



**Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Российский государственный социальный университет»**

**Факультет информационных технологий
Кафедра информационных систем, сетей и безопасности**

Направление подготовки – 09.03.02 Информационные системы и технологии

Квалификация: бакалавр

Выпускная квалификационная работа

Тема: Разработка информационной системы складского учета для ООО «ФИНЧ»

Обучающийся

_____ подпись

Потапов Антон Ильич

Дата _____

Руководитель

_____ подпись

ст. пр. Елисеева Д.Ю.

_____ (ученая степень, ученое звание, фамилия, инициалы)

ВКР допущена к защите «___» _____ 20__ г.

И.о. заведующего кафедрой
информационных систем,
сетей и безопасности
канд. экон. наук

_____ подпись

Веретехина С.В.

Москва, 2019

Содержание

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ.....	4
1.1 АНАЛИЗ СУЩЕСТВУЮЩИХ РЕШЕНИЙ СИСТЕМ СКЛАДСКОГО УЧЁТА	4
1.2 АНАЛИЗ ДЕЯТЕЛЬНОСТИ КОМПАНИИ.....	12
1.3 ОПИСАНИЕ И ВЫБОР ИНСТРУМЕНТАЛЬНЫХ СРЕДСТВ ДЛЯ РЕАЛИЗАЦИИ ПРОЕКТА	14
Выводы по главе 1	22
2. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ МОДЕЛЕЙ И РАЗРАБОТКА СИСТЕМЫ СКЛАДСКОГО УЧЁТА НА ПРИМЕРЕ КОМПАНИИ ООО «ФИНЧ»	25
2.1 ВЫБОР МЕТОДОВ И СРЕДСТВ ПРОЕКТИРОВАНИЯ	25
2.2 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ СИСТЕМЫ.....	27
2.3 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ СИСТЕМЫ	28
2.4 ПРОЕКТИРОВАНИЕ МАКЕТА ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА	29
2.5 ОПИСАНИЕ КОНЕЧНОГО ВИДА ИНТЕРФЕЙСА	33
2.6 АЛГОРИТМЫ РАБОТЫ.....	42
Выводы по главе 2	44
3 РАЗРАБОТКА КОМПОНЕНТОВ ИНФОРМАЦИОННОЙ СИСТЕМЫ.....	44
3.1 ОПИСАНИЕ СРЕДЫ РАЗРАБОТКИ И ПРОГРАММЫ	44
3.2 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ИНФОРМАЦИОННЫХ МОДЕЛЕЙ.....	46
Выводы по главе 3	55
4 ОБОСНОВАНИЕ ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ.....	56
Выводы по главе 4	59
ЗАКЛЮЧЕНИЕ.....	60
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	63
ПРИЛОЖЕНИЕ А. ФРАГМЕНТЫ ИСХОДНОГО КОДА ПРОГРАММЫ.....	68

ВВЕДЕНИЕ

Тема выпускной квалификационной работы «Разработка информационной системы складского учета для ООО «ФИНЧ».

Актуальность темы выпускной квалификационной работы обусловлена необходимостью выгрузки сводной информации, высокие затраты времени на проведение ревизии и взаимодействие с инвентарём в сервисе KeerTeam. Внедрение разработанной информационной системы позволит сократить затраты времени на проведение ревизии, обеспечить быстрый поиск хранимого инвентаря в системе и избавить компанию от нужды покупать и поддерживать стороннюю дорогостоящую информационную систему.

Объект исследования: процесс разработки информационной системы складского учета.

Предмет исследования: содержательные и технологические особенности разработки информационной системы складского учета.

Цель работы: разработать информационную систему складского учёта для хранения инвентаря в компании ООО «ФИНЧ».

Для реализации поставленной цели сформулированы следующие задачи:

- 1) провести анализ предметной области и инструментальных средств для реализации проекта;
- 2) разработать информационную модель пользовательского интерфейса системы складского учёта для компании ООО «ФИНЧ»;
- 3) осуществить программную реализацию информационных моделей и алгоритмов системы складского учёта;
- 4) провести обоснование экономической эффективности информационной системы для компании ООО «ФИНЧ».

Практическая апробация результатов исследований докладывалось на XIII международной конференции «Современные информационные технологии в образовании, науке и промышленности».

Выпускная квалификационная работа содержит 4 главы, шестьдесят семь страниц, одну таблицу, один график, одно приложение.

1. Анализ предметной области

1.1 Анализ существующих решений систем складского учёта

На сегодняшний день существует большое количество систем складского учёта, как коммерческих, так и свободно распространяемых, каждая из которых может быть удобна в различных ситуациях.

Крупные компании могут позволить себе более сложные и дорогие системы складского учёта такие как «1С-Предприятие», «Бизнес.Ру» или облачный сервис «Мой склад».

На рынке представлены различные программы складского учета.

- 1С: Торговля и склад (см. рис. 1.1) - программа для учета товаров и автоматизации складского учета[26].



Рисунок 1.1 - 1С: Торговля и склад

- Простой Склад (см. рис. 1.2) - это программа складского учета, позволяющая легко выполнять операции управления складом, очень проста в работе и конфигурировании[28].

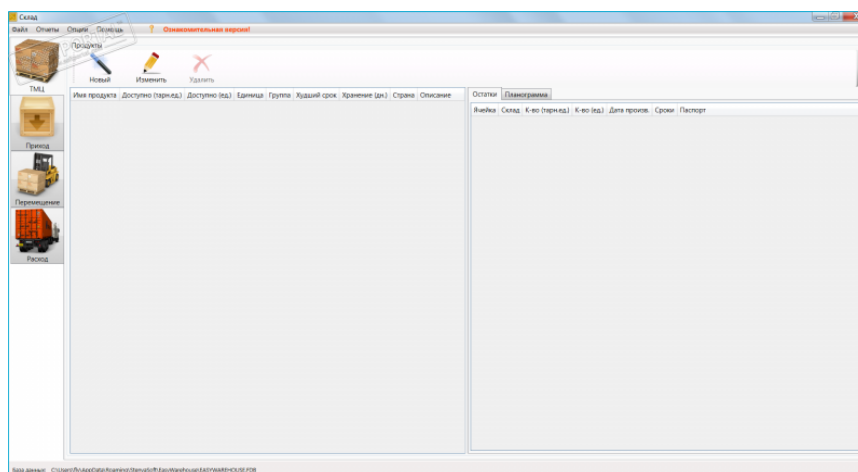


Рисунок 1.2 - Простой Склад

- Склад в Excel (см. рис. 1.3) - это довольно простая складская программа с простым управлением для ведения учета и контроля на производстве; осуществляет учет остатков склада на всех этапах от оприходования до отгрузки продукции заказчику, не требует специального обучения или каких-либо специальных навыков, пригодна для разных видов производства[36].

Артикул	Наименование	Ед	Остатки			Цена продажа	Цена закуп	Поставщик
			Доступно	Факт.	Бронь			
Л333-47	Лист r/k 10XCHД 12*1500	шт	100	100		703,00	403,00	ООО "МетСнаб"
Л333-48	Лист r/k 10XCHД 14*1500	шт	100	100		704,00	404,00	ИП СеверМет
Л333-49	Лист r/k 10XCHД 16*1500	шт	100	100		705,00	405,00	ТрансКо
Л333-50	Лист r/k 10XCHД 20*1500	шт	100	100		705,00	406,00	ТрансКо
Л333-51	Лист r/k 10XCHД 30*1500	шт	96	100	4	705,00	407,00	ТрансКо
Л333-52	Лист r/k 30XГСА 12*1200	шт	100	100		705,00	408,00	ТрансКо
Л333-53	Лист r/k 30XГСА 14*1200	шт	100	100		705,00	409,00	ТрансКо
Л333-54	Лист r/k 30XГСА 16*1200	шт	100	100		705,00	410,00	ИП СеверМет
Л333-55	Лист r/k 30XГСА 20*1200	шт	100	100		705,00	411,00	ИП СеверМет
Л333-56	Лист r/k 30XГСА 30*1200	шт	100	100		705,00	412,00	ИП СеверМет
Л333-57	Лист r/k 30XГСА 40*1200	шт	93	100	7	705,00	413,00	ИП СеверМет
Л333-58	Лист r/k 30XГСА 50*1200	шт	100	100		705,00	414,00	ООО "МетСнаб"
Л333-59	Лист r/k 30XГСА 60*1200	шт	100	100		705,00	415,00	ООО "МетСнаб"
Л333-60	Лист r/k 30XГСА 75*1200	шт	100	100		705,00	416,00	ООО "МетСнаб"
Швеллер								
Ш123-12	Швеллер гнутый ст3сп/nc-5 100*50	куб.м	97	100	3	201,00	151,00	ИП СеверМет
Ш123-13	Швеллер гнутый ст3сп/nc-5 100*60	шт	100	100		202,00	152,00	ТрансКо
Ш123-14	Швеллер гнутый ст3сп/nc-5 120*50	п.м.	100	100		203,00	153,00	ООО "МетСнаб"
Ш123-15	Швеллер гнутый ст3сп/nc-5 140*50	шт	100	100		204,00	154,00	ИП СеверМет
Ш123-16	Швеллер гнутый ст3сп/nc-5 160*80	шт	100	100		205,00	155,00	ТрансКо
Ш123-17	Швеллер гнутый ст3сп/nc-5 160*90	шт	100	100		206,00	156,00	ТрансКо
Ш123-18	Швеллер гнутый ст3сп/nc-5 170*80	шт	100	100		207,00	157,00	ТрансКо
Ш123-19	Швеллер гнутый ст3сп/nc-5 180*80	шт	100	100		208,00	158,00	ТрансКо
Ш123-20	Швеллер гнутый ст3сп/nc-5 190*80	шт	100	100		209,00	159,00	ТрансКо
Ш123-21	Швеллер гнутый ст3сп/nc-5 200*80	шт	100	100		210,00	160,00	ИП СеверМет
Ш123-22	Швеллер гнутый ст3сп/nc-5 200*90	шт	100	100		211,00	161,00	ИП СеверМет
Ш123-23	Швеллер гнутый ст3сп/nc-5 200*100	шт	100	100		212,00	162,00	ИП СеверМет

Рисунок 1.3 - Склад в Excel

- Класс365 (см. рис. 1.4) -онлайн сервис для автоматизации малого бизнеса. Класс365 ориентирован на представителей малого бизнеса. Программа позволяет вести полноценный торговый и складской учет, включает многофункциональную CRM систему (запись истории взаимоотношений, встроенный почтовый клиент и сервис SMS), банк, поддерживает интеграцию с интернет-магазинами [18].



Рисунок 1.4 - Класс365

- Promarket(см. рис. 1.5) - Система Интерактивных Бизнес Коммуникаций 3.0. - интерактивная торгово-закупочная система для специалистов и профессионалов из различных отраслей экономики, работать на Безлимитной Версии Торговой Платформы - Promarket (приложение к программе Skype), что дает возможность быстрого и эффективного

наращивания клиентской базы. Торговля происходит в режиме Обмена Мгновенными Сообщениями. запрос - ответ. Участники, находящийся за Сотни километров друг от друга, сразу видят поступающую информацию на экране своего монитора и могут предложить свою продукцию/услугу или сделать запрос о покупке интересующего товара. А транспортные компании доставят груз до склада [51].



Рисунок 1.5 - Promarket

- Склад производства(см. рис. 1.6) - свободно распространяемая программа для учета на складе; осуществляет учет прихода, расхода со склада, состояние склада, контроль минимальных остатков, контроль своевременного списания товара[38] .

№ Док	Дата	Поставщик	Склад	Сумма	Закрывает
A-28	12.10.2013	ООО "Курьер-Сервис Омск"	Склад в цехе	165 000,00	<input type="checkbox"/>
A-28	09.10.2013	ООО "МЕТАЛЛ-ДИЗАЙН"	Склад в цехе	360 000,00	<input checked="" type="checkbox"/>
A-25	03.10.2013	ООО "Евровосток"	Склад сбыта	643 058,00	<input checked="" type="checkbox"/>
14	02.10.2013	ООО "Группа компаний Старт"	Склад полуфабриката	231 587,00	<input checked="" type="checkbox"/>
114	01.10.2013	ООО "Матушка рукодельница"	Уличный склад	185 000,00	<input checked="" type="checkbox"/>
118	12.09.2013	ООО "Копи-Центр"	Склад крепежных элементов	175 000,00	<input checked="" type="checkbox"/>
119	21.08.2013	ООО "Индустрия сервис"	Склад полуфабриката	200 000,00	<input checked="" type="checkbox"/>
У-8	14.08.2013	ООО "ИТЦ Меандр-Автоматик"	Заказ № 28163	360 000,00	<input checked="" type="checkbox"/>
0421	26.07.2013	ООО "Изумруд"	Склад в цехе	120 000,00	<input checked="" type="checkbox"/>
541	25.06.2013	ООО "ВЕСТА"	Склад полуфабриката	5 000,00	<input checked="" type="checkbox"/>
A9823	13.06.2013	ООО "ДП - Композит"	Склад в цехе	30 000,00	<input checked="" type="checkbox"/>
28	06.06.2013	ООО "Группа "ПСМК"	Уличный склад	72 000,00	<input checked="" type="checkbox"/>
M23	08.05.2013	ООО "Ижорский резервуарный завод"	Уличный склад	600 000,00	<input checked="" type="checkbox"/>
O63	01.05.2013	ООО "Медицинские системы"	Уличный склад	18 000,00	<input checked="" type="checkbox"/>
C54	10.04.2013	ООО "НПП "АВИАГАЗ-СОЮЗ+"	Склад крепежных элементов	240 000,00	<input checked="" type="checkbox"/>
K4	16.02.2013	ООО "Элодея"	Склад сбыта	30 000,00	<input checked="" type="checkbox"/>

Рисунок 1.6 Склад производства

- Storage(см. рис. 1.7) - простая в использовании программа для организации складского учета, производства и рознично-оптовой торговли,

имеет интерфейс, интуитивно понятный даже начинающему пользователю[55].

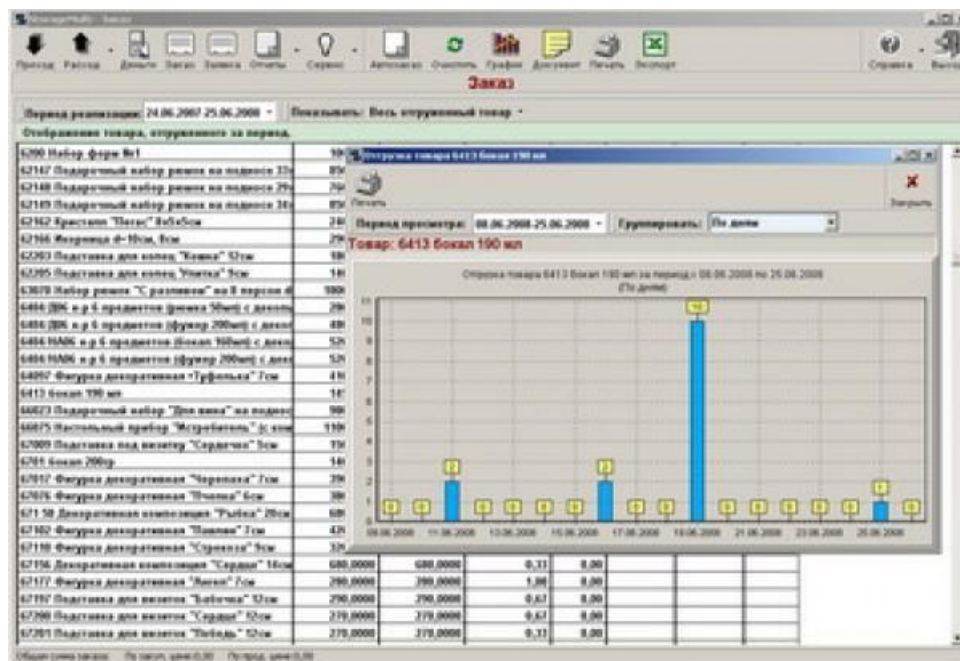


Рисунок 1.7 Storage

- Лайт-учёт (см. рис. 1.8) - мощная и простая программа для автоматизации товарного складского и финансового учета, подходит для всех видов предприятий торговли, оптовых и мелкооптовых фирм, оптово-розничных, розничных и частных предпринимателей, позволяет вести складской учет товаров в торговле, а также финансовый учет; включает в себя склад, кассу, банк и счета, единый справочник товаров и клиентов и многое другое; полная поддержка штрих-кодирования; печать на ценниках и документах, поиск[19]. Возможности программы:

- отдельный учет накладных, товарных чеков и счетов-фактур, одновременное ведение оптового и розничного учета;
- настраиваемая фасовка товаров и единиц измерения, удобная работа, как с фасованным товаром, так и с весовым товаром;
- возможность использования процентных скидок, учет товаров по срокам годности, учет транспортных и оказанных услуг;
- возможность ведения учета по 99-ти складам или отделам, авансовые отчеты, справочник поставщиков и покупателей;

- учет накладных на реализацию, учет бартерных операций, расчет продажи товара для расчетов с поставщиками;
- одновременное и независимое ведение складского и бухгалтерского учета;
- формирование, хранение и печать огромного количества форм и документов.

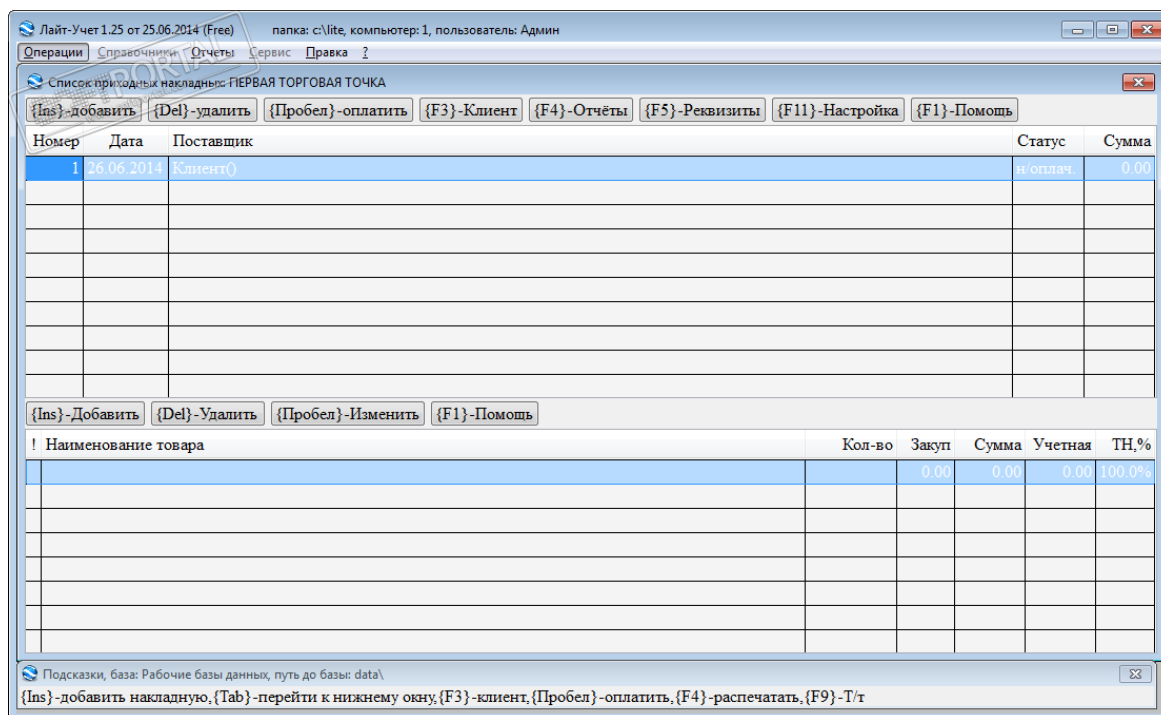


Рисунок 1.8 Лайт-учёт

- Storecalc(см. рис. 1.9) - простая складская программа, складской калькулятор. Выполняет лишь самые необходимые операции. Разработана для неподготовленных пользователей, у которых есть потребность вести учет на складе “для себя”. Сохраняет код, наименование, цену, дату последнего движения товара и остатки на складе (количество товара, стоимость). Для каждого товара сохраняются все его движения. Оптимизирована для работы с ноутбуком [56] .

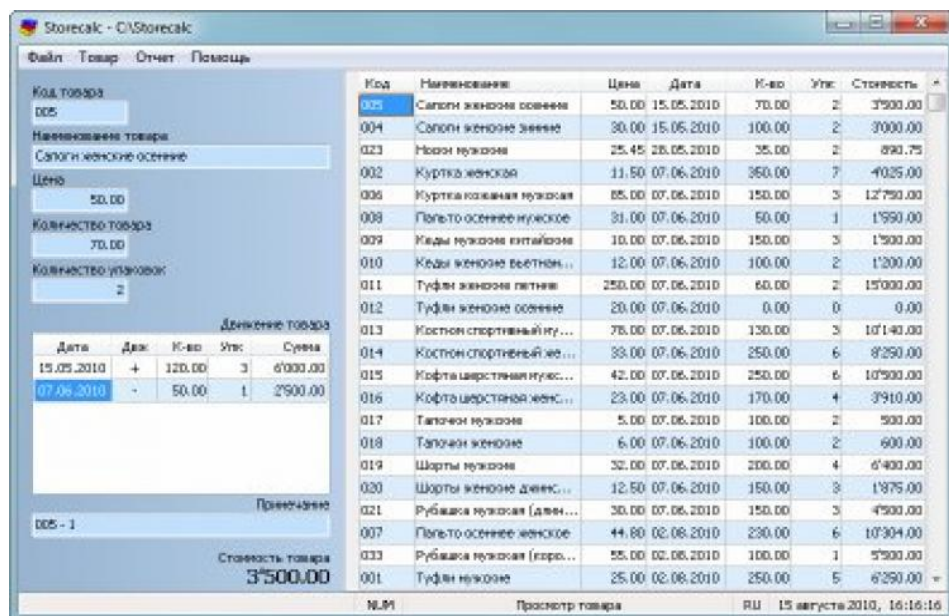


Рисунок 1.9 Storecalc

- ТЦУ Старт (см. рис. 1.10) -свободно распространяемая программа учета для склада и торговли под задачи малого и среднего бизнеса; позволяет контролировать неограниченное количество складов, торговых точек, касс, счетов, вести взаиморасчеты с клиентами, планирование и финансовый анализ[42].

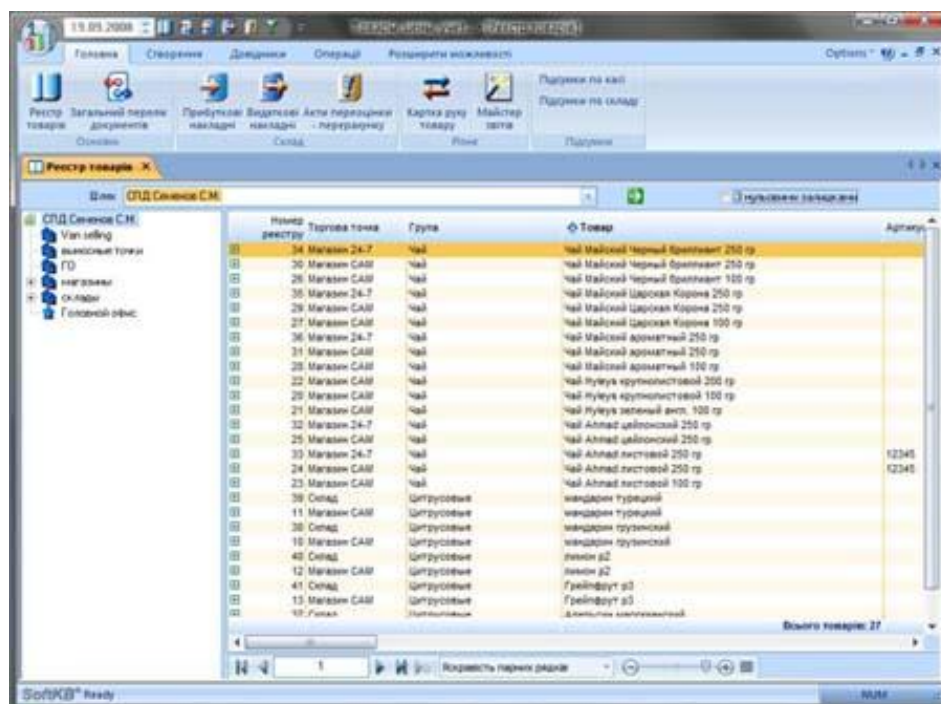


Рисунок 1.10 ТЦУ Старт

- SLS-Склад (см. рис. 1.11) -программные продукты серии SLS-Склад предназначены для автоматизации оперативного и финансового учета на предприятиях, работающих в сфере оптово-розничной торговли, а также в

сфере услуг[53].Серия состоит из 5 программ разного функционального уровня, при этом каждая программа текущего уровня содержит все функции программ более низких уровней. Самой простой является SLS-Склад Лайт, с помощью которой небольшая организация может вести оперативный учет товаров и услуг на складе, а наиболее сложной - программа SLS-Склад Супер, обеспечивающая полноценный контроль товарооборота, документооборота, учет денежных средств, анализ финансово-хозяйственной деятельности предприятия и многое другое.

Первая программа SLS-Склад вышла в 1993 году. В 1997 компания SoftLandSystems, разработчик программы, была признана победителем VI Международного конкурса "Бизнес-Софт 97" в классе «Торговый Склад». В 1998 году программа награждена медалью "Лауреат ВВЦ" в номинации "Управление". Впоследствии программа модифицировалась и расширялась, в результате превратившись в целую серию программ. Серия SLS-Склад выгодно отличается от аналогичных систем:

- простым и удобным интерфейсом;
- компактностью и надежностью базы данных;
- высоким быстродействием.

Существует пять различных по цене и функциональным возможностям вариантов программы SLS-Склад [53] :

- SLS-Склад Лайт;
- SLS-Склад Стандарт;
- SLS-Склад Мастер;
- SLS-Склад Профи;
- SLS-Склад Супер.

Рисунок 1.11 SLS-Склад

- Склад Предприниматель (см. рис. 1.12) - программа предназначена для упрощения складского учета в магазине, имеет очень простой и понятный интерфейс пользователя[37]. Среди отличий программы "СкладПредприниматель" можно выделить следующие функции:

- поступление, списание и отпуск товара;
- учет расхода по кассе;
- переоценка товара;
- справочник фактур, поддерживающий экспорт в Excel;
- установка остатков;
- отчеты;
- уценка и наценка товара;
- расходы по кассе;
- получение товара;
- списание товара;
- долги по всем покупателям и на сегодняшний день.

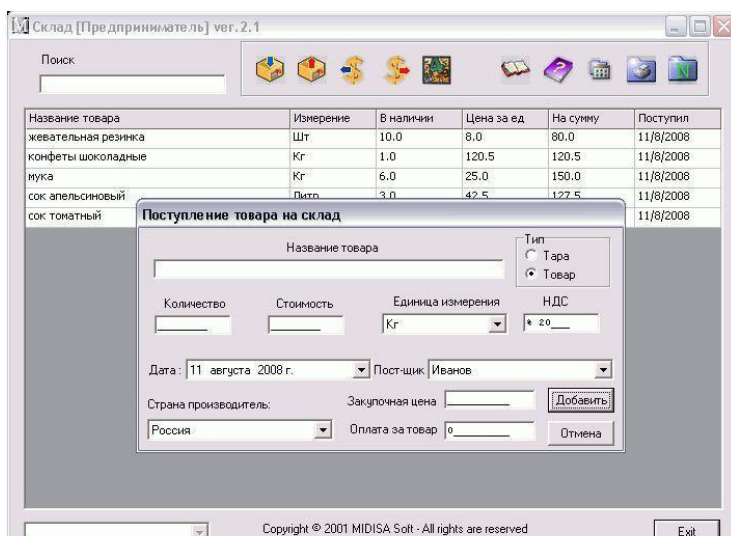


Рисунок 1.12 Склад Предприниматель

Из перечисленных систем складского учёта можно выявить минимальную функциональность системы автоматизации склада.

Рассматривая тему реализации системы складского учёта можно увидеть все разнообразие средств разработки, которое представляется разработчикам сегодня. Среди них языки программирования (1C, C#, PHP, Python, Java, Swift, Objective-C), такие базы данных как BorlandInterBase 7.5, NET, SQLite, фреймворки (CoreData, Symphony, RubyonRails, Spring MVC, Django) и большая часть из них предоставляется бесплатно в виде open-source проектов под свободными лицензиями.

1.2 Анализ деятельности компании

Компания ООО «ФИНЧ», является поставщиком услуг по разработке мобильных приложений, сайтов, социальных сетей и решений для бизнеса начиная с 2010 года.

За 8 лет компания ООО «ФИНЧ» разработала около 300 проектов для , примерно, тридцати клиентов, среди которых: ТНТ, СТС, ВГТРК, МТРК Мир, ТВ3-LiteСтолото, банк Открытие, ФК Спартак, ГК Независимость и многие другие. Одним из сильных направлений бизнеса компании сегодня является акцент на качество предоставляемых услуг и долгосрочные отношения с клиентом[22].

В части разработки мобильных приложений активно применяются новые технологии, команды разработчиков пристально следят за качеством написания кода и выпускаемым продуктом. Вне рабочего времени в ФИНЧ проводятся митапы и встречи на которых разработчики делятся знаниями и рассказывает о том, что появилось нового в мире IT. Суммарные показатели компании по проектам за последний год: более 60 миллионов пользователей, 2 миллиона установок приложений, 3 миллиарда рублей обработанных платежей.

Компания ООО «ФИНЧ» разработала множество интересных и сложных проектов, каждый из которых по своему уникален и тщательно спроектирован.

Приложение TB3-Lite - простой, быстрый и безопасный способ прикоснуться своими руками к таинственному и опасному миру снов, магии и паранормального.

В рамках проекта TNTClub был создан сайт и разработаны мобильные приложения для IOS, Android и WindowsPhone.

Проект Столото является одним из самых больших в компании, в рамках которого было создано большая backend составляющая, приложение под iOS и Android, которые активно поддерживаются и развиваются.

Сайт проекта Дом 2 был создан в стенах компании ФИНЧ и продолжает активно развиваться.

На данный момент ООО «ФИНЧ» активно развивается, увеличивается число сотрудников и рабочих мест, что влияет на увеличение количества техники и оборудования компании, учёт которых ведётся в онлайн-сервисе KeepTeam, что, в свою очередь, ведёт к усложнению процесса инвентаризации, поиска, учёта соответствующего оборудования.

Keepteam - это онлайн-сервис для управления и подбора персонала, быстрого оформления необходимых кадровых документов, согласования отпусков, рабочих графиков, а также учёта инвентаря и других задач [50].

Возможности Keepteam [50]:

- хранение рабочей и личной информации о сотрудниках;
- лента событий компании;
- управление вакансиями и соискателями;
- автоматический сбор откликов и импорт данных с рабочих сайтов;
- воронка подбора персонала;
- график отпусков и других отсутствий;
- конструктор отчётов и дашборды;
- учёт выданного инвентаря;
- навыки и цели (KPI) для сотрудников;
- хранение копий документов и заявлений;
- настройка прав доступа;

- инструменты для совместной работы и HR-самообслуживания.

Сервис автоматизирует рутинные кадровые процессы и экономит время сотрудников для решения значимых задач.

Из-за ограниченного функционала доступного для ведения складского учёта возникают сложности во время поиска соответствующих позиций и проведение инвентаризации, данные проблемы влекут за собой большое количество рабочих часов потраченное на их решение.

В сервисе KeepTeam присутствует раздел «Инвентарь» (см. рис. 1.13).

Название	Идентификатор	Принято	Категория	Ответственный
<input type="checkbox"/> Мышь компьютерная	8847	07.07.2015	Оборудование	Кабанов Ефим Бизнес-аналитик
<input type="checkbox"/> стул	4522	03.07.2014	Мебель	Кабанов Ефим Бизнес-аналитик
<input type="checkbox"/> Стол	4534	03.07.2014	Мебель	Кабанов Ефим Бизнес-аналитик
<input type="checkbox"/> Пропуск	56	03.07.2014	Пропуск	Кабанов Ефим Бизнес-аналитик
<input type="checkbox"/> стул				
<input type="checkbox"/> стол				
<input type="checkbox"/> Своя тарелка	4439671	10.07.2014	Корпоративная карта	Кабанов Ефим Бизнес-аналитик
<input type="checkbox"/> Фитнес	45456646544	30.03.2015	Корпоративная карта	Пупин Владимир Бухгалтер
<input type="checkbox"/> Карточка сотрудника	0100807	03.04.2015	Пропуск	Белякова Александра Руководитель отдела кадров
<input type="checkbox"/> LG	45EF22	01.04.2015	Монитор	Антонов Аристарх Маркетолог
<input type="checkbox"/> Стол	45645	12.03.2014	Мебель	Борисов Якуп Менеджер проекта
<input type="checkbox"/> Стол компьютерный	6748	10.09.2014	Мебель	Белякова Александра Руководитель отдела кадров
<input type="checkbox"/> Переводчик	567	01.08.2011	Онлайн сервис	Иванова Дария Ведущий разработчик
<input type="checkbox"/> Настольная лампа	7598	01.05.2011	Мебель	Иванова Дария Ведущий разработчик

Рисунок 1.13 Раздел «Инвентарь»

В данном разделе есть возможность добавления инвентаря, присваивание ему категории и ответственного. Так же присутствует возможность сортировки списка по нескольким критериям, таким как время, сотрудник, категории.

1.3 Описание и выбор инструментальных средств для реализации проекта

На сегодняшний день очень популярен рынок мобильных устройств, так как компания ООО «ФИНЧ» акцентирует внимание на разработке мобильных приложений и каждый сотрудник имеет доступ к мобильному устройству, то актуально будет разработать Систему складского

учёта для мобильного устройства, которая будет дополнять уже существующую систему KeerTeam.

Наиболее оптимальной операционной системой является iOS, ввиду простоты разработки в программе XCode и высокого уровня оптимизации системы.

iOS - мобильная операционная система для смартфонов, электронных планшетов, носимых проигрывателей и некоторых других устройств, разрабатываемая и выпускаемая американской компанией Apple. Была выпущена в 2007 году; первоначально - для iPhone и iPodtouch, позже - для таких устройств, как iPad и Apple TV. В 2014 году появилась поддержка автомобильных мультимедийных систем AppleCarPlay. В отличие от WindowsPhone (Microsoft) и Android (Google), выпускается только для устройств, производимых фирмой Apple. Данная информация получена с официального ресурса корпорации Apple[46].

Xcode — интегрированная среда разработки (IDE) программного обеспечения для платформ macOS, iOS, watchOS и tvOS, разработанная корпорацией Apple. Стабильные версии распространяются бесплатно через MacAppStore. Пакет Xcode включает в себя изменённую версию свободного набора компиляторов GNUCompilerCollection и поддерживает языки C, C++, Objective-C, Objective-C++ (англ.)русск., Swift, Java, AppleScript, Python и Ruby с различными моделями программирования, включая (но не ограничиваясь) Cocoa, Carbon и Java. Сторонними разработчиками реализована поддержка GNUPascal, FreePascal, Ada, C#, Perl, Haskell и D. Пакет Xcode использует GDB в качестве back-end'a для своего отладчика. Данная информация получена с официального ресурса корпорации Apple[46].

Для iOS приложения можно использовать разные базы данных, но самым оптимальным решением является SQLite. Работать с этой базой данных позволяет бесплатный фреймворк от компании Apple под названием CoreData, он позволяет разработчику взаимодействовать с базой данных.

SQLite – это компактная встраиваемая СУБД. Слово «встраиваемый» (embedded) означает, что SQLite не использует парадигму клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а представляет собой библиотеку, с которой программа компонуется, и движок становится составной частью программы. Таким образом, в качестве протокола обмена используются вызовы функций (API) библиотеки SQLite. Такой подход уменьшает накладные расходы, время отклика и упрощает программу. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором выполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; ACID-функции достигаются в том числе за счёт создания файла журнала [54].

Многопоточность позволяет избежать проблем при чтении данных с одной базы при условии, что никаких других запросов в данный момент не обслуживается иначе запрос завершится с ошибкой и программа вернёт код ошибки. Так же есть другой вариант получить доступ к данным – это автоматическое повторение попыток записи в течение заданного интервала времени.

Возможности SQLite [54].

1. Поддерживает большую часть SQL92, включая триггеры и транзакции. Существует система отката, защищающая целостность данных
2. Файлы БД могут совместно использоваться между машинами с различными запросами
3. Поддерживаются БД до 2 терабайт, занимая при этом небольшую часть памяти
4. Открытые исходные коды
5. Транзакции атомарны, последовательны, изолированы и долговечны (ACID) даже после сбоев системы и сбоев питания.

6. Нулевая конфигурация - не требуется настройка или администрирование.
7. Реализует большую часть SQL92. (Функции не поддерживаются)
Полная база данных хранится в одном файле межплатформенного диска.
Поддержка терабайтных размеров баз данных и Gigabyte.
8. Компактность кода: меньше, чем 300KiB полностью или менее 180KiB с дополнительными функциями.
9. Быстрее, чем популярные клиент-серверные ядра баз данных для большинства распространенных операций.
10. Простой, легкий в использовании API.
11. Написано в ANSI-C. TCL привязки включены.
12. Привязки для десятков других языков доступны отдельно.
13. Хорошо прокомментированный исходный код с более чем 99% тестовым покрытием.
14. Доступен в виде одного файла исходного кода ANSI-C, который можно легко перенести в другой проект.
15. Автономный: нет внешних зависимостей.
16. Кросс-платформенный: Unix (Linux и MacOSX), OS/2 и Windows (Win32 и WinCE) поддерживаются из коробки.
17. Легко портировать на другие системы.
18. Источники находятся в открытом доступе.
19. Используйте для любых целей.
20. Поставляется с автономным клиентом интерфейса командной строки (CLI), который может использоваться для администрирования баз данных SQLite.

SQLite накладывает ряд ограничений [29]:

- максимальная длина строки или BLOB поля 1 000 000 000;
- максимальное количество колонок 2 000;
- максимальная длина SQL-выражения 1000000000;
- максимальное количество таблиц в выражениях с JOIN 64;

- максимальная глубина дерева выражений 1000;
- максимальное количество аргументов функции 127;
- максимальное количество термов в объединённом выражении с SELECT 500;
- максимальная длина шаблона как аргумента операторов LIKE или GLOB 50000;
- максимальное количество символов-заменителей в одном SQL-выражении 999;
- максимальная глубина рекурсии триггеров 1000;
- максимальное количество присоединённых баз 10;
- максимальный размер страницы базы данных 65536;
- максимальное количество страниц в файле базы данных.

Для более быстрого и эффективного взаимодействия с базой данных и отправки запросов нужно использовать многопоточность. Apple предоставляет нам несколько инструментов для работы с многопоточным приложением: GCD и Operation.

GCD (GrandCentralDispatch) - фреймворк, который предоставляет и управляет очередями FIFO, в которых приложение может отправлять задачи в виде блочных объектов. Задача, отправляемая в очереди диспетчеризации, выполняется в пуле потоков, полностью управляемых системой. Нет гарантии относительно потока, на котором выполняется задача [49].

Operation - представляет собой законченную задачу и является абстрактным классом, который предоставляет потока-безопасную структуру для моделирования состояния операции, ее приоритета, зависимостей (dependencies) от других Operations и управления этой операцией [20].

Хорошим решением является использование GCD, так как данный фреймворк достаточно лёгок для использования и удовлетворяет всем нашим потребностям.

Для общения сервисом KeepTeam будет использоваться технология RESTAPI.

REST - это стиль архитектуры программного обеспечения для построения распределенных масштабируемых веб-сервисов. Данная технология стала популярной, когда она была подробно описана и представлена Роем Филдингом в его докторской диссертации под названием *Architectural Styles and the Design of Network-based Software Architectures* в 2000 году. Рой известен своими вкладами в развитие веба, в особенности HTTP. Рой выступал за использование стандартных HTTP методов так, чтобы придавать запросам определённый смысл [52].

Таким образом, данные HTTP-запросы будут иметь различный смысловую нагрузку в REST:

- GET /object/list;
- POST /object/list;
- PUT /object/list.

API - описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой. Обычно входит в описание какого-либо интернет-протокола (например, RFC), программного каркаса (фреймворка) или стандарта вызовов функций операционной системы. Часто реализуется отдельной программной библиотекой или сервисом операционной системы. Используется программистами при написании всевозможных приложений [52].

Для разработки приложения для iOS существует два языка программирования Swift и Objective-C, но наиболее популярным и актуальным, является Swift версии 4.2.

Swift- открытый мультипарадигмальный компилируемый язык программирования общего назначения (см. рис. 1.14) [46].



Рисунок 1.14 Язык программирования Swift

Создан компанией Apple в первую очередь для разработчиков iOS и macOS. Swift работает с фреймворками Cocoa и CocoaTouch и совместим с основной кодовой базой Apple, написанной на Objective-C. Разработка текущего варианта языка Swift началась в 2010 году Крисом Латтнером, руководителем отдела разработки инструментов для создания программного обеспечения Apple и одним из основных разработчиков LLVM. Swift заимствовал идеи из «Objective-C, Rust, Haskell, Ruby, Python, C#, CLU, и ещё из стольких многих языков, что сложно перечислить» [46]. Swift бесплатно доступен для использования разработчиками, преподавателями и студентами по лицензии на распространение ПО с открытым исходным кодом Apache 2.0.

Для упрощения процесса разработки можно использовать сторонние внешние библиотеки загружаемые в проект с помощью CocoaPods.

CocoaPods - это менеджер зависимостей уровня приложений для Objective-C, Swift и любых других языков, которые работают во время выполнения Objective-C, например RubyMotion, который предоставляет стандартный формат для управления внешними библиотеками (см. рис 1.15)[47].



Рисунок 1.15 Менеджер зависимостей CocoaPods

CocoaPods помогает решить ряд проблем, вызванных копированием кода библиотек в проект, это является весомым доводом для использования этого менеджера.

Стоит определиться с выбранными зависимостями для проекта, для этой задачи подходит сервис [Cocoapods.org](https://cocoapods.org), в котором можно выполнять поиск среди всех существующих зависимостей. Такие зависимости как HUD, QRCodeReader.swift, Fabric, Alomfire, Charts должны обеспечить минимальный функционал и позволить сосредоточиться на проекте.

Fabric - обеспечивает простой способ управления всеми SDK, что позволяет не беспокоиться о утомительных конфигурациях или манипуляциями различными учетными записями.

HUD – позволяет легко и быстро показывать индикатор загрузки и успеха из любой части приложения.

Alomfire- библиотека реализующая лёгкое взаимодействие с сетью.

QRCodeReader.swift - это простой считыватель кода iOS в Swift. Он основан на AVFoundation от Apple, чтобы заменить ZXing или ZBar для iOS 8.0 и более. Он может декодировать MachineObject. QRCodeReader.swift предоставляет контроллер представления по умолчанию для отображения вида камеры с наложением области сканирования, а также предоставляет кнопку для переключения между передней и задней камерами.

Charts – библиотека используемая для создания разного типа графиков и диаграмм.

Все эти зависимости окажут большую помощь в разработке и помогут в реализации проекта.

Разработку можно распространять как с открытым исходным кодом, так и закрытым в магазине приложений AppStore на бесплатной основе.

Для написания чистого, аккуратного, легко поддерживаемого кода следует выбрать архитектуру, которую в дальнейшем нужно будет использовать во всём проекте. Самым подходящим решением является SOA (Сервис-ориентированная архитектура, см. рис. 1.16).

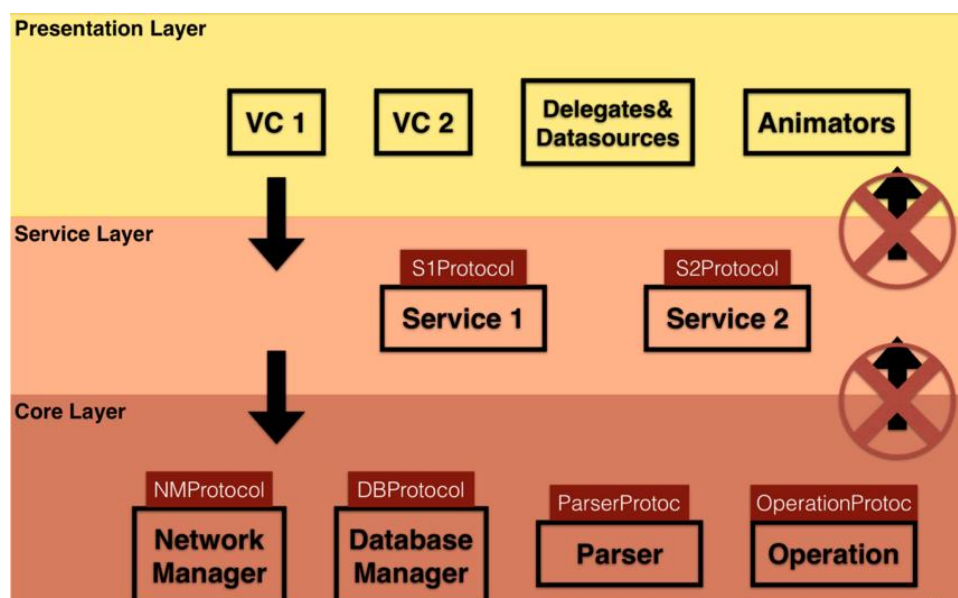


Рисунок 1.16 Сервис-ориентированная архитектура

Сервис-ориентированная архитектура - модульный подход к разработке программного обеспечения, основанный на использовании распределённых, слабо связанных заменяемых компонентов, оснащённых стандартизированными интерфейсами для взаимодействия по стандартизированным протоколам. Программные комплексы, разработанные в соответствии с сервис-ориентированной архитектурой, обычно реализуются как набор веб-служб, взаимодействующих по протоколу SOAP, но существуют и другие реализации (например, на базе jini, CORBA, на основе REST). Интерфейсы компонентов в сервис-ориентированной архитектуре инкапсулируют детали реализации (операционную систему, платформу, язык программирования) от остальных компонентов, таким образом обеспечивая комбинирование и многократное использование компонентов для построения сложных распределённых программных комплексов, обеспечивая независимость от используемых платформ и инструментов разработки, способствуя масштабируемости и управляемости создаваемых систем [12].

Выводы по главе 1

Был проведен анализ предметной области, который показал актуальность осуществления разработки с использованием выбранного инструментария.

Исходя из проведенного анализа существующих инструментальных средств для разработки были выбраны следующие компоненты.

- Открытый мультипарадигмальный компилируемый язык программирования Swift.
- Интегрированная среда разработки XCode.
- Компактная встраиваемая СУБД SQLite.
- Менеджер зависимостей уровня приложений CocoaPods.

Данный выбор вполне очевиден, т.к. это стандартный и в тоже время популярный набор инструментальных средств разработки большинства iOS приложений, и прослеживается выгода - быстрая интеграция в компанию, лёгкость в использовании, достаточно мощная и отказоустойчивая, легко поддерживаемая система.

Минимальная функциональность системы автоматизации склада должна будет включать в себя:

- инструменты для обеспечения добавления/удаления/редактирования инвентаря;
- поиск и функционал для проведение ревизии;
- поддержка чтения QR-Кодов и вывод сводной отчётности.

Главным преимуществом автоматизации складского учёта станет сокращение количества хранимых данных, из этого следует уменьшение затрат на содержание предприятия, устранение возможности возникновения ошибки при хранении информации в разных местах, устранение затрат на обновление большого количества копий.

Ещё одним достоинством станет уменьшение большого количества промежуточных файлов, папок, документов, а так же ощутимое сокращение затраченного времени на поиск информации, который производится в специальном окне программы, где есть возможность указывать критерии поиска или при использовании специальных QR-кодов.

Есть большое количество возможностей для реализаций системы складского учёта, например, программа для персонального компьютера, мобильное приложение или программа на облачном хранилище.

Если говорить о разработке системы складского учёта, то стоит принимать во внимание современность и ясность интерфейса для пользователя. Если разрабатывать доступный и бесплатный продукт, то нужно учитывать, что пользователь может быть не ознакомлен с системами складского учёта, соответственно интерфейс должен быть интуитивен. Пользователь должен иметь возможность манипулировать создавать/удалять складское помещение, добавлять/убирать те или иные необходимые позиции.

Вся хранимая и отображаемая информация находится на удалённом сервисе KeepTeam, взаимодействие с которым происходит посредством API.

2.

Проектирование информационных моделей и разработка системы складского учёта на примере компании ООО «ФИНЧ»

2.1 Выбор методов и средств проектирования

Дизайн программного обеспечения - это процесс выбора и документирования наиболее эффективных элементов, которые в совокупности будут реализовать требования к программной системе. Дизайн определяет специфический, логический подход к удовлетворению программных требований.

Дизайн программного обеспечения обычно включает в себя решение проблем и планирование программного решения. Это включает в себя как низкоуровневый компонент и алгоритм проектирования, так и высокоуровневый архитектурный дизайн.

Все элементы интерфейса для iOS приложения обеспечивает стандартная библиотека UIKit.

UIKit - платформа UIKit предоставляет необходимую инфраструктуру для приложений iOS или tvOS. Он предоставляет архитектуру окна и представления для реализации вашего интерфейса, инфраструктуру обработки событий для доставки мультитач и других типов ввода в приложение, а также основной цикл выполнения, необходимый для управления взаимодействиями между пользователем, системой и приложением. Другие функции, предлагаемые платформой, включают поддержку анимации, поддержку документов, поддержку рисования и печати, информацию о текущем устройстве, управление текстом и отображением, поддержку поиска, поддержку специальных возможностей, поддержку расширений приложений и управление ресурсами.

Проектирование ПО - процесс определения архитектуры, компонентов, интерфейсов и других характеристик системы.

Для реализации процесса проектирования существуют различные нотации такие как блок-схемы, UML-диаграммы, ER-диаграммы, а так же макеты.

Информационная модель- модель объекта, представленная в виде информации, описывающей существенные для данного рассмотрения параметры и переменные величины объекта, связи между ними, входы и выходы объекта, и позволяющая путем подачи на модель информации об изменениях входных величин моделировать возможные состояния объекта.

Разработка информационной модели объекта управления включает определение: какую информацию (или информацию о чем) выдавать человеку и в какой форме (или в каком виде).

Для решения первого вопроса, прежде всего, необходимо знать, какие сведения об управляемом объекте являются существенными с точки зрения задачи управления и предполагаемых способов ее решения. Если это неизвестно, то вопрос решается просто в принципиальном отношении, но сложно в отношении технического обеспечения: оператору надо представить всю возможную информацию об объекте, стремясь максимально приблизить информационную модель к натуре. Во всех остальных случаях проектирование информационной модели должно определяться структурой потока информации в системе управления, которая включает в себя и психофизиологическую структуру деятельности человека-оператора.

Для решения второго вопроса (о форме представления информации) необходимо знать, с каким материалом в феноменальной (психической) сфере удобнее всего человеку решать задачу управления. Например, для общей ориентации летчика в пространстве информацию о курсе лучше всего выдавать в наглядной форме. Если же требуется производить навигационные вычисления, та же информация должна быть выдана в цифровом виде.

Блок-схема - распространенный тип схем (описывающих алгоритмы или процессы, в которых отдельные шаги изображаются в виде блоков различной формы, соединенных между собой линиями, указывающими направление последовательности[15].

Данный вид схемы помогает составить и визуализировать алгоритм, что поможет в разработке архитектуры информационной системы.

UML (унифицированный язык моделирования) - это универсальный, развивающийся язык моделирования в области разработки программного обеспечения, который призван обеспечить стандартный способ визуализации дизайна системы [58].

Первоначально, создание UML было мотивировано желанием стандартизировать разрозненные нотационные системы и подходы к разработке программного обеспечения, разработанные ГрэдиБучем, Иваром Джекобсоном и Джеймсом Рамбо в RationalSoftware в 1994-1995 годах, с дальнейшим развитием, проведенным ими в 1996 году [58].

В 1997 году UML был принят в качестве стандарта Группой управления объектами (OMG), и до сих пор ею руководит эта организация. В 2005 году UML был также опубликован Международной организацией по стандартизации (ISO) в качестве утвержденного стандарта ISO. С тех пор он периодически пересматривался, чтобы охватить последний выпуск UML. UML предлагает способ визуализации архитектурных чертежей системы на диаграмме, включая такие элементы, как:

- любые виды деятельности;
- отдельные компоненты системы;
- их взаимодействие с другими программными компонентами;
- как система будет работать;
- как объекты взаимодействуют с другими (компоненты и интерфейсы);
- внешний пользовательский интерфейс.

Хотя первоначально он был предназначен для объектно-ориентированной проектной документации, UML был расширен до более широкого набора проектной документации (как указано выше) и был признан полезным во многих контекстах.

2.2 Логическое проектирование системы

В соответствие с поставленной задачей нужно провести проектировку функциональной модели, результат изображён на рисунке 2.3.

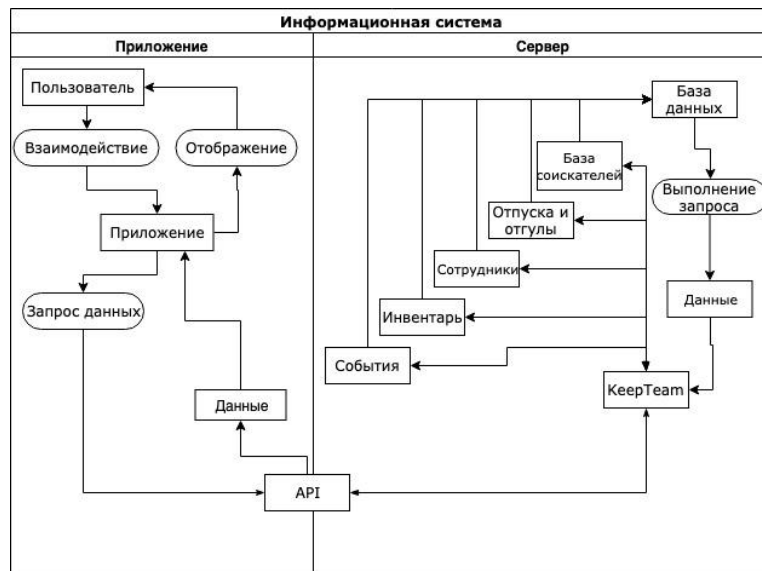


Рисунок 2.2 Функциональная модель приложения

Данная диаграмма выполнена спомощью унифицированного языка моделированияUML. Прямоугольником представлены сущности, выполняемыефункции представлены в качестве овалов.

2.3 Физическое проектирование системы

При первом запуске приложения инициализируется один склад, под названием ФИНЧ т.к. в KeepTeam нет фильтрации инвентаря в соответствие складу. По переходу на экран склада загружается информацияо существующем в нём инвентаре. У инвентаря есть возможность редактирования и удаления из текузего склада, каждому из которых можно создать уникальный QR-код.

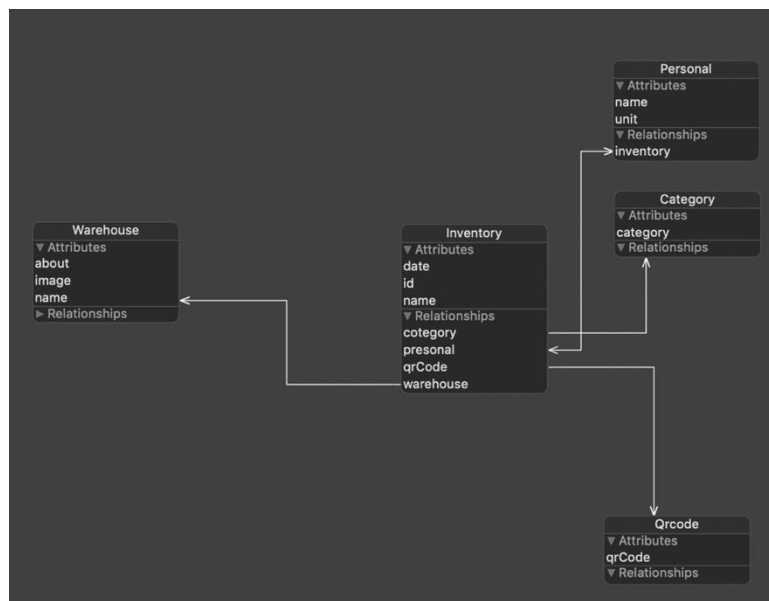


Рисунок 2.3 Схема используемой БД

2.4 Проектирование макета пользовательского интерфейса

Для выполнения проектирования пользовательского интерфейса стоит выделить ключевые действия, доступные пользователю. В рамках приложения это будет:

- просмотр инвентаря,
- редактирование инвентаря,
- добавление инвентаря,
- создание отчётов.

Данные действия предусмотрены в рамках сервиса KeerTeam, но реализация пользовательского интерфейса будет адаптирована под специфику операционной системы iOS. Действие пользователя «Просмотр инвентаря» может быть выполнено в соответствии с определёнными функциями, такими как: поиск, конфигурирование фильтра. Таким образом пользователь может получить список инвентаря с параметрами заданными в поиске (поиск производится по полю «Название») или фильтре.

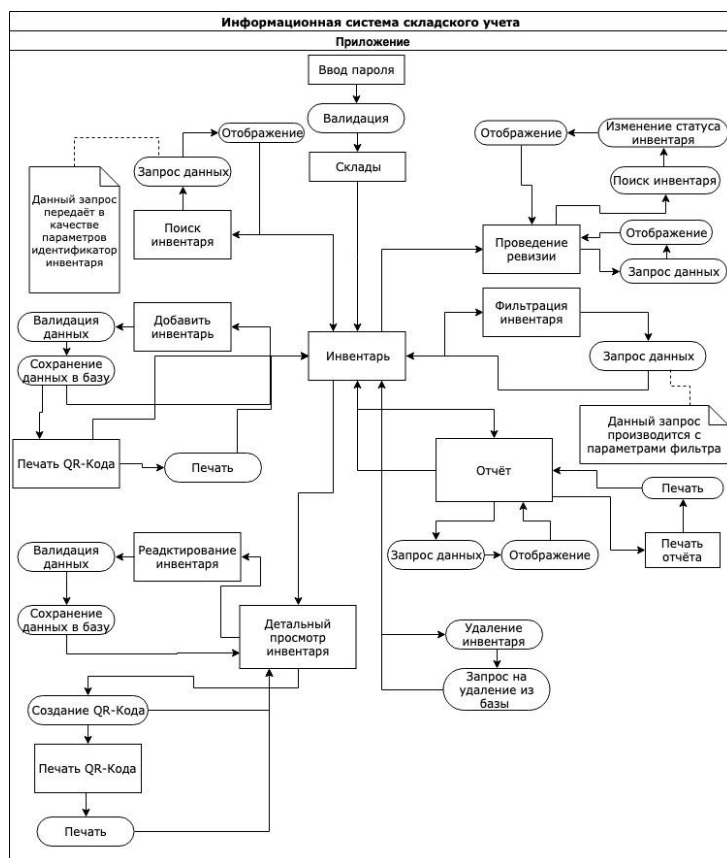


Рисунок 2.4 Схема пользовательского интерфейса информационной системы
складского учёта

Склады

Склад ООО "Finch"

Рисунок 2.5 Макет главного экрана

Инвентарь	Добавить	Отчёт	Фильтр
Строка поиска по имени			
Таблица инвентаря			

Рисунок 2.6 Макет экрана инвентаря

Название	Сохранение
Идентификатор	
Принято	
Категория	
Ответственный	
<div>QR-Code</div>	

Рисунок 2.7 Макет экрана детального просмотра и редактирования инвентаря

Добавить инвентарь	Сохранение
Добавить	
Идентификатор	
Принято	
Категория	
Ответственный	
Флаг генерации QR-Кода	

Рисунок 2.8 Макет экрана добавления инвентаря

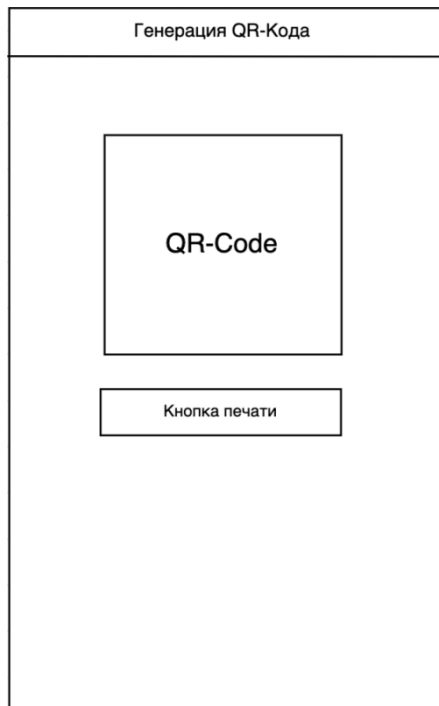


Рисунок 2.9 Макет экрана генерации QR-Кода

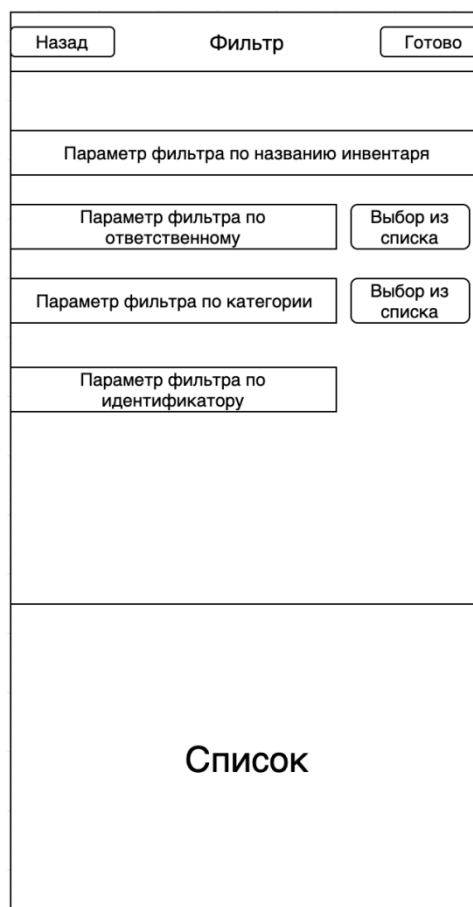


Рисунок 2.10 Макет экрана фильтра

Таблица

Текстовое поле

Учесть

Сканировать

Рисунок 2.11 Макет экрана ревизии

Печать

Рисунок 2.12 Макет экрана отчёта

2.5 Описание конечного вида интерфейса

На экране «Ввод пароля»у пользователя запрашивается пароль, предоставляется возможность его изменения и возможность авторизоваться

используя биометрические данные (см. рис. 2.13). Чтение биометрических данных происходит с использованием двух технологий: FaceID, TouchID. Используемая технология зависит от устройства. В случае ошибки, допущенной пользователем во время ввода пароля, отображается соответствующий текст. При вводе последнего символа, автоматически происходит считывание пароля. В случае, если пользователь разрешил использование биометрических данных и создал пароль, то запрос на сканирование будет появляться при запуске приложения. Создать новый пароль можно только после нажатие на кнопку «Создать новый пароль» и успешного ввода количества сотрудников добавленных в KeepTeam.

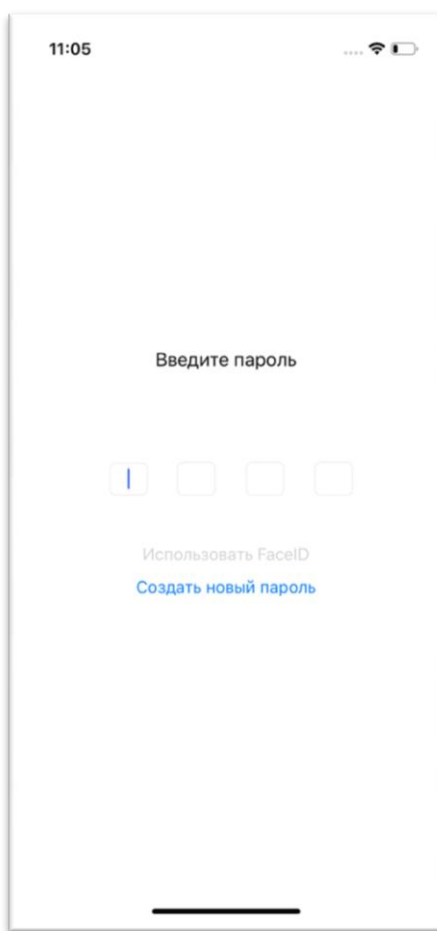


Рисунок 2.13 Вид интерфейса экрана «Ввод пароля»



Рисунок 2.14 Вид интерфейса экрана «Инвентарь»

Экран «Инвентарь» отображает весь список инвентаря компании ООО «ФИНЧ» (см. рис. 2.14), на нём можно детально просмотреть необходимую информацию, отфильтровать список инвентаря, добавить новый инвентарь, выгрузить сводную статистику, провести ревизию и отсканировать QR-Код.

Метод конфигурирования экрана инвентарь:

```
private func drawSelf() {
    title = "Инвентарь"

    let addItem = UIBarButtonItem(barButtonSystemItem: .add, target: self,
    action: #selector(addAction))

    let reportItem = UIBarButtonItem(image: #imageLiteral(resourceName:
    "report"), style: .plain, target: self, action: nil)

    let filterItem = UIBarButtonItem(image: #imageLiteral(resourceName:
    "filter"), style: .plain, target: self, action: nil)

    tableView.sectionHeaderHeight = 30

    navigationItem.largeTitleDisplayMode = .never

    navigationItem.rightBarButtonItemItems = [filterItem, reportItem, addItem]
}
```

При нажатии на кнопку добавления инвентаря, открывается экран «Добавление инвентаря» (см. рис 2.15).

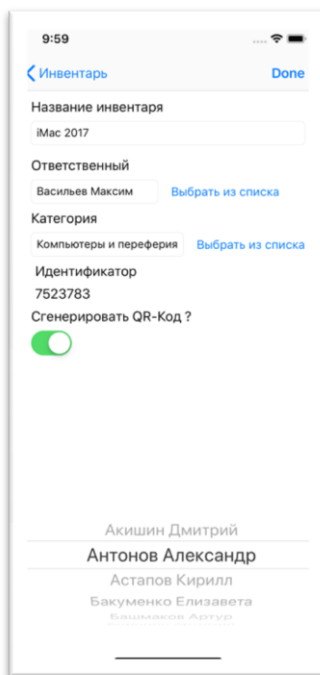


Рисунок 2.15 Добавление инвентаря

Экран «Добавления инвентаря» позволяет пользователю указать название, ответственного и категорию инвентаря, а так же согласиться или отказаться от генерации QR-Кода после добавления.

Метод сохранения инвентаря:

```
@objcprivatefunc saveAction() {  
    if nameField.text == "" || fio.text == "" || category.text == "" {  
        self.errorNotice("Незаполнено", autoClear: true) return  
    }  
    self.pleaseWait()  
    guard let name = nameField.text, let category = category.text, let  
        employee = fio.text else { return }  
    if !employee.contains(employee) {  
        warehouseStorageFacade?.saveEmployee(employee: employee,  
            position: employee.count, completionClosure: { _ in }) }  
    if !categoryArray.contains(category) {
```

```

        warehouseStorageFacade?.saveCategory(category: category,
position: categoryArray.count, completionClosure: { _ in }) }
        if self.qrSwitch.isOn {
            let view =
                AppDelegate.rootAssembly.presentationAssembly.qrCod
eViewController(id: String(id), delegate: self)
            let nav = UINavigationController(rootViewController:
view)
            self.navigationController?.present(nav, animated: true)
            sleep(1)
            self.navigationController?.popViewController(animated:
true)
        } else {
            let model = InventoryModel(name: name, id: String(id),
date: Int(Date().timeIntervalSince1970), category: category,
personal: employee, qrCode: nil)
            warehouseStorageFacade?.saveInventory(model: model) {
                success in guard success else { self.errorNotice("Save
error", autoClear: true) return }
                self.clearAllNotice() self.noticeSuccess("Save success",
autoClear: true, autoClearTime: 1)
            }
            self.navigationController?.popViewController(animated: true)
        }
    }
}

```

В случае успешного завершения процесса добавления инвентаря, отображается экран печати сгенерированного QR-Кода на основе его id (см. рисунок 2.16)



Рисунок 2.16 Печать QR-Кода

Нажатие на кнопку «Печать» запускает процесс выбора формата и настройки печати QR-Кода. После успешной печати пользователь увидит соответствующее уведомление и приложение вернётся к экрану «Инвентарь».

В случае если на этапе создания была пропущена генерация QR-Кода, то повторить этот процесс можно на экране «Детальный просмотр инвентаря» (см. рис. 2.17).

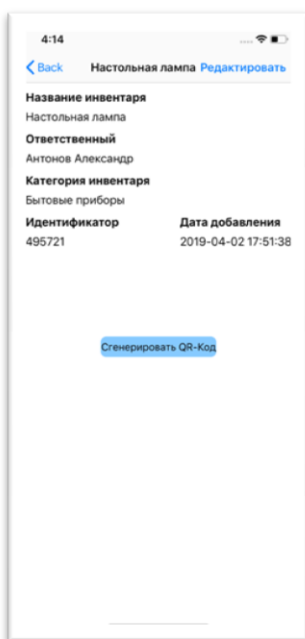


Рисунок 2.17 Детальный просмотр инвентаря

По нажатию на кнопку «Сгенерировать QR-Код» на её месте появится созданный QR-Код и кнопка печати, поведение которой будет аналогично поведению кнопки «Печать» на экране «Печать QR-Кода». Кнопка «Редактировать» ведёт на «Редактирование».

Функция генерации QR-Кода:

```
private func createQrCode() {  
    guard let id = id else { return }  
    let data = id.data(using: String.Encoding.isoLatin1,  
allowLossyConversion: false)  
    let filter = CIFilter(name: "CIQRCodeGenerator")  
    filter?.setValue(data, forKey: "inputMessage")  
    filter?.setValue("Q", forKey: "inputCorrectionLevel")  
    guard let ciImage = filter?.outputImage else { return }  
    let scaleX = qrImageView.frame.size.width /  
ciImage.extent.size.width  
    let scaleY = qrImageView.frame.size.height /  
ciImage.extent.size.height  
    let transformedImage = ciImage.transformed(by:  
CGAffineTransform(scaleX: scaleX, y: scaleY))  
    let qrCodeImage = UIImage(ciImage: transformedImage)  
    qrImageView.image = qrCodeImage  
    delegate?.qrCodeGenerated(qrCode: qrCodeImage)  
}
```

Нажатие на элемент «Фильтр» открывает экран «Параметры фильтра» (см. рис. 2.18).

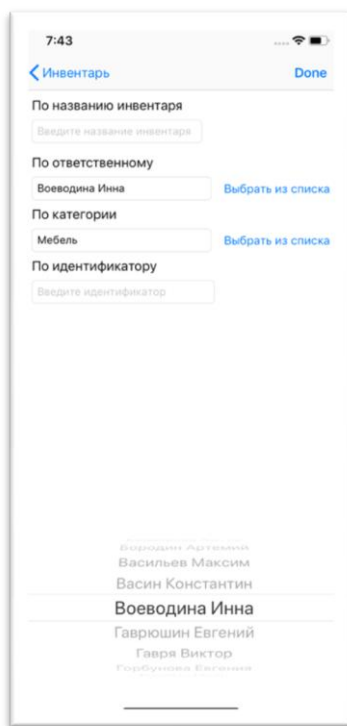


Рисунок 2.18 Параметры фильтра

В по нажатию на кнопку «Готово», создаётся экземпляр структуры Filter и передаётся в блок завершения, обработка которого происходит на экране «Инвентарь».

Экран «Редактирование» (см. рис. 2.19) инициализируется экземпляром InventoryModel, после изменения одного из текстовых полей, по нажатию на кнопку «Готово» происходит обновление данной модели в базе данных.

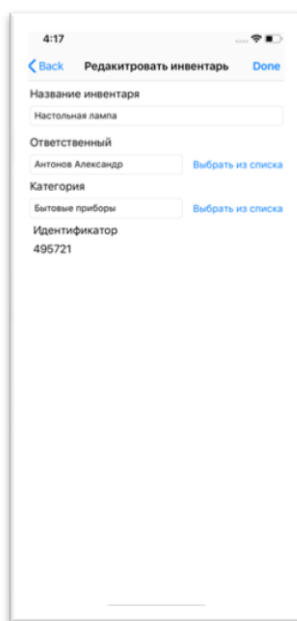


Рисунок 2.19 Экран редактирование инвентаря

После успешного обновления модели происходит возврат на экран «Детальный просмотр инвентаря».

При нажатие кнопки отчёта открывается экран «Отчёт» (см. рис. 2.20).

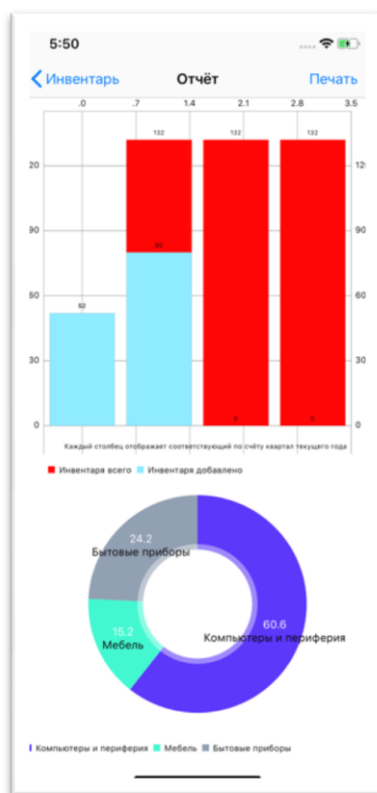


Рисунок 2.20 Отчёт

Данный экран отображает сводную статистику, размещённую на двух графиках. Первый график состоит из четырёх столбцов, которые соответствуют четырём кварталам текущего года. Столбцы голубого цвета отображают количество добавленного инвентаря в соответствующем квартале. Столбцы красного цвета отображают суммарное количество инвентаря существующего период соответствующего квартала.

Второй график содержит круговую диаграмму, сегментами которой являются категории инвентаря, а их значениями – количество инвентаря в этой категории на данный момент. Нажатие на кнопку «Печать» запускает процесс печати двух графиков.

По нажатию кнопки ревизии произведётся открытие экрана «Ревизия» (см. рис. 2.21).

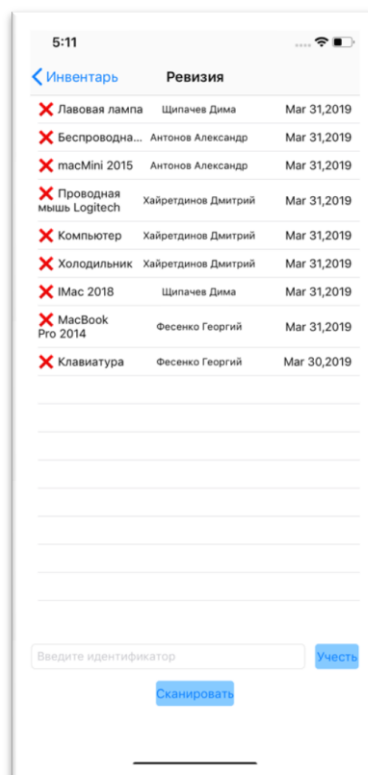


Рисунок 2.21 Ревизия

Данный экран состоит из таблицы, поля для ввода идентификатора, кнопок учёта и сканирования. Таблица содержит только тот инвентарь, у которого истекла инвентаризация. Что бы учесть нужный инвентарь нужно ввести его идентификатор в поле и нажать кнопку «Учесть» или «Сканировать», если данные будут совпадать с идентификатором инвентаря находящимся в таблице, соответствующий инвентарь пройдёт ревизию и изменит статус.

2.6 Алгоритмы работы

Алгоритм представляет систему решений и правил, которая направлена на переход от входных, допустимых данных к конечному решению. Задача алгоритма заключается в том, что бы за наименьший промежуток времени прийти к решению поставленной задачи, при этом затратив на её выполнение наименьшее количество ресурсов. В системе складского учёта присутствует множество алгоритмов, но из ключевых можно выделить алгоритм отображения инвентаря и алгоритм взаимодействия с инвентарём.

В целом алгоритм отображения инвентаря можно выразить в схеме (см. рис. 2.22):

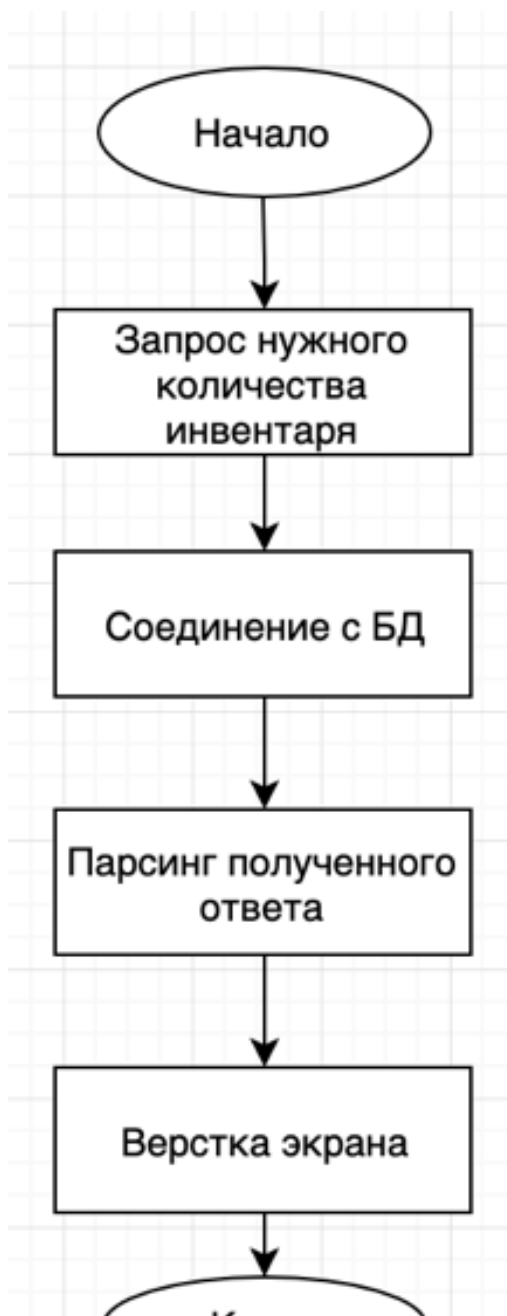


Рисунок 2.22 Алгоритм получения отображаемой информации

Результатом исполнения данного алгоритма является отображение всего набора инвентаря на экране «Инвентарь» (см. рис. 2.14). Верстка экрана происходит путём размещения полученных данных в отображаемую ячейку, все ранее заданные ограничения (Constraints) рассчитываются системой.

Алгоритм взаимодействия с информационной системой (см. рис. 2.23):

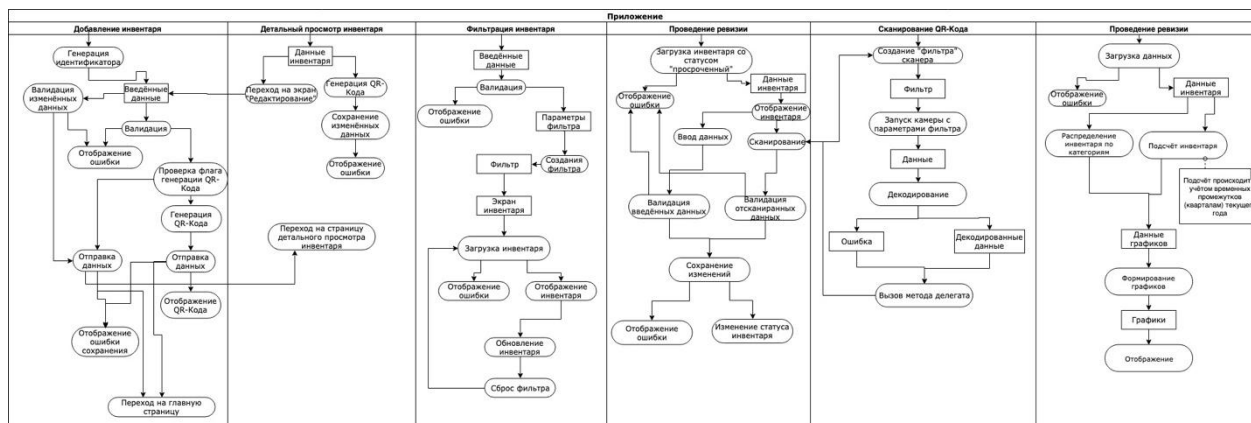


Рисунок 2.23 Алгоритм взаимодействия с информационной системой

Выводы по главе2

В рамках данной главы проделана работа по разработке макетов пользовательского интерфейса и проектированию алгоритмов работы приложения. Предполагаемый функционал соответствует всем требованиям компании ООО «ФИНЧ» и удовлетворяет современным требованиям для подобного рода разработок. Предполагается реализация таких функций как: проведение ревизии, быстрой проверки и поиска инвентаря, а так же вывод сводного отчёта, добавление и удаление инвентаря, редактирование инвентаря, добавление дополнительного склада, фильтрация инвентаря. Учтена возможность расширения текущих функций, что позволит в кратчайшие сроки улучшить существующую систему.

3 Разработка компонентов информационной системы

3.1 Описание среды разработки и программы

Среда разработки Xcode состоит из нескольких окон: окно навигации, окно утилит, зона отладки, окно редактора. Для отладки программы Xcode позволяет компилировать и запускать проект в программе simulator на любом актуальном устройстве с системой iOS, а так же на реальном устройстве.

Вся программа состоит файлов формата .swift, .xcworkspace, .plist, .xib, .storyboard. Файлы формата .swift содержит программный код, который

существует в рамках проекта, файл формата `.xcworkspace`— это сам проект, созданный на основе стандартного `.xcodeproj` файла, с подключенным набором зависимостей. Файлы формата `.plist` – словарь с набором. ключей и значений используемых для конфигурации проекта, файлы формата `.xib` – это файл содержащий `xml`код, который формирует визуальный интерфейс (см. рис. 3.1).

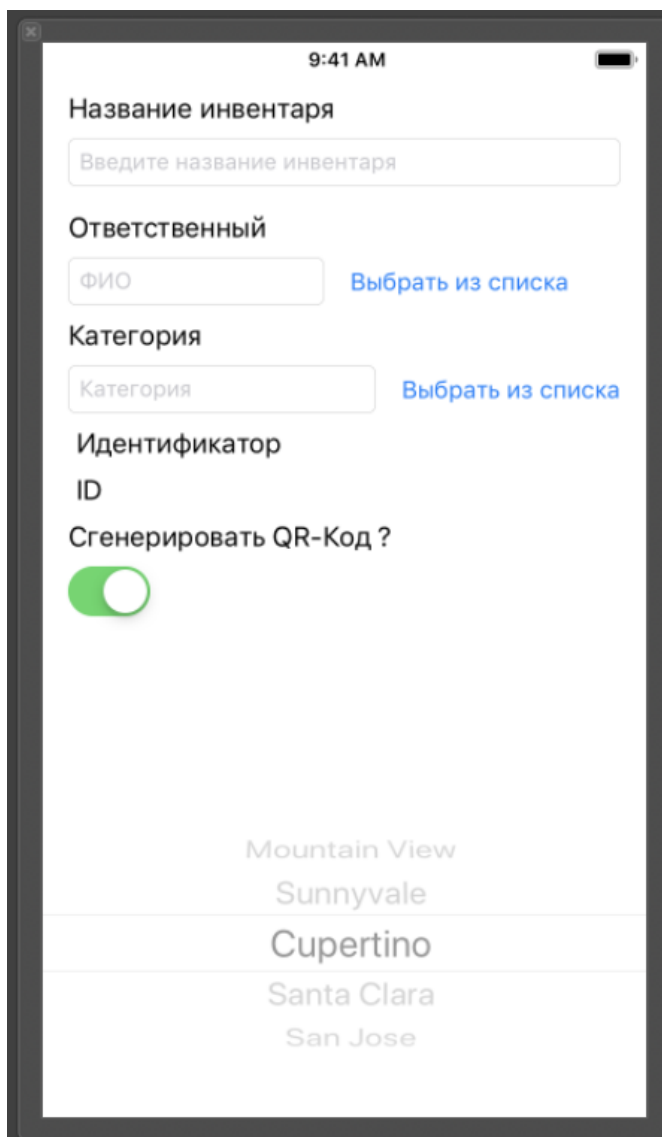


Рисунок 3.1 `xib`файл

Файл формата `.storyboard` аналогичен формату `.xib`, но их отличие в том, что `storyboard` содержит в себе несколько экранов и связи между ними (см. рис. 3.2).

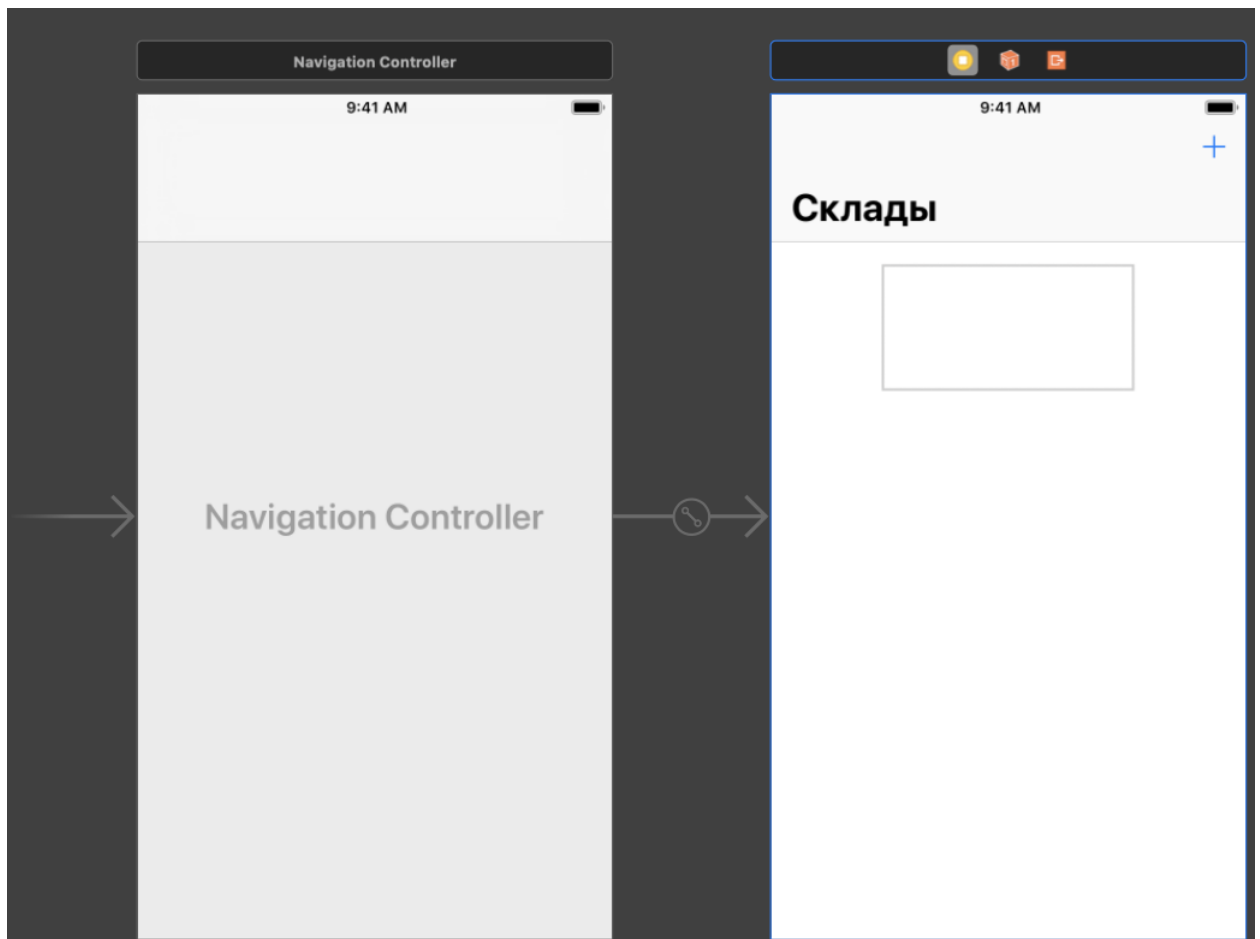


Рисунок 3.2 Storyboardфайл

В файле `pod.txt` содержится набор зависимостей, для записи используется язык Ruby.

3.2 Программная реализация информационных моделей

Для инициализации зависимостей в терминале используется команда `podinstall`.

Код файла `Pod.txt`:

```
platform :ios, '8.0'
source 'https://github.com/CocoaPods/Specs.git'

target 'Warehouse' do

  frameworks
  use_frameworks!
  pod 'HUD'
```

```

pod 'Alomfire'
pod 'Fabric'
pod 'QRCodeReader.swift', '~> 10.0.0'
end

```

В данном программном коде указывается минимальная версия системы iOS и исходная ссылка на репозиторий содержащий файлы podSpecs, а так же targetиспользуемый в проекте, далее список всех подключаемых зависимостей.

Первым экраном программы является экран «Ввод пароля». Классом экрана «Ввод пароля» является `PinCodeViewController` унаследованный от `UIViewController`, его задача заключается в создании или проверке введённого пароля. При открытии экрана проверяется существование сохранённого пароля в хранилище `Keychain` и происходит загрузка массива сотрудников. За взаимодействие с хранилищем отвечает класс `KeychainService`, который имеет методы:

- `updatePassword(data: String)`
- `removePassword()`
- `savePassword(data: String)`
- `loadPassword() ->String`

Проверка на существование пароля:

```

if let password = KeychainService.loadPassword() {
mainLabel.text = "Введите пароль"
self.existPassword = password
}

firstChar.becomeFirstResponder()

```

В случае если пароль существует, то после ввода всех символов будет осуществляться сравнение с ранее сохранённым паролем или если доступны биометрические данные, то пользователь увидит запрос на разрешение использования, в случае если запрос одобрен, то произойдёт процесс считывания. Если пользователь верно ввёл пароль или биометрические

данные соответствуют ранее добавленным в систему iOS, происходит переход на экран «Склад».

Проверка биометрических данных:

```
let policy: LAPolicy = .deviceOwnerAuthenticationWithBiometrics
var error: NSError?
let reason = "Ввод пароля"
if context.canEvaluatePolicy(policy, error: &error) {
    context.evaluatePolicy(policy, localizedReason: reason, reply: { (success,
error) in
        if success {
            DispatchQueue.main.async {
                self.performSegue(withIdentifier: "ShowMainNavigation", sender: nil)
            }
        } else {
            DispatchQueue.main.async {
                print("recognize error")
            }
        }
    })
} else {
    biometricBtn.isEnabled = false
}
```

По нажатию на кнопку «Создать новый пароль» пользователю будет предложено ввести количество сотрудников компании, к примеру, если сотрудников 32, то пользователь должен ввести 0032. В момент инициализации контроллера запускается загрузка актуального списка сотрудников, на основе которого происходит сравнение введённого числа. В случае совпадения запускается процедура создания нового пароля.

Если пользователь не создавал пароль, то ему будет предложено ввести новый пароль. После успешного ввода, введённый пароль сохраняется в переменную `firstEntered`, после ему предлагается повторить введённый пароль, после повтора пароля, новый, введённый пароль сравнивается с переменной `firstEntered`, в случае совпадения данный пароль сохраняется в хранилище с помощью метода `savePassword(data: String)` и приложение отображает экран «Склады».

Экран

«Склады» является классом `MainViewController` унаследованным от `UIViewController` и состоит из элемента `UIKit` под названием `UICollectionView`, который содержит один элемент `WarehouseCell` унаследованный от `UICollectionViewCell`. Данный класс принимает три визуальных элемента, такие как `UILabel`, `UIImageView`.

```
@IBOutlet weak var title: UILabel!  
@IBOutlet weak var about: UILabel!  
@IBOutlet weak var image: UIImageView!
```

Параметр `image` переменной `image` принимает картинка на которую применяется `blurEffect`:

```
let blurEffect = UIBlurEffect(style: UIBlurEffect.Style.dark)  
let blurEffectView = UIVisualEffectView(effect: blurEffect)  
blurEffectView.frame = image.bounds  
blurEffectView.autoresizingMask = [.flexibleWidth, .flexibleHeight]
```

Все данные для ячеек хранятся в массиве `warehouse`, количество отображаемых ячеек регулируется количеством элементов данного массива. По нажатию на ячейку данными ячейки инициализируется класс `InventoryTableViewController` унаследованный от `UIViewController`. Вся навигация между экранами управляется классом `UINavigationController`. Для отображения нового экрана нужно вызвать функцию:

```
navigationController?.pushViewController(vc, animated: true).
```

В качестве параметров данная функция принимает новый `UIViewController`, и значение типа `Bool` на основе которого принимается решение, будет ли анимирован переход между представлениями.

Экран «Инвентарь» состоит из `UITableView`, ячейки которой содержат информацию о загруженном инвентаре. Для загрузки данных используется метод:

```
private func loadData() {  
    self.pleaseWait()  
    warehouseStorageFacade?.fetchInventory() { [weak self] (inventory, error) in  
        guard let inventory = inventory, error == nil else {
```

```

        print(error ?? "")
        self?.clearAllNotice()
        return
    }

    self?.models = inventory
        self?.models.sort(by: { $0.date > $1.date })
        self?.tableView.reloadData()
        self?.clearAllNotice()
    }
}

```

В данном методе вызывается функция `fetchInventory` у экземпляра класса имплементирующего протокол `IWarehouseStorageFacade`, в блоке завершения идёт проверка на отсутствие ошибки и массиву `models` присваивается полученный результат. Результат сортируется по убыванию и запускается обновление табличного представления. Количество и данные строк таблицы регулируется количеством и данными элементов в массиве `models`. Массив `models` хранит в себе экземпляры класса `InventoryModel`.

Класс `InventoryModel`:

```

class InventoryModel {
    var name: String
    var id: String
    var date: Int
    var revisionDate: Int
    var category: String
    var personal: String
    var qrCode: String?

    init(name: String, id: String, date: Int, revisionDate: Int, category: String, personal: String, qrCode: String?) {

        self.name = name
        self.id = id
        self.date = date
        self.revisionDate = revisionDate
        self.category = category
        self.personal = personal
        self.qrCode = qrCode
    }
}

```

```

func sendModel() -> [String : Any] {
    var dict = ["name" : name as Any,
                "date" : date as Any,
                "category" : category as Any,
                "personal" : personal as Any]

    if let qrCode = qrCode {
        dict["qrCode"] = qrCode as Any
    }

    return dict
}

```

Каждая из переменных хранит в себе соответствующие названию данные, инициализируемые при создании класса. Метод `sendModel` используется для конвертации хранимых данных словарь типа `[String : Any]`.

В навигационном меню экрана «Инвентарь» присутствует несколько элементов `UINavigationController`, которые создаются в методе `drawSelf`:

```

private func drawSelf() {

    title = "Инвентарь"

    self.refreshControl = customRefreshControl

    let cameraItem = UIBarButtonItem(barButtonSystemItem: .camera, target:
self, action: #selector(showScanner))

    let addItem = UIBarButtonItem(barButtonSystemItem: .add, target:
self, action: #selector(addAction))
    let reportItem = UIBarButtonItem(image: imageLiteral(resourceName:
"report"), style: .plain, target: self, action: #selector(showReport))
    let filterItem = UIBarButtonItem(image: imageLiteral(resourceName:
"filter"), style: .plain, target: self, action: #selector(showFilter))
    let revisionItem = UIBarButtonItem(barButtonSystemItem: .bookmarks,
target: self, action: #selector(revision))

    tableView.estimatedRowHeight = 65
    tableView.sectionHeaderHeight = 30
}

```

```

navigationItem.largeTitleDisplayMode = .never
navigationItem.rightBarButtonItemItems = [cameraItem, reportItem, filterItem,
revisionItem, addItem]
}

```

Нажатие на элемент «addItem» ведёт на экран «Добавление инвентаря» классом которого является `AddInventoryViewController` унаследованный от `UIViewController`. По нажатию на кнопку подтверждения вызывается функция `saveAction`. В данной функции производится проверка на пустоту строк и отсутствие введённой категории или персонала, если что-то из введённого отсутствует в БД, то вызывается метод `saveCategory(category: , position: , completionClosure:)` или `saveEmployee(employee: , position: , completionClosure:)` соответственно. В случае если функция `isOn`, элемента `UISwitch`, возвращает `true`, выполняется переход на экран «Печать QR-Кода», классом которого является `QRCodeViewController` унаследованный от `UIViewController`. Его инициализация происходит на основе параметра `paid`. Представление данного контролера происходит модально (данный контроллер существует вне навигационного стека). В случае если метод `isOn` вернул `false`, то создаётся экземпляр `InventoryModel` и вызывается метод `saveInventory(model:, completion: (Bool) ->Void)` принимающий на вход объект типа `InventoryModel`, в замыкании проверяется объект типа `Bool`, в случае если он равен `true` отображается уведомление об успешном сохранении и происходит возврат на предыдущий контроллер в навигационном стеке, иначе отображается уведомление об ошибке.

Нажатие на элемент «cameraItem» ведёт на экран «Сканирование» классом которого является `QRCodeReaderViewController` унаследованный от `UIViewController`. Данный экран отображается модально и существует вне навигационного стека. Делегатом данного контроллера является экземпляр класса `InventoryTableViewController` унаследованный от протокола `QRCodeReaderViewControllerDelegate` и реализующий методы `reader(_ reader: QRCodeReaderViewController, didScanResult result: QRCodeReaderResult)` и `readerDidCancel(_ reader: QRCodeReaderViewController)`. В методе `reader(_`

reader: `QRCodeReaderViewController`, `didScanResult`:
`QRCodeReaderResult`) происходит проверка на существование
`InventoryModel` параметром `id`равным отсканированному числу, если такой
экземпляр существует, то отображается экран «Детальный просмотр
инвентаря» классом которогоявляется
`DetailInventoryViewController`унаследованный от `UIViewController`.
Инициализация `DetailInventoryViewController` требует объект типа
`InventoryModel`. Еслинеудётсянайтиобъекттипа`InventoryModel`спараметром
`id`равным отсканированному, показывается уведомление о ошибке. В методе
`readerDidCancel(_ reader: QRCodeReaderViewController)` происходит
остановка сканирования и закрытие экземпляра `QRCodeReaderViewController`
из представление методом `dismiss(animated:, completion:)`.Нажатие на кнопку
«Редактировать» ведёт на экран «Добавление инвентаря». Благодаря
переходу на данный экран с предварительной инициализацией объекта
`model`экран отображается без возможности генерации QR-кода и всеми
заломленными полями из модели `model`. Инициализация в зависимости от
существования `model`:

```
if let model = model {
    title = "Редактировать инвентарь"

    idLabel.text = model.id
    nameField.text = model.name
    fio.text = model.personal
    category.text = model.category
    qrSwitch.isHidden = true
    generateTitle.isHidden = true
} else {
    title = "Добавить инвентарь"
    idLabel.text = String(id)
}
```

Нажатие на элемент `reportItem` ведёт на экран «Отчёт»
классом которогоявляется `ChartViewController`унаследованный от `UIViewController`.
Данный экран отображается в навигационном стеке.

Экран инициализируется массивом моделей типа `InventoryModel` и отображает два UI-элемента:

```
@IBOutlet weak var lineChartView: PieChartView!
```

```
@IBOutlet weak var chartView: BarChartView!
```

Первый берёт данные из массива моделей `InventoryModel`, второй из словаря типа `[String:[InventoryModel]]` ключами которого является категория, а значением массив соответствующих моделей.

Нажатие на элемент «`revisionItem`» ведёт на экран «Ревизия» классом которого является `RevisionViewController` унаследованный от `UIViewController` (см. рис. 2.21). Таблица данного представления заполняется инвентарём, который нуждается в проведении ревизии. Необходимость в ревизии можно узнать из вычисляемого свойства `var isNeedRevision: Bool`. Данное свойство возвращает `true` в случае, если текущая дата и время вышла из диапазона времени того квартала, в котором находится параметр `var revisionDate: Int`. При нажатии на кнопку «Учесть» проверяется текстовое поле на существование введённых символов, в случае если символы существуют, то производится поиск и сопоставление введённых символов с идентификаторами инвентаря в таблице. Если введённый номер совпадает с идентификатором инвентаря из таблицы, то происходит перезапись поля `revisionDate` и перерасчёт поля `title` соответствующей ячейке. При нажатии кнопки «Сканировать» происходит открытие сканера QR-Кода. В методе делегата данного экрана производится сопоставление отсканированного значения и существующих идентификаторов инвентарей из таблицы.

Нажатие на элемент «`filterItem`» ведёт на экран «Параметры фильтра» классом которого является `FilterViewController` унаследованный от `UIViewController` (см. рис. 2.18). На данный класс инициализируется с блоком завершения в котором происходит фильтрация по возвращению на экран «Инвентарь». По нажатию на кнопку «готово» заполняются создаётся структура `Filter`.

Структура `Filter`:

```

struct Filter {
    let name: String?
    let personal: String?
    let category: String?
    let id: String?
}

```

Данная структура заполняется из текстовых полей представленных на экране «Параметры фильтра», если поле является пустым, соответствующая константа принимает значение nil. В блоке завершения параметры структуры проверяются на nil. Экземпляр данной структуры передаётся в блок завершения, в рамках которого происходит проверка каждого свойства данной структуры на nil, в случае, если поле не равно nil, происходит фильтрация копии массива моделей. По завершению всех проверок происходит отображение отфильтрованного массива.

Выводы по главе 3

В рамках данной главы проведена работа по созданию информационных моделей и алгоритмов системы складского учёта, а так же проверка работоспособности информационной системы посредством ручного тестирования. Разработанная информационная система предусматривает обработку ввода некорректных данных, что позволяет избежать нестабильной работы информационной системы.

4 Обоснование экономической эффективности

На разработку приложения было затрачено 30 дней, в течение которых были проделаны работы по созданию приложения, разработке архитектуры, моделированию дизайна, его тестированию и проверке в реальных условиях.

Расчет себестоимости работ проводят по тем расходам, которые были затрачены в процессе создания. Затраты определяются с помощью сложения статей калькуляции себестоимости: затраты на электроэнергию и интернет, амортизационные расходы на технику, затраты на работы, которые выполняются программистом, дизайнером, другие прямые затраты. Расчет приведем с учетом восьмичасового рабочего дня (см. табл. 1).

Таблица 1

Наименование	Единица измерения	Количество	Цена, руб.	Сумма, руб.
Электроэнергия: компьютер освещение	КВт·ч	144 19,2	3,3 3,3	$72 \cdot 3,3 = 238$ $9,6 \cdot 3,3 = 32$
Амортизационные расходы на технику	Руб.	1	35000	35000
Интернет	Безлимит	1	450	450
Зарплата программиста	Руб.	1	35000	35000
Всего				70720 руб.

Формула для расчета амортизации оборудования:

$$A = \frac{C}{T * 12},$$

где A – ежемесячная сумма износа;

C – цена приобретения компьютера;

T – период службы;

12 – количество месяцев.

Зарплата программиста была выявлена путем поиска таковых людей на разных биржах для работы и переговоров с ними с предоставлением необходимого для разработки. Приведенные результаты являются усредненными.

Заработная плата программиста за разработку приложения 35000 рублей.

Проводя анализ экономических показателей стоимости проекта, будут важны такие показатели как: разница во времени затраченном на проведение ревизии, стоимость разработки программы. В сумму стоимости разработки включены все затраты, которые будут произведены во время создания программы. Разница во времени затраченном на проведение ревизии вычисляется между средним временем затраченном на её проведение до внедрения программы и временем, затраченном на проведение после внедрения программы.

Следует учитывать, что разработанную информационную систему нужно сопровождать и поддерживать, так как сбой в системе понесет за собой материальные потери для предприятия.

Рассчитаем конечную сумму прибыли от внедрения проекта за 1 квартал.

Формула для расчета конечной суммы прибыли от внедрения проекта за 1 квартал:

$$V = S_k - S_0,$$

где V - конечная сумма прибыли от внедрения проекта;

S_k -конечная прибыль компании;

S_0 – обслуживание информационной системы за текущий период.

$$V = 1845000 \text{ руб.} - 30000 \text{ руб.} = 1815000 \text{ руб.}$$

Количество времени, затраченное сотрудником на проведение ревизии в четвёртом квартале 2018 года было равно 1ч. 45мин. Количество времени, затраченное сотрудником на проведение ревизии, после внедрения программы, в первом квартале 2019 года составило 54 минуты (см. рис. 4.1).

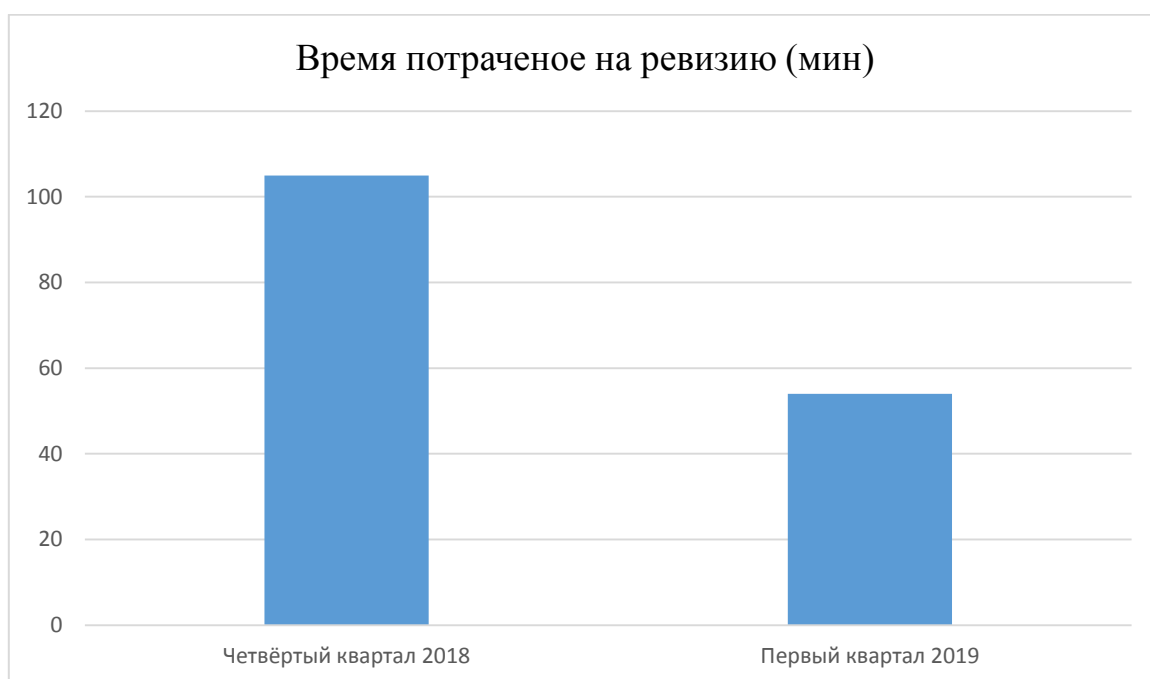


Рисунок 4.1 Время потраченное на ревизию

Таким образом, сотрудник экономит 50% от затраченного ранее времени. Взаимодействие с данной программой упрощает ведение учёта и позволяет вести учёт с нескольких устройств. Основной выгодой от внедрения информационной системы будет, ускорение проведения ревизии, быстрому формированию отчётности, более удобный и быстрый доступ к инвентарю, что позволит уменьшить количество затрачиваемого времени на инвентаризацию тем самым увеличив прибыль компании.

Выводы по главе 4

Внедрение разработанной информационной системы будет способствовать уменьшению рабочего времени затраченного на проведение ревизии и ведения складского учёта, за счет быстрого доступа к управлению инвентаря и сканеру QR-Кодов в любой момент времени с любого устройства компании.

Положительным эффектом от внедрения разработанной информационной системы в процесс деятельности компании станет ускорение проведения ревизии, возможность быстрого формирования отчётности, удобный и быстрый доступ к инвентарю, что позволит уменьшить количество затрачиваемого времени на инвентаризацию тем самым увеличив прибыль компании.

Заключение

В настоящей выпускной квалификационной работе разработана информационная система складского учёта для компании ООО «ФИНЧ», позволяющая уменьшить затраченное время на проведение ревизии и просматривать сводную статистику хранимого инвентаря.

В первой главе выпускной квалификационной работы проведён анализ предметной области и инструментальных средств для реализации проекта в рамках которого были изучены существующие информационные системы и их функциональная составляющая. Рассмотренные системы достаточно функциональны, большая часть которого может быть не использована, а так же достаточно дороги для внедрения. Произведено описание и выбор инструментальных средств для реализации проекта. В качестве среды разработки выбрана программа Xcode (язык программирования Swift), в качестве менеджера зависимостей выбрана CocoaPods, в качестве зависимостей используемых в информационной системе выбраны QRCodeReader.swift, Alamofire, HUD, Fabric, Charts. Данный набор инструментальных обеспечивает быструю и эффективную разработку.

Во второй главе разработаны информационные модели пользовательского интерфейса системы складского учёта для компании ООО «ФИНЧ» и выбраны методы и средства проектирования, сформированы алгоритмы работы и алгоритм взаимодействия с инвентарём.

В третьей главе осуществлена программная реализация информационных моделей и алгоритмов системы складского учёта с учётом возможности возникновения логических ошибок. Произведено речное тестирование всевозможных реальных сценариев.

В четвёртой главе работы рассчитаны затраты на разработку информационной системы, конечная сумма прибыли после интеграции информационной системы. Стоимость разработки 70720 рублей что включает в себя все необходимые затраты. Внедрение информационной системы позволит сократить время затраченное на проведение ревизии и выводить сводную информацию по компании.

Разработанная информационная система способна дать экономический эффект обусловленный сокращением затрачиваемого времени на косвенные обязанности сотрудника и возможностью чётче контролировать состояние инвентаря находящегося на складе благодаря сводным графиками.

Целью данной выпускной квалификационной работы являлась разработка информационной системы складского учёта для ООО «ФИНЧ». В результате работы был получен полноценное приложение соответствующее современным требованиям, которое используется в деятельности компании. Данное приложение обладает следующим функционалом:

- добавления инвентаря;
- удаление инвентаря;
- редактирование инвентаря;
- создание QR-Кода;
- чтение QR-Кода;
- фильтрация инвентаря;
- добавление категории инвентаря;
- добавление сотрудника;
- вывод отчёта.

Безопасность приложения обеспечивается входом по пин-коду, Face-Id или Touch-ID. Пароль хранится в локальном Keychain хранилище. Так же пользователь может менять пароль ответив на вопрос о текущем количестве сотрудников, добавленных в сервис KeepTeam. Безопасность хранения пароля и данных об отпечатке пальца и модели лица обеспечивается технологией DataProtection и SecureEnclave соответственно. В ходе работы, для

максимально эффективной и качественной разработки были изучены основы проектирования баз данных и информационных систем, требования безопасности, защиты конфиденциальности и разграничения доступа к ним. Были использованы основные средства, предоставляемые стандартным инструментарием, что позволило максимально четко сформировать картину необходимого функционала и максимально объективно оценить требования и работоспособность будущего приложения. При внедрении в деятельность компании данное приложение позволит повысить эффективность фирмы, так как оно ускоряет и упрощает работу с инвентарём, учётом и инвентаризацией.

Таким образом, все задачи выпускной квалификационной работы выполнены, цель достигнута.

Список использованных источников

1. ПНСТ 277-2018 - Сравнительные испытания мобильных приложений для смартфонов.
2. ГОСТ Р 51904-2002 - Программное обеспечение встроенных систем.
3. Васильев А. Е. Микроконтроллеры. Разработка встраиваемых приложений (+ CD-ROM); БХВ-Петербург - Москва, 2012. - 304 с.
4. Волгин, В. В. Логистика хранения товаров. Практическое пособие / В.В. Волгин. - М.: Дашков и Ко, 2017. - 368 с.
5. Вроблевски, Люк Сначала мобильные! / Люк Вроблевски. - М.: Манн, Иванов и Фербер, 2012. - 300 с.
6. Голицына О.Л., Попов И.И. Основы алгоритмизации и программирования; Форум - Москва, 2010. - 432 с.
7. Делессио, К. Создание приложений для Android за 24 часа / К. Делессио, Л. Дарси, Ш. Кондер. - Москва: Гостехиздат, 2015. - 528 с.
8. Емельянова Н.З., Партыка Т.Л., Попов И.И. Проектирование информационных систем; Форум - Москва, 2009. - 432 с.
9. Марк, Дэйв iOS 6 SDK. Разработка приложений для iPhone, iPad и iPodtouch / Дэйв Марк и др. - Москва: СПб. [и др.] : Питер, 2013. - 672 с.
10. Склад и логистика. - М.: Издательство Гревцова, 2016. - 360 с.
11. Усов Василий Swift. Основы разработки приложений под iOS и macOS. — СПб.: Питер, 2017. - С. 180-192.
12. Чугреев В. Л. Разработка сервис-ориентированной архитектуры в ИСЭРТ РАН - 2013. - С. 23-28.
13. Архитектура iOS приложения [Электронный ресурс]. URL: <https://habr.com/ru/post/246877/> (дата обращения: 03.02.2019).

14. Архитектура iOS приложения SOA [Электронный ресурс]. URL: https://www.redmadrobot.ru/fyi/architecture_ios (дата обращения: 05.02.2019).
15. Блок-схема [Электронный ресурс]. URL: <http://samzan.ru/32152> (дата обращения: 16.03.2019).
16. Добавление элемента UI [Электронный ресурс]. URL: <https://ru.stackoverflow.com/questions/811807/Как-добавить-в-header-кнопку-action-и-почему-то-происходит-ошибка-сохранения-кл> (дата обращения: 17.03.2019).
17. Использование CoreData [Электронный ресурс]. URL: <https://habr.com/ru/post/303512/> (дата обращения: 15.03.2019).
18. Класс365 [Электронный ресурс]. URL: <http://freeanalogs.ru/Класс365> (дата обращения 7.02.2019)
19. Лайт-учет [Электронный ресурс]. URL: <https://www.softportal.com/software-35862-lajt-uchyot.html> (дата обращения 13.02.2019)
20. Многопоточность [Электронный ресурс]. URL: <https://www.coursera.org/lecture/ios-multithreading/osnovy-operationsqueue-tIdBi> (дата обращения 5.03.2019)
21. Нововведения iOS 12 [Электронный ресурс]. URL: https://vk.com/@iphone_optom-obzor-11-glavnyh-novovvedenii-ios-122 (дата обращения: 28.03.2019).
22. ООО «ФИНЧ» [Электронный ресурс]. URL: <https://funch.fm> (дата обращения 24.02.2019)
23. Описание и работа с JSON [Электронный ресурс]. URL: <https://ruseller.com/lessons.php?id=1212> (дата обращения: 07.02.2019).
24. Организация системы складирования [Электронный ресурс]. URL: <http://diplomba.ru/work/26146> (дата обращения: 28.12.2018).
25. Организация системы складского учёта [Электронный ресурс]. URL: <https://habr.com/ru/company/trinion/blog/296718/> (дата обращения: 29.03.2019).

26. Программа «1С: Торговля и склад» – обучение и возможности [Электронный ресурс]. URL: <https://www.ekam.ru/blogs/pos/programma-1s-torgovlya-i-sklad-obuchenie> (дата обращения 7.02.2019)
27. Проектирование ПО [Электронный ресурс]. URL: <https://megaobuchalka.ru/3/11623.html> (дата обращения: 24.03.2019).
28. Простой Склад [Электронный ресурс]. URL: <https://www.softportal.com/software-35147-prostoj-sklad.html> (дата обращения 7.02.2019)
29. Работа с сетью [Электронный ресурс]. URL: <https://habr.com/ru/post/330760/> (дата обращения: 30.03.2019).
30. Работа с JSON [Электронный ресурс]. URL: <https://habr.com/ru/post/276247/> (дата обращения: 09.02.2019).
31. Работа с Storyboard [Электронный ресурс]. URL: <https://habr.com/ru/post/312766/> (дата обращения: 27.03.2019).
32. Разработка ПО [Электронный ресурс]. URL: <https://habr.com/ru/post/255991/> (дата обращения: 24.03.2019).
33. Разработка iOS приложения [Электронный ресурс]. URL: <https://myapples.ru/ios/razrabotka> (дата обращения: 19.03.2019).
34. Руководство по дизайну iOS приложения [Электронный ресурс]. URL: <http://www.blog.jazov.com/mobile-design/polnoe-rukovodstvo-po-dizajnu-dlya-ios.html> (дата обращения: 19.12.2018).
35. Система складирования [Электронный ресурс]. URL: https://studwood.ru/885048/marketing/sistema_skladirovaniya (дата обращения: 07.11.2018).
36. Склад в Excel [Электронный ресурс]. URL: <http://sklad-excel.ru/sklad-excel/> (дата обращения 7.02.2019)
37. Склад Предприниматель [Электронный ресурс]. URL: <http://www.cdmail.ru/business/bookoffice/sklad-predprinimatel.htm> (дата обращения 24.02.2019)

38. Склад производства UP [Электронный ресурс]. URL: <https://www.dso55.ru/goods/1447033/> (дата обращения 12.02.2019)
39. Создание SQLite [Электронный ресурс]. URL: <https://zametkinapolyah.ru/zametki-o-mysql/chast-2-2-pervoe-znakomstvo-s-sqlite3-sozдание-baz-dannyx-sqlite3-rasshirenie-fajlov-baz-dannyx-v-sqlite3.html> (дата обращения: 24.03.2019).
40. Создание UML диаграмм [Электронный ресурс]. URL: <http://msugvnua000.web710.discountasp.net/Posts/Details/3787> (дата обращения: 13.03.2019).
41. Среда разработки Xcode [Электронный ресурс]. URL: <http://www.tryobj.com/87-soft-xcode.html> (дата обращения: 17.11.2018).
42. ТЦУ Старт [Электронный ресурс]. URL: <http://www.bizguru.ru/node/460> (дата обращения 18.02.2019)
43. Управление памятью [Электронный ресурс]. URL: <https://riptutorial.com/ru/swift/topic/745/memory-management> (дата обращения: 14.12.2018).
44. Уроки разработки [Электронный ресурс]. URL: <https://www.raywenderlich.com> (дата обращения: 03.12.2018).
45. Формирование идею для разработки [Электронный ресурс]. URL: <https://habr.com/ru/post/250171/> (дата обращения: 23.03.2019).
46. Apple[Электронный ресурс]. URL: <https://developer.apple.com>(дата обращения 02.03.2019)
47. CocoaPods[Электронный ресурс]. URL: <https://itnan.ru/post.php?c=1&p=331780> (дата обращения 13.02.2019)
48. ERP система [Электронный ресурс]. URL: <https://habr.com/ru/company/trinion/blog/333018/> (дата обращения: 23.03.2018).
49. Grand Central Dispatch [Электронныйресурс]. URL: <http://qaru.site/questions/186310/grand-central-dispatch-vs-nsthreads> (дата обращения 2.03.2019)

50. Keepteam [Электронный ресурс]. URL: <https://startpack.ru/application/keepteam> (дата обращения 27.02.2019)
51. Promarket - Интерактивные Бизнес Коммуникации [Электронный ресурс]. URL: <https://professional.ru/Soobschestva/promarket--interaktivnye-biznes-kommunikatsii/> (дата обращения 7.02.2019)
52. RESTfulAPI [Электронный ресурс]. URL: <https://habr.com/ru/post/265845/> (дата обращения 10.03.2019)
53. SLS-Склад [Электронный ресурс]. URL: <http://www.sls.ru/produce/sklad.php> (дата обращения 18.02.2019)
54. SQLite[Электронный ресурс]. URL: <https://lecturesdb.readthedocs.io/databases/sqlite.html> (дата обращения 05.02.2019)
55. Storage [Электронный ресурс]. URL: <http://best-soft.ru/programs/3101.html> (дата обращения 12.02.2019)
56. Storecalc [Электронный ресурс]. URL: <https://www.softportal.com/software-20053-storecalc.html> (дата обращения 16.02.2019)
57. UIKit [Электронный ресурс]. URL: <https://habr.com/ru/company/badoo/blog/341542/> (дата обращения: 21.03.2018).
58. UML [Электронный ресурс]. URL: <https://arhivinfo.ru/1-43460.html> (дата обращения: 19.03.2019).

Приложение А. Фрагменты исходного кода программы

```
class RootAssembly {
  lazy varpresentationAssembly: IPresentationAssembly =
    PresentationAssembly(serviceAssembly: self.serviceAssembly)
  private lazy varserviceAssembly: IServicesAssembly =
    ServicesAssembly(coreAssembly: self.coreAssembly)
  private lazy varcoreAssembly: ICoreAssembly = CoreAssembly()
}

protocol Setupable {
  func setup(_ model: Any)
}

extension Date {

  funcfullDate() -> String {
    let dateFormatter = DateFormatter()
    dateFormatter.dateFormat = "yyyy-MM-ddHH:mm:ss"

    let date = self
    return dateFormatter.string(from: date)
  }

  funcshortDate() -> String {
    let dateFormatter = DateFormatter()
    dateFormatter.dateFormat = "MMM dd,yyyy"

    let date = self
    return dateFormatter.string(from: date)
  }
}
```

```
}
```

```
extension UIViewController {  
    @objcfunc keyboardWillShow(notification: NSNotification) {  
        if let keyboardSize =  
            (notification.userInfo?[UIResponder.keyboardFrameBeginUserInfoKey] as?  
             NSValue)?.cgRectValue {  
            if self.view.frame.origin.y == 0 {  
                self.view.frame.origin.y -= keyboardSize.height  
            }  
        }  
    }  
}
```

```
    @objcfunc keyboardWillHide(notification: NSNotification) {  
        if self.view.frame.origin.y != 0 {  
            self.view.frame.origin.y = 0  
        }  
    }  
}
```

```
final class RevisionViewController: UIViewController {
```

```
    @IBOutlet weak var tableView: UITableView!  
    @IBOutlet weak var idField: UITextField!  
    @IBOutlet weak var checkButton: UIButton!  
    @IBOutlet weak var scanButton: UIButton!
```

```

var models = [InventoryModel]()
var existingModels: [InventoryModel]?
private var warehouseStorageFacade: IWarehouseStorageFacade?

override func viewDidLoad() {
    super.viewDidLoad()

    tableView.delegate = self
    tableView.dataSource = self

    warehouseStorageFacade =
        AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade

    NotificationCenter.default.addObserver(self, selector:
        #selector(keyboardWillShow), name:
        UIResponder.keyboardWillShowNotification, object: nil)
    NotificationCenter.default.addObserver(self, selector:
        #selector(keyboardWillHide), name: UIResponder.keyboardWillHideNotification,
        object: nil)

    title = "Ревизия"
    let nib = UINib(nibName: "InventoryTableViewCell", bundle: nil)
    tableView.register(nib, forCellReuseIdentifier: "InventoryTableViewCell")
    guard let existingModels = existingModels else {
        return
    }
    existingModels.forEach { model in
        if model.isNeedRevision {
            models.append(model)
        }
    }
}

```

```
}
```

```
scanButton.layer.cornerRadius = 4
```

```
checkButton.layer.cornerRadius = 4
```

```
tableView.reloadData()
```

```
}
```

```
@IBActionfunctapOnVew(_ sender: Any) {
```

```
view.endEditing(true)
```

```
}
```

```
@IBActionfuncscanAction(_ sender: Any) {
```

```
view.endEditing(true)
```

```
idField.text = ""
```

```
let builder = QRCodeReaderViewControllerBuilder {
```

```
$0.reader = QRCodeReader(metadataObjectTypes: [.qr], captureDevicePosition:  
.back)
```

```
}
```

```
let vc = QRCodeReaderViewController(builder: builder)
```

```
vc.delegate = self
```

```
vc.modalPresentationStyle = .formSheet
```

```
present(vc, animated: true, completion: nil)
```

```
}
```

```
@IBActionfunccheckAction(_ sender: Any) {
```

```
models.forEach { model in
```

```
let id = idField.text ?? "0"
```

```

if model.id == id {
    idField.text = ""
    setRevision(model: model)
    view.endEditing(true)
}
}
}

/*
// MARK: - Navigation

// In a storyboard-based application, you will often want to do a little preparation
before navigation
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    // Get the new view controller using segue.destination.
    // Pass the selected object to the new view controller.
}
*/

}

extension RevisionViewController: UITableViewDelegate,
UITableViewDataSource {
    func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int)
->Int {
        return models.count
    }
}

```



```

func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) -
> UITableViewCell {
    let model = models[indexPath.row]
    let cell = tableView.dequeueReusableCell(withIdentifier:
    "InventoryTableViewCell", for: indexPath)

    (cell as? Setupable)?.setup(model as Any)
    cell.selectionStyle = .none
    return cell
}

```

```

func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath)
{
    tableView.deselectRow(at: indexPath, animated: true)
}

```

```

private func setRevision(model: InventoryModel) {
    let oldValue = model.revisionDate
    model.revisionDate = Int(Date().timeIntervalSince1970)
    var index: Int? = nil
    for (i,m) in models.enumerated() {
        if model.id == m.id {
            index = i
        }
    }
}

```

```

guard let i = index else {
    let alert = UIAlertController(title: "Данный инвентарь не нуждается в ревизии.",
    message: "Весь инвентарь нуждающийся в ревизии находится в таблице",
    preferredStyle: .alert)
}

```

```
alert.addAction(UIAlertAction(title: "Ок", style: .default, handler: nil))
present(alert, animated: true, completion: nil)
return
}
```

```
let indexPath = IndexPath(row: i, section: 0)
guard let cell = tableView.cellForRow(at: indexPath) as? InventoryTableViewCell
else { return }
```

```
warehouseStorageFacade?.saveInventory(model: model, completionClosure: {
[weak self] result in
guard let self = self else {
model.revisionDate = oldValue
return
}
guard result else {
model.revisionDate = oldValue
self.errorNotice("Ошибка", autoClear: true)
return
}
self.tableView.scrollToRow(at: indexPath, at: .none, animated: true)
self.tableView.beginUpdates()
self.tableView.insertRows(at: [indexPath], with: .automatic)
cell.reloadTitle()
self.tableView.beginUpdates()
}))
}
}
```

```

extension RevisionViewController: QRCodeReaderViewControllerDelegate {

func reader(_ reader: QRCodeReaderViewController, didScanResult result:
QRCodeReaderResult) {
    reader.stopScanning()
    let id = result.value
    dismiss(animated: true, completion: nil)
    if let model = existingModels?.first(where: { model -> Bool in
        model.id == id
    }) {
        setRevirion(model: model)
    } else {
        let alert = UIAlertController(title: "Упс!", message: "Инвентарь не найден",
        preferredStyle: .alert)
        alert.addAction(UIAlertAction(title: "Ок", style: .default, handler: nil))
        present(alert, animated: true, completion: nil)
    }

}

func readerDidCancel(_ reader: QRCodeReaderViewController) {
    reader.stopScanning()
    reader.dismiss(animated: true, completion: nil)
}

}

```

```

class ChartViewController: UIViewController {

    @IBOutlet weak var lineChartView: PieChartView!
    @IBOutlet weak var chartView: BarChartView!
    var models: [InventoryModel]?
    var sortedModel: [String:[InventoryModel]] = [:]
    private var warehouseStorageFacade: IWarehouseStorageFacade?
    private var categoryArray = [String]() {
        didSet {
            categoryArray.forEach { category in
                let filterModels = models?.filter({ $0.category == category })
                sortedModel[category] = filterModels
            }
            let defaultModel = [InventoryModel(name: "", id: "", date: 1, revisionDate: 1,
            category: "", personal: "", qrCode: nil)]
            let first = models?.filter { model -> Bool in
                return (model.date > 1546300800 && model.date < 1554076800)
            } ?? defaultModel
            let second = models?.filter { model -> Bool in
                return (model.date > 1554076800 && model.date < 1561939200)
            } ?? defaultModel

            let third = models?.filter { model -> Bool in
                return (model.date > 1561939200 && model.date < 1569888000)
            } ?? defaultModel

            let ch = models?.filter { model -> Bool in
                return (model.date > 1569888000 && model.date < 1577836800)
            } ?? defaultModel

```

```

let unitsSold: [Double] = [Double(first.count)*4, Double(second.count)*4,
Double(thirt.count)*4, Double(ch.count)*4]
let summCount: [Double] = [Double(first.count)*4, Double(second.count +
first.count)*4, Double(thirt.count + second.count + first.count)*4, Double(
ch.count + thirt.count + second.count + first.count)*4]
let months = ["Первый квартал", "Второй квартал", "Третий квартал",
"Четвёртый квартал"]
setChart(dataPoints: months, values1: unitsSold, values2: summCount)
}
}

```

```

override func viewDidLoad() {
super.viewDidLoad()

```

```

guard let _ = models else { return }
title = "Отчёт"
navigationItem.rightBarButtonItem = UIBarButtonItem(title: "Печать", style:
.plain, target: self, action: nil)

```

```

self.pleaseWait()
warehouseStorageFacade =
AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade
warehouseStorageFacade?.fetchOrgData(completionClosure: { [weak self] (_,
category, error) in
guard let self = self else { return }
guard let category = category, error == nil else {
self.clearAllNotice()
self.errorNotice(error ?? "", autoClear: true)
return

```

```

}
self.clearAllNotice()
self.categoryArray = category
})

chartView.noDataText = "Нетдопустимыхдиаграмм"
// Do any additional setup after loading the view.
}

funcsetChart(dataPoints: [String], values1: [Double], values2: [Double]) {
vardataEntries: [BarChartDataEntry] = []
var dataEntries2: [BarChartDataEntry] = []

for i in 0..

```

```
charDataSet2.colors = [.red]
```

```
let chartData = BarChartData(dataSets: [charDataSet2, chartDataSet])
```

```
chartView.data = chartData
```

```
chartView.xAxis.labelHeight = 0
```

```
let desc = Description()
```

```
desc.text = "Каждый столбец отображает соответствующий по счёту квартал  
текущего года"
```

```
chartView.chartDescription = desc
```

```
var dataPieEntries: [PieChartDataEntry] = []
```

```
for i in 0..  
sorterdModel.count {
```

```
let category = categoryArray[i]
```

```
let dataEntry = PieChartDataEntry(value:
```

```
Double((Int(sorterdModel[category]?.count ?? 0)*4)))
```

```
dataEntry.label = category
```

```
dataPieEntries.append(dataEntry)
```

```
}
```

```
let pieChartDataSet = PieChartDataSet(values: dataPieEntries, label: nil)
```

```
var colors: [UIColor] = []
```

```
for _ in 0..  
sorterdModel.count {
```

```
let red = Double(arc4random_uniform(256))
```

```
let green = Double(arc4random_uniform(256))
```

```
let blue = Double(arc4random_uniform(256))
```

```

let color = UIColor(red: CGFloat(red/255), green: CGFloat(green/255), blue:
CGFloat(blue/255), alpha: 1)
colors.append(color)
}

```

```

pieChartDataSet.entryLabelColor = .black
pieChartDataSet.colors = colors
let pieChartData = PieChartData(dataSet: pieChartDataSet)

```

```

lineChartView.data = pieChartData
lineChartView.usePercentValuesEnabled = true

```

```

}

```

```

}

```

```

extension ChartViewController {

```

```

    private class BarChartFormatter: NSObject, IAxisValueFormatter {

```

```

        var labels: [String] = []

```

```

        func stringForValue(_ value: Double, axis: AxisBase?) -> String {
            return labels[Int(value)]
        }

```

```

        init(labels: [String]) {

```



```
super.init()
self.labels = labels
}
}
}
```

```
class ReportCell: UITableViewCell {

    @IBOutlet weak varcountLabel: UILabel!
    @IBOutlet weak varcategoryLabel: UILabel!
    var model: Model?

    override funcawakeFromNib() {
        super.awakeFromNib()
        // Initialization code

    }

    funcdrawSelf(model: Model) {
        categoryLabel.text = model.name
        countLabel.text = String(model.count)
    }

    override funcsetSelected(_ selected: Bool, animated: Bool) {
        super.setSelected(selected, animated: animated)

        // Configure the view for the selected state
    }
}
```

```
}
```

```
extension ReportCell: Setupable {
```

```
    struct Model {
```

```
        let name: String
```

```
        let count: Int
```

```
    }
```

```
    func setup(_ model: Any) {
```

```
        guard let model = model as? Model else { return }
```

```
        self.model = model
```

```
        drawSelf(model: model)
```

```
    }
```

```
}
```

```
class ReportTableViewController: UITableViewController {
```

```
    var models: [InventoryModel]?
```

```
    var sorterdModel: [String:[InventoryModel]] = [:]
```

```
    private var warehouseStorageFacade: IWarehouseStorageFacade?
```

```
    private var categoryArray = [String]() {
```

```
        didSet {
```

```
            categoryArray.forEach { category in
```

```
                let filterModels = models?.filter({ $0.category == category })
```

```
                sorterdModel[category] = filterModels
```

```
            }
```

```
            tableView.reloadData()
```

```
        }
```

```
}
```

```
override func viewDidLoad() {  
    super.viewDidLoad()
```

```
    let nib = UINib(nibName: "ReportCell", bundle: nil)  
    tableView.register(nib, forCellReuseIdentifier: "ReportCell")
```

```
    guard let _ = models else { return }  
    self.pleaseWait()  
    warehouseStorageFacade =  
        AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade  
    warehouseStorageFacade?.fetchOrgData(completionClosure: { [weak self] (_,  
        category, error) in  
        guard let self = self else { return }  
        guard let category = category, error == nil else {  
            self.clearAllNotice()  
            self.errorNotice(error ?? "", autoClear: true)  
            return  
        }  
        self.clearAllNotice()  
        self.categoryArray = category  
    })
```

```
    // Uncomment the following line to preserve selection between presentations  
    // self.clearsSelectionOnViewWillAppear = false
```

```
    // Uncomment the following line to display an Edit button in the navigation bar for  
    this view controller.
```

```
    // self.navigationItem.rightBarButtonItem = self.editButtonItem
```

```
}
```

```
// MARK: - Table view data source
```

```
override funcnumberOfSections(in tableView: UITableView) ->Int {  
    // #warning Incomplete implementation, return the number of sections  
    return 1  
}
```

```
override funtableView(_ tableView: UITableView, numberOfRowsInSection  
section: Int) ->Int {  
    // #warning Incomplete implementation, return the number of rows  
    return sortedModel.count  
}
```

```
override funtableView(_ tableView: UITableView, cellForRowAtindexPath:  
IndexPath) ->UITableViewCell {  
    let category = categoryArray[indexPath.row]  
    guard let cell = tableView.dequeueReusableCell(withIdentifier: "ReportCell", for:  
indexPath) as? ReportCell,  
    let countCellModels = sortedModel[category]?.count else { return  
UITableViewCell() }
```

```
    let model = ReportCell.Model(name: category, count: countCellModels)  
    cell.setup(model)
```

```
    return cell
```

```
}
```

```
}
```

```
class FilterViewController: UIViewController {
```

```

@IBOutlet weak var picker: UIPickerView!
@IBOutlet weak varnameField: UITextField!
@IBOutlet weak varpersonalField: UITextField!
@IBOutlet weak varcategoryField: UITextField!
@IBOutlet weak varidFiled: UITextField!

private varwarehouseStorageFacade: IWarehouseStorageFacade?
private var chose: mySet = .none
private var employee = [String]()
private varcategoryArray = [String]()
var completion: ((Filter) -> Void)?
var filter: Filter?

override funcviewDidLoad() {
super.viewDidLoad()
warehouseStorageFacade =
AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade
picker.delegate = self
picker.dataSource = self
drawSelf()
}

@IBActionfuncpersonalAction(_ sender: Any) {
view.endEditing(true)

if chose == .personal {
chose = .none
picker.isHidden = !picker.isHidden
return
}
}

```

```

picker.isHidden = false
chose = .personal
picker.reloadAllComponents()

}

```

```

@IBActionfunccategoryAction(_ sender: Any) {
view.endEditing(true)

```

```

if chose == .category {
chose = .none
picker.isHidden = !picker.isHidden
return
}

```

```

picker.isHidden = false
chose = .category
picker.reloadAllComponents()

}

```

```

funcdrawSelf() {
warehouseStorageFacade?.fetchOrgData(completionClosure: { [weak self]
(personal, category, error) in

```

```

guard let self = self else { return }
guard let personal = personal, let category = category, error == nil else {
self.errorNotice(error ?? "")
return
}
}

```

```
self.employees = personal
self.categoryArray = category
```

```
}}
```

```
title = "Фильтр"
```

```
let doneItem = UIBarButtonItem(barButtonSystemItem: .done, target: self, action:
#selector(filterAction))
navigationItem.rightBarButtonItem = doneItem
}
```

```
@objc private funcfilterAction() {
```

```
let name = nameField.text == "" ? nil : nameField.text
```

```
let personal = personalField.text == "" ? nil : personalField.text
```

```
let category = categoryField.text == "" ? nil : categoryField.text
```

```
let id = idFiled.text == "" ? nil : idFiled.text
```

```
let filter = Filter(name: name, personal: personal, category: category, id: id)
```

```
guard let completion = completion else { return }
```

```
completion(filter)
```

```
navigationController?.popViewController(animated: true)
```

```
}
```

```
/*
```

```
// MARK: - Navigation
```

```
// In a storyboard-based application, you will often want to do a little preparation
before navigation
```

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
```

```
// Get the new view controller using segue.destination.
```

```
// Pass the selected object to the new view controller.
```

```
}
```

```
*/
```

```
}
```

```
extension FilterViewController: UIPickerViewDelegate, UIPickerViewDataSource  
{  
funcnumberOfComponents(in pickerView: UIPickerView) ->Int {  
return 1  
}
```

```
funcpickerView(_ pickerView: UIPickerView, numberOfRowsInComponent  
component: Int) ->Int {  
switch chose {  
case .category:  
return categoryArray.count  
case .personal:  
return employee.count  
case .none:  
return 0  
}  
}
```

```
funcpickerView(_ pickerView: UIPickerView, titleForRow row: Int,  
forComponent component: Int) -> String? {  
switch chose {  
case .category:  
return categoryArray[row]  
case .personal:  
return employee[row]  
case .none:
```



```

return nil
}
}

```

```

func pickerView(_ pickerView: UIPickerView, didSelectRow row: Int,
inComponent component: Int) {
switch chose {
case .category:
categoryField.text = categoryArray[row]
case .personal:
personalField.text = employee[row]
case .none:
return
}
}
}

```

```

class PinCodeViewController: UIViewController {

```

```

@IBOutlet weak varfirstChar: UITextField!
@IBOutlet weak varsecondChar: UITextField!
@IBOutlet weak varthirdChar: UITextField!
@IBOutlet weak varfourthChar: UITextField!
@IBOutlet weak varmainLabel: UILabel!
@IBOutlet weak varbiometricBtn: UIButton!

```

```

private let context = LAContext()
private varwarehouseStorageFacade: IWarehouseStorageFacade?

```

```

varexistPassword: String?
varenteredPassword: String?
varfirstEnterd: String?
varsecondEnterd: String?
var employee = [String]()

override funcviewDidLoad() {
super.viewDidLoad()

self.warehouseStorageFacade =
AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade

firstChar.delegate = self
secondChar.delegate = self
thirdChar.delegate = self
fourthChar.delegate = self

warehouseStorageFacade?.fetchOrgData { [weak self] (employee, _, error ) in
guard let self = self, let employee = employee, error == nil else { return }
self.employee = employee
}

if let password = KeychainService.loadPassword() {
mainLabel.text = "Введитепароль"
self.existPassword = password
biometricAction(self)
}

firstChar.becomeFirstResponder()
}

```

```

@IBAction func createNewPassword(_ sender: Any) {
    KeychainService.removePassword()
    clearAll()
    mainLabel.text = "Введите количество сотрудников"
    existPassword = nil
    firstEntered = nil
    secondEntered = nil
    firstChar.becomeFirstResponder()
}

```

```

private func clearAll() {
    firstChar.text = ""
    secondChar.text = ""
    thirdChar.text = ""
    fourthChar.text = ""
}

```

```

@IBAction func tapToView(_ sender: Any) {
    view.endEditing(true)
}

```

```

@IBAction func biometricAction(_ sender: Any) {

    let policy: LAPolicy = .deviceOwnerAuthenticationWithBiometrics
    var error: NSError?
    let reason = "Ввод пароля"

    if context.canEvaluatePolicy(policy, error: &error) {

```

```

context.evaluatePolicy(policy, localizedReason: reason, reply: { (success, error) in
if success {

DispatchQueue.main.async {

self.performSegue(withIdentifier: "ShowMainNavigation", sender: nil)
}

} else {
DispatchQueue.main.async {
print("recognize error")
}

}

}))

} else {
biometricBtn.isEnabled = false
}
}

}

funccreateSting(char: [Character]) -> String {
var string = ""
char.forEach({string.append($0)})
return string
}

extension PinCodeViewController: UITextFieldDelegate {

```

```

func textField(_ textField: UITextField, shouldChangeCharactersIn range:
 NSRange, replacementString string: String) -> Bool {
    guard string.count == 1 else {
        return false
    }

    if textField == firstChar {
        secondChar.becomeFirstResponder()
        if existPassword == nil, firstEntered == nil {
            firstEntered = string
        } else if existPassword == nil, firstEntered?.count == 4, secondEntered == nil {
            secondEntered = string
        } else if existPassword != nil {
            enteredPassword = string
        }
        firstChar.text = string
        return true
    } else if textField == secondChar {

        thirdChar.becomeFirstResponder()
        if existPassword == nil, firstEntered?.count == 1 {
            firstEntered?.append(string)
        } else if existPassword == nil, firstEntered?.count == 4, secondEntered?.count == 1 {
            secondEntered?.append(string)
        } else if existPassword != nil {
            enteredPassword?.append(string)
        }
        secondChar.text = string
        return true
    }
}

```

```

    } else if textField == thirdChar {
    fourthChar.becomeFirstResponder()
    if existPassword == nil, firstEntered?.count == 2 {
    firstEntered?.append(string)
    } else if existPassword == nil, firstEntered?.count == 4, secondEntered?.count == 2 {
    secondEntered?.append(string)
    } else if existPassword != nil {
    enteredPassword?.append(string)
    }
    thirdChar.text = string
    return true
    } else if textField == fourthChar {

    if existPassword == nil, firstEntered?.count == 3 {
    firstEntered?.append(string)
    } else if existPassword == nil, firstEntered?.count == 4, secondEntered?.count == 3 {
    secondEntered?.append(string)
    } else if existPassword != nil {
    enteredPassword?.append(string)
    }

    if existPassword == nil, firstEntered?.count == 4, secondEntered == nil {
    ifmainLabel.text == "Введите количество сотрудников" {
    if let str = firstEntered {
    varcharSet = [Character("0")]
    str.forEach { char in
    if char != "0" {
    if charSet.first == "0" {
    charSet = [char]
    } else {
    charSet.append(char)

```

```

}
} else if charSet.first != "0" {
charSet.append(char)
}
}
if Int(createSting(char: charSet)) == employee.count {
mainLabel.text = "Введите новый пароль"
}
firstEntered = nil
clearAll()
firstChar.becomeFirstResponder()
return false
}
}
mainLabel.text = "Повторите пароль"
firstChar.becomeFirstResponder()
clearAll()
} else if existPassword == nil, firstEntered?.count == 4, secondEntered?.count == 4 {
if firstEntered == secondEntered {
guard let password = secondEntered else { return false }
KeychainService.savePassword(data: password)
self.performSegue(withIdentifier: "ShowMainNavigation", sender: nil)
} else {
firstChar.becomeFirstResponder()
mainLabel.text = "Ошибка, повторите ввод пароля"
secondEntered = nil
clearAll()
return false
}
} else if existPassword != nil, enteredPassword?.count == 4 {

```

```

if existPassword == enteredPassword {
self.performSegue(withIdentifier: "ShowMainNavigation", sender: nil)
} else {
firstChar.becomeFirstResponder()
mainLabel.text = "Выввели неверный пароль"
enteredPassword = nil
clearAll()
return false
}
}
return true
} else {
return false
} }
}

```

```

class DetailInventoryViewController: UIViewController {

```

```

@IBOutlet weak var printBtn: UIButton!
@IBOutlet weak var qrCodeGenerateButton: UIButton!
@IBOutlet weak var nameLabel: UILabel!
@IBOutlet weak var personalLabel: UILabel!
@IBOutlet weak var categoryLabel: UILabel!
@IBOutlet weak var idLabel: UILabel!
@IBOutlet weak var dateLabel: UILabel!
@IBOutlet weak var qrCodeImageView: UIImageView!

```

```

private let warehouseStorageFacade =
AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade

```

```

var model: InventoryModel?

```



```
override funcviewDidLoad() {  
    super.viewDidLoad()
```

```
drawSelf()  
    // Do any additional setup after loading the view.  
}
```

```
override funcviewWillAppear(_ animated: Bool) {  
    super.viewWillAppear(animated)  
    drawSelf()  
}
```

```
private funcdrawSelf() {
```

```
    guard let model = model else { return }
```

```
    let reductBarButtonItem = UIBarButtonItem(title: "Редактировать", style: .done,  
    target: self, action: #selector(reduct))
```

```
    navigationItem.rightBarButtonItem = reductBarButtonItem
```

```
    title = model.name
```

```
    printBtn.isHidden = true
```

```
    nameLabel.text = model.name
```

```
    personalLabel.text = model.personal
```

```
    categoryLabel.text = model.category
```

```
    idLabel.text = model.id
```

```
    let timeInterval = TimeInterval(exactly: model.date) ?? TimeInterval()
```

```
let date = Date(timeIntervalSince1970: TimeInterval)
```

```
dateLabel.text = date.fullDate()
```

```
qrCodeGenerateButton.layer.cornerRadius = 8
```

```
printBtn.layer.cornerRadius = 8
```

```
if let qrCodeString = model.qrCode, let data = Data(base64Encoded:  
qrCodeString) {
```

```
let image = UIImage(data: data)
```

```
printBtn.isHidden = false
```

```
qrCodeImageView.isHidden = false
```

```
qrCodeGenerateButton.isHidden = true
```

```
qrCodeImageView.image = image
```

```
}
```

```
}
```

```
@objcfunc reduct() {
```

```
let vc =
```

```
AppDelegate.rootAssembly.presentationAssembly.addInventoryViewController(m  
odel: model)
```

```
navigationController?.pushViewController(vc, animated: true)
```

```
}
```

```
@IBAction func generateQRCode(_ sender: Any) {
```

```
guard let id = model?.id, let ciImage = QRCodeViewController.generate(id: id), let  
image = ciImage.toImage(frame: qrCodeImageView.frame) else { return }
```

```
self.pleaseWait()
```

```
let dataString = image.ciImage?.recreated?.base64EncodedString()
```

```

model?.qrCode = dataString
qrCodeImageView.isHidden = false
printBtn.isHidden = false
qrCodeGenerateButton.isHidden = true

if let model = model {
warehouseStorageFacade.saveInventory(model: model) { [weak self] success in

guard let self = self else { return }

self.clearAllNotice()
if success {
self.qrCodeImageView.image = image
self.successNotice("Сохранено")
} else {
self.successNotice("Ошибка сохранения")
}
self.clearAllNotice()
}
}
}

@IBAction func printAction(_ sender: Any) {
guard let image = qrCodeImageView.image, let id = model?.id else { return }
let printController = UIPrintInteractionController.shared
let printInfo = UIPrintInfo(dictionary: nil)
printInfo.outputType = .general
printInfo.jobName = "Печать QR-Кода"
printInfo.orientation = .landscape
printController.showsNumberOfCopies = true

```

```

printController.printInfo = printInfo
printController.printingItems = [image, "Finch: \(String(describing: id))"]
printController.present(animated: true) { (controller, success, errorMsg) in
if success {
print("***** Print Successfully")
} else {
print("***** Print Failed : \(String(describing:
errorMsg?.localizedDescription))")
}
}
}
}

```

```

class WarehouseStorageModel {
varwarehouseImage: UIImage
varwarehouseName: String
varwarehouseDescription: String
varwarehouseNameWasEdited: Bool = false
vardescriptionWasEdited: Bool = false
varwarehouseImageWasEdited: Bool = false
init(name: String, description: String, image: UIImage) {
self.warehouseName = name
self.warehouseDescription = description
self.warehouseImage = image
}
}

class InventoryModel {

```

```

var name: String

```

```

var id: String
var date: Int
var category: String
var revisionDate: Int
var personal: String
var qrCode: String?

var isNeedRevision: Bool {
    let range: [ClosedRange<Int>] = [1546300800...1554076800,
    1554076800...1561939200, 1561939200...1569888000,
    1569888000...1577836800]

    for (index, value) in range.enumerated() {
        if value.contains(revisionDate), index + 1 != range.count, range[index +
        1].contains(Int(Date().timeIntervalSince1970)) {
            return true
        }
    }
    return false
}

init(name: String, id: String, date: Int, revisionDate: Int, category: String, personal:
String, qrCode: String?) {

    self.name = name
    self.id = id
    self.date = date
    //Временный костыль
    if revisionDate == 0 {
        self.revisionDate = date
    }
}

```

```

    } else {
    self.revisionDate = revisionDate
    }
    self.category = category
    self.personal = personal
    self.qrCode = qrCode
    }

    funcsendModel() -> [String : Any] {
    vardict = ["name" : name as Any,
    "date" : date as Any,
    "revisionDate" : revisionDate as Any,
    "category" : category as Any,
    "personal" : personal as Any]

    if let qrCode = qrCode {
    dict["qrCode"] = qrCode as Any
    }

    return dict
    }

    struct Filter {
    let name: String?
    let personal: String?
    let category: String?
    let id: String?
    }

```

```

class WarehouseStorageFacade: IWarehouseStorageFacade {

    private var warehouseStorageService: IWarehouseStorageService =
        WarehouseGCDStorageService()
    private var warehouseCoreDataStorage: WarehouseCoreDataStorage

    init(warehouseStorageService: IWarehouseStorageService,
        warehouseCoreDataStorage: WarehouseCoreDataStorage) {
        self.warehouseStorageService = warehouseStorageService
        self.warehouseCoreDataStorage = warehouseCoreDataStorage
    }

    func fetchInventory(completionClosure: @escaping ([InventoryModel]?, String?) ->
        Void) {
        self.warehouseCoreDataStorage.fetchInventory(completionClosure:
            completionClosure)
    }

    func fetchOrgData(completionClosure: @escaping ([String]?, [String]?, String?) ->
        Void) {
        self.warehouseCoreDataStorage.fetchOrgData(completionClosure:
            completionClosure)
    }

    func saveInventory(model: InventoryModel, completionClosure: @escaping (Bool)
        -> Void) {
        self.warehouseCoreDataStorage.saveInventoryToFirebase(model: model,
            completionClosure: completionClosure)
    }
}

```

```

funcsaveCategory(category: String, position: Int, completionClosure: @escaping
(Bool) -> Void) {
self.warehouseCoreDataStorage.saveCategory(category: category, position:
position, completionClosure: completionClosure)
}

```

```

funcsaveEmployee(employee: String, position: Int, completionClosure: @escaping
(Bool) -> Void) {
self.warehouseCoreDataStorage.saveEmployee(employee: employee, position:
position, completionClosure: completionClosure)
}

```

```

funcdeleteInventory(model: InventoryModel, completionClosure: @escaping
(Bool) -> Void) {
self.warehouseCoreDataStorage.deleteInventoryFromFirebase(model: model,
completionClosure: completionClosure)
}

```

```

funcupdateWarehouse(warehouses: WarehouseStorageModel, completionClosure:
@escaping (Bool) -> Void) {
self.warehouseStorageService.saveWarehouse(warehouse: warehouses,
completionClosure: completionClosure)
}

```

```

funcgetWarehouse(completionClosure: @escaping ([WarehouseStorageModel]) ->
Void) {
self.warehouseStorageService.loadWarehouse(completionClosure:
completionClosure)
}

```



```
}
```

```
protocol IWarehouseStorageFacade {  
    funcupdateWarehouse(warehouses: WarehouseStorageModel,  
        completionClosure: @escaping (_ withError: Bool) -> Void)  
    funcgetWarehouse(completionClosure: @escaping (_ warehouses:  
        [WarehouseStorageModel]) -> Void)  
    funcfetchInventory(completionClosure: @escaping (_ Inventory:  
        [InventoryModel]?, _ error: String?) -> Void)  
    funcfetchOrgData(completionClosure: @escaping (_ personal: [String]?, _  
        category: [String]?, _ error: String?) -> Void)  
    funcsaveInventory(model: InventoryModel, completionClosure: @escaping (_  
        success: Bool) -> Void)  
    funcsaveCategory(category: String, position: Int, completionClosure: @escaping  
        (_ success: Bool) -> Void)  
    funcsaveEmployee(employee: String, position: Int, completionClosure: @escaping  
        (_ success: Bool) -> Void)  
    funcdeleteInventory(model: InventoryModel, completionClosure: @escaping (_  
        success: Bool) -> Void)  
}  
  
class InventoryTableViewCell: UITableViewCell {  
  
    @IBOutlet weak var name: UILabel!  
    @IBOutlet weak var personal: UILabel!  
    @IBOutlet weak var date: UILabel!  
  
    private var model: InventoryModel?  
  
    override funcawakeFromNib() {  
        super.awakeFromNib()
```

```

// Initialization code
}

override fun setSelected(_ selected: Bool, animated: Bool) {
    super.setSelected(selected, animated: animated)

    // Configure the view for the selected state
}

fun reloadTitle() {
    guard let model = model else { return }
    self.name.text = model.isNeedRevision ? "□ \$(model.name)" : "□ \$(model.name)"
}

}

extension InventoryTableViewCell: Setupable {

    fun setup(_ model: Any) {

        guard let model = model as? InventoryModel else { return }

        self.model = model
        self.name.text = model.isNeedRevision ? "□ \$(model.name)" : "□ \$(model.name)"
        self.personal.text = model.personal

        let timeInterval = TimeInterval(exactly: model.date) ?? TimeInterval()
        let date = Date(timeIntervalSince1970: timeInterval)

        self.date.text = date.shortDate()
    }
}

```

```
}
```

```
}
```

```
class InventoryTableViewController: UITableViewController {
```

```
    let customRefreshControl = UIRefreshControl()
```

```
    var warehouseStorageFacade: IWarehouseStorageFacade?
```

```
    var models = [InventoryModel]()
```

```
    override func viewDidLoad() {
```

```
        super.viewDidLoad()
```

```
        tableView.delegate = self
```

```
        tableView.dataSource = self
```

```
        warehouseStorageFacade =
```

```
        AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade
```

```
        drawSelf()
```

```
        refreshControl?.attributedTitle = NSAttributedString(string: "Pull to refresh")
```

```
        refreshControl?.addTarget(self, action: #selector(loadData), for:
```

```
        UIControl.Event.valueChanged)
```

```
        let nib = UINib(nibName: "InventoryTableViewCell", bundle: nil)
```

```
        tableView.register(nib, forCellReuseIdentifier: "InventoryTableViewCell")
```

```
        loadData()
```

```
    }
```

```

@objcfuncloadData() {
self.pleaseWait()
warehouseStorageFacade?.fetchInventory() { [weak self] (inventory, error) in

guard let inventory = inventory, error == nil else {
print(error ?? "")
self?.clearAllNotice()
return
}

self?.models = inventory
self?.models.sort(by: { $0.date > $1.date })
self?.tableView.reloadData()
self?.refreshControl?.endRefreshing()
self?.clearAllNotice()
}
}

override funcviewWillAppear(_ animated: Bool) {
super.viewWillAppear(animated)
tableView.reloadSections(IndexSet(arrayLiteral: 0), with: .automatic)
}

private funcdrawSelf() {

title = "ИНВЕНТАРЬ"

self.refreshControl = customRefreshControl

```

```

let cameraItem = UIBarButtonItem(barButtonSystemItem: .camera, target: self,
action: #selector(showScanner))

let addItem = UIBarButtonItem(barButtonSystemItem: .add, target: self, action:
#selector(addAction))

let reportItem = UIBarButtonItem(image: imageLiteral(resourceName: "report"),
style: .plain, target: self, action: #selector(showReport))

let filterItem = UIBarButtonItem(image: imageLiteral(resourceName: "filter"),
style: .plain, target: self, action: #selector(showFilter))

let revisionItem = UIBarButtonItem(barButtonSystemItem: .bookmarks, target:
self, action: #selector(revision))

tableView.estimatedRowHeight = 65
tableView.sectionHeaderHeight = 30

navigationItem.largeTitleDisplayMode = .never
navigationItem.rightBarButtonItem = [cameraItem, reportItem, filterItem,
revisionItem, addItem]
}

private funcdrawTableHeaderView() ->UIView {

return UINib(nibName: "HeaderView", bundle: nil).instantiate(withOwner: nil,
options: nil)[0] as! UIView
}

private funcshowDetail(with model: InventoryModel) {

```

```

let view =
AppDelegate.rootAssembly.presentationAssembly.detailInventoryViewController(
model: model)
self.navigationController?.pushViewController(view, animated: true)
}

```

```

@objcfunc revision() {
let view =
AppDelegate.rootAssembly.presentationAssembly.revisionViewController(models
: models)
self.navigationController?.pushViewController(view, animated: true)
}

```

```

@objcfuncshowReport() {
let view = AppDelegate.rootAssembly.presentationAssembly.reportChar(models:
models)
navigationController?.pushViewController(view, animated: true)
}

```

```

@objc private funcshowFilter() {
loadData()
let view = AppDelegate.rootAssembly.presentationAssembly.filerViewController
{ [weak self] filter in

guard let self = self else { return }

if let fileredByPersonal = filter.personal {
self.models = self.models.filter({ model -> Bool in
model.personal == fileredByPersonal
})
}
}

```

```
}
```

```
if let filteredByName = filter.name {  
    self.models = self.models.filter({ model -> Bool in  
        model.name == filteredByName  
    })  
}
```

```
if let filterByCategory = filter.category {  
    self.models = self.models.filter({ model -> Bool in  
        model.category == filterByCategory  
    })  
}
```

```
if let filterByID = filter.id {  
    self.models = self.models.filter({ model -> Bool in  
        model.id == filterByID  
    })  
}  
self.tableView.reloadData()  
}  
navigationController?.pushViewController(view, animated: true)  
}
```

```
@objc private func showScanner() {
```

```
    let builder = QRCodeReaderViewControllerBuilder {  
        $0.reader = QRCodeReader(metadataObjectTypes: [.qr], captureDevicePosition:  
            .back)  
    }  
}
```

```
let vc = QRCodeReaderViewController(builder: builder)
```

```
vc.delegate = self
```

```
vc.modalPresentationStyle = .formSheet
```

```
present(vc, animated: true, completion: nil)
```

```
}
```

```
@objc private func addAction() {
```

```
let vc =
```

```
AppDelegate.rootAssembly.presentationAssembly.addInventoryViewController(model: nil)
```

```
let randomId = Int.random(in: 100000...100000000)
```

```
guard !models.contains(where: { Int($0.id) == randomId }) else { return }
```

```
vc.id = randomId
```

```
self.navigationController?.pushViewController(vc, animated: true)
```

```
}
```

```
override func numberOfSections(in tableView: UITableView) -> Int {
```

```
// #warning Incomplete implementation, return the number of sections
```

```
return 1
```

```
}
```

```
override func tableView(_ tableView: UITableView, viewForHeaderInSection section: Int) -> UIView? {
```

```
let header = drawTableHeaderView()
```

```
return header
```

```
}
```

```
override func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
```

```
// #warning Incomplete implementation, return the number of rows
```



```
return models.count
}
```

```
override fun tableView(_ tableView: UITableView, willDisplay cell:
UITableViewCell, forRowAtindexPath: IndexPath) {
    guard let cell = cell as? InventoryTableViewCell else { return }
    cell.reloadTitle()
}
```

```
override fun tableView(_ tableView: UITableView, cellForRowAtindexPath:
IndexPath) -> UITableViewCell {
```

```
    let model = models[indexPath.row]
    let cell = tableView.dequeueReusableCell(withIdentifier:
        "InventoryTableViewCell", for: indexPath)
```

```
    (cell as? Setupable)?.setup(model as Any)
```

```
    return cell
}
```

```
override fun tableView(_ tableView: UITableView, didSelectRowAtindexPath:
IndexPath) {
```

```
    let model = models[indexPath.row]
    showDetail(with: model)
}
```

```
/*
```

```
// Override to support conditional editing of the table view.
```

```

override fun tableView(_ tableView: UITableView, canEditRowAtIndexPath:
IndexPath) -> Bool {
// Return false if you do not want the specified item to be editable.
return true
}
*/

// Override to support editing the table view.
override fun tableView(_ tableView: UITableView, commit editingStyle:
UITableViewCellEditingStyle, forRowAtIndexPath: IndexPath) {
if editingStyle == .delete {
// Delete the row from the data source
self.pleaseWait()

let model = models[indexPath.row]
warehouseStorageFacade?.deleteInventory(model: model) { [weak self] success in

self?.clearAllNotice()
guard success else {
self?.errorNotice("Delete error", autoClear: true)

return
}

self?.noticeSuccess("Delete success", autoClear: true, autoClearTime: 1)
self?.models.removeAll(where: {$0.id == model.id})
tableView.beginUpdates()
tableView.deleteRows(at: [indexPath], with: .fade)
tableView.endUpdates()

```

```

}
} else if editingStyle == .insert {
// Create a new instance of the appropriate class, insert it into the array, and add a
new row to the table view
}
}

/*
// Override to support rearranging the table view.
override func tableView(_ tableView: UITableView, moveRowAtfromIndexPath:
IndexPath, to: IndexPath) {

}
*/

/*
// Override to support conditional rearranging of the table view.
override func tableView(_ tableView: UITableView, canMoveRowAtindexPath:
IndexPath) -> Bool {
// Return false if you do not want the item to be re-orderable.
return true
}
*/

/*
// MARK: - Navigation

// In a storyboard-based application, you will often want to do a little preparation
before navigation

```

```

override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    // Get the new view controller using segue.destination.
    // Pass the selected object to the new view controller.
}
*/

}

```

```

extension InventoryTableViewController: QRCodeReaderViewControllerDelegate
{

```

```

func reader(_ reader: QRCodeReaderViewController, didScanResult result:
QRCodeReaderResult) {

```

```

    reader.stopScanning()
    let id = result.value
    dismiss(animated: true, completion: nil)
    if let model = models.first(where: { model -> Bool in
        model.id == id
    }) {
        showDetail(with: model)
    } else {
        let alert = UIAlertController(title: "Упс!", message: "Инвентарь не найден",
        preferredStyle: .alert)
        alert.addAction(UIAlertAction(title: "Ок", style: .default, handler: nil))
        present(alert, animated: true, completion: nil)
    }

}

```

```

funcreaderDidCancel(_ reader: QRCodeReaderViewController) {

    reader.stopScanning()
    reader.dismiss(animated: true, completion: nil)
}

}protocol QRCodeViewControllerDelegate: class {
funcqrCodeGenerated(qrCode: UIImage)
}

class QRCodeViewController: UIViewController {

    @IBOutlet weak varprintButton: UIButton!
    @IBOutlet weak varqrImageView: UIImageView!
    weak var delegate: QRCodeViewControllerDelegate?
    var id: String?

    override funcviewDidLoad() {
        super.viewDidLoad()

        let readyItem = UIBarButtonItem(barButtonItemSystemItem: .done, target: self, action:
        #selector(doneAction))
        self.navigationItem.rightBarButtonItem = readyItem
        drawSelf()
        ceateQrCode()
        // Do any additional setup after loading the view.
    }

```

```

@IBAction func printAction(_ sender: Any) {

    let printController = UIPrintInteractionController.shared
    let printInfo = UIPrintInfo(dictionary: nil)
    printInfo.outputType = .general
    printInfo.jobName = "Печать QR-Кода"
    printInfo.orientation = .landscape
    printController.showsNumberOfCopies = true

    printController.printInfo = printInfo
    printController.printingItems = [qrImageView.image!, "Finch: \(String(describing:
id))"]
    printController.present(animated: true) { (controller, success, errorMsg) in
        if success {
            print("***** Print Successfully")
        } else {
            print("***** Print Failed : \(String(describing:
errorMsg?.localizedDescription))")
        }
    }
}

private func drawSelf() {
    printButton.layer.cornerRadius = 12
}

static func generate(id: String) -> UIImage? {
    let data = id.data(using: String.Encoding.isoLatin1, allowLossyConversion: false)

    let filter = CIFilter(name: "CIQRCodeGenerator")

```

```

filter?.setValue(data, forKey: "inputMessage")
filter?.setValue("Q", forKey: "inputCorrectionLevel")

guard let ciImage = filter?.outputImage else { return nil }
return ciImage
}

private func createQRCode() {

guard let id = id, let ciImage = QRCodeViewController.generate(id: id) else {
return }

qrImageView.image = ciImage.toImage(frame: qrImageView.frame)

delegate?.qrCodeGenerated(qrCode: ciImage)
}

@objc func doneAction() {
self.dismiss(animated: true, completion: nil)
}
/*
// MARK: - Navigation

// In a storyboard-based application, you will often want to do a little preparation
before navigation
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
// Get the new view controller using segue.destination.
// Pass the selected object to the new view controller.

```

```

}
*/

}
enummySet {
case category
case personal
case none
}

class AddInventoryViewController: UIViewController {

@IBOutlet weak varnameField: UITextField!
@IBOutlet weak varfio: UITextField!
@IBOutlet weak var category: UITextField!
@IBOutlet weak varidLabel: UILabel!
@IBOutlet weak varqrSwitch: UISwitch!
@IBOutlet weak var picker: UIPickerView!
@IBOutlet weak vargenerateTitle: UILabel!

var id = 0
var model: InventoryModel?
private var employee = [String]()
private varcategoryArray = [String]()

private var chose: mySet = .none
private varwarehouseStorageFacade: IWarehouseStorageFacade?

override funcviewDidLoad() {
super.viewDidLoad()

```



```
warehouseStorageFacade =  
AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade
```

```
picker.delegate = self  
picker.dataSource = self
```

```
drawSelf()  
}
```

```
private funcdrawSelf() {
```

```
if let model = model {  
title = "Редактироватьинвентарь"
```

```
idLabel.text = model.id  
nameField.text = model.name  
fio.text = model.personal  
category.text = model.category  
qrSwitch.isHidden = true  
generateTitle.isHidden = true  
} else {  
title = "Добавитьинвентарь"  
idLabel.text = String(id)  
}
```

```
warehouseStorageFacade?.fetchOrgData(completionClosure: { [weak self]  
(personal, category, error) in
```

```
guard let self = self, category?.first != "" else { return }
```

```

guard let personal = personal, let category = category, error == nil else {
    self.errorNotice(error ?? "")
    return
}

self.employee = personal
self.categoryArray = category

})

let action = model != nil ? #selector(resave):#selector(saveAction)
let addItem = UIBarButtonItem(barButtonSystemItem: .done, target: self, action:
action)
self.navigationItem.rightBarButtonItem = addItem
}

@IBAction func hideKeyboard(_ sender: Any) {
    view.endEditing(true)
}

@objc func resave() {
    guard let model = model else { return }

    if nameField.text == "" || fio.text == "" || category.text == "" {
        self.errorNotice("Незаполнено", autoClear: true)
        return
    }

    self.pleaseWait()

```

```
guard let name = nameField.text, let category = category.text, let employee =  
fio.text else { return }
```

```
if !employee.contains(employee) {  
warehouseStorageFacade?.saveEmployee(employee: employee, position:  
employee.count, completionClosure: { _ in })  
}
```

```
if !categoryArray.contains(category) {  
warehouseStorageFacade?.saveCategory(category: category, position:  
categoryArray.count, completionClosure: { _ in })  
}
```

```
let newModel = InventoryModel(name: name, id: model.id, date: model.date,  
revisionDate: model.revisionDate, category: category, personal: employee,  
qrCode: model.qrCode)  
warehouseStorageFacade?.saveInventory(model: newModel) { success in  
guard success else {  
self.errorNotice("Save error", autoClear: true)  
return  
}  
self.clearAllNotice()  
self.noticeSuccess("Save success", autoClear: true, autoClearTime: 1)  
self.navigationController?.children.forEach({ vc in  
if let vc = vc as? DetailInventoryViewController {  
vc.model = newModel  
}  
})  
self.navigationController?.children.forEach({ view in  
if let vc = view as? InventoryTableViewController {
```

```

vc.loadData()
}
}))
self.navigationController?.popViewController(animated: true)
}
}

@objc private func saveAction() {

if nameField.text == "" || category.text == "" {
self.errorNotice("Незаполнено", autoClear: true)
return
}

self.pleaseWait()
let fioStr = fio.text == "" ? "-":nameField.text
guard let name = nameField.text, let category = category.text, let employee =
fioStr else { return }

if !employee.contains(employee), employee != "-" {
warehouseStorageFacade?.saveEmployee(employee: employee, position:
employee.count, completionClosure: { _ in })
}

if !categoryArray.contains(category) {
warehouseStorageFacade?.saveCategory(category: category, position:
categoryArray.count, completionClosure: { _ in })
}

if self.qrSwitch.isOn {

```

```

let view =
AppDelegate.rootAssembly.presentationAssembly.qrCodeViewController(id:
String(id), delegate: self)
let nav = UINavigationController(rootViewController: view)
self.navigationController?.present(nav, animated: true)

navigationController?.children.forEach({ vc in
if let vc = vc as? InventoryTableViewController {
vc.loadData()
}
})
self.navigationController?.popViewController(animated: true)
} else {
let date = Int(Date().timeIntervalSince1970)
let model = InventoryModel(name: name, id: String(id), date: date, revisionDate:
date, category: category, personal: employee, qrCode: nil)
warehouseStorageFacade?.saveInventory(model: model) { success in
guard success else {
self.errorNotice("Save error", autoClear: true)
return
}
self.clearAllNotice()
self.noticeSuccess("Save success", autoClear: true, autoClearTime: 1)
self.navigationController?.popViewController(animated: true)
}
}

}

@IBActionfuncchooseFio(_ sender: Any) {

```

```
view.endEditing(true)
```

```
if chose == .personal {  
    chose = .none  
    picker.isHidden = !picker.isHidden  
    return  
}  
picker.isHidden = false  
chose = .personal  
picker.reloadAllComponents()  
  
}
```

```
@IBAction func chooseCategory(_ sender: Any) {
```

```
    view.endEditing(true)  
  
    if chose == .category {  
        chose = .none  
        picker.isHidden = !picker.isHidden  
        return  
    }  
    picker.isHidden = false  
    chose = .category  
    picker.reloadAllComponents()  
  
}
```

```

/*
// MARK: - Navigation

// In a storyboard-based application, you will often want to do a little preparation
before navigation
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
// Get the new view controller using segue.destination.
// Pass the selected object to the new view controller.
}
*/
}

```

```

extension AddInventoryViewController: UIPickerViewDelegate,
UIPickerViewDataSource {

```

```

funcnumberOfComponents(in pickerView: UIPickerView) ->Int {
return 1
}

```

```

funcpickerView(_ pickerView: UIPickerView, numberOfRowsInComponent
component: Int) ->Int {

```

```

switch chose {
case .category:
return categoryArray.count
case .personal:
return employee.count
case .none:

```

```
return 0
}
}
```

```
func pickerView(_ pickerView: UIPickerView, titleForRow row: Int,
forComponent component: Int) -> String? {
```

```
switch chose {
case .category:
return categoryArray[row]
case .personal:
return employee[row]
case .none:
return nil
}
}
```

```
func pickerView(_ pickerView: UIPickerView, didSelectRow row: Int,
inComponent component: Int) {
switch chose {
case .category:
category.text = categoryArray[row]
case .personal:
fio.text = employee[row]
case .none:
return
}
}
}
```



```

extension AddInventoryViewController: QRCodeViewControllerDelegate {

funcqrCodeGenereted(qrCode: CIImage) {

guard let name = nameField.text, let category = category.text, let employee =
fio.text, let image = qrCode.toIamge(frame: CGRect(origin: .zero, size:
CGSize(width: 250, height: 250))) else { return }

let dataString = image.ciImage?.recreated?.base64EncodedString()

let date = Int(Date().timeIntervalSince1970)

let model = InventoryModel(name: name, id: String(id), date: date, revisionDate:
date, category: category, personal: employee, qrCode: dataString)

warehouseStorageFacade?.saveInventory(model: model) { success in
guard success else {
self.errorNotice("Save error", autoClear: true)
return
}
self.clearAllNotice()
self.noticeSuccess("Save success", autoClear: true, autoClearTime: 1)
}

}

}

extension CIImage {
var recreated: Data? {

```

```

 UIGraphicsBeginImageContextWithOptions(CGSize(width: 250, height: 250),
 false, 1)
 defer { UIGraphicsEndImageContext() }
 UIImage(ciImage: self).draw(in: CGRect(origin: .zero, size: CGSize(width: 250,
 height: 250)))
 guard let redraw = UIGraphicsGetImageFromCurrentImageContext() else { return
 nil }
 return redraw.pngData()
 }

```

```

 func toImage(frame: CGRect) -> UIImage? {
 let scaleX = frame.size.width / self.extent.size.width
 let scaleY = frame.size.height / self.extent.size.height

```

```

 let transformedImage = self.transformed(by: CGAffineTransform(scaleX: scaleX,
 y: scaleY))
 return UIImage(ciImage: transformedImage)
 }
 }

```

```

 class AddWarehouseViewController: UIViewController {

```

```

 @IBOutlet weak var image: UIImageView!
 @IBOutlet weak var nameLabel: UITextField!
 @IBOutlet weak var descriptionTextView: UITextView!

```

```

 // MARK: - Private property

```

```

 private var warehouseStorageFacade: IWarehouseStorageFacade
 private var warehouse: WarehouseStorageModel?

```

```

private varrepeatSaveBlock: (() -> Void)?
private varimagePickerController: UIImagePickerController?
private varpreparingModel = WarehouseStorageModel(name: "", description: "",
image: UIImage())

override funcviewDidLoad() {
super.viewDidLoad()
navigationItem.rightBarButtonItem = UIBarButtonItem(title: "Готово", style:
.done, target: self, action: #selector(doneButton))
// Do any additional setup after loading the view.
}
@IBActionfuncselectImage(_ sender: Any) {
selectAlert()
}

init(warehouseStorageFacade: IWarehouseStorageFacade) {
self.warehouseStorageFacade = warehouseStorageFacade
super.init(nibName: "AddWarehouse", bundle: nil)
}
required init?(coder aDecoder: NSCoder) {
fatalError("init(coder:) has not been implemented")
}

//MARK: - Private methods

private funcselectAlert() {

imagePickerController = UIImagePickerController()
imagePickerController?.delegate = self

```

```

let alertController = UIAlertController(title: "Выбрать изображение склада",
message: nil, preferredStyle: .actionSheet)
alertController.addAction(UIAlertAction(title: "Установить из галереи",
style: .default, handler: self.chooseFromGallery))
alertController.addAction(UIAlertAction(title: "Сделать фото", style: .default,
handler: self.makePhoto))
alertController.addAction(UIAlertAction(title: "Отмена", style: .cancel, handler:
nil))
self.present(alertController, animated: true, completion: nil)
}

```

```

private func chooseFromGallery(action: UIAlertAction) {
if UIImagePickerController.isSourceTypeAvailable(.photoLibrary) {
self.imagePickerController?.sourceType = .photoLibrary
guard let imagePicker = self.imagePickerController else { return }
self.present(imagePicker, animated: true, completion: nil)
} else {
print("Нет доступа к галереи")
}
}

```

```

private func makePhoto(action: UIAlertAction) {
if UIImagePickerController.isSourceTypeAvailable(.camera) {
self.imagePickerController?.sourceType = .camera
guard let imagePicker = self.imagePickerController else { return }
self.present(imagePicker, animated: true, completion: nil)
} else {
print("Нет доступа к камере")
}
}

```

```
}
```

```
private funcsaveWarehouse() {
    self.repeatSaveBlock = {
        GradientLoadingBar.shared.show()
        //      self.activityIndicator.startAnimating()
        //      self.setEnabledButton(isEnabled: false)
        //      UserDefaults.standard.set(self.userNameField.text, forKey: "userName")
        guard let name = self.nameLabel.text,
            let description = self.descriptionTextView.text,
            let image = self.image.image else { return }
        self.warehouse = WarehouseStorageModel(name: name, description: description,
            image: image)

        guard let warehouse = self.warehouse else { return }
        self.warehouseStorageFacade.updateWarehouse(warehouses: warehouse,
            completionClosure: { (withError: Bool) in
            //      self.activityIndicator.stopAnimating()
            if withError {
                print("error")
                //      self.presentErrorAlertController()
            } else {
                self.successLoad()
                //      self.presentSuccessAlertController()
                //      self.profileButton.isHidden = true
            }
            //      self.setEnabledButton(isEnabled: true)
            //      self.editingMode = withError
        })
    }
}
```

```
self.repeatSaveBlock?()
}
```

```
private func successLoad() {
    GradientLoadingBar.shared.hide()
    self.navigationController?.popViewController(animated: true)
}
```

```
@objc private func doneButton() {
    saveWarehouse()
}
```

```
/*
```

```
// MARK: - Navigation
```

```
// In a storyboard-based application, you will often want to do a little preparation
before navigation
```

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
```

```
// Get the new view controller using segue.destination.
```

```
// Pass the selected object to the new view controller.
```

```
}
```

```
*/
```

```
}
```

```
// MARK: - UIImagePickerControllerDelegate
```

```
extension AddWarehouseViewController: UIImagePickerControllerDelegate,
    UINavigationControllerDelegate {
```

```
func imagePickerController(_ picker: UIImagePickerController,
```

```

didFinishPickingMediaWithInfo info: [UIImagePickerController.InfoKey: Any]) {
if let image = info[UIImagePickerController.InfoKey.originalImage] as? UIImage
{
self.image.image = image
preparingModel.warehouseImage = image
preparingModel.warehouseImageWasEdited = true

} else {
print("Не удалось выбрать изображение")
}
picker.dismiss(animated: true, completion: nil)
}

```

```

func collectionViewPickerController(_ picker: AddWarehouseViewController,
didFinishPickingImage image: UIImage) {

```

```

self.image.image = image
preparingModel.warehouseImage = image
preparingModel.warehouseImageWasEdited = true
picker.dismiss(animated: true, completion: nil)
}
}

```

```

class MainViewController: UIViewController {

```

```

@IBOutlet weak var mainCollectionView: UICollectionView!

```

```

var warehouseStorageFacade: IWarehouseStorageFacade?
var warehouses = [WarehouseStorageModel]()

```

```
var reference: DatabaseReference?
```

```
lazy varbackgroundView: UIView = {  
    let bgView = UIView(frame: mainCollectionView.frame)  
    let title = UILabel()  
    title.frame = CGRect(x: mainCollectionView.bounds.size.width / 2, y: 50, width:  
        mainCollectionView.frame.width, height: 50)  
    title.text = "У вас нет складов"  
    title.textAlignment = .center  
    title.font = UIFont.systemFont(ofSize: 18)  
    title.center = view.center
```

```
    bgView.addSubview(title)  
    return bgView  
}()
```

```
override funcviewDidLoad() {  
    super.viewDidLoad()
```

```
    self.warehouseStorageFacade =  
        AppDelegate.rootAssembly.presentationAssembly.warehouseStorageFacade  
    mainCollectionView.register(UINib(nibName: "WarehouseCell", bundle: nil),  
        forCellWithReuseIdentifier: "Warehouse")  
    mainCollectionView.delegate = self  
    mainCollectionView.dataSource = self
```

```
    //    configureDatabase()  
    //    sendMessage(data: ["name":"Finch"])  
}
```



```

override funcviewWillAppear(_ animated: Bool) {
super.viewWillAppear(animated)
//    self.pleaseWait()

setupDefaultData()
self.navigationController?.navigationBar.prefersLargeTitles = true
self.navigationItem.rightBarButtonItem?.isEnabled = false
}

private funcsetupStorageData() {
warehouseStorageFacade?.getWarehouse(completionClosure: { warehouses in
self.warehouses = warehouses
self.mainCollectionView.reloadData()
if warehouses.count == 0 {
self.mainCollectionView.backgroundColor = self.backgroundColor
} else {
self.mainCollectionView.backgroundColor = nil
}
}

self.clearAllNotice()
}))

}

private funcsetupDefaultData() {

guard let image = UIImage(named: "logo") else { return }
let warehouse = WarehouseStorageModel(name: "Finch", description:
"ГлавныйскладООО «Finch»", image: image)
self.warehouses = [warehouse]

```

```

}
@IBActionfuncAddAction(_ sender: Any) {
let addWarehouseViewController =
AppDelegate.rootAssembly.presentationAssembly.addWarehouseViewController(
)
self.navigationController?.pushViewController(addWarehouseViewController,
animated: true)
}
}

```

```

extension MainViewController: UICollectionViewDelegate,
UICollectionViewDataSource {
funccollectionView(_ collectionView: UICollectionView,
numberOfItemsInSection section: Int) ->Int {
return warehouses.count
}
}

```

```

funccollectionView(_ collectionView: UICollectionView,
cellForItemAtindexPath: IndexPath) ->UICollectionViewCell {
guard let cell = mainCollectionView.dequeueReusableCell(withReuseIdentifier:
"Warehouse", for: indexPath) as? WarehouseCell else { return
UICollectionViewCell() }
let warehouse = warehouses[indexPath.row]
cell.title.text = warehouse.warehouseName
cell.about.text = warehouse.warehouseDescription
cell.image.image = warehouse.warehouseImage
return cell
}
}

```

```
func collectionView(_ collectionView: UICollectionView,
didSelectItemAtindexPath: IndexPath) {

let vc =
AppDelegate.rootAssembly.presentationAssembly.inventoryTableViewController(
)
self.navigationController?.pushViewController(vc, animated: true)
}
}
```

```
extension WarehouseCell {
struct Model {
let warehouseImage: UIImage
let warehouseName: String
let warehouseDescription: String
}
}
```

```
class WarehouseCell: UICollectionViewCell {
```

```
@IBOutlet weak var title: UILabel!
@IBOutlet weak var about: UILabel!
@IBOutlet weak var image: UIImageView!
```

```
override func awakeFromNib() {
super.awakeFromNib()
```

```
let blurEffect = UIBlurEffect(style: UIBlurEffect.Style.dark)
```

```

let blurEffectView = UIVisualEffectView(effect: blurEffect)
blurEffectView.frame = image.bounds
blurEffectView.autoresizingMask = [.flexibleWidth, .flexibleHeight]
image.addSubview(blurEffectView)
image.layer.cornerRadius = 12
}

}

}

```

```

protocol IPresentationAssembly {

```

```

    varwarehouseStorageFacade: IWarehouseStorageFacade { get }

```

```

    funcmainViewController() ->MainViewController

```

```

    funcaddWarehouseViewController() ->AddWarehouseViewController

```

```

    funcaddInventoryViewController(model: InventoryModel?) -

```

```

    >AddInventoryViewController

```

```

    funcqrCodeViewController(id: String, delegate: QRCodeViewControllerDelegate)

```

```

    ->QRCodeViewController

```

```

    funcdetailInventoryViewController(model: InventoryModel) -

```

```

    >DetailInventoryViewController

```

```

    funcfilerViewController(completion: @escaping (_ model: Filter) -> Void) -

```

```

    >FilterViewController

```

```

    funcreportTableViewController(models: [InventoryModel]) -

```

```

    >ReportTableViewController

```

```

    funcreportChar(models: [InventoryModel]) ->ChartViewController

```

```
func revisionViewController(models: [InventoryModel]) -> RevisionViewController {
    func inventoryTableViewController() -> UITableViewController {
    }
}
```

```
class PresentationAssembly: IPresentationAssembly {
```

```
    lazy var warehouseStorageFacade: IWarehouseStorageFacade =
        WarehouseStorageFacade(warehouseStorageService:
            self.serviceAssembly.warehouseStorageService, warehouseCoreDataStorage:
            self.serviceAssembly.warehouseStorageService as! WarehouseCoreDataStorage)
```

```
    private let serviceAssembly: IServicesAssembly
```

```
    init(serviceAssembly: IServicesAssembly) {
        self.serviceAssembly = serviceAssembly
    }
}
```

```
//MARK: - View Controllers
```

```
func mainViewController() -> MainViewController {
    let view = MainViewController()
    view.warehouseStorageFacade = warehouseStorageFacade
    return view
}
```

```
func addWarehouseViewController() -> AddWarehouseViewController {
```

```
let view = AddWarehouseViewController(warehouseStorageFacade:  
warehouseStorageFacade)
```

```
return view  
}
```

```
funcaddInventoryViewController(model: InventoryModel?) -
```

```
>AddInventoryViewController {
```

```
let view = AddInventoryViewController()
```

```
view.model = model
```

```
return view
```

```
}
```

```
funcdetailInventoryViewController(model: InventoryModel) -
```

```
>DetailInventoryViewController {
```

```
let view = DetailInventoryViewController()
```

```
view.model = model
```

```
return view
```

```
}
```

```
funcqrCodeViewController(id: String, delegate: QRCodeViewControllerDelegate)
```

```
->QRCodeViewController {
```

```
let view = QRCodeViewController()
```

```
view.id = id
```

```
view.delegate = delegate
```

```
return view
```

```
}
```

```
funcinventoryTableViewController() ->UIViewController {
    let storyboard = UIStoryboard(name: "Inventory", bundle: nil)

    guard let view = storyboard.instantiateInitialViewController() else {
        return InventoryTableViewController()
    }
}
```

```
return view
}
```

```
funcreportTableViewController(models: [InventoryModel]) -
>ReportTableViewController {
    let view = ReportTableViewController()
    view.models = models
    return view
}
```

```
funcreportChar(models: [InventoryModel]) ->ChartViewController {
    let view = ChartViewController()
    view.models = models
    return view
}
```

```
funcfilerViewController(completion: @escaping (Filter) -> Void) -
>FilterViewController {
    let view = FilterViewController()
    view.completion = completion
    return view
}
```

```
func revisionViewController(models: [InventoryModel]) -
```

```
> RevisionViewController {
```

```
    let view = RevisionViewController()
```

```
    view.existingModels = models
```

```
    return view
```

```
}
```

```
}
```

```
let service = "www.finch.fm"
```

```
let kSecClassValue = NSString(format: kSecClass)
```

```
let kSecAttrAccountValue = NSString(format: kSecAttrAccount)
```

```
let kSecValueDataValue = NSString(format: kSecValueData)
```

```
let kSecClassGenericPasswordValue = NSString(format:  
kSecClassGenericPassword)
```

```
let kSecAttrServiceValue = NSString(format: kSecAttrService)
```

```
let kSecMatchLimitValue = NSString(format: kSecMatchLimit)
```

```
let kSecReturnDataValue = NSString(format: kSecReturnData)
```

```
let kSecMatchLimitOneValue = NSString(format: kSecMatchLimitOne)
```

```
public class KeychainService: NSObject {
```

```
    class func updatePassword(data: String) {
```

```
        if let dataFromString: Data = data.data(using: String.Encoding.utf8,  
        allowLossyConversion: false) {
```

```
            // Instantiate a new default keychain query
```

```
            let keychainQuery: NSMutableDictionary = NSMutableDictionary(objects:
```

```
            [kSecClassGenericPasswordValue, service, "Finch"], forKeys: [kSecClassValue,  
            kSecAttrServiceValue, kSecAttrAccountValue])
```



```
let status = SecItemUpdate(keychainQuery as CFDictionary,  
[kSecValueDataValue:dataFromString] as CFDictionary)
```

```
if (status != errSecSuccess) {  
if let err = SecCopyErrorMessageString(status, nil) {  
print("Read failed: \(err)")  
}  
}  
}  
}
```

```
class func removePassword() {
```

```
// Instantiate a new default keychain query
```

```
let keychainQuery: NSMutableDictionary = NSMutableDictionary(objects:  
[kSecClassGenericPasswordValue, service, "Finch", kCFBooleanTrue], forKeys:  
[kSecClassValue, kSecAttrServiceValue, kSecAttrAccountValue,  
kSecReturnDataValue])
```

```
// Delete any existing items
```

```
let status = SecItemDelete(keychainQuery as CFDictionary)  
if (status != errSecSuccess) {  
if let err = SecCopyErrorMessageString(status, nil) {  
print("Remove failed: \(err)")  
}  
}  
  
}
```

```

class funcsavePassword(data: String) {
    if let dataFromString = data.data(using: String.Encoding.utf8,
        allowLossyConversion: false) {

        // Instantiate a new default keychain query
        let keychainQuery: NSMutableDictionary = NSMutableDictionary(objects:
            [kSecClassGenericPasswordValue, service, "Finch", dataFromString], forKeys:
            [kSecClassValue, kSecAttrServiceValue, kSecAttrAccountValue,
            kSecValueDataValue])

        // Add the new keychain item
        let status = SecItemAdd(keychainQuery as CFDictionary, nil)

        if (status != errSecSuccess) {    // Always check the status
            if let err = SecCopyErrorMessageString(status, nil) {
                print("Write failed: \(err)")
            }
        }
    }
}

class funcloadPassword() -> String? {
    // Instantiate a new default keychain query
    // Tell the query to return a result
    // Limit our results to one item
    let keychainQuery: NSMutableDictionary = NSMutableDictionary(objects:
        [kSecClassGenericPasswordValue, service, "Finch", kCFBooleanTrue,
        kSecMatchLimitOneValue], forKeys: [kSecClassValue, kSecAttrServiceValue,
        kSecAttrAccountValue, kSecReturnDataValue, kSecMatchLimitValue])

```

```

var dataTypeRef : AnyObject?

// Search for the keychain items
let status: OSStatus = SecItemCopyMatching(keychainQuery, &dataTypeRef)
var contentsOfKeychain: String?

if status == errSecSuccess {
    if let retrievedData = dataTypeRef as? Data {
        contentsOfKeychain = String(data: retrievedData, encoding: String.Encoding.utf8)
    }
    } else {
        print("Nothing was retrieved from the keychain. Status code \(status)")
    }

return contentsOfKeychain
}

}

class WarehouseGCDStorageService: IWarehouseStorageService {
    private let warehouseManager: WarehouseFileStorage = WarehouseFileStorage()

    func saveWarehouse(warehouse: WarehouseStorageModel, completionClosure:
        @escaping (Bool) -> Void) {
        let globalQueue = DispatchQueue.global(qos: .utility)
        globalQueue.async {
            let hasError = self.warehouseManager.save(warehouses: warehouse)
            DispatchQueue.main.async {
                completionClosure(hasError)
            }
        }
    }
}

```

```
}  
}
```

```
funcsaveCategory(category: String, position: Int, completionClosure: @escaping  
(Bool) -> Void) {  
    print("WarehouseGCDStorageService")  
}
```

```
funcsaveEmployee(employee: String, position: Int, completionClosure: @escaping  
(Bool) -> Void) {  
    print("WarehouseGCDStorageService")  
}
```

```
funcsaveInventoryToFirebase(model: InventoryModel, completionClosure:  
@escaping (Bool) -> Void) {  
    print("WarehouseGCDStorageService")  
}
```

```
funcsaveInventoryToFirebase(completionHandler: (() -> Void)?) {  
    print("WarehouseGCDStorageService")  
}
```

```
funcfetchOrgData(completionClosure: @escaping ([String]?, [String]?, String?) ->  
Void) {  
    print("WarehouseGCDStorageService")  
}
```

```
funcfetchInvetry(completionClosure: @escaping ([InventoryModel]?, _ error:  
String?) -> Void) {
```

```
print("WarehouseGCDStorageService")
}
```

```
funcdeleteInventoryFromFirebase(model: InventoryModel, completionClosure:
@escaping (Bool) -> Void) {
print("WarehouseGCDStorageService")
}
```

```
funcloadWarehouse(completionClosure: @escaping ([WarehouseStorageModel]) -
> Void) {
let globalQueue = DispatchQueue.global(qos: .utility)
globalQueue.async {
let warehouses = self.warehouseManager.get()
DispatchQueue.main.async {
//MARK: - TODO
guard let warehouse = warehouses else { return }
completionClosure([warehouse])
}
}
}

}
```

```
class WarehouseCoreDataStorage: IWarehouseStorageService {
```

```
private vardataStack: ICoreDataStack
private varwarehouseReference: DatabaseReference
private varcategoryReference: DatabaseReference
private varpersonalReference: DatabaseReference
```

```

init(dataStack: ICoreDataStack, reference: DatabaseReference, categoryReference:
DatabaseReference, personalReference: DatabaseReference) {
self.dataStack = dataStack
self.warehouseReference = reference
self.categoryReference = categoryReference
self.personalReference = personalReference
}

```

```

funcsaveWarehouse(warehouse: WarehouseStorageModel, completionClosure:
@escaping (Bool) -> Void) {
let context = self.dataStack.saveContext
if let waregouseArray = Warehouse.findOrInsertWarehouses(in: context), let last =
waregouseArray.last {
last.name = warehouse.warehouseName
last.about = warehouse.warehouseDescription
last.image = warehouse.warehouseImage
self.dataStack.performSave(context: context) {
DispatchQueue.main.async {
completionClosure(false)
}
}
}
}

```

```

funcfetchInvetory(completionClosure: @escaping ([InventoryModel]?, _ error:
String?) -> Void) {
var models = [InventoryModel]() {

```

```

didSet {
  completionClosure(models, nil)
}
}

warehouseReference.observe(.childAdded, with: { snap in

  let id = snap.key

  guard let dict = snap.value as? [String : Any],
        let name = dict["name"] as? String,
        let date = dict["date"] as? Int,
        let category = dict["category"] as? String,
        let personal = dict["personal"] as? String else {
    completionClosure(nil, "Parse error")
    return
  }
  let qrCodeString = dict["qrCode"] as? String
  var revisionDate = 0

  if let revDate = dict["revisionDate"] as? Int {
    revisionDate = revDate
  }

  let model = InventoryModel(name: name, id: id, date: date, revisionDate:
    revisionDate, category: category, personal: personal, qrCode: qrCodeString)

  models.append(model)
}) { error in
  completionClosure(nil, error.localizedDescription)
}

```

```
}
```

```
}
```

```
fun saveCategory(category: String, position: Int, completionClosure: @escaping  
(Bool) -> Void) {  
    categoryReference.child("category").child(String(position)).setValue(category)  
    completionClosure(true)  
}
```

```
fun saveEmployee(employee: String, position: Int, completionClosure: @escaping  
(Bool) -> Void) {  
    personalReference.child("employee").child(String(position)).setValue(employee)  
    completionClosure(true)  
}
```

```
fun fetchOrgData(completionClosure: @escaping ([String]?, [String]?, String?) ->  
Void) {
```

```
    personalReference.observe(.childAdded, with: { personalSnap in  
        guard let personal = personalSnap.value as? [String] else { return }  
        self.categoryReference.observe(.childAdded, with: { categorySnap in  
            guard let category = categorySnap.value as? [String] else { return }  
            completionClosure(personal, category, nil)  
        }) { error in  
            completionClosure(nil, nil, error.localizedDescription)  
        }  
    })
```



```
}
```

```
funcsaveInventoryToFirebase(model: InventoryModel, completionClosure:  
@escaping (Bool) -> Void) {  
    warehouseReference.child(model.id).setValue(model.sendModel())  
    completionClosure(true)  
}
```

```
funcdeleteInventoryFromFirebase(model: InventoryModel, completionClosure:  
@escaping (Bool) -> Void) {
```

```
    warehouseReference.observe(.childAdded) { snap in  
        let id = snap.key  
        if model.id == id {  
            self.warehouseReference.child(id).removeValue()  
            completionClosure(true)  
        }  
    }  
}
```

```
funcloadWarehouse(completionClosure: @escaping ([WarehouseStorageModel]) -  
> Void) {  
    let context = self.dataStack.saveContext  
    if let warehouseArray = Warehouse.findOrInsertWarehouses(in: context) {  
        var array: [WarehouseStorageModel] = []  
  
        warehouseArray.forEach({ model in  
            if let name = model.name,  
                let about = model.about,  
                let image = model.image as? UIImage {
```

```

let warehouse = WarehouseStorageModel(name: name,
description: about,
image: image)
array.append(warehouse)
} else {
completionClosure([WarehouseStorageModel]())
}
})
DispatchQueue.main.async {
completionClosure(array)
}
}
}

}

```

```

protocol IWarehouseStorageService {
funcfetchInventory(completionClosure: @escaping (_ Inventory:
[InventoryModel]?, _ error: String?) -> Void)
funcfetchOrgData(completionClosure: @escaping (_ personal: [String]?, _
category: [String]?, _ error: String?) -> Void)
funcsaveInventoryToFirebase(model: InventoryModel, completionClosure:
@escaping (Bool) -> Void)
funcsaveCategory(category: String, position: Int, completionClosure: @escaping
(Bool) -> Void)
funcsaveEmployee(employee: String, position: Int, completionClosure: @escaping
(Bool) -> Void)
funcdeleteInventoryFromFirebase(model: InventoryModel, completionClosure:
@escaping (Bool) -> Void)

```

```

funcsaveWarehouse(warehouse: WarehouseStorageModel, completionClosure:
@escaping (_ withError: Bool) -> Void)
funcloadWarehouse(completionClosure: @escaping (_ warehouses:
[WarehouseStorageModel]) -> Void)
}

```

```

class WarehouseFileStorage {

private let imageFilename = "titleImage.png"
private let warehouseFilename = "warehouseName.txt"
private let descriptionFilename = "description.txt"
func save(warehouses: WarehouseStorageModel) -> Bool {
do {
let filePath = self.getDocumentsDirectory()
if warehouses.warehouseNameWasEdited {
try warehouses.warehouseName.write(to:
filePath.appendingPathComponent(self.warehouseFilename),
atomically: false,
encoding: String.Encoding.utf8)
}

if warehouses.descriptionWasEdited {
try warehouses.warehouseDescription.write(to:
filePath.appendingPathComponent(self.descriptionFilename),
atomically: false,
encoding: String.Encoding.utf8)
}

if warehouses.warehouseImageWasEdited {

```

```

let imageData = warehouses.warehouseImage.pngData()
try imageData?.write(to: filePath.appendingPathComponent(self.imageFilename),
options: .atomic)
}
return false
} catch {
return true
}
}

```

```

func get() -> WarehouseStorageModel? {
do {
let filePath = getDocumentsDirectory()
let warehouseName = try String(contentsOf:
filePath.appendingPathComponent(self.warehouseFilename))
let warehouseDescription = try String(contentsOf:
filePath.appendingPathComponent(self.descriptionFilename))
let warehouseImage = UIImage(contentsOfFile:
filePath.appendingPathComponent(self.imageFilename).path) ?? UIImage()
return WarehouseStorageModel(name: warehouseName, description:
warehouseDescription, image: warehouseImage)
} catch {
return nil
}
}

private func getDocumentsDirectory() -> URL {
return FileManager.default.urls(for: .documentDirectory, in: .userDomainMask)[0]
}
}

```

```

protocol ICoreDataService {
    varmainContext: NSManagedObjectContext {get}
    varsaveContext: NSManagedObjectContext {get}
    funcperformSave(context: NSManagedObjectContext,
        completionHandler: (() -> Void)?)
}

class CoreDataService: ICoreDataService {

    private varcoreData: ICoreDataStack
    varmainContext: NSManagedObjectContext {
        get { return self.coreData.mainContext }
        set {}
    }

    varsaveContext: NSManagedObjectContext { return self.coreData.saveContext }

    init(coreData: ICoreDataStack) {
        self.coreData = coreData
    }

    funcperformSave(context: NSManagedObjectContext, completionHandler: (() ->
        Void)?) {
        self.coreData.performSave(context: context, completionHandler:
            completionHandler)
    }

}

protocol IServicesAssembly {
    varwarehouseStorageService: IWarehouseStorageService {get}

```

```
varcoreDataService: ICoreDataService {get}
}
```

```
class ServicesAssembly: IServicesAssembly {
```

```
private var reference: DatabaseReference
private varcategoryReference: DatabaseReference
private varpersonalReference: DatabaseReference
private let coreAssembly: ICoreAssembly
```

```
init(coreAssembly: ICoreAssembly) {
self.coreAssembly = coreAssembly
let ref = Database.database().reference()
self.reference = ref.child("warehouse-145b8")
self.personalReference = ref.child("Employee")
self.categoryReference = ref.child("Category")
}
```

```
lazy varwarehouseStorageService: IWarehouseStorageService =
WarehouseCoreDataStorage(dataStack: self.coreAssembly.coreDataStack,
reference: reference, categoryReference: categoryReference, personalReference:
personalReference)
lazy varcoreDataService: ICoreDataService = CoreDataService(coreData:
self.coreAssembly.coreDataStack)
}
```

```
extension Warehouse {
```

```
static funcinsertWarehouses(in context: NSManagedObjectContext) ->
Warehouse? {
```

```

if let warehouse = NSEntityDescription.insertNewObject(forEntityName:
"Warehouse", into: context) as? Warehouse {
return warehouse
}
return nil
}

```

```

static func findOrInsertWarehouses(in context: NSManagedObjectContext) ->
[Warehouse]? {
guard let model = context.persistentStoreCoordinator?.managedObjectModel else
{
print ("Model is not available in context")
assert(false)
return nil
}
var warehouse: [Warehouse] = []
guard let fetchRequest = Warehouse.fetchRequestWarehouse(model: model) else {
return nil
}

```

```

do {
let results = try context.fetch(fetchRequest)
results.forEach({ warehouse.append($0) })
if let newWarehouse = Warehouse.insertWarehouses(in: context) {
warehouse.append(newWarehouse)
}
} catch {
print ("Failed to fetch warehouse: \(error)")
}

```

```

if warehouse.count == 0, let newWarehouse = Warehouse.insertWarehouses(in:
context) {
warehouse.append(newWarehouse)
}
return warehouse
}

```

```

static funcfetchRequestWarehouse(model: NSManagedObjectModel) -
>NSFetchRequest<Warehouse>? {

```

```

let templateName = "Warehouse"
guard let fetchRequest = model.fetchRequestTemplate(forName: templateName)
as? NSFetchRequest<Warehouse> else {
assert(false, "No template with name \"(templateName)\")
return nil
}
return fetchRequest
}
}

```

```

protocol ICoreDataStack {
varmasterContext: NSManagedObjectContext {get}
varmainContext: NSManagedObjectContext {get}
varsaveContext: NSManagedObjectContext {get}
funcperformSave(context: NSManagedObjectContext, completionHandler: (() ->
Void)?)
}

```

```

class CoreDataStack: ICoreDataStack {
let dataModelName = "Warehouse"

```



```

let dataModelExtension = "momd"
lazy varmanagedObjectModel: NSManagedObjectModel = {
let modelURL = Bundle.main.url(forResource: self.dataModelName,
withExtension: self.dataModelExtension)!
return NSManagedObjectModel(contentsOf: modelURL)!
}()
lazy varpersistentStoreCoordinator: NSPersistentStoreCoordinator = {
let coordinator = NSPersistentStoreCoordinator(managedObjectModel:
self.managedObjectModel)
do {
try coordinator.addPersistentStore(ofType: NSSQLiteStoreType,
configurationName: nil,
at: self.storeURL,
options: nil)
} catch {
assert(false, "Error adding store \(error)")
}
return coordinator
}()
varstoreURL: URL {
let documentsURL = FileManager.default.urls(for: .documentDirectory, in:
.userDomainMask).first!
return documentsURL.appendingPathComponent("Warehouse.sqlite")
}
lazy varmasterContext: NSManagedObjectContext = {
varmasterContext = NSManagedObjectContext(concurrencyType:
.privateQueueConcurrencyType)
masterContext.persistentStoreCoordinator = self.persistentStoreCoordinator
masterContext.mergePolicy = NSOverwriteMergePolicy
return masterContext
}

```

```

}()
lazy varmainContext: NSManagedObjectContext = {
varmainContext = NSManagedObjectContext(concurrencyType:
.mainQueueConcurrencyType)
mainContext.parent = self.masterContext
mainContext.mergePolicy = NSOverwriteMergePolicy
return mainContext
}()
lazy varsaveContext: NSManagedObjectContext = {
varsaveContext = NSManagedObjectContext(concurrencyType:
.privateQueueConcurrencyType)
saveContext.parent = self.mainContext
saveContext.mergePolicy = NSOverwriteMergePolicy
return saveContext
}()
funcperformSave(context: NSManagedObjectContext, completionHandler: (() ->
Void)?) {
if context.hasChanges {
context.perform { [weak self] in
do {
try context.save()
} catch {
print("Context save error: \(error)")
}
if let parent = context.parent {
self?.performSave(context: parent, completionHandler: completionHandler)
} else {
completionHandler?()
}
}
}
}
}

```

```

    } else {
    completionHandler?()
    }
    }
    }

```

```

protocol ICoreAssembly {
varcoreDataStack: ICoreDataStack {get}
}

```

```

class CoreAssembly: ICoreAssembly {
lazy varcoreDataStack: ICoreDataStack = CoreDataStack()
}

```

```

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

```

```

var window: UIWindow?
static let rootAssembly = RootAssembly()

```

```

func application(_ application: UIApplication,
didFinishLaunchingWithOptionslaunchOptions:
[UIApplication.LaunchOptionsKey: Any]?) -> Bool {

```

```

FirebaseApp.configure()
GradientLoadingBar.shared = GradientLoadingBar(
height: 3.0,
durations: Durations(fadeIn: 1.0, fadeOut: 2.0, progress: 3.00),
gradientColorList: [

```

```

        .red, .yellow, .green
    ]
)
setNavigarionColor()

```

```

return true
}

```

```

private funcsetNavigarionColor() {
//    UINavigationController.appearance().setBackgroundImage(UIColor(), for:
UIBarMetrics.default)
//    UINavigationController.appearance().shadowImage = UIImage()
//    UINavigationController.appearance().isTranslucent = true
//    UINavigationController.appearance().backgroundColor = UIColor.clear
}

```

```

funcapplicationWillResignActive(_ application: UIApplication) {
// Sent when the application is about to move from active to inactive state. This can
occur for certain types of temporary interruptions (such as an incoming phone call
or SMS message) or when the user quits the application and it begins the transition
to the background state.
// Use this method to pause ongoing tasks, disable timers, and invalidate graphics
rendering callbacks. Games should use this method to pause the game.
}

```

```

funcapplicationDidEnterBackground(_ application: UIApplication) {
// Use this method to release shared resources, save user data, invalidate timers,
and store enough application state information to restore your application to its
current state in case it is terminated later.
}

```

// If your application supports background execution, this method is called instead of applicationWillTerminate: when the user quits.

}

func applicationWillEnterForeground(_ application: UIApplication) {

// Called as part of the transition from the background to the active state; here you can undo many of the changes made on entering the background.

}

func applicationDidBecomeActive(_ application: UIApplication) {

// Restart any tasks that were paused (or not yet started) while the application was inactive. If the application was previously in the background, optionally refresh the user interface.

}

func applicationWillTerminate(_ application: UIApplication) {

// Called when the application is about to terminate. Save data if appropriate. See also applicationDidEnterBackground:.

// Saves changes in the application's managed object context before the application terminates.

self.saveContext()

}

// MARK: - Core Data stack

lazy var persistentContainer: NSPersistentContainer = {

/*

The persistent container for the application. This implementation creates and returns a container, having loaded the store for the application to it. This property is optional since there are legitimate

error conditions that could cause the creation of the store to fail.

```
*/
```

```
let container = NSPersistentContainer(name: "Warehouse")
```

```
container.loadPersistentStores(completionHandler: { (storeDescription, error) in
```

```
if let error = error as NSError? {
```

```
// Replace this implementation with code to handle the error appropriately.
```

```
// fatalError() causes the application to generate a crash log and terminate. You  
should not use this function in a shipping application, although it may be useful  
during development.
```

```
/*
```

Typical reasons for an error here include:

- * The parent directory does not exist, cannot be created, or disallows writing.

- * The persistent store is not accessible, due to permissions or data protection when the device is locked.

- * The device is out of space.

- * The store could not be migrated to the current model version.

Check the error message to determine what the actual problem was.

```
*/
```

```
fatalError("Unresolved error \(error), \(error.userInfo)")
```

```
}
```

```
})
```

```
return container
```

```
()
```

```
// MARK: - Core Data Saving support
```

```
func saveContext () {
```

```
let context = persistentContainer.viewContext
```

```
if context.hasChanges {
```

```

do {
try context.save()
} catch {
// Replace this implementation with code to handle the error appropriately.
// fatalError() causes the application to generate a crash log and terminate. You
should not use this function in a shipping application, although it may be useful
during development.
let nserror = error as NSError
fatalError("Unresolved error \(nserror), \(nserror.userInfo)")
}
}
}

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
<key>NSFaceIDUsageDescription</key>
<string>$(PRODUCT_NAME) запрашивает доступ к использованию FaceID
для аутентификации пользователя</string>
<key>NSCameraUsageDescription</key>
<string>$(PRODUCT_NAME) Разрешите использование камеры для чтения QR-
Кода</string>
<key>NSPhotoLibraryUsageDescription</key>
<string>$(PRODUCT_NAME) Using the library to set the warehouse
image</string>
<key>CFBundleDevelopmentRegion</key>
<string>$(DEVELOPMENT_LANGUAGE)</string>

```

```
<key>CFBundleExecutable</key>
<string>$(EXECUTABLE_NAME)</string>
<key>CFBundleIdentifier</key>
<string>$(PRODUCT_BUNDLE_IDENTIFIER)</string>
<key>CFBundleInfoDictionaryVersion</key>
<string>6.0</string>
<key>CFBundleName</key>
<string>$(PRODUCT_NAME)</string>
<key>CFBundlePackageType</key>
<string>APPL</string>
<key>CFBundleShortVersionString</key>
<string>1.0</string>
<key>CFBundleVersion</key>
<string>1</string>
<key>LSRequiresIPhoneOS</key>
<true/>
<key>UILaunchStoryboardName</key>
<string>LaunchScreen</string>
<key>UIMainStoryboardFile</key>
<string>Main</string>
<key>UIRequiredDeviceCapabilities</key>
<array>
<string>armv7</string>
</array>
<key>UISupportedInterfaceOrientations</key>
<array>
<string>UIInterfaceOrientationPortrait</string>
<string>UIInterfaceOrientationLandscapeLeft</string>
<string>UIInterfaceOrientationLandscapeRight</string>
</array>
```



```

<key>UISupportedInterfaceOrientations~ipad</key>
<array>
<string>UIInterfaceOrientationPortrait</string>
<string>UIInterfaceOrientationPortraitUpsideDown</string>
<string>UIInterfaceOrientationLandscapeLeft</string>
<string>UIInterfaceOrientationLandscapeRight</string>
</array>
</dict>
</plist>

```

Uncomment the next line to define a global platform for your project

platform :ios, '8.0'

source 'https://github.com/CocoaPods/Specs.git'

target 'Warehouse' do

Comment the next line if you're not using Swift and don't want to use dynamic frameworks

use_frameworks!

pod 'QRCodeReader.swift', '~> 10.0.0'

pod 'GradientLoadingBar', '~> 1.0'

pod 'Firebase/Core'

pod 'Firebase/Database'

pod 'Charts'

end

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
```

```
toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"
```

```
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
```

```
useSafeAreas="YES" colorMatched="YES">
```

```

<device id="retina6_1" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="RevisionViewController" customModule="Warehouse"
customModuleProvider="target">
<connections>
<outlet property="checkButton" destination="cXl-YM-Efw" id="79n-aO-tZz"/>
<outlet property="idField" destination="wir-zX-iSY" id="ABJ-FS-Jlg"/>
<outlet property="scanButton" destination="W1G-S9-Afu" id="AdU-IS-aUQ"/>
<outlet property="tableView" destination="dYb-uD-WZb" id="j07-r8-Fai"/>
<outlet property="view" destination="Kjf-uF-cnC" id="WC1-gX-coa"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view contentMode="scaleToFill" id="Kjf-uF-cnC">
<rect key="frame" x="0.0" y="0.0" width="414" height="896"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>

```

```

<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="W1G-S9-Afu">
<rect key="frame" x="160.5" y="786" width="93" height="30"/>
<color key="backgroundColor" red="0.51372549019607838"
green="0.78582151200000006" blue="1" alpha="1" colorSpace="custom"
customColorSpace="sRGB"/>
<state key="normal" title="Сканировать"/>
<connections>
<action selector="scanAction:" destination="-1" eventType="touchUpInside"
id="gh5-wm-nQi"/>
</connections>
</button>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="Введитеидентификатор"
textAlignment="natural" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="wir-zX-iSY">
<rect key="frame" x="12" y="742" width="326" height="30"/>
<constraints>
<constraint firstAttribute="width" relation="greaterThanOrEqual" constant="200"
id="Bf8-cX-GIi"/>
</constraints>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits" keyboardType="numberPad"
returnKeyType="done"/>
</textField>

```

```

<button opaque="NO".contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="cXl-YM-Efw">
<rect key="frame" x="350" y="742" width="52" height="30"/>
<color key="backgroundColor" red="0.51372549019607838"
green="0.78431372549019607" blue="1" alpha="1" colorSpace="custom"
customColorSpace="sRGB"/>
<constraints>
<constraint firstAttribute="width" constant="52" id="2Z0-xy-11v"/>
</constraints>
<state key="normal" title="Учесть"/>
<connections>
<action selector="checkAction:" destination="-1" eventType="touchUpInside"
id="izf-Y7-Ya3"/>
</connections>
</button>
<tableView clipsSubviews="YES".contentMode="scaleToFill"
alwaysBounceVertical="YES" style="plain" separatorStyle="default"
rowHeight="-1" estimatedRowHeight="-1" sectionHeaderHeight="28"
sectionFooterHeight="28" translatesAutoresizingMaskIntoConstraints="NO"
id="dYb-uD-WZb">
<rect key="frame" x="0.0" y="44" width="414" height="670"/>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
</tableView>
</subviews>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<gestureRecognizers/>

```

```

<constraints>
<constraint firstItem="wir-zX-iSY" firstAttribute="centerY" secondItem="cXl-
YM-Efw" secondAttribute="centerY" id="2wG-jq-1T2"/>
<constraint firstItem="Q5n-Ol-QOJ" firstAttribute="trailing" secondItem="cXl-
YM-Efw" secondAttribute="trailing" constant="12" id="AVs-lq-3qP"/>
<constraint firstItem="dYb-uD-WZb" firstAttribute="top" secondItem="Q5n-Ol-
QOJ" secondAttribute="top" id="EdR-dX-x68"/>
<constraint firstItem="wir-zX-iSY" firstAttribute="top" secondItem="dYb-uD-
WZb" secondAttribute="bottom" constant="28" id="LX7-ys-qUB"/>
<constraint firstItem="W1G-S9-Afu" firstAttribute="centerX" secondItem="Q5n-
Ol-QOJ" secondAttribute="centerX" id="LeV-oS-xgV"/>
<constraint firstItem="W1G-S9-Afu" firstAttribute="top" secondItem="cXl-YM-
Efw" secondAttribute="bottom" constant="14" id="MrL-0g-LYc"/>
<constraint firstItem="wir-zX-iSY" firstAttribute="leading" secondItem="Q5n-Ol-
QOJ" secondAttribute="leading" constant="12" id="RdM-K3-azu"/>
<constraint firstItem="cXl-YM-Efw" firstAttribute="leading" secondItem="wir-
zX-iSY" secondAttribute="trailing" constant="12" id="VOM-1O-Uke"/>
<constraint firstItem="Q5n-Ol-QOJ" firstAttribute="trailing" secondItem="dYb-
uD-WZb" secondAttribute="trailing" id="Z70-OB-R2R"/>
<constraint firstItem="Q5n-Ol-QOJ" firstAttribute="bottom" secondItem="W1G-
S9-Afu" secondAttribute="bottom" constant="46" id="p3l-gg-jLU"/>
<constraint firstItem="dYb-uD-WZb" firstAttribute="leading" secondItem="Q5n-
Ol-QOJ" secondAttribute="leading" id="u28-Of-MSS"/>
</constraints>
<viewLayoutGuide key="safeArea" id="Q5n-Ol-QOJ"/>
<connections>
<outletCollection property="gestureRecognizers" destination="CYi-A3-JN1"
appends="YES" id="2wz-Lc-pbA"/>
</connections>

```

```

<point key="canvasLocation" x="47.826086956521742"
y="35.491071428571423"/>
</view>
<tapGestureRecognizer id="CYi-A3-JN1">
<connections>
<action selector="tapOnView:" destination="-1" id="koA-Gl-wJ6"/>
</connections>
</tapGestureRecognizer>
</objects>
</document>

<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina6_1" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>

```

```

<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="ChartViewController" customModule="Warehouse"
customModuleProvider="target">
<connections>
<outlet property="chartView" destination="1eC-q7-xj9" id="5Eg-1u-jLq"/>
<outlet property="lineChartView" destination="cdv-iG-a1a" id="ssT-Sa-wah"/>
<outlet property="view" destination="i5M-Pr-FkT" id="sfx-zR-JGt"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view clearsContextBeforeDrawing="NO" contentMode="scaleToFill" id="i5M-
Pr-FkT">
<rect key="frame" x="0.0" y="0.0" width="414" height="896"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<view contentMode="scaleToFill"
translatesAutoresizingMaskIntoConstraints="NO" id="1eC-q7-xj9"
customClass="BarChartView" customModule="Charts">
<rect key="frame" x="0.0" y="44" width="414" height="448"/>
<color key="backgroundColor" white="0.8599999999999999"
alpha="0.56000000000000005" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
</view>
<view contentMode="scaleToFill"
translatesAutoresizingMaskIntoConstraints="NO" id="cdv-iG-a1a"
customClass="PieChartView" customModule="Charts">
<rect key="frame" x="0.0" y="492" width="414" height="370"/>

```

```

<color key="backgroundColor" white="0.85866152968036513"
alpha="0.55963720034246578" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
</view>
</subviews>
<color key="backgroundColor" red="1" green="1" blue="1" alpha="1"
colorSpace="custom" customColorSpace="sRGB"/>
<constraints>
<constraint firstItem="fnl-2z-Ty3" firstAttribute="trailing" secondItem="1eC-q7-
xj9" secondAttribute="trailing" id="BLd-EM-MCj"/>
<constraint firstItem="fnl-2z-Ty3" firstAttribute="bottom" secondItem="cdv-iG-
a1a" secondAttribute="bottom" id="WHS-hN-Ge8"/>
<constraint firstItem="1eC-q7-xj9" firstAttribute="leading" secondItem="fnl-2z-
Ty3" secondAttribute="leading" id="YUZ-Oo-5Mf"/>
<constraint firstItem="cdv-iG-a1a" firstAttribute="leading" secondItem="fnl-2z-
Ty3" secondAttribute="leading" id="j2s-S7-6M8"/>
<constraint firstItem="cdv-iG-a1a" firstAttribute="top" secondItem="1eC-q7-xj9"
secondAttribute="bottom" id="jTi-j1-VwT"/>
<constraint firstItem="fnl-2z-Ty3" firstAttribute="trailing" secondItem="cdv-iG-
a1a" secondAttribute="trailing" id="mqK-Gx-2M7"/>
<constraint firstItem="1eC-q7-xj9" firstAttribute="height" secondItem="i5M-Pr-
FkT" secondAttribute="height" multiplier="0.5" id="pQE-L0-cM6"/>
<constraint firstItem="1eC-q7-xj9" firstAttribute="top" secondItem="fnl-2z-Ty3"
secondAttribute="top" id="yLY-5K-WqQ"/>
</constraints>
<viewLayoutGuide key="safeArea" id="fnl-2z-Ty3"/>
</view>
</objects>
</document>

```



```

<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina6_1" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner"/>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<tableViewCellcontentMode="scaleToFill" selectionStyle="default"
indentationWidth="10" reuseIdentifier="ReportCell" id="KGk-i7-Jjw"
customClass="ReportCell" customModule="Warehouse"
customModuleProvider="target">
<rect key="frame" x="0.0" y="0.0" width="320" height="44"/>
<autoresizingMask key="autoresizingMask" flexibleMaxX="YES"
flexibleMaxY="YES"/>

```

```

<tableViewCellContentView key="contentView" opaque="NO"
clipsSubviews="YES" multipleTouchEnabled="YES" contentMode="center"
tableViewCell="KGk-i7-Jjw" id="H2p-sc-9uM">
<rect key="frame" x="0.0" y="0.0" width="320" height="43.5"/>
<autoresizingMask key="autoresizingMask"/>
<subviews>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Category"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="YTe-W1-VCh">
<rect key="frame" x="16" y="11.5" width="71" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Count"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="x1l-7l-qGh">
<rect key="frame" x="257" y="11.5" width="47" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
</subviews>
<constraints>
<constraint firstAttribute="trailing" secondItem="x1l-7l-qGh"
secondAttribute="trailing" constant="16" id="9rL-Vu-gB0"/>

```

```

<constraint firstItem="YTe-W1-VCh" firstAttribute="centerY" secondItem="H2p-sc-9uM" secondAttribute="centerY" id="Ae0-J5-6Zu"/>
<constraint firstItem="YTe-W1-VCh" firstAttribute="leading" secondItem="H2p-sc-9uM" secondAttribute="leading" constant="16" id="ExA-Be-OiR"/>
<constraint firstItem="x1l-7l-qGh" firstAttribute="centerY" secondItem="H2p-sc-9uM" secondAttribute="centerY" id="V0f-X4-HMo"/>
</constraints>
</tableViewCellContentView>
<viewLayoutGuide key="safeArea" id="njF-e1-oar"/>
<connections>
<outlet property="categoryLabel" destination="YTe-W1-VCh" id="ut4-dg-LCy"/>
<outlet property="countLabel" destination="x1l-7l-qGh" id="cBx-bX-hOf"/>
</connections>
</tableViewCell>
</objects>
</document>

<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="13142" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<dependencies>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="12042"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>

```

```

<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="ReportTableViewController"
customModuleProvider="target">
<connections>
<outlet property="view" destination="i5M-Pr-FkT" id="sfx-zR-JGt"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<tableView opaque="NO" clipsSubviews="YES"
clearsContextBeforeDrawing="NO" contentMode="scaleToFill"
bouncesZoom="NO" style="plain" separatorStyle="default" rowHeight="44"
sectionHeaderHeight="22" sectionFooterHeight="22" id="i5M-Pr-FkT">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<color key="backgroundColor" red="1" green="1" blue="1" alpha="1"
colorSpace="custom" customColorSpace="sRGB"/>
<viewLayoutGuide key="safeArea" id="vLr-E1-eTs"/>
<connections>
<outlet property="dataSource" destination="-1" id="Tng-2m-Rnh"/>
<outlet property="delegate" destination="-1" id="9aC-8N-iBw"/>
</connections>
</tableView>
</objects>
</document>

<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"

```

```

propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina6_1" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="FilterViewController" customModule="Warehouse"
customModuleProvider="target">
<connections>
<outlet property="categoryField" destination="mqh-nz-JVs" id="f3p-tC-D0z"/>
<outlet property="idFiled" destination="T4J-NT-tQw" id="FRa-za-pn3"/>
<outlet property="nameField" destination="Tom-ma-IR1" id="MLW-oL-p67"/>
<outlet property="personalField" destination="E9G-xy-2Ge" id="1jy-QU-6LG"/>
<outlet property="picker" destination="8Lh-uM-ax0" id="B6H-tp-ASf"/>
<outlet property="view" destination="i5M-Pr-FkT" id="sfx-zR-JGt"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view clearsContextBeforeDrawing="NO" contentMode="scaleToFill" id="i5M-
Pr-FkT">

```

```

<rect key="frame" x="0.0" y="0.0" width="414" height="896"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Поназваниюинвентаря" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="ICV-rO-VaB">
<rect key="frame" x="16" y="54" width="193" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="Введитеназваниеинвентаря"
textAlignment="natural" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="Tom-ma-IR1">
<rect key="frame" x="16" y="83" width="218" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
<textField opaque="NO" contentMode="scaleToFill"
horizontalHuggingPriority="249" horizontalCompressionResistancePriority="749"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="ВведитеФИО" textAlignment="natural"

```

```

minimumFontSize="17" translatesAutoresizingMaskIntoConstraints="NO"
id="E9G-xy-2Ge">
<rect key="frame" x="16" y="157" width="230" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="ПООТВЕТСТВЕННОМУ" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="CEG-h4-LbG">
<rect key="frame" x="16" y="128" width="156" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Покаатегории" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="gUA-Ho-BYS">
<rect key="frame" x="16" y="195" width="109" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"

```

```

buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="V9p-cy-f7m">
<rect key="frame" x="262" y="157" width="136" height="30"/>
<state key="normal" title="Выбратьизсписка"/>
<connections>
<action selector="personalAction:" destination="-1" eventType="touchUpInside"
id="5Hx-Tp-KF4"/>
</connections>
</button>
<textField opaque="NO" contentMode="scaleToFill"
horizontalHuggingPriority="249" horizontalCompressionResistancePriority="749"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="Введитекатегорию"
textAlignment="natural" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="mqh-nz-JVs">
<rect key="frame" x="16" y="224" width="230" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="aT7-br-UPf">
<rect key="frame" x="262" y="224" width="136" height="30"/>
<state key="normal" title="Выбратьизсписка"/>
<connections>
<action selector="categoryAction:" destination="-1" eventType="touchUpInside"
id="jxH-Vc-Qv7"/>
</connections>

```



```

</button>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Поидентификатору" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="GN0-Pe-Tym">
<rect key="frame" x="16" y="262" width="161" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<pickerView hidden="YES" contentMode="scaleToFill"
translatesAutoresizingMaskIntoConstraints="NO" id="8Lh-uM-ax0">
<rect key="frame" x="0.0" y="646" width="414" height="216"/>
</pickerView>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="Введитеидентификатор"
textAlignment="natural" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="T4J-NT-tQw">
<rect key="frame" x="16" y="291" width="234" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
</subviews>
<color key="backgroundColor" red="1" green="1" blue="1" alpha="1"
colorSpace="custom" customColorSpace="sRGB"/>
<constraints>

```

<constraint firstItem="CEG-h4-LbG" firstAttribute="top" secondItem="Tom-ma-IR1" secondAttribute="bottom" constant="15" id="5Fe-fO-bji"/>
 <constraint firstItem="aT7-br-UPf" firstAttribute="leading" secondItem="mqh-nz-JVs" secondAttribute="trailing" constant="16" id="6Ed-g5-Qoi"/>
 <constraint firstItem="8Lh-uM-ax0" firstAttribute="leading" secondItem="fnl-2z-Ty3" secondAttribute="leading" id="88f-01-fuU"/>
 <constraint firstItem="gUA-Ho-BYS" firstAttribute="top" secondItem="E9G-xy-2Ge" secondAttribute="bottom" constant="8" id="8Ff-M9-GOZ"/>
 <constraint firstItem="8Lh-uM-ax0" firstAttribute="bottom" secondItem="fnl-2z-Ty3" secondAttribute="bottom" id="9s9-5l-YbT"/>
 <constraint firstItem="T4J-NT-tQw" firstAttribute="top" secondItem="GN0-Pe-Tym" secondAttribute="bottom" constant="8" id="BN2-Yc-iDH"/>
 <constraint firstItem="Tom-ma-IR1" firstAttribute="top" secondItem="ICV-rO-VaB" secondAttribute="bottom" constant="8" id="Bx8-rG-HcX"/>
 <constraint firstItem="gUA-Ho-BYS" firstAttribute="leading" secondItem="fnl-2z-Ty3" secondAttribute="leading" constant="16" id="JR3-Wj-7FG"/>
 <constraint firstItem="E9G-xy-2Ge" firstAttribute="top" secondItem="CEG-h4-LbG" secondAttribute="bottom" constant="8" id="KOb-xz-2kK"/>
 <constraint firstItem="E9G-xy-2Ge" firstAttribute="leading" secondItem="fnl-2z-Ty3" secondAttribute="leading" constant="16" id="McG-f9-kPh"/>
 <constraint firstItem="fnl-2z-Ty3" firstAttribute="trailing" secondItem="V9p-cy-f7m" secondAttribute="trailing" constant="16" id="Np3-XK-ykt"/>
 <constraint firstItem="GN0-Pe-Tym" firstAttribute="leading" secondItem="fnl-2z-Ty3" secondAttribute="leading" constant="16" id="XMK-un-Nsz"/>
 <constraint firstItem="aT7-br-UPf" firstAttribute="top" secondItem="mqh-nz-JVs" secondAttribute="top" id="YLq-lD-yQU"/>
 <constraint firstItem="8Lh-uM-ax0" firstAttribute="trailing" secondItem="fnl-2z-Ty3" secondAttribute="trailing" id="btE-bX-Q5j"/>
 <constraint firstItem="ICV-rO-VaB" firstAttribute="top" secondItem="fnl-2z-Ty3" secondAttribute="top" constant="10" id="cBk-Yi-uaa"/>

```

<constraint firstItem="ICV-rO-VaB" firstAttribute="leading" secondItem="fnl-2z-
Ty3" secondAttribute="leading" constant="16" id="iCy-74-j08"/>
<constraint firstItem="fnl-2z-Ty3" firstAttribute="trailing" secondItem="aT7-br-
UPf" secondAttribute="trailing" constant="16" id="p0H-0O-1iq"/>
<constraint firstItem="Tom-ma-IR1" firstAttribute="leading" secondItem="fnl-2z-
Ty3" secondAttribute="leading" constant="16" id="pcC-yf-UDg"/>
<constraint firstItem="CEG-h4-LbG" firstAttribute="leading" secondItem="fnl-
2z-Ty3" secondAttribute="leading" constant="16" id="qLR-Rw-pXt"/>
<constraint firstItem="V9p-cy-f7m" firstAttribute="top" secondItem="CEG-h4-
LbG" secondAttribute="bottom" constant="8" id="sw8-vN-4M2"/>
<constraint firstItem="mqh-nz-JVs" firstAttribute="top" secondItem="gUA-Ho-
BYS" secondAttribute="bottom" constant="8" id="tMG-sY-DWU"/>
<constraint firstItem="T4J-NT-tQw" firstAttribute="leading" secondItem="fnl-2z-
Ty3" secondAttribute="leading" constant="16" id="tUw-Ph-fCW"/>
<constraint firstItem="V9p-cy-f7m" firstAttribute="leading" secondItem="E9G-
xy-2Ge" secondAttribute="trailing" constant="16" id="tZC-kF-dSZ"/>
<constraint firstItem="fnl-2z-Ty3" firstAttribute="trailing" secondItem="T4J-NT-
tQw" secondAttribute="trailing" constant="164" id="vPw-eN-N9B"/>
<constraint firstItem="GN0-Pe-Tym" firstAttribute="top" secondItem="mqh-nz-
JVs" secondAttribute="bottom" constant="8" id="zPw-3g-bdG"/>
<constraint firstItem="mqh-nz-JVs" firstAttribute="leading" secondItem="fnl-2z-
Ty3" secondAttribute="leading" constant="16" id="zgR-5R-luS"/>
</constraints>
<viewLayoutGuide key="safeArea" id="fnl-2z-Ty3"/>
<point key="canvasLocation" x="752" y="12"/>
</view>
</objects>
</document>

<?xml version="1.0" encoding="UTF-8"?>

```

```

<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="DetailInventoryViewController"
customModule="Warehouse" customModuleProvider="target">
<connections>
<outlet property="categoryLabel" destination="5Vd-fv-16Q" id="Xxq-Tw-
SwU"/>
<outlet property="dateLabel" destination="haF-nv-qUF" id="V94-xb-F9X"/>
<outlet property="idLabel" destination="F0d-3q-JGJ" id="bum-Tv-3UQ"/>
<outlet property="nameLabel" destination="f9S-Y4-dhZ" id="Ryw-WW-MxD"/>
<outlet property="personalLabel" destination="c5b-N3-lnm" id="xPs-yX-Gfl"/>
<outlet property="printBtn" destination="pOZ-t4-Sid" id="U3f-x8-0Lb"/>
<outlet property="qrCodeGenerateButton" destination="61t-2p-vPo" id="eCi-vy-
HYv"/>

```

```

<outlet property="qrCodeImageView" destination="66G-25-68s" id="7tv-hf-
c8U"/>
<outlet property="view" destination="i5M-Pr-FkT" id="sfx-zR-JGt"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view clearsContextBeforeDrawing="NO" contentMode="scaleToFill" id="i5M-
Pr-FkT">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<scrollView clipsSubviews="YES" multipleTouchEnabled="YES"
contentMode="scaleToFill" translatesAutoresizingMaskIntoConstraints="NO"
id="uPU-b3-GX2">
<rect key="frame" x="0.0" y="20" width="375" height="647"/>
</scrollView>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Названиеинвентаря" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="G7r-uN-UZL">
<rect key="frame" x="12" y="32" width="177" height="20.5"/>
<fontDescription key="fontDescription" type="boldSystem" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

```

```

<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Name"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="f9S-Y4-dhZ">
<rect key="frame" x="12" y="60.5" width="45" height="20"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="ОТВЕТСТВЕННЫЙ" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="CQL-Zj-3ld">
<rect key="frame" x="12" y="92.5" width="132.5" height="20.5"/>
<fontDescription key="fontDescription" type="boldSystem" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Personal"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="c5b-N3-lnm">
<rect key="frame" x="12" y="121" width="66" height="20.5"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

```

```

<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Категорияинвентаря" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="LFS-IM-eXQ">
<rect key="frame" x="12" y="153.5" width="183" height="20"/>
<fontDescription key="fontDescription" type="boldSystem" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Category"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="5Vd-fv-16Q">
<rect key="frame" x="12" y="181.5" width="71" height="20.5"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Идентификатор" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="PY9-Ms-tUH">
<rect key="frame" x="12" y="214" width="137" height="20.5"/>
<fontDescription key="fontDescription" type="boldSystem" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>

```

```

</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="id"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="F0d-3q-JGJ">
<rect key="frame" x="12" y="242.5" width="14" height="21.5"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Датадобавления" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="boW-0e-jPg">
<rect key="frame" x="215.5" y="214" width="147.5" height="21"/>
<fontDescription key="fontDescription" type="boldSystem" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Date"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="haF-nv-qUF">
<rect key="frame" x="215.5" y="243" width="36" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>

```



```

<nil key="highlightedColor"/>
</label>
<imageView hidden="YES" userInteractionEnabled="NO"
contentMode="scaleToFill" horizontalHuggingPriority="251"
verticalHuggingPriority="251" translatesAutoresizingMaskIntoConstraints="NO"
id="66G-25-68s">
<rect key="frame" x="67.5" y="288" width="240" height="240"/>
<constraints>
<constraint firstAttribute="width" priority="750" constant="240" id="36R-e2-
ncF"/>
<constraint firstAttribute="width" secondItem="66G-25-68s"
secondAttribute="height" multiplier="1:1" id="LKs-J0-qeI"/>
</constraints>
</imageView>
<button opaque="NO" contentMode="scaleToFill" misplaced="YES"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="61t-2p-vPo">
<rect key="frame" x="67" y="393" width="240" height="30"/>
<color key="backgroundColor" red="0.5150729997999999"
green="0.78582151200000006" blue="1" alpha="1" colorSpace="custom"
customColorSpace="sRGB"/>
<state key="normal" title="Сгенерировать QR-Код">
<color key="titleColor" white="0.0" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
</state>
<connections>
<action selector="generateQRCode:" destination="-1"
eventType="touchUpInside" id="eQa-p7-ETZ"/>
</connections>

```

```

</button>
<button opaque="NO".contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="pOZ-t4-Sid">
<rect key="frame" x="67.5" y="536" width="240" height="30"/>
<color key="backgroundColor" red="0.51372549019607838"
green="0.78582151200000006" blue="1" alpha="1" colorSpace="custom"
customColorSpace="sRGB"/>
<state key="normal" title="Печать">
<color key="titleColor" white="0.0" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
</state>
<connections>
<action selector="generateQRCode:" destination="-1"
eventType="touchUpInside" id="pyg-6O-19T"/>
<action selector="printAction:" destination="-1" eventType="touchUpInside"
id="hFp-hp-upa"/>
</connections>
</button>
</subviews>
<color key="backgroundColor" red="1" green="1" blue="1" alpha="1"
colorSpace="custom" customColorSpace="sRGB"/>
<constraints>
<constraint firstItem="F0d-3q-JGJ" firstAttribute="leading" secondItem="uPU-b3-
GX2" secondAttribute="leading" constant="12" id="0GV-q1-xBo"/>
<constraint firstItem="c5b-N3-lnm" firstAttribute="leading" secondItem="uPU-
b3-GX2" secondAttribute="leading" constant="12" id="3B9-rV-Qyh"/>
<constraint firstItem="F0d-3q-JGJ" firstAttribute="top" secondItem="PY9-Ms-
tUH" secondAttribute="bottom" constant="8" id="3eF-4t-I3b"/>

```

<constraint firstItem="haF-nv-qUF" firstAttribute="top" secondItem="boW-0e-jPg" secondAttribute="bottom" constant="8" id="5WR-Mm-kcr"/>
 <constraint firstItem="f9S-Y4-dhZ" firstAttribute="top" secondItem="G7r-uN-UZL" secondAttribute="bottom" constant="8" id="9qI-mh-7tL"/>
 <constraint firstItem="66G-25-68s" firstAttribute="top" secondItem="F0d-3q-JGJ" secondAttribute="bottom" constant="24" id="ChV-AV-co6"/>
 <constraint firstItem="LFS-IM-eXQ" firstAttribute="top" secondItem="c5b-N3-lnm" secondAttribute="bottom" constant="12" id="Dao-0B-bvt"/>
 <constraint firstItem="fnl-2z-Ty3" firstAttribute="leading" secondItem="uPU-b3-GX2" secondAttribute="leading" id="EQe-5j-ptb"/>
 <constraint firstItem="uPU-b3-GX2" firstAttribute="trailing" secondItem="boW-0e-jPg" secondAttribute="trailing" constant="12" id="GAc-v1-UBz"/>
 <constraint firstItem="G7r-uN-UZL" firstAttribute="leading" secondItem="uPU-b3-GX2" secondAttribute="leading" constant="12" id="GYu-eN-8yW"/>
 <constraint firstItem="f9S-Y4-dhZ" firstAttribute="leading" secondItem="uPU-b3-GX2" secondAttribute="leading" constant="12" id="HUx-nr-FmV"/>
 <constraint firstItem="G7r-uN-UZL" firstAttribute="top" secondItem="uPU-b3-GX2" secondAttribute="top" constant="12" id="IO4-nE-j6A"/>
 <constraint firstItem="c5b-N3-lnm" firstAttribute="top" secondItem="CQL-Zj-3ld" secondAttribute="bottom" constant="8" id="J7n-Fl-e3W"/>
 <constraint firstItem="PY9-Ms-tUH" firstAttribute="leading" secondItem="uPU-b3-GX2" secondAttribute="leading" constant="12" id="Mpl-3I-ADv"/>
 <constraint firstItem="CQL-Zj-3ld" firstAttribute="leading" secondItem="uPU-b3-GX2" secondAttribute="leading" constant="12" id="NF0-UV-V20"/>
 <constraint firstItem="PY9-Ms-tUH" firstAttribute="top" secondItem="5Vd-fv-16Q" secondAttribute="bottom" constant="12" id="PHI-47-Cvd"/>
 <constraint firstItem="CQL-Zj-3ld" firstAttribute="top" secondItem="f9S-Y4-dhZ" secondAttribute="bottom" constant="12" id="SKL-R5-fOc"/>
 <constraint firstItem="uPU-b3-GX2" firstAttribute="centerX" secondItem="fnl-2z-Ty3" secondAttribute="centerX" id="UyR-xt-owQ"/>

<constraint firstItem="LFS-IM-eXQ" firstAttribute="leading" secondItem="uPU-b3-GX2" secondAttribute="leading" constant="12" id="WjO-Tx-Gms"/>
 <constraint firstItem="uPU-b3-GX2" firstAttribute="centerY" secondItem="fnl-2z-Ty3" secondAttribute="centerY" id="Xhc-rR-Jbo"/>
 <constraint firstItem="61t-2p-vPo" firstAttribute="centerX" secondItem="66G-25-68s" secondAttribute="centerX" id="aDC-AQ-XOf"/>
 <constraint firstItem="boW-0e-jPg" firstAttribute="centerY" secondItem="PY9-Ms-tUH" secondAttribute="centerY" id="c0e-bi-z8F"/>
 <constraint firstItem="haF-nv-qUF" firstAttribute="leading" relation="greaterThanOrEqual" secondItem="F0d-3q-JGJ" secondAttribute="trailing" constant="24" id="dUy-6U-Jrw"/>
 <constraint firstItem="boW-0e-jPg" firstAttribute="leading" relation="greaterThanOrEqual" secondItem="PY9-Ms-tUH" secondAttribute="trailing" constant="24" id="f3N-p6-B4i"/>
 <constraint firstItem="pOZ-t4-Sid" firstAttribute="width" secondItem="66G-25-68s" secondAttribute="width" id="h9U-1e-uHq"/>
 <constraint firstItem="5Vd-fv-16Q" firstAttribute="leading" secondItem="uPU-b3-GX2" secondAttribute="leading" constant="12" id="jy9-Wt-jyy"/>
 <constraint firstItem="haF-nv-qUF" firstAttribute="leading" secondItem="boW-0e-jPg" secondAttribute="leading" id="oue-tL-9cs"/>
 <constraint firstItem="pOZ-t4-Sid" firstAttribute="centerX" secondItem="uPU-b3-GX2" secondAttribute="centerX" id="ptO-OR-IqJ"/>
 <constraint firstItem="pOZ-t4-Sid" firstAttribute="top" secondItem="66G-25-68s" secondAttribute="bottom" constant="8" id="rGL-WH-8MT"/>
 <constraint firstItem="uPU-b3-GX2" firstAttribute="trailing" relation="greaterThanOrEqual" secondItem="haF-nv-qUF" secondAttribute="trailing" constant="12" id="rUT-Eq-uE5"/>
 <constraint firstItem="66G-25-68s" firstAttribute="centerX" secondItem="i5M-Pr-FkT" secondAttribute="centerX" id="sCE-AE-aDE"/>

```

<constraint firstItem="uPU-b3-GX2" firstAttribute="top" secondItem="fnl-2z-
Ty3" secondAttribute="top" id="uzO-iC-s2g"/>
<constraint firstItem="61t-2p-vPo" firstAttribute="centerY" secondItem="66G-25-
68s" secondAttribute="centerY" id="vq6-Yg-PQc"/>
<constraint firstItem="5Vd-fv-16Q" firstAttribute="top" secondItem="LFS-IM-
eXQ" secondAttribute="bottom" constant="8" id="x9q-2V-gIb"/>
<constraint firstItem="66G-25-68s" firstAttribute="top" secondItem="haF-nv-
qUF" secondAttribute="bottom" constant="24" id="xwv-rF-WvT"/>
</constraints>
<viewLayoutGuide key="safeArea" id="fnl-2z-Ty3"/>
<point key="canvasLocation" x="138.40000000000001"
y="152.47376311844079"/>
</view>
</objects>
</document>

<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14460.31" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14460.20"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>

```

```

<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner"/>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<tableViewCellcontentMode="scaleToFill" selectionStyle="default"
indentationWidth="10" reuseIdentifier="InventoryTableViewCell" id="KGk-i7-
Jjw" customClass="InventoryTableViewCell" customModule="Warehouse"
customModuleProvider="target">
<rect key="frame" x="0.0" y="0.0" width="320" height="44"/>
<autoresizingMask key="autoresizingMask" flexibleMaxX="YES"
flexibleMaxY="YES"/>
<tableViewCellContentView key="contentView" opaque="NO"
clipsSubviews="YES" multipleTouchEnabled="YES" contentMode="center"
tableViewCell="KGk-i7-Jjw" id="H2p-sc-9uM">
<rect key="frame" x="0.0" y="0.0" width="320" height="43.5"/>
<autoresizingMask key="autoresizingMask"/>
<subviews>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="252" text="Name"
textAlignment="natural" lineBreakMode="tailTruncation" numberOfLines="2"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="2eu-eY-zcJ">
<rect key="frame" x="16" y="11" width="116.5" height="21.5"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>

```

```

</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Date"
textAlignment="natural" lineBreakMode="tailTruncation" numberOfLines="2"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="eQX-zJ-pp6">
<rect key="frame" x="266" y="11" width="38" height="21.5"/>
<constraints>
<constraint firstAttribute="width" relation="greaterThanOrEqualTo" constant="38"
id="cQn-tV-ZCf"/>
</constraints>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Personal"
textAlignment="natural" lineBreakMode="tailTruncation" numberOfLines="2"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="K5n-Nc-Oa9">
<rect key="frame" x="140.5" y="14.5" width="49" height="14.5"/>
<fontDescription key="fontDescription" type="system" pointSize="12"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
</subviews>
<constraints>
<constraint firstItem="eQX-zJ-pp6" firstAttribute="trailing" secondItem="H2p-sc-
9uM" secondAttribute="trailingMargin" id="4On-T7-5sF"/>

```

```

<constraint firstItem="2eu-eY-zcJ" firstAttribute="top" secondItem="H2p-sc-
9uM" secondAttribute="topMargin" id="8cA-KD-lv7"/>
<constraint firstItem="K5n-Nc-Oa9" firstAttribute="centerX" secondItem="H2p-
sc-9uM" secondAttribute="centerX" constant="5" id="PNL-km-hor"/>
<constraint firstItem="eQX-zJ-pp6" firstAttribute="top" secondItem="H2p-sc-
9uM" secondAttribute="topMargin" id="Pov-WJ-ID0"/>
<constraint firstItem="eQX-zJ-pp6" firstAttribute="centerY" secondItem="H2p-
sc-9uM" secondAttribute="centerY" id="QEF-PM-pbf"/>
<constraint firstItem="2eu-eY-zcJ" firstAttribute="leading" secondItem="H2p-sc-
9uM" secondAttribute="leadingMargin" id="VkJ-VB-WIq"/>
<constraint firstItem="K5n-Nc-Oa9" firstAttribute="centerY" secondItem="H2p-
sc-9uM" secondAttribute="centerY" id="dj7-CV-PSU"/>
<constraint firstItem="2eu-eY-zcJ" firstAttribute="centerY" secondItem="H2p-sc-
9uM" secondAttribute="centerY" id="k7g-HB-azK"/>
<constraint firstItem="K5n-Nc-Oa9" firstAttribute="leading" secondItem="2eu-
eY-zcJ" secondAttribute="trailing" constant="8" id="zZt-PA-YNC"/>
</constraints>
</tableViewCellContentView>
<viewLayoutGuide key="safeArea" id="njF-e1-oar"/>
<connections>
<outlet property="date" destination="eQX-zJ-pp6" id="tuV-oC-EAH"/>
<outlet property="name" destination="2eu-eY-zcJ" id="T2e-sm-rDB"/>
<outlet property="personal" destination="K5n-Nc-Oa9" id="e7w-53-0nC"/>
</connections>
<point key="canvasLocation" x="137.59999999999999"
y="153.82308845577214"/>
</tableViewCell>
</objects>
</document>

```



```

<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.Storyboard.XIB"
version="3.0" toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES" initialViewController="qbZ-zP-
APn">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<scenes>
<!--Tab Bar Controller-->
<scene sceneID="LkE-oU-dch">
<objects>
<tabBarController id="RmI-eB-ZXr" sceneMemberID="viewController">
<tabBar key="tabBar" contentMode="scaleToFill"
insetsLayoutMarginsFromSafeArea="NO" id="6m1-xp-ZxT">
<rect key="frame" x="0.0" y="0.0" width="414" height="49"/>
<autoresizingMask key="autoresizingMask"/>
<color key="backgroundColor" white="0.0" alpha="0.0" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
</tabBar>
</tabBarController>

```

```

<placeholder placeholderIdentifier="IBFirstResponder" id="xBW-Sv-efN"
userLabel="First Responder" sceneMemberID="firstResponder"/>
</objects>
<point key="canvasLocation" x="-1541" y="-218"/>
</scene>
<!--Item-->
<scene sceneID="n4q-nw-gXx">
<objects>
<placeholder placeholderIdentifier="IBFirstResponder" id="CCA-VD-lWk"
userLabel="First Responder" sceneMemberID="firstResponder"/>
<tableViewController id="qbZ-zP-APn"
customClass="InventoryTableViewController" customModule="Warehouse"
customModuleProvider="target" sceneMemberID="viewController">
<tableView key="view" clipsSubviews="YES" contentMode="scaleToFill"
alwaysBounceVertical="YES" dataMode="prototypes" style="plain"
separatorStyle="default" rowHeight="-1" estimatedRowHeight="-1"
sectionHeaderHeight="28" sectionFooterHeight="28" id="mDJ-9p-Hc5">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<prototypes>
<tableViewCell clipsSubviews="YES" contentMode="scaleToFill"
preservesSuperviewLayoutMargins="YES" selectionStyle="default"
indentationWidth="10" reuseIdentifier="InventoryTableViewCell" id="H3n-ru-
jDq">
<rect key="frame" x="0.0" y="28" width="375" height="44"/>
<autoresizingMask key="autoresizingMask"/>

```

```

<tableViewCellContentView key="contentView" opaque="NO"
clipsSubviews="YES" multipleTouchEnabled="YES" contentMode="center"
preservesSuperviewLayoutMargins="YES"
insetsLayoutMarginsFromSafeArea="NO" tableViewCell="H3n-ru-jDq"
id="4Mk-FT-2BC">
<rect key="frame" x="0.0" y="0.0" width="375" height="43.5"/>
<autoresizingMask key="autoresizingMask"/>
</tableViewCellContentView>
</tableViewCell>
</prototypes>
<connections>
<outlet property="dataSource" destination="qbZ-zP-APn" id="DvY-Ho-azI"/>
<outlet property="delegate" destination="qbZ-zP-APn" id="38V-jr-eIZ"/>
</connections>
</tableView>
<tabBarItem key="tabBarItem" title="Item" id="YnP-Ag-hHF"/>
</viewController>
</objects>
<point key="canvasLocation" x="-252" y="-668"/>
</scene>
<!--Item-->
<scene sceneID="aoX-L5-ubK">
<objects>
<viewController id="IN1-JA-KgU" sceneMemberID="viewController">
<view key="view" contentMode="scaleToFill" id="qZ7-pb-Qih">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>

```

```

<button opaque="NO".contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="zdo-cb-Qnb">
<rect key="frame" x="141" y="591" width="93" height="30"/>
<state key="normal" title="Сканировать"/>
</button>
<textField opaque="NO".contentMode="scaleToFill" misplaced="YES"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" textAlignment="natural" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="4kY-Cv-ZIq">
<rect key="frame" x="12" y="498" width="284" height="30"/>
<constraints>
<constraint firstAttribute="width" relation="lessThanOrEqualTo" constant="289"
id="LLx-sV-IrS"/>
</constraints>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
<button opaque="NO".contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="x9L-pa-qxw">
<rect key="frame" x="314" y="547" width="49" height="30"/>
<state key="normal" title="Учесть"/>
</button>
</subviews>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>

```

```

<constraints>
<constraint firstItem="80V-Og-Z9z" firstAttribute="bottom" secondItem="zdo-cb-
Qnb" secondAttribute="bottom" constant="46" id="O7p-uV-scJ"/>
<constraint firstItem="4kY-Cv-ZIq" firstAttribute="centerY" secondItem="x9L-
pa-qxw" secondAttribute="centerY" id="Qa4-0f-BCP"/>
<constraint firstItem="4kY-Cv-ZIq" firstAttribute="leading" secondItem="80V-
Og-Z9z" secondAttribute="leading" constant="12" id="Ru5-DY-aRL"/>
<constraint firstItem="zdo-cb-Qnb" firstAttribute="centerX" secondItem="80V-
Og-Z9z" secondAttribute="centerX" id="XxU-tb-UgB"/>
<constraint firstItem="zdo-cb-Qnb" firstAttribute="top" secondItem="x9L-pa-
qxw" secondAttribute="bottom" constant="14" id="cZN-8F-9W1"/>
<constraint firstItem="x9L-pa-qxw" firstAttribute="leading"
relation="greaterThanOrEqual" secondItem="4kY-Cv-ZIq"
secondAttribute="trailing" constant="12" id="ec4-et-Wiv"/>
<constraint firstItem="80V-Og-Z9z" firstAttribute="trailing" secondItem="x9L-
pa-qxw" secondAttribute="trailing" constant="12" id="gCC-2B-1Zh"/>
</constraints>
<viewLayoutGuide key="safeArea" id="80V-Og-Z9z"/>
</view>
<tabBarItem key="tabBarItem" title="Item" id="mDq-eS-GwR"/>
</viewController>
<placeholder placeholderIdentifier="IBFirstResponder" id="MHY-I0-251"
userLabel="First Responder" sceneMemberID="firstResponder"/>
</objects>
<point key="canvasLocation" x="-252" y="112.89355322338831"/>
</scene>
</scenes>
</document>

<?xml version="1.0" encoding="UTF-8"?>

```

```

<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14460.31" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14460.20"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner"/>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view contentMode="scaleToFill" id="iN0-l3-epB">
<rect key="frame" x="0.0" y="0.0" width="375" height="30"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Названиеинвентаря" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"

```

```

adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="5wM-BF-cbJ">
<rect key="frame" x="18" y="9" width="104" height="12"/>
<fontDescription key="fontDescription" type="system" pointSize="10"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="ОТВЕТСТВЕННЫЙ" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="XMC-2d-ZbC">
<rect key="frame" x="149" y="9" width="77" height="12"/>
<fontDescription key="fontDescription" type="system" pointSize="10"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Датадобавления" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="Pap-jk-Rqh">
<rect key="frame" x="270.5" y="9" width="86.5" height="12"/>
<fontDescription key="fontDescription" type="system" pointSize="10"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

```

```

<view.contentMode="scaleToFill"
translatesAutoresizingMaskIntoConstraints="NO" id="mej-4O-olp"
userLabel="Separator">
<rect key="frame" x="0.0" y="29" width="375" height="1"/>
<color key="backgroundColor" white="0.18324593321917809"
alpha="0.25056185787671231" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<constraints>
<constraint firstAttribute="height" constant="1" id="tJQ-34-tgq"/>
</constraints>
</view>
</subviews>
<color key="backgroundColor" red="1" green="1" blue="1" alpha="1"
colorSpace="custom" customColorSpace="sRGB"/>
<constraints>
<constraint firstItem="XMC-2d-ZbC" firstAttribute="centerX"
secondItem="vUN-kp-3ea" secondAttribute="centerX" id="4wi-Ez-Mh3"/>
<constraint firstItem="5wM-BF-cbJ" firstAttribute="leading" secondItem="vUN-
kp-3ea" secondAttribute="leading" constant="18" id="5Bv-cK-sfS"/>
<constraint firstItem="5wM-BF-cbJ" firstAttribute="centerY" secondItem="vUN-
kp-3ea" secondAttribute="centerY" id="9J8-RS-4uD"/>
<constraint firstItem="vUN-kp-3ea" firstAttribute="trailing" secondItem="Pap-jk-
Rqh" secondAttribute="trailing" constant="18" id="RXl-fZ-2gD"/>
<constraint firstItem="Pap-jk-Rqh" firstAttribute="centerY" secondItem="vUN-
kp-3ea" secondAttribute="centerY" id="Rje-fh-s2V"/>
<constraint firstItem="mej-4O-olp" firstAttribute="width" secondItem="iN0-l3-
epB" secondAttribute="width" id="THq-Sa-R13"/>
<constraint firstItem="XMC-2d-ZbC" firstAttribute="centerY"
secondItem="vUN-kp-3ea" secondAttribute="centerY" id="UZo-xo-nPk"/>

```



```

<constraint firstAttribute="trailing" secondItem="mej-4O-olp"
secondAttribute="trailing" id="Vty-3J-nuh"/>
<constraint firstItem="vUN-kp-3ea" firstAttribute="bottom" secondItem="mej-
4O-olp" secondAttribute="bottom" id="z1H-Ah-eao"/>
</constraints>
<freeformSimulatedSizeMetrics key="simulatedDestinationMetrics"/>
<viewLayoutGuide key="safeArea" id="vUN-kp-3ea"/>
<point key="canvasLocation" x="61.600000000000001"
y="47.676161919040482"/>
</view>
</objects>
</document>

<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14460.31" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14460.20"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>

```

```

<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="QRCodeViewController" customModule="Warehouse"
customModuleProvider="target">
<connections>
<outlet property="printButton" destination="mOi-ch-4D3" id="Bgd-dp-0Bl"/>
<outlet property="qrImageView" destination="exb-tY-E2f" id="lGd-pk-Ere"/>
<outlet property="view" destination="i5M-Pr-FkT" id="sfx-zR-JGt"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view clearsContextBeforeDrawing="NO" contentMode="scaleToFill"
restorationIdentifier="QRCodeViewController" id="i5M-Pr-FkT">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<imageView userInteractionEnabled="NO" contentMode="scaleToFill"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
translatesAutoresizingMaskIntoConstraints="NO" id="exb-tY-E2f">
<rect key="frame" x="67.5" y="106" width="240" height="240"/>
<color key="backgroundColor" red="1" green="1" blue="0.5215686274999999"
alpha="1" colorSpace="calibratedRGB"/>
<constraints>
<constraint firstAttribute="width" constant="240" id="1ru-H4-Gdq"/>
<constraint firstAttribute="width" secondItem="exb-tY-E2f"
secondAttribute="height" multiplier="1:1" id="xAl-wG-TX7"/>
</constraints>
</imageView>

```

```

<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="mOi-ch-4D3">
<rect key="frame" x="67.5" y="358" width="240" height="30"/>
<color key="backgroundColor" red="0.51507299977351007"
green="0.78582151195335714" blue="1" alpha="1" colorSpace="custom"
customColorSpace="sRGB"/>
<state key="normal" title="Печать">
<color key="titleColor" white="0.0" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
</state>
<connections>
<action selector="printAction:" destination="-1" eventType="touchUpInside"
id="IPj-wD-5fc"/>
</connections>
</button>
</subviews>
<color key="backgroundColor" red="1" green="1" blue="1" alpha="1"
colorSpace="custom" customColorSpace="sRGB"/>
<constraints>
<constraint firstItem="mOi-ch-4D3" firstAttribute="width" secondItem="exb-tY-
E2f" secondAttribute="width" id="7cb-oh-GgM"/>
<constraint firstItem="exb-tY-E2f" firstAttribute="centerX" secondItem="fnl-2z-
Ty3" secondAttribute="centerX" id="DIl-ks-Erm"/>
<constraint firstItem="exb-tY-E2f" firstAttribute="top" secondItem="fnl-2z-Ty3"
secondAttribute="top" constant="86" id="jaF-f6-q1Z"/>
<constraint firstItem="mOi-ch-4D3" firstAttribute="top" secondItem="exb-tY-
E2f" secondAttribute="bottom" constant="12" id="qfU-YM-BGl"/>

```

```

<constraint firstItem="mOi-ch-4D3" firstAttribute="centerX" secondItem="fnl-2z-
Ty3" secondAttribute="centerX" id="rYX-Rc-Hpi"/>
</constraints>
<viewLayoutGuide key="safeArea" id="fnl-2z-Ty3"/>
<point key="canvasLocation" x="138.40000000000001"
y="153.37331334332833"/>
</view>
</objects>
</document>
<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="AddInventoryViewController"
customModule="Warehouse" customModuleProvider="target">
<connections>

```

```

<outlet property="category" destination="d3Z-5H-f9B" id="EHQ-K0-yk6"/>
<outlet property="fio" destination="ZG3-c3-PJN" id="YFP-0d-PXO"/>
<outlet property="generateTitle" destination="X6l-RJ-wLP" id="1kZ-Kk-mxc"/>
<outlet property="idLabel" destination="jq6-g0-iso" id="npT-0K-Hbj"/>
<outlet property="nameField" destination="MkB-TU-WEL" id="d1q-MR-lYv"/>
<outlet property="picker" destination="sKd-zi-JAc" id="0lx-If-hCJ"/>
<outlet property="qrSwitch" destination="1nx-vL-9SM" id="ivy-2Y-qZg"/>
<outlet property="view" destination="bLd-av-6CV" id="J4u-RT-eHp"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view contentMode="scaleToFill" id="bLd-av-6CV">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Названиеинвентаря" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="XGa-hj-UNe">
<rect key="frame" x="16" y="30" width="166" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"

```

```

borderStyle="roundedRect" placeholder="Введите название инвентаря"
textAlignment="natural" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="MkB-TU-WEL">
<rect key="frame" x="16" y="59" width="343" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
<textField opaque="NO" contentMode="scaleToFill"
horizontalHuggingPriority="249" horizontalCompressionResistancePriority="749"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="ФИО" textAlignment="natural"
minimumFontSize="17" translatesAutoresizingMaskIntoConstraints="NO"
id="ZG3-c3-PJN">
<rect key="frame" x="16" y="133" width="191" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="ОТВЕТСТВЕННЫЙ" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="cg1-f4-ZCG">
<rect key="frame" x="16" y="104" width="124" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

```

```

<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Категория" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="WW8-v7-xKw">
<rect key="frame" x="16" y="171" width="83" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>

<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="B9D-hB-JvF">
<rect key="frame" x="223" y="133" width="136" height="30"/>
<state key="normal" title="Выбрать из списка"/>
<connections>
<action selector="chooseFio:" destination="-1" eventType="touchUpInside"
id="5Bu-sj-5pH"/>
</connections>
</button>

<textField opaque="NO" contentMode="scaleToFill"
horizontalHuggingPriority="249" horizontalCompressionResistancePriority="749"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="Категория" textAlignment="natural"
minimumFontSize="17" translatesAutoresizingMaskIntoConstraints="NO"
id="d3Z-5H-f9B">
<rect key="frame" x="16" y="200" width="191" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>

```

```

<textInputTraits key="textInputTraits"/>
</textField>
<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="BZn-4P-BbU">
<rect key="frame" x="223" y="200" width="136" height="30"/>
<state key="normal" title="Выбрать из списка"/>
<connections>
<action selector="chooseCategory:" destination="-1" eventType="touchUpInside"
id="fxc-eq-m7u"/>
</connections>
</button>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Идентификатор" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="q5y-4x-ZTA">
<rect key="frame" x="21" y="238" width="128" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="ID"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="jq6-g0-iso">
<rect key="frame" x="21" y="267" width="17" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>

```



```

<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<pickerView hidden="YES" contentMode="scaleToFill"
translatesAutoresizingMaskIntoConstraints="NO" id="sKd-zi-JAc">
<rect key="frame" x="0.0" y="451" width="375" height="216"/>
</pickerView>
<switch opaque="NO" contentMode="scaleToFill"
horizontalHuggingPriority="750" verticalHuggingPriority="750"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
on="YES" translatesAutoresizingMaskIntoConstraints="NO" id="1nx-vL-9SM">
<rect key="frame" x="16" y="325" width="51" height="31"/>
</switch>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Сгенерировать QR-Код ?" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="X6l-RJ-wLP">
<rect key="frame" x="16" y="296" width="201" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
</subviews>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<gestureRecognizers/>
<constraints>

```

<constraint firstItem="XGa-hj-UNe" firstAttribute="top" secondItem="LV4-8T-CsK" secondAttribute="top" constant="10" id="00K-yo-CnS"/>
 <constraint firstItem="jq6-g0-iso" firstAttribute="top" secondItem="q5y-4x-ZTA" secondAttribute="bottom" constant="8" id="3Hh-vD-Fnz"/>
 <constraint firstItem="d3Z-5H-f9B" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="5xc-D3-sfj"/>
 <constraint firstItem="MkB-TU-WEL" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="75E-dN-ZRD"/>
 <constraint firstItem="ZG3-c3-PJN" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="7ew-Am-MSe"/>
 <constraint firstItem="BZn-4P-BbU" firstAttribute="top" secondItem="d3Z-5H-f9B" secondAttribute="top" id="9tC-U2-uWA"/>
 <constraint firstItem="XGa-hj-UNe" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="9tK-CG-85l"/>
 <constraint firstItem="MkB-TU-WEL" firstAttribute="top" secondItem="XGa-hj-UNe" secondAttribute="bottom" constant="8" id="ALk-YU-zbp"/>
 <constraint firstItem="X6l-RJ-wLP" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="D8U-OH-DgL"/>
 <constraint firstItem="ZG3-c3-PJN" firstAttribute="top" secondItem="cg1-f4-ZCG" secondAttribute="bottom" constant="8" id="DpW-Uf-eS5"/>
 <constraint firstItem="1nx-vL-9SM" firstAttribute="top" secondItem="X6l-RJ-wLP" secondAttribute="bottom" constant="8" id="GPx-LP-vUb"/>
 <constraint firstItem="sKd-zi-JAc" firstAttribute="bottom" secondItem="LV4-8T-CsK" secondAttribute="bottom" id="GZo-86-mhG"/>
 <constraint firstItem="WW8-v7-xKw" firstAttribute="top" secondItem="ZG3-c3-PJN" secondAttribute="bottom" constant="8" id="LZJ-wu-SZy"/>
 <constraint firstItem="sKd-zi-JAc" firstAttribute="trailing" secondItem="LV4-8T-CsK" secondAttribute="trailing" id="NUJ-Sa-Hnh"/>

<constraint firstItem="LV4-8T-CsK" firstAttribute="trailing" secondItem="BZn-4P-BbU" secondAttribute="trailing" constant="16" id="Tly-0W-yzM"/>
 <constraint firstItem="d3Z-5H-f9B" firstAttribute="top" secondItem="WW8-v7-xKw" secondAttribute="bottom" constant="8" id="V4X-Vo-W3l"/>
 <constraint firstItem="LV4-8T-CsK" firstAttribute="trailing" secondItem="MkB-TU-WEL" secondAttribute="trailing" constant="16" id="bRl-lh-OfE"/>
 <constraint firstItem="cg1-f4-ZCG" firstAttribute="top" secondItem="MkB-TU-WEL" secondAttribute="bottom" constant="15" id="bWX-8T-Jfu"/>
 <constraint firstItem="sKd-zi-JAc" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" id="bX0-EZ-HWe"/>
 <constraint firstItem="WW8-v7-xKw" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="cbs-kZ-abR"/>
 <constraint firstItem="lnx-vL-9SM" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="eR3-YL-TfH"/>
 <constraint firstItem="B9D-hB-JvF" firstAttribute="leading" secondItem="ZG3-c3-PJN" secondAttribute="trailing" constant="16" id="eb7-8G-wCn"/>
 <constraint firstItem="B9D-hB-JvF" firstAttribute="top" secondItem="cg1-f4-ZCG" secondAttribute="bottom" constant="8" id="fRS-Tf-tfM"/>
 <constraint firstItem="q5y-4x-ZTA" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="21" id="g2p-3i-tzr"/>
 <constraint firstItem="BZn-4P-BbU" firstAttribute="leading" secondItem="d3Z-5H-f9B" secondAttribute="trailing" constant="16" id="iFD-ro-utt"/>
 <constraint firstItem="cg1-f4-ZCG" firstAttribute="leading" secondItem="LV4-8T-CsK" secondAttribute="leading" constant="16" id="jvj-UF-L4O"/>
 <constraint firstItem="X6l-RJ-wLP" firstAttribute="top" secondItem="jq6-g0-iso" secondAttribute="bottom" constant="8" id="n1v-0P-a15"/>
 <constraint firstItem="q5y-4x-ZTA" firstAttribute="top" secondItem="d3Z-5H-f9B" secondAttribute="bottom" constant="8" id="oi5-tA-M4a"/>
 <constraint firstItem="LV4-8T-CsK" firstAttribute="trailing" secondItem="B9D-hB-JvF" secondAttribute="trailing" constant="16" id="sGw-kg-49y"/>

```

<constraint firstItem="jq6-g0-iso" firstAttribute="leading" secondItem="LV4-8T-
CsK" secondAttribute="leading" constant="21" id="tK1-6N-Qa4"/>
</constraints>
<viewLayoutGuide key="safeArea" id="LV4-8T-CsK"/>
<connections>
<outletCollection property="gestureRecognizers" destination="S1j-UN-XjB"
appends="YES" id="6bi-JB-yAY"/>
</connections>
<point key="canvasLocation" x="-31.199999999999999"
y="74.212893553223395"/>
</view>
<tapGestureRecognizer id="S1j-UN-XjB">
<connections>
<action selector="hideKeyboard:" destination="-1" id="hzq-5M-GJL"/>
<outlet property="delegate" destination="bLd-av-6CV" id="G0Q-1S-xgp"/>
</connections>
</tapGestureRecognizer>
</objects>
</document>
<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14460.31" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>

```

```

<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14460.20"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner" customClass="AddWarehouseViewController"
customModule="Warehouse" customModuleProvider="target">
<connections>
<outlet property="descriptionTextView" destination="N92-dt-r9I" id="2Mj-qA-
b6Z"/>
<outlet property="image" destination="Wyb-Dh-mV7" id="lVn-so-seP"/>
<outlet property="nameLabel" destination="frk-Gh-gzZ" id="H72-9z-Kow"/>
<outlet property="view" destination="MDj-wt-AAI" id="C3m-2S-dUo"/>
</connections>
</placeholder>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<view contentMode="scaleToFill" id="MDj-wt-AAI">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<imageView userInteractionEnabled="NO" contentMode="scaleToFill"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
image="noImage" translatesAutoresizingMaskIntoConstraints="NO" id="Wyb-
Dh-mV7">
<rect key="frame" x="87.5" y="28" width="200" height="200"/>

```

```

<constraints>
<constraint firstAttribute="width" constant="200" id="hUW-yP-egF"/>
<constraint firstAttribute="height" constant="200" id="yar-TM-inL"/>
</constraints>
</imageView>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Название склада" textAlignment="natural"
lineBreakMode="tailTruncation" baselineAdjustment="alignBaselines"
adjustsFontSizeToFit="NO" translatesAutoresizingMaskIntoConstraints="NO"
id="n7W-2B-4kk">
<rect key="frame" x="16" y="264" width="138" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Описание" textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="IMh-Ya-tmY">
<rect key="frame" x="16" y="339" width="81" height="21"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>
<nil key="highlightedColor"/>
</label>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" placeholder="Введите название склада"

```

```

textAlignment="natural" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="frk-Gh-gzZ">
<rect key="frame" x="16" y="293" width="343" height="30"/>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits"/>
</textField>
<textView clipsSubviews="YES" multipleTouchEnabled="YES"
contentMode="scaleToFill" text="Введите описание склада"
translatesAutoresizingMaskIntoConstraints="NO" id="N92-dt-r9I">
<rect key="frame" x="16" y="368" width="343" height="180"/>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<color key="textColor" white="0.66666666666666663" alpha="1"
colorSpace="custom" customColorSpace="genericGamma22GrayColorSpace"/>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<textInputTraits key="textInputTraits" autocapitalizationType="sentences"/>
</textView>
<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="npE-Z3-1oa">
<rect key="frame" x="168.5" y="201" width="119" height="30"/>
<state key="normal" title="Обложка склада"/>
<connections>
<action selector="selectImage:" destination="-1" eventType="touchUpInside"
id="zUZ-xg-jNC"/>
</connections>
</button>
</subviews>

```

```

<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<constraints>
<constraint firstItem="n7W-2B-4kk" firstAttribute="leading" secondItem="qsL-
uy-eww" secondAttribute="leading" constant="16" id="J8H-du-3ce"/>
<constraint firstItem="qsL-uy-eww" firstAttribute="trailing" secondItem="N92-dt-
r9I" secondAttribute="trailing" constant="16" id="Oe6-pK-VqX"/>
<constraint firstItem="Wyb-Dh-mV7" firstAttribute="trailing" secondItem="npE-
Z3-1oa" secondAttribute="trailing" id="Owi-6H-FBH"/>
<constraint firstItem="frk-Gh-gzZ" firstAttribute="top" secondItem="n7W-2B-
4kk" secondAttribute="bottom" constant="8" id="R8M-8n-phR"/>
<constraint firstItem="n7W-2B-4kk" firstAttribute="top" secondItem="Wyb-Dh-
mV7" secondAttribute="bottom" constant="36" id="RaN-kN-qI0"/>
<constraint firstItem="N92-dt-r9I" firstAttribute="top" secondItem="IMh-Ya-
tmY" secondAttribute="bottom" constant="8" id="U87-JX-EiU"/>
<constraint firstItem="Wyb-Dh-mV7" firstAttribute="top" secondItem="qsL-uy-
eww" secondAttribute="top" constant="8" id="ZAk-yX-d98"/>
<constraint firstItem="N92-dt-r9I" firstAttribute="height" secondItem="MDj-wt-
AAI" secondAttribute="height" multiplier="180:667" id="ZFe-on-bmJ"/>
<constraint firstItem="n7W-2B-4kk" firstAttribute="top" secondItem="Wyb-Dh-
mV7" secondAttribute="bottom" constant="36" id="b1D-7R-MCx"/>
<constraint firstItem="qsL-uy-eww" firstAttribute="trailing" secondItem="frk-Gh-
gzZ" secondAttribute="trailing" constant="16" id="ccb-z1-yvB"/>
<constraint firstItem="Wyb-Dh-mV7" firstAttribute="centerX" secondItem="qsL-
uy-eww" secondAttribute="centerX" id="dw1-q1-XQW"/>
<constraint firstItem="Wyb-Dh-mV7" firstAttribute="bottom" secondItem="npE-
Z3-1oa" secondAttribute="bottom" constant="-3" id="gGA-O8-qVE"/>
<constraint firstItem="N92-dt-r9I" firstAttribute="leading" secondItem="qsL-uy-
eww" secondAttribute="leading" constant="16" id="kY2-UV-GyR"/>

```



```

<constraint firstItem="frk-Gh-gzZ" firstAttribute="leading" secondItem="qsL-uy-eww" secondAttribute="leading" constant="16" id="srU-ew-cVx"/>
<constraint firstItem="IMh-Ya-tmY" firstAttribute="leading" secondItem="qsL-uy-eww" secondAttribute="leading" constant="16" id="tO4-eS-WeL"/>
<constraint firstItem="IMh-Ya-tmY" firstAttribute="top" secondItem="frk-Gh-gzZ" secondAttribute="bottom" constant="16" id="z9w-Im-hGg"/>
</constraints>
<viewLayoutGuide key="safeArea" id="qsL-uy-eww"/>
<point key="canvasLocation" x="-31.199999999999999"
y="74.212893553223395"/>
</view>
</objects>
<resources>
<image name="noImage" width="192" height="192"/>
</resources>
</document>
<?xml version="1.0" encoding="UTF-8"?>
<document type="com.apple.InterfaceBuilder3.CocoaTouch.XIB" version="3.0"
toolsVersion="14460.31" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14460.20"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>

```

```

<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<objects>
<placeholder placeholderIdentifier="IBFilesOwner" id="-1" userLabel="File's
Owner"/>
<placeholder placeholderIdentifier="IBFirstResponder" id="-2"
customClass="UIResponder"/>
<collectionViewCell opaque="NO" clipsSubviews="YES"
multipleTouchEnabled="YES" contentMode="center"
reuseIdentifier="Warehouse" id="gTV-IL-0wX" customClass="WarehouseCell"
customModule="Warehouse" customModuleProvider="target">
<rect key="frame" x="0.0" y="0.0" width="320" height="150"/>
<autoresizingMask key="autoresizingMask"/>
<view key="contentView" opaque="NO" clipsSubviews="YES"
multipleTouchEnabled="YES" contentMode="center">
<rect key="frame" x="0.0" y="0.0" width="320" height="150"/>
<autoresizingMask key="autoresizingMask" flexibleMaxX="YES"
flexibleMaxY="YES"/>
<subviews>
<imageViewclipsSubviews="YES" userInteractionEnabled="NO"
contentMode="scaleAspectFill" horizontalHuggingPriority="251"
verticalHuggingPriority="251" translatesAutoresizingMaskIntoConstraints="NO"
id="9He-sg-94v">
<rect key="frame" x="0.0" y="0.0" width="320" height="150"/>
</imageView>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Имя"
textAlignment="center" lineBreakMode="tailTruncation"

```

```

baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="aII-Q0-0mn">
<rect key="frame" x="12" y="12" width="296" height="42"/>
<fontDescription key="fontDescription" type="system" weight="semibold"
pointSize="35"/>
<color key="textColor"
cocoaTouchSystemColor="tableCellGroupedBackgroundColor"/>
<nil key="highlightedColor"/>
</label>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251" text="Label"
textAlignment="natural" lineBreakMode="tailTruncation"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="XTt-bf-eZV">
<rect key="frame" x="12" y="62" width="296" height="20.5"/>
<fontDescription key="fontDescription" type="system" weight="light"
pointSize="17"/>
<color key="textColor"
cocoaTouchSystemColor="tableCellGroupedBackgroundColor"/>
<nil key="highlightedColor"/>
</label>
</subviews>
</view>
<color key="backgroundColor" white="0.0" alpha="0.0" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<constraints>
<constraint firstItem="9He-sg-94v" firstAttribute="trailing" secondItem="XTt-bf-
eZV" secondAttribute="trailing" constant="12" id="4PT-KE-jfj"/>
<constraint firstItem="ZTg-uK-7eu" firstAttribute="bottom" secondItem="9He-sg-
94v" secondAttribute="bottom" id="65J-Mr-Pl7"/>

```

```

<constraint firstItem="9He-sg-94v" firstAttribute="leading" secondItem="gTV-IL-
0wX" secondAttribute="leading" id="6WA-un-WEI"/>
<constraint firstItem="9He-sg-94v" firstAttribute="top" secondItem="gTV-IL-
0wX" secondAttribute="top" id="JaA-YN-GfA"/>
<constraint firstItem="aII-Q0-0mn" firstAttribute="top" secondItem="gTV-IL-
0wX" secondAttribute="top" constant="12" id="Sil-tI-XXt"/>
<constraint firstItem="aII-Q0-0mn" firstAttribute="leading" secondItem="9He-sg-
94v" secondAttribute="leading" constant="12" id="YET-gw-QHM"/>
<constraint firstItem="XTt-bf-eZV" firstAttribute="top" secondItem="aII-Q0-
0mn" secondAttribute="bottom" constant="8" id="e5C-MT-mcU"/>
<constraint firstAttribute="trailing" secondItem="9He-sg-94v"
secondAttribute="trailing" id="gA1-tR-R3I"/>
<constraint firstItem="XTt-bf-eZV" firstAttribute="leading" secondItem="9He-sg-
94v" secondAttribute="leading" constant="12" id="h48-5r-DQp"/>
<constraint firstItem="aII-Q0-0mn" firstAttribute="centerX" secondItem="ZTg-
uK-7eu" secondAttribute="centerX" id="wO0-q3-5Pe"/>
</constraints>
<viewLayoutGuide key="safeArea" id="ZTg-uK-7eu"/>
<size key="customSize" width="417" height="289"/>
<connections>
<outlet property="about" destination="XTt-bf-eZV" id="S5l-68-Fep"/>
<outlet property="image" destination="9He-sg-94v" id="o90-zN-Mrx"/>
<outlet property="title" destination="aII-Q0-0mn" id="TN1-aC-tTn"/>
</connections>
<point key="canvasLocation" x="430.39999999999998"
y="260.86956521739131"/>
</collectionViewCell>
</objects>
</document>
<?xml version="1.0" encoding="UTF-8"?>

```

```

<document type="com.apple.InterfaceBuilder3.CocoaTouch.Storyboard.XIB"
version="3.0" toolsVersion="14490.70" targetRuntime="iOS.CocoaTouch"
propertyAccessControl="none" useAutolayout="YES" useTraitCollections="YES"
useSafeAreas="YES" colorMatched="YES" initialViewController="JWe-oc-
uB1">
<device id="retina4_7" orientation="portrait">
<adaptation id="fullscreen"/>
</device>
<dependencies>
<deployment identifier="iOS"/>
<plugIn identifier="com.apple.InterfaceBuilder.IBCocoaTouchPlugin"
version="14490.49"/>
<capability name="Safe area layout guides" minToolsVersion="9.0"/>
<capability name="documents saved in the Xcode 8 format"
minToolsVersion="8.0"/>
</dependencies>
<scenes>
<!--Склады-->
<scene sceneID="hBD-i9-hNu">
<objects>
<viewControllerstoryboardIdentifier="Warehouse" id="CX1-us-lmP"
customClass="MainViewController" customModule="Warehouse"
customModuleProvider="target" sceneMemberID="viewController">
<view key="view" contentMode="scaleToFill" id="Pql-C8-gCI">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>

```

```

<collectionViewclipsSubviews="YES" multipleTouchEnabled="YES"
contentMode="scaleToFill" dataMode="prototypes"
translatesAutoresizingMaskIntoConstraints="NO" id="ZiN-l2-y1k">
<rect key="frame" x="0.0" y="116" width="375" height="551"/>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<collectionViewFlowLayout key="collectionViewLayout"
minimumLineSpacing="10" minimumInteritemSpacing="18" id="vRH-8g-x1c">
<size key="itemSize" width="340" height="100"/>
<size key="headerReferenceSize" width="0.0" height="0.0"/>
<size key="footerReferenceSize" width="0.0" height="0.0"/>
<inset key="sectionInset" minX="0.0" minY="18" maxX="0.0" maxY="18"/>
</collectionViewFlowLayout>
<cells>
<collectionViewCell opaque="NO" clipsSubviews="YES"
multipleTouchEnabled="YES" contentMode="center"
reuseIdentifier="Warehouse" id="46w-kD-Nsr">
<rect key="frame" x="87.5" y="18" width="200" height="100"/>
<autoresizingMask key="autoresizingMask" flexibleMaxX="YES"
flexibleMaxY="YES"/>
<view key="contentView" opaque="NO" clipsSubviews="YES"
multipleTouchEnabled="YES" contentMode="center"
insetsLayoutMarginsFromSafeArea="NO">
<rect key="frame" x="0.0" y="0.0" width="200" height="100"/>
<autoresizingMask key="autoresizingMask"/>
</view>
<size key="customSize" width="200" height="100"/>
</collectionViewCell>
</cells>
</collectionView>

```

```

</subviews>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<constraints>
<constraint firstItem="ZiN-l2-y1k" firstAttribute="top" secondItem="8Jm-CE-
IOo" secondAttribute="top" id="6cx-Qy-TdN"/>
<constraint firstItem="ZiN-l2-y1k" firstAttribute="bottom" secondItem="8Jm-CE-
IOo" secondAttribute="bottom" id="d1b-4K-NBw"/>
<constraint firstItem="ZiN-l2-y1k" firstAttribute="trailing" secondItem="8Jm-CE-
IOo" secondAttribute="trailing" id="mzk-WT-Htf"/>
<constraint firstItem="ZiN-l2-y1k" firstAttribute="leading" secondItem="8Jm-CE-
IOo" secondAttribute="leading" id="wqI-3D-fLL"/>
</constraints>
<viewLayoutGuide key="safeArea" id="8Jm-CE-IOo"/>
</view>
<navigationItem key="navigationItem" title="Склады" id="Fsu-CN-Ff4">
<barButtonItem key="rightBarButtonItem" systemItem="add" id="sLz-Hm-fmS">
<connections>
<action selector="AddAction:" destination="CX1-us-lmP" id="cV1-k4-rFC"/>
</connections>
</barButtonItem>
</navigationItem>
<connections>
<outlet property="mainCollectionView" destination="ZiN-l2-y1k" id="g2U-5a-
nZJ"/>
</connections>
</viewController>
<placeholder placeholderIdentifier="IBFirstResponder" id="Q7W-zK-Unj"
userLabel="First Responder" sceneMemberID="firstResponder"/>
</objects>

```

```

<point key="canvasLocation" x="42.39999999999999" y="-
932.38380809595208"/>
</scene>
<!--Pin Code View Controller-->
<scene sceneID="gcz-eW-olZ">
<objects>
<viewController id="JWe-oc-uB1" customClass="PinCodeViewController"
customModule="Warehouse" customModuleProvider="target"
sceneMemberID="viewController">
<view key="view" contentMode="scaleToFill" id="bdR-ZF-qpi">
<rect key="frame" x="0.0" y="0.0" width="375" height="667"/>
<autoresizingMask key="autoresizingMask" widthSizable="YES"
heightSizable="YES"/>
<subviews>
<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="fpK-4h-edT">
<rect key="frame" x="109" y="396.5" width="157" height="30"/>
<state key="normal" title="ИспользоватьFaceID"/>
<connections>
<action selector="biometricAction:" destination="JWe-oc-uB1"
eventType="touchUpInside" id="Wh9-FJ-4j0"/>
</connections>
</button>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" textAlignment="center" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="zmK-mB-UvA">
<rect key="frame" x="82.5" y="329" width="34" height="29"/>

```



```

<constraints>
<constraint firstAttribute="height" constant="29" id="ZQn-vJ-H4E"/>
<constraint firstAttribute="width" constant="34" id="aoN-zZ-V1k"/>
</constraints>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits" keyboardType="numberPad"/>
</textField>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" textAlignment="center" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="8Vf-mu-IDT">
<rect key="frame" x="267.5" y="328.5" width="34" height="30"/>
<constraints>
<constraint firstAttribute="width" constant="34" id="3wx-Zv-VyJ"/>
<constraint firstAttribute="height" constant="30" id="mtj-3y-Q0R"/>
</constraints>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits" keyboardType="numberPad"/>
</textField>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" textAlignment="center" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="3aa-25-vxg">
<rect key="frame" x="205.5" y="328.5" width="34" height="30"/>
<constraints>
<constraint firstAttribute="width" constant="34" id="0uR-K0-7B4"/>
<constraint firstAttribute="height" constant="30" id="LNW-tc-7oh"/>
</constraints>

```

```

<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits" keyboardType="numberPad"/>
</textField>
<textField opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="left" contentVerticalAlignment="center"
borderStyle="roundedRect" textAlignment="center" minimumFontSize="17"
translatesAutoresizingMaskIntoConstraints="NO" id="BIt-6Q-LMZ">
<rect key="frame" x="143.5" y="328.5" width="34" height="30"/>
<constraints>
<constraint firstAttribute="width" constant="34" id="I8G-Pd-oGy"/>
<constraint firstAttribute="height" constant="30" id="K41-Ic-cad"/>
</constraints>
<nil key="textColor"/>
<fontDescription key="fontDescription" type="system" pointSize="14"/>
<textInputTraits key="textInputTraits" keyboardType="numberPad"/>
</textField>
<label opaque="NO" userInteractionEnabled="NO" contentMode="left"
horizontalHuggingPriority="251" verticalHuggingPriority="251"
text="Запегестрируйтеновыйпароль" textAlignment="center"
lineBreakMode="tailTruncation" numberOfLines="2"
baselineAdjustment="alignBaselines" adjustsFontSizeToFit="NO"
translatesAutoresizingMaskIntoConstraints="NO" id="cyS-Y6-cnX">
<rect key="frame" x="60.5" y="225" width="254.5" height="21"/>
<constraints>
<constraint firstAttribute="height" relation="greaterThanOrEqual" constant="21"
id="gDl-rM-P5H"/>
</constraints>
<fontDescription key="fontDescription" type="system" pointSize="17"/>
<nil key="textColor"/>

```

```

<nil key="highlightedColor"/>
</label>
<button opaque="NO" contentMode="scaleToFill"
contentHorizontalAlignment="center" contentVerticalAlignment="center"
buttonType="roundedRect" lineBreakMode="middleTruncation"
translatesAutoresizingMaskIntoConstraints="NO" id="P97-SF-Ku4">
<rect key="frame" x="106" y="426" width="163" height="30"/>
<state key="normal" title="Создатьновыйпароль"/>
<connections>
<action selector="createNewPassword:" destination="JWe-oc-uB1"
eventType="touchUpInside" id="JAc-xE-lQM"/>
</connections>
</button>
</subviews>
<color key="backgroundColor" white="1" alpha="1" colorSpace="custom"
customColorSpace="genericGamma22GrayColorSpace"/>
<gestureRecognizers/>
<constraints>
<constraint firstItem="P97-SF-Ku4" firstAttribute="top" secondItem="fpK-4h-
edT" secondAttribute="bottom" constant="-0.5" id="Ebn-1p-q4C"/>
<constraint firstItem="cyS-Y6-cnx" firstAttribute="centerX" secondItem="fpK-
4h-edT" secondAttribute="centerX" id="Gm6-2d-ckP"/>
<constraint firstItem="zmK-mB-UvA" firstAttribute="centerY"
secondItem="6dC-qS-Ff1" secondAttribute="centerY" id="PsD-FH-VEp"/>
<constraint firstItem="BI6-6Q-LMZ" firstAttribute="centerX" secondItem="6dC-
qS-Ff1" secondAttribute="centerX" constant="-27" id="RWc-jF-KWP"/>
<constraint firstItem="cyS-Y6-cnx" firstAttribute="leading"
relation="greaterThanOrEqualTo" secondItem="6dC-qS-Ff1"
secondAttribute="leading" constant="20" id="TWo-Ky-fpQ"/>

```

```

<constraint firstItem="fpK-4h-edT" firstAttribute="leading" secondItem="6dC-qS-
Ff1" secondAttribute="leading" constant="109" id="Thh-a9-Def"/>
<constraint firstItem="cyS-Y6-cnx" firstAttribute="centerX" secondItem="bdR-
ZF-qpi" secondAttribute="centerX" id="U8p-W5-dVT"/>
<constraint firstItem="8Vf-mu-IDT" firstAttribute="centerY" secondItem="BI-
6Q-LMZ" secondAttribute="centerY" id="UXw-hL-dng"/>
<constraint firstItem="8Vf-mu-IDT" firstAttribute="leading" secondItem="3aa-
25-vxg" secondAttribute="trailing" constant="28" id="WM0-7W-Ur8"/>
<constraint firstItem="P97-SF-Ku4" firstAttribute="centerX" secondItem="6dC-
qS-Ff1" secondAttribute="centerX" id="Zen-EM-i3r"/>
<constraint firstItem="cyS-Y6-cnx" firstAttribute="top"
relation="greaterThanOrEqual" secondItem="6dC-qS-Ff1" secondAttribute="top"
constant="20" id="d9z-ID-DvN"/>
<constraint firstItem="3aa-25-vxg" firstAttribute="leading" secondItem="BI-6Q-
LMZ" secondAttribute="trailing" constant="28" id="ex0-mb-3Bg"/>
<constraint firstItem="BI-6Q-LMZ" firstAttribute="leading" secondItem="zmK-
mB-UvA" secondAttribute="trailing" constant="27" id="h8H-zX-5Ot"/>
<constraint firstItem="zmK-mB-UvA" firstAttribute="centerY" secondItem="BI-
6Q-LMZ" secondAttribute="centerY" id="hOQ-Zf-viJ"/>
<constraint firstItem="zmK-mB-UvA" firstAttribute="top" secondItem="cyS-Y6-
cnx" secondAttribute="bottom" constant="83" id="iAB-GO-Tjk"/>
<constraint firstItem="fpK-4h-edT" firstAttribute="top" secondItem="BI-6Q-
LMZ" secondAttribute="bottom" constant="38" id="qAR-0w-UNL"/>
<constraint firstItem="3aa-25-vxg" firstAttribute="centerY" secondItem="8Vf-
mu-IDT" secondAttribute="centerY" id="qE3-XT-CY7"/>
</constraints>
<viewLayoutGuide key="safeArea" id="6dC-qS-Ff1"/>
<connections>
<outletCollection property="gestureRecognizers" destination="bzk-Zx-RcP"
appends="YES" id="Cax-tb-FaH"/>

```

```

</connections>

</view>

<connections>

<outlet property="biometricBtn" destination="fpK-4h-edT" id="YiP-yD-3bQ"/>
<outlet property="firstChar" destination="zmK-mB-UvA" id="Ub5-Cw-WTs"/>
<outlet property="fourthChar" destination="8Vf-mu-IDT" id="4i7-6K-yK2"/>
<outlet property="mainLabel" destination="cyS-Y6-cnx" id="nmU-ai-e0s"/>
<outlet property="secondChar" destination="BIt-6Q-LMZ" id="ujV-PK-Qmw"/>
<outlet property="thirdChar" destination="3aa-25-vxg" id="TU9-1a-0WY"/>
<segue destination="9WI-Pw-g7q" kind="show"
identifier="ShowMainNavigation" id="Ote-5Q-ZaO"/>

</connections>

</viewController>

<placeholder placeholderIdentifier="IBFirstResponder" id="ttb-0E-UYb"
userLabel="First Responder" sceneMemberID="firstResponder"/>

<tapGestureRecognizer id="bzk-Zx-RcP">

<connections>

<action selector="tapToView:" destination="JWe-oc-uB1" id="qqy-UM-Pck"/>

</connections>

</tapGestureRecognizer>

</objects>

<point key="canvasLocation" x="-1636" y="-932.38380809595208"/>

</scene>

<!--Navigation Controller-->

<scene sceneID="wSO-tR-QvI">

<objects>

<navigationController id="9WI-Pw-g7q" sceneMemberID="viewController">
<navigationBar key="navigationBar" contentMode="scaleToFill"
insetsLayoutMarginsFromSafeArea="NO" largeTitles="YES" id="doJ-3Z-2XQ">
<rect key="frame" x="0.0" y="20" width="375" height="96"/>

```

```

<autoresizingMask key="autoresizingMask"/>
</navigationBar>
<toolbar key="toolbar" opaque="NO" clearsContextBeforeDrawing="NO"
contentMode="scaleToFill" insetsLayoutMarginsFromSafeArea="NO" id="CQ0-
JN-ebx">
<autoresizingMask key="autoresizingMask"/>
</toolbar>
<connections>
<segue destination="CX1-us-lmP" kind="relationship"
relationship="rootViewController" id="m75-sg-hEW"/>
</connections>
</navigationController>
<placeholder placeholderIdentifier="IBFirstResponder" id="xec-Pv-syS"
userLabel="First Responder" sceneMemberID="firstResponder"/>
</objects>
<point key="canvasLocation" x="-758" y="-932"/>
</scene>
</scenes>
</document>

```