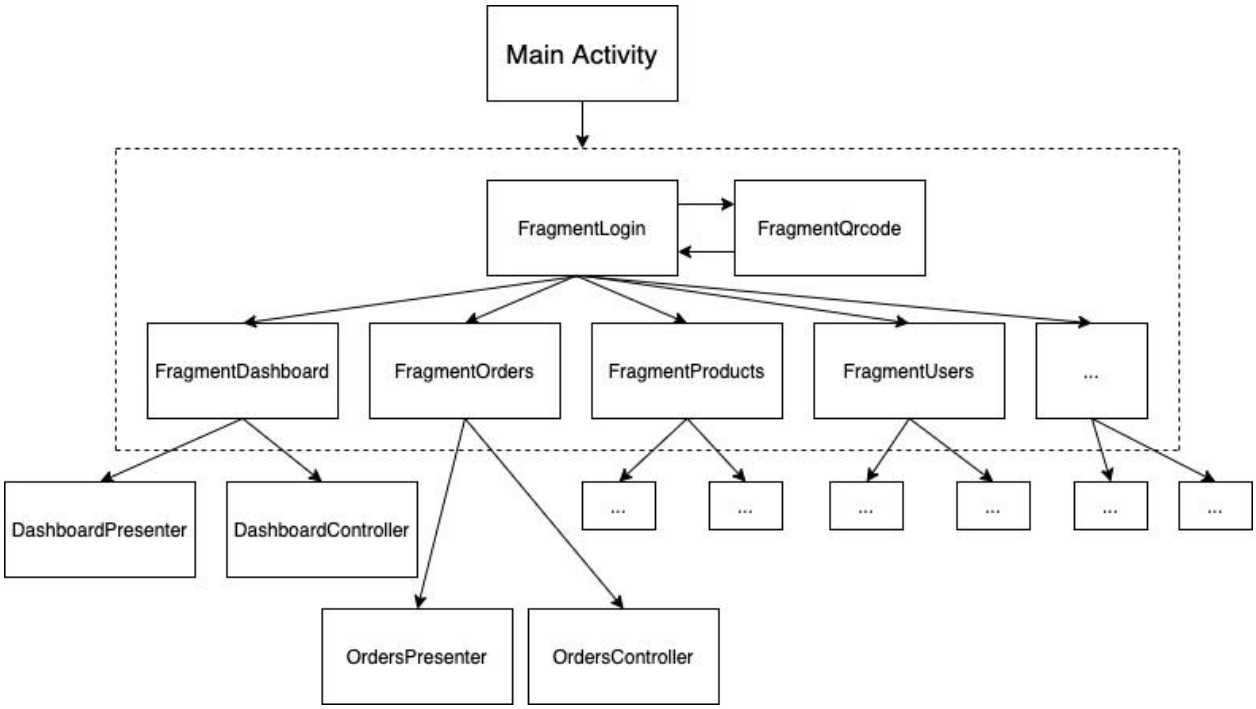

Рис. 7. Общая схема приложения.

Каждый фрагмент включается в себя Presenter и Controller. Controller отвечает за загрузку и обработку информации. Presenter отвечает за отображение данной информации.

К примеру, FragmentOrders – фрагмент страницы списка заказов. OrdersController – загружает заказы и формирует список. OrdersPresenter – отображает сформированный список заказов.

В качестве основного архитектурного паттерна выбрана связка MVP и MVC. Такая связка обоснована следующими критериями:

- Масштабируемость;
- сопровождаемость;
- надежность;
- многоплатформенность;
- разделение ответственности;
- повторное использование кода;
- тестируемость.

На данном этапе разработки не использованы автоматические тесты для тестирования функционала приложения. Тестирование происходило полностью вручную частями, после каждого этапа разработки.

Этапы разработки разбиты по функционалу приложения. Каждый этап разработки отвечал за конкретную страницу приложения. Такой способ тестирования и разработки позволяет разделить ответственность, не нарушая функционала других страниц приложения.

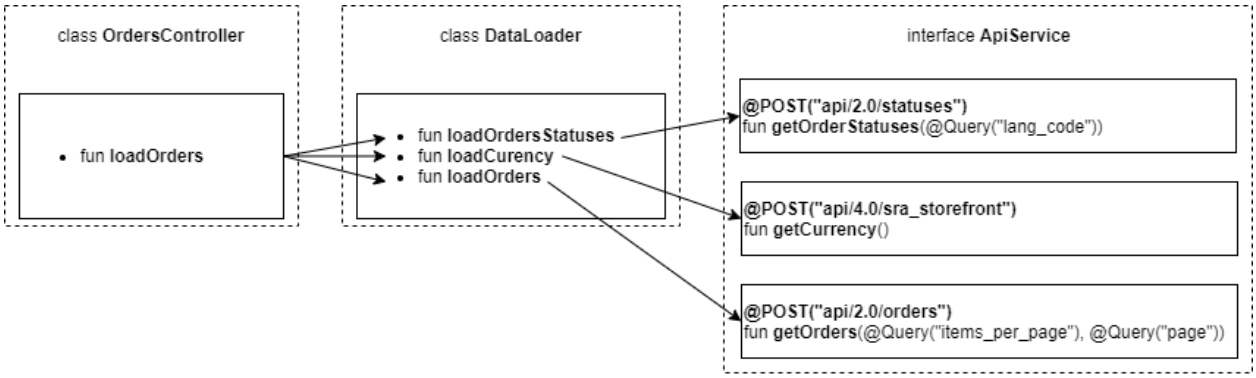

Рис. 8. Схема загрузки списка заказов.

На рис. 8 изображен пример схемы реализации создания API запроса. В классе контроллера реализована одна функция, которая загружает в себе всю необходимую для заказов информацию (статусы заказов, валюта магазина, список заказов). Для загрузки данной информации создан класс **DataLoader**. Данный класс универсален и доступен для каждого из контроллеров. Класс хранит в себе функции создания API запросов.

Каждая функция создает API запрос и обрабатывает результат. В случае успеха возвращает результат в начальный класс (OrdersController), в случае ошибки – выводит ошибку на экран.

Все API запросы описываются в классе-интерфейсе **ApiService**. Каждая функция состоит из двух строк. Первая строка – это аннотация запроса, состоящая из типа запроса и метода. Вторая строка является вызовом запроса и принимает параметры, которые отправляются вместе с API запросом.

Данный функционал реализован с помощью библиотеки **Retrofit2**. Благодаря ему код становится меньше и более понятен. Retrofit2 в себе использует библиотеку **okhttp3.**

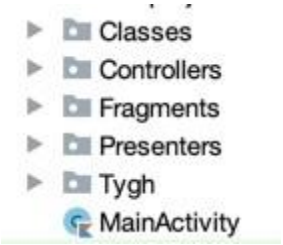

Рис. 9. Навигация классов приложения.

Основным критерием проектирования приложения являлась легкая и понятная навигация по коду. Таким образом, каждый класс приложения находится в отдельной тематической папке. На рис.9 изображен состав пакета приложения. Все контроллеры лежат в папке **Controllers**, все фрагменты лежат в папке **Fragments** и т.д. Также имеется папка **Tygh**,

которая содержит в себе дополнительные классы. Они могут быть доступны из любого класса. Папка **Classes** хранит в себе все сущности, которые используются в приложении.

Заключение

В настоящее время многие люди совершают свои покупки через Интернет. Чтобы не терять клиентуру (или даже увеличить ее) большинство сфер бизнеса переходят в онлайн-режим. Для ускорения перехода CMS CS-Cart предоставляет удобные средства, которые поддерживают как просто обычный магазин (CS-Cart), так и маркет-плейс – место, где располагается много представителей разных магазинов и представляют свои товары (Multi-Vendor).

В результате для популярного продукта CMS CS-Cart разработан удобный интерфейс для работы администратора интернет-магазина в любое время и в любом месте.

В ближайшее время планируется расположить данное приложение в магазине Google Play, после чего планируется портировать приложение на мобильную ОС iOS.

Функционал приложения будет расширяться для привлечения большего числа потребителей. Планируется вывести большую часть функционала сайта в мобильное приложение, что позволит заменить использование сайта.

Данный программный продукт может быть использован не только владельцем интернет-магазина, но и его сотрудниками благодаря распределению привилегий, которые реализованы в CMS CS-Cart и поддерживаемые мобильным приложением.

Мобильное приложение поддерживает два языка: русский и английский, что увеличивает рынок использования. В дальнейшем планируется расширить список языков, чтобы позволить большему числу потребителей использовать его с удобством и легкостью.

Мир технологий растет и развивается. Развитие направлено на облегчение и улучшение качества жизни каждого жителя нашей планеты. Так и это приложение разработано с целью облегчения работы владельцам интернет-магазинов и ускорения получения своих заказов потребителями этих магазинов.

Список литературы

1. Гриффитс Д., *Head First. Программирование для Android*. СПб.: Питер, 2016. 704 с.
2. Ретабоуил С., *Android NDK. Руководство для начинающих*. М.: ДМК Пресс, 2016. 518 с.
3. Скин Д., Гринхол Д. *Kotlin. Программирование для профессионалов.* СПб.: Питер, 2020. 464 с.
4. Стюарт К., Филлипс Б., Марсикано К. *Android. Программирование для профессионалов*. СПб.: Питер, 2017. 688 с.
5. Фримен Э., Робсон Э. *Head First. Паттерны проектирования. Обновленное юбилейное издание.* СПб.: Питер, 2018. 656 с.
6. Iuanu Adelekan. *Kotlin Programming by Example*. Packt, 2018. 502 p.