Санкт-Петербургский политехнический университет
Петра Великого

—

Институт компьютерных наук и технологий
Кафедра «Распределенные вычисления и компьютерные сети»

КУРСОВОЙ ПРОЕКТ

Проект iGarage

по дисциплине «Программная инженерия»

Выполнил
студент гр. 33507/1 А. В. Шаланкин

Руководитель
профессор, к.т.н А. В. Самочадин

Санкт-Петербург
2016

Оглавление

Постановка задачи

Требовалось создать приложение, позволяющее быстро и эффективно создавать, дополнять и использовать некую базу хранимых товаров (различных вещей, материалов, инструментов и др.). Продукт должен работать стабильно (не вылетать например), а также справляться с объемом работы качественно и надежно.

О проекте

Проект iGarage предназначен для хранения информации об инвентаре гаража. Суть заключается в следующем. Предположим, что в гараже есть несколько шкафов, в каждом из которых лежит нечто материальное. Для каждого шкафа генерируется QR-код, которые распечатывается и помещается в любое удобное для считывания место. Наше приложение включается в себя сканер QR-кодов. После распознавания QR-кода, наше приложение сверяется с базой данных, и, если находится соответствие между распознанным идентификатором и идентификатором, хранящемся в базе данных, приложение выводит список вещей, лежащих в шкафу.

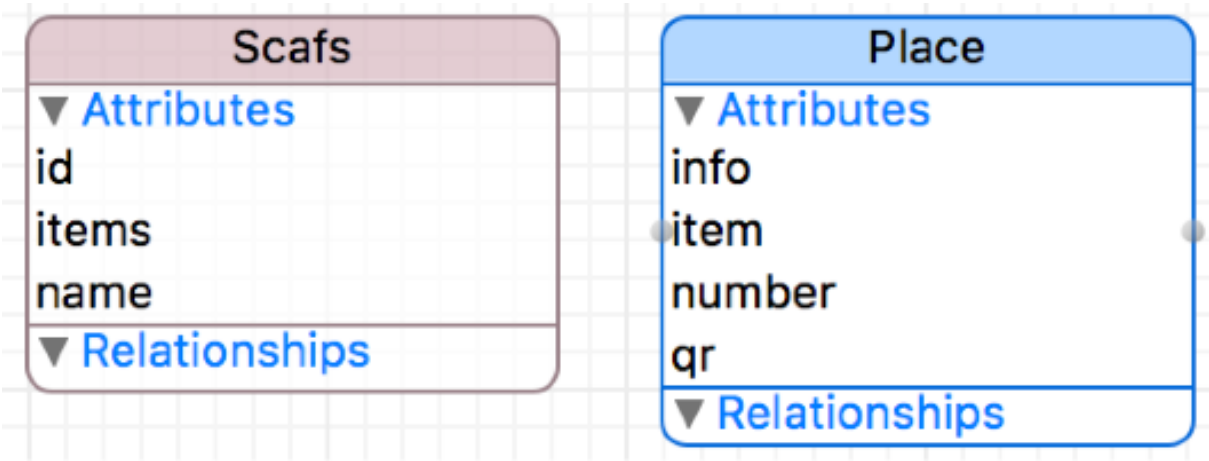
База данных

Для хранения информации о шкафах и содержащихся в нем материальных объектах, к нашему проекту была прикручена база данных. Для идентификации шкафа в базе данных было решено обращаться к нему не по имени, так как это может вызвать ошибки в работе программы в будущем, когда будет добавлена поддержка нескольких пользователей, а по уникальному идентификатору. Для генерации идентификатора было решено использовать библиотеку NSUUID, разработанную компанией Apple. Библиотека NSUUID позволяет генерировать уникальные идентификаторы, причем Apple гарантирует, что генерация двух одинаковых идентификаторов невозможна.

После считывания QR-кода, в котором зашифрован идентификатор шкафа, программа сопоставляет значение распознанного идентификатора с идентификаторами, хранящимися в базе данных. В случае совпадения, программа выводит список предметов из шкафа, соответствующему распознанному идентификатору.

База данных хранится на устройстве локально, что проедотвращает попадание данных о расположении личных вещей в чужие руки (кроме случаев, когда ваш смартфон украден, но это тема для отдельного разговора).

После создания макета проекта, а также всех необходимых файлов, я принялся за разработку базы данных. Нами была построена модель данных и представлена в проекте при помощи фреймворка «Core Data» (как и говорилось раннее). Суть работы данного фреймворка заключается не в формировании постоянных запросов, а в хранении данных в формате модели и запросов к ней,



для выгрузки данных. То есть мы обращаемся к данным только про инициализации переменных (массива) в котором они будут хранится, а затем уже обращаемся к переменной (массиву). Вместе с командой разработки мы пришли к выводу, что данные удобнее всего будет хранить в двух моделях, модели товара и модели шкафа, таким образом мы получили базу данных с двумя таблицами. Таблица «Place» стала базой информации о товаре, а таблица «Scafs» стала базой информации о шкафах. В общем случае данная база используется, как массив переменных некого класса, содержащего поля с информацией. Базы связывали два поля: «id» - «qr» и «name» - «number». Первое поле содержит уникальный идентификатор, а второе - название шкафа. В поле «items» хранится информация о количестве вещей, содержащихся в шкафу, поле «item» хранит название товара, а поле «info» - его описание (например номер полки).

Подведение итогов

Надежность

Как я уже и говорил, в QR коде хранится значение поля «id», в котором, в свою очередь, хранится уникальный идентификационный номер, генерируемый функцией «UUID», лежащей в библиотеке «UIKit». Данный код представляет собой последовательность из 32 чисел и букв, разбитых на 5 групп, которые в свою очередь разбиты дефисом (пример кода: 159CAFBC-36DC-4E84-92F5-D7137A7875FC). Функция «UUID» генерирует уникальный ключ, а его сложность и длина позволяют исключить вероятность повторения. Получаем, что имя шкафа никто, кроме хозяина не узнает, считая QR-код.

База данных, в которой хранится вся информация, лежит локально в папке с приложением, к ней нет удаленного доступа. Таким образом утечка данных исключена. В нашем приложении можно хранить расположение личных вещей, не переживая при этом за попадание этой в информации в чужие руки.

Считывание QR кода происходит при поддержке стандартной библиотеки, а это значит, что вероятность ошибки в данном процессе настолько невелико, насколько качественно в Apple создают библиотеки для Swift.

После нескольких дней использования приложения на различных устройства (соответсвенно после последнего исправления багов) ошибок не было выявлено, что означает, что при регулярном использовании, приложение работает верно, а также стабильно.

Вывод

Итогом всей проделанной работы стало приложение, позволяющее качественно, быстро и надежно создать собственное виртуальное хранилище вещей, а также управлять всей этой базой, например в случае перемещения объектов. Также, наше приложение позволяет наиболее эффективно использовать возможности современных мобильных устройств, а именно их камеры, при помощи которых пользователю открывается возможность, не открывая шкаф, узнать о всех товарах, лежащих в нем. Конечно, пользователем предварительно должна быть составлена вся его база верно, кроме того QR коды должны располагаться также верно. Ошибки пользователя могут привести к ошибкам в работе программы.

Подводя итоги, я считаю, что созданное нашей командой приложение прекрасно справляется со своими задачами, упрощая банальный ежедневный поиск предметов.