УДК 004.4
DOI: 10.18101/978-5-9793-1397-9-88-91

РАЗРАБОТКА БЕССЕРВЕРНЫХ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ

© Мархакшинов Аюр Лувсаншаравович
кандидат технических наук, старший преподаватель,
Бурятский государственный университет имени Доржи Банзарова
Россия, 670000, г. Улан-Удэ, ул. Смолина, 24а
E-mail: ayurmar@yandex.ru

Описаны принципы создания мобильных приложений, не требующих написания программного кода серверной части. Базу данных, функции аутентификации пользователей и отправки уведомлений предлагается реализовывать по модели BaaS (Back end as a Service — бэкенд как услуга). В качестве примера продемонстрирована архитектура бессерверного мобильного приложения, использующего облачные сервисы Google Firebase. Приведено краткое описание основных взаимодействий между компонентами бессерверной архитектуры.

Ключевые слова: мобильные приложения; бессерверная архитектура; облачные сервисы; разработка приложений.

Введение

В настоящее время неотъемлемой частью практически любого мобильного приложения стали функции предоставления пользователям индивидуальных учетных записей, хранения и обработки информации с помощью единой базы данных, рассылки push-уведомлений для оповещения о различных событиях. Как правило, реализация этих функций подразумевает написание дополнительного программного кода серверной части приложения (в современной разработке ПО часто называемой бэкенд-частью приложения или просто бэкендом) помимо непосредственно мобильного клиента.

Очевидно, что уменьшение затрат на создание бэкенд-части или даже полный ее перенос в готовые облачные сервисы позволит существенно сократить сроки выпуска приложения и/или освободить ресурсы для работы над клиентской частью приложения, что особенно актуально для небольших команд разработчиков.

К требованиям, предъявляемым к сервисам BaaS, можно отнести:

- независимость от конкретной платформы клиентского приложения;
- масштабируемость предлагаемых решений;
- наличие гибкого тарифного плана.

Наиболее известными поставщиками BaaS-услуг сегодня являются Google Firebase, Microsoft Azure, Oracle Cloud [1]. Далее в статье рассматривается бессерверная архитектура приложения, основанная на использовании сервисов Google Firebase. Выбор поставщика обосновывается наличием наиболее полного набора бэкенд-услуг, таких как аутентификация пользователей, облачное файловое хранилище, база данных, облачные функции, ана-

литика сбоев приложения и его производительности, сервис тестирования приложений, рассылка уведомлений и др. Особый интерес представляет база данных Cloud Firestore, имеющая возможность автоматической синхронизации информации на подключенных клиентских устройствах.

Архитектура бессерверного мобильного приложения

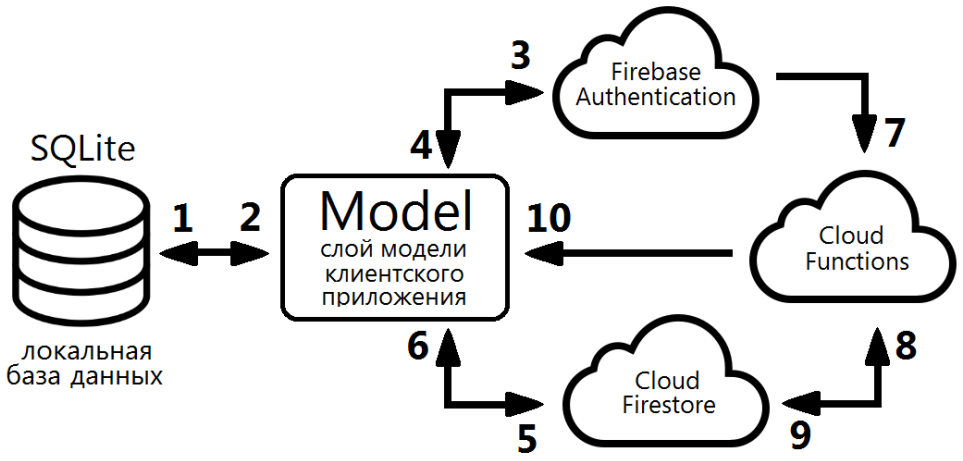

Рис. 1. Пример архитектуры бессерверного мобильного приложения

На рис. 1 показана диаграмма возможных взаимодействий в бессерверном мобильном приложении, построенном на базе сервисов Google Firebase. В приложениях, реализованных по одному из наиболее распространенных паттернов MVC, MVP или MVVM, обработка бизнес-логики выносится в отдельный слой, называемый слоем модели. Модель отвечает за различные операции над данными приложения, которые затем некоторым образом, зависящим от выбранного паттерна, визуализируются в пользовательском интерфейсе.

В архитектуре, использующей BaaS-подход, взаимодействия с локальной базой данных на устройстве клиента, отмеченные на рис. 1 цифрами *1* и *2*, фактически могут понадобиться лишь для кэширования данных с целью предоставления возможности работы с приложением в период отсутствия подключения к сети Интернет. С учетом того, что сервис Cloud Firestore также поддерживает функцию кэширования часто используемых данных, возможен вариант архитектуры без локальной базы данных.

Множество современных приложений предлагает своим пользователям персонализированный контент, для этого используется система учетных записей для идентификации и различные методы аутентификации для подтверждения достоверности учетных записей. Сервис Firebase Authentication предлагает готовое решение для создания учетных записей на основе адреса электронной почты или номера телефона. Кроме того, существует возможность использовать учетные записи некоторых других популярных сайтов (Google, Facebook, Twitter, Github).

В процессе аутентификации (*3*) введенные пользователем логин и пароль пересылаются по защищенному соединению в облачный сервис Firebase, где выполняется их сверка с базой существующих учетных записей приложения. Результат запроса на аутентификацию (*4*) возвращается в клиент приложения. При положительном результате становится доступен уникальный идентификационный номер пользователя и данные его учетной записи. Эти сведения затем могут использоваться для обращения к базе данных и получения информации, актуальной для конкретного пользователя.

В качестве базы данных в предложенной архитектуре используется сервис Cloud Firestore, представляющий собой облачную NoSQL базу данных. В Firestore данные имеют гибкую структуру и хранятся в виде документов, которые индексируются по умолчанию и организованы в коллекции и субколлекции. Преимуществами Firestore, по заявлению разработчика, являются система запросов с удобной фильтрацией и сортировкой результатов, синхронизация с клиентскими устройствами в реальном времени, поддержка офлайн-режима работы с базой данных и легкая масштабируемость[1].

Обращения к Firestore (*5*) делятся на два типа: однократный запрос данных и подписка на события изменения данных в определенном узле базы. Результатом обращения (*6*) в обоих случаях могут являться как отдельный документ, так и группа документов, отобранная и отсортированная в соответствии с параметрами запроса. При однократном запросе существует возможность выбрать источник данных: только облачная база данных, только локальный кэш или облачная база с возвратом результата из кэша в случае недоступности базы данных. При подписке на изменения в определенном документе, коллекции или группе коллекций сервис Firestore в автоматическом режиме отслеживает события изменения, удаления или создания данных, присылая на клиент обновленную информацию.

При работе с сервисами Firebase Authentication и Cloud Firestore доступен API для наиболее популярных языков программирования (Java, Swift, Objective-C, Kotlin, PHP, Python, Go, C#, Ruby), а также REST API.

Сервис Cloud Functions позволяет автоматически запускать серверный код по триггеру или по HTTPS-запросу. Код функций пишется на языке JavaScript и исполняется на облачных серверах Google, избавляя разработчиков от необходимости запускать и поддерживать собственные серверы.

Триггерами для Cloud Functions являются различные события сервисов Firebase. Например, для Firebase Authentication триггерами (*7*) являются события создания и удаления учетных записей, для Cloud Firestore — события (*8*) создания, изменения и удаления документов в базе данных. Функции, вызываемые из Cloud Functions, могут использоваться для обслуживания базы данных Firestore (*9*) либо для взаимодействия с клиентскими приложениями пользователей (*10*).

[1] Firebase Documentation [Электронный ресурс]. URL: https://firebase.google.com/docs (дата обращения: 12.05.2019).

Литература

1. Baldini I., Castro P., Cheng P. et al. Serverless Computing: Current Trends and Open Problems // Proceedings of the International Workshop on Mobile Software Engineering and Systems. 2017. № 1. Pp. 287–306.

SERVERLESS MOBILE APPLICATIONS DEVELOPMENT

Ayur L. Marhakshinov
Cand. Sci. (Engineering), Senior Lecturer,
Dorzhi Banzarov Buryat State University
24a Smolina St., Ulan-Ude 670000, Russia
E-mail: ayurmar@yandex.ru

Principles of mobile applications development, requiring no server code, are described. Database, user authentication and notification sending functions are suggested to be implemented by BaaS model (Backend as a Service). Example architecture of serverless mobile application using Google Firebase cloud services is shown. Brief description of basic interactions between serverless architecture components is given.

Keywords: mobile applications; serverless architecture; cloud services; application development.