МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра Математического и компьютерного моделирования

«Проектирование и разработка ИС учета рабочего времени»

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студентки 4 курса 441 группы

направление 09.03.03 — Прикладная математика и информатика

механико-математического факультета

Голованової Юлии Олеговны

Научный руководитель
доцент, к.ф.-м.н О. М. Ромакина

Зав. кафедрой
зав. каф., д.ф.-м.н., доцент Ю. А. Блинков

Саратов 2019

Введение. Главное направление совершенствования мобильных телефонов можно определить одним понятием: конвергенция технологий. Мобильные телефоны объединили в себе едва ли не всё, что можно и разумно объединять под корпусом одного устройства.

Основные задачи телефона прошлого — совершать и принимать звонки, писать SMS. Сегодня эти задачи дополнилась работой с интернет - ресурсами, прослушиванием музыки, фотосъемкой, использованием игр и приложений. Мобильный уже добавил к списку основных задач просмотр телепрограмм, управление различной техникой, функции контроля состояния здоровья своего владельца и многое другое.

Эксперты обращают внимание потребителей на тот факт, что сейчас на рынке доступно несколько мобильных платформ, стремительную популярность среди которых в последнее время приобретает iOS и Android. На рынке США компания Apple уже давно заняла пальму первенства в сегменте дорогих и презентабельных мобильных устройств, постоянно удивляя клиента интересными новинками. На примере Apple, видно, как кардинально может измениться тот или иной рынок с появлением новых устройств.

Что бы ни говорили, а мобильность сегодня — явление глобальное, проникающее во многие сферы нашей жизни. Почти половина всех пользователей мобильных услуг, проживают в пяти странах, среди которых и Россия. Ныне потребитель ищет не просто мобильный телефон, он внимательно изучает условия, соотнося их с требованиями современного рынка. Именно поэтому можно утверждать, что развитие мобильных технологий будет только ускоряться.

Для выполнения бакалаврской работы было необходимо разработать мобильное приложение для операционной системы iOS компании Apple, версии выше 10.0. Приложение должно функционировать в портретной ориентации на устройствах Apple iPhone.

Основная часть. Необходимо разработать мобильное приложение для операционной системы iOS компании Apple, версии выше 10.0. Приложение должно функционировать в портретной ориентации на устройствах Apple iPhone.

Приложение должно представлять собой клиент для сайта http://cms.profsoft.online, который является сайтом для сотрудников компании Profsoft. На этом сайте пользователи могут осуществлять вход в свой личный кабинет, отслеживать часы работы, смотреть количество бонусов, проверять есть ли определенный сотрудник в офисе.

Так как для работы на сайте необходима авторизация, в приложении требуется реализовать функционал, позволяющий производить авторизацию зарегистрированным сотрудникам.

После авторизации текущая сессия пользователя должна быть сохранена. Это значит, что после авторизации при запуске приложения не нужно снова вводить имя пользователя и пароль, вход будет осуществлен автоматически.

1.Выбранные средства разработки

1.1 Среда разработки

Для разработки приложения выбор был остановлен на пакете

инструментов XCode, разработанном компанией Apple. В пакет входит интегрированная среда разработки и набор библиотек Cocoa.

Cocoa — объектно-ориентированный API для операционной системы macOS производства компании Apple. Это один из пяти основных API, доступных в Mac OS X, — Cocoa, Carbon, Toolbox (для работы старых приложений Mac OS 9), POSIX и Java. Такие языки, как Perl, Python и Ruby, не считаются основными, так как на них пока что пишется не так много серьёзных приложений для Mac OS X.

Xcode — интегрированная среда разработки (IDE) программного обеспечения для платформ macOS, iOS, watchOS и tvOS, разработанная корпорацией Apple. Первая версия выпущена в 2001 году. Стабильные версии распространяются бесплатно через Mac App Store, а зарегистрированные разработчики также имеют доступ к бета-сборкам через сайт Apple Develope.

2. Язык

Swift — молодой, быстроразвивающийся язык с большим количеством нововведений. Это первый язык программирования промышленного качества, который так же понятен и увлекателен, как скриптовый язык.

Swift исключает большой пласт распространенных программных ошибок при помощи применения современных программных паттернов:

- Переменные всегда инициализированы до того, как будут использованы.
- Индексы массивов всегда проверяются на out-of-bounds ошибки.
- Целые числа проверяются на переполнение.
- Опционалы гарантируют, что значения nil будут явно обработаны.
- Автоматическое управление памятью
- инструкции, методики, разработанные в процессе выполнения работы;
- Обработка ошибок позволяет осуществлять контролируемое восстановление от непредвиденных ошибок.

2.1 База данных сервера и API

API (Application Programming Interface) - набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) или операционной системой для использования во внешних программных продуктах.

2.2 Взаимодействие с API и парсинг данных

В общем случае, парсинг строит шаблон последовательности символов. Например, может использоваться древовидная структура. Она показывает, в какой последовательности в строке встречаются символы. Может указывать на приоритет, если речь идет о математическом выражении. Такие структуры нужны для анализа данных.

2.3 COREDATA, Interface Builder и Table View

Core Data – это фреймворк, который управляет и хранит данные в приложении. Можно рассматривать Core Data, как оболочку над физическим реляционным хранилищем, представляющую данные в виде объектов, при этом сама Core Data не является базой данных.

Процесс создания любого приложения можно условно разделить на три этапа: создание интерфейса, непосредственное написание кода и отладка. Interface Builder (далее просто IB) — средство для визуального создания и тестирования интерфейсов, входящей в состав SDK разработчика под Mac OS.

Table View - табличное видовое представление. Табличное представление представляет данные в виде списка с прокруткой с множеством строк, которые могут быть разделены на секции. Оно представляет данные в одноколо-

ночном многострочном списке, и позволяет отображать и редактировать его иерархию.

3 Проектирование

3.1 MVC

При проектировании приложения использовался шаблон проектирования «Модель-Вид-Контроллер» (MVC). Это классический шаблон, он прост в понимании и отлично применяется при проектировании небольших проектов. Данная концепция позволяет разделить представление, данные и обработку пользовательских действий на три отдельных компонента:

1. Модель. Модель представляет данные и методы для работы с ними, изменения их состояния и реакции на запросы.
2. Вид, представление. Отвечает за представление информации пользователю
3. Контроллер. Позволяет обеспечить связь между пользователем и системой, используя модель и вид, и реализует необходимую реакцию модели на действия пользователя.

3.2 Кэширование данных и настройка Table View

В сфере вычислительной обработки данных кэш – это высокоскоростной уровень хранения, на котором требуемый набор данных, как правило, временного характера. Доступ к данным на этом уровне осуществляется значительно быстрее, чем к основному месту их хранения. С помощью кэширования становится возможным эффективное повторное использование ранее полученных или вычисленных данных.

Table View — это табличное представление представляет данные в виде списка с прокруткой с множеством строк, которые могут быть разделены на секции. Оно представляет данные в одноколоночном многострочном списке.

Настройка таблиц производится в Interface Builder, в секции Table View инспектора атрибутов. Некоторые настройки недоступны в инспекторе атрибутов, и они должны выполниться с помощью программного кода.

Для того, чтобы отображать контент, табличное представление должно иметь источник данных. Источник данных является посредником между моделью данных приложения и таблицей. Источник данных Table View должен поддерживать UITableViewDataSource протокол.

Каждая отдельно взятая ячейка может отображать различные варианты контента. Ячейки могут использовать стиль по умолчанию, могут отображать изображения и текстовые метки.

4 Программная реализация

4.1 Клиент API

Главная роль в механизме доступа к данным в проекте отдана классу APIClient. Это класс, реализующий шаблоны «Одиночка» и «Фасад», скрывающий за собой настройки подключения к API сервера, методы запросов по ключам API, настройки отражения ответов сервера с данными в объекты, правила кэширования получаемых данных и логика выбора принудительной загрузки данных с сервера или из базы кэшированных данных. В то же время предоставляет единый интерфейс работы с данными в проекте: достаточно вызвать метод для получения определенных данных, а внутренняя логика либо сделает запрос к серверу и отразит ответ в набор объектов, либо загрузит данные из базы кэшированных данных.

4.2 Навигация

Интерфейс приложения описан в специальном файле с расширением storyboard. В нем содержится все экраны приложения, кроме создаваемых программно. К каждому экрану привязывается файл-наследник класса UIViewController. Этот класс отвечает за загрузку и отображение данных на экране и реагирование на действия пользователей.

Основные переходы по экранам обеспечивает класс UINavigationController. Он представляет собой некий стек экранов (контроллеров), в который можно «положить» следующий экран и тем самым перейти вперёд по навигации, либо убрать последний экран, тем самым перейти назад.

За параллельную схему интерфейса отвечает контроллер вкладок – UITabBarController. В представлении данного контроллера в роли каждой вкладки может выступать как простой экран UIViewController, так и набор экранов UINavigationController.

4.3 Сессия пользователя и локализация

Авторизация пользователя с помощью адреса электронной почты и пароля, пользователь вводит данные в необходимые поля и по нажатию на кнопку «Войти» на сервер отправляется запрос.

Успешным ответом сервера будет объект типа UserProfile, содержащий в себе всю информацию о профиле пользователя.

Для локализации приложения используется стандартный механизм библиотеки UIKit. В проекте присутствуют файлы с расширением strings, относящиеся к определенным языкам (отношение устанавливается в среде XCode). Каждая строка файла состоит из ключа и значения, которое будет использоваться в локализации.

5 Руководство пользователя

5.1 Требования ПО

Для запуска приложения требуется Apple iPod, iPhone или iPad с операционной системой iOS выше версии 10.0.

5.2 Вход в приложение

Запустить приложение из меню приложений. После открытия приложения пользователю предстоит вход в систему. Вход осуществляется по личной почте пользователя и пароля, которые выдаются ему после того, как его зарегистрировали в специальной офисной системе

5.3 Раздел «Список сотрудников»

Далее после входа в систему пользователю открывается раздел под названием «Список сотрудников». В этом разделе находится список всех сотрудников. Рядом с каждой фотографией того или иного сотрудника находится значок, которой показывает, есть сотрудник в офисе в данный момент.

Так же рядом с фамилией и именем сотрудника находится значок под названием «Лог». При нажатии на этот значок пользователь данного приложения может увидеть расписание интересующего его сотрудника. В этом расписании представлены часы работы сотрудника. Часы работы пользователя можно просматривать за:

1. День
2. Месяц
3. Всего

Если на экране «Список сотрудников» приложения кликнуть по фамилии и имени, то откроется профиль сотрудника, где отображается аватарка, мобильный телефон, почта, день рождения и четыре ссылки на социальные сети и мессенджеры. Если выбранный нами для просмотра сотрудник заполнил эти ссылки или ссылку, то в его профиле значок подсвечивается голубым цветом. Если пользователь приложения кликает на этот значок, то в браузере или другом приложении (зависит от того, есть ли это приложение на используемом устройстве) открывается профиль сотрудника.

5.4 Раздел «Общий лог»

На данном экране приложения отображаются все входы и выходы сотрудников, этот раздел нужен для удобства и быстрого отслеживания опозданий и проверки на то, был ли тот или иной сотрудник в офисе в или нет, чтобы не заходить специально в профиль каждого работника.

5.5 Раздел «Сотрудник»

Данный раздел приложения представляет собой личный профиль пользователя. Тут доступна информация о бонусах и количестве дополнительных выходных пользователя, также доступны дополнительные экраны, такие как: смена пароля, лог (расписание пользователя), просмотр и редактирование профиля.

При клике на экране «Сотрудник» на кнопку «Сменить пароль», открывается дополнительный экран для смены пароля с тремя полями, доступными для ввода. Далее, после заполнения всех трех окон для смены пароля, новые данные отправляются на сервер, где перезаписываются за место старых.

Из экрана «Сотрудник» пользователь может попасть в свой профиль, если кликнет на свою фамилию и имя. Экран профиля схож с экраном профиля других сотрудников, описанным раннее, но добавляется одна деталь – это редактирование профиля. В редактирование профиля пользователю доступно внесение всех четырех ссылок на социальные сети и мессенджеры, изменение фотографии и фонового изображения профиля. Затем после нажатия на кнопку «Готово» новые данные отправляются на сервер и сохраняются.

На экране «Сотрудник» доступна кнопка «Выход», при нажатии на которую, пользователя возвращает на экран авторизации приложения.

6 Тестирование

Тестирование приложение – это важный этап, на котором проверяется поведение приложения на большом количестве входных данных, включая неверные.

На тестирование в проекте было отведено сорок часов. Задачей занималась команда тестировщиков, состоящая из двух человек. Тестирование проводилось вручную.

В процессе тестирования был выявлен ряд ошибок в алгоритмах приложения, которые были успешно исправлены, а также были внесены некоторые изменения в графический интерфейс

6.1 Особенности тестирования мобильных приложений

Мобильные приложения — это приложения, тестирование которых значительно отличается от тестирования на компьютере.

Мобильное приложение должно быть интуитивно понятным, удобным, работать бесперебойно, безопасно, круглосуточно и достаточно быстро реагировать на действия пользователя. Итак, рассмотрим основные моменты и особенности тестирования мобильных приложений, на которые стоит обратить внимание при функциональном и GUI тестировании мобильных приложений

Основными и опорными положениями для тестирования мобильных приложений является:

1. Размер экрана
2. Touch-интерфейс
3. Утечки памяти
4. Проверка работы приложений на ретина экранах
5. Проверка работы приложений для разных версий OS
6. Проверка типа покупок (восстанавливаемые, не восстанавливаемые)
7. Проверка работы обратной связи
8. Проверка работы обновлений
9. Проверка реакции приложения на внешние прерывания
10. Реклама в мобильном приложении
11. Проверка локализации
12. Проверка энергопотребления

Заключение. В ходе подготовки бакалаврской работы были приобретены знания о разработке мобильных приложений для платформы iOS и осо-

бенностях работы API сервисов, а также разработано мобильное приложение-клиент для сайта, которое позволяет с легкостью пользоваться сервисом с высокой степенью мобильности.