1. РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ
   1. Анализ предметной области
   2. Общие требования

1.1 Назначение документа

Настоящее Техническое задание определяет требования и порядок разработки мобильного приложения «Склад» пополняемого репозитория фанатского творчества.

1.2 Наименование Исполнителя и Заказчика

* Заказчик – ИП MAMS; Адрес: с. Ольгино, ул. Лесная, 5;
* Исполнитель – ИП Baimors; Адрес: г. Н. Новгород, Московское ш., 1.

1.3 Информация о компании

1. Название: ИП MAMS;
   1. Отрасль и направления деятельности: развлечения;
   2. Перечень услуг:
      * Хранение репозитория текстовых работ;
      * Организация публикации текстовых работ;
      * Аналитика;
      * Организация подписок.
   3. Публичные контактные данные: +79625062815;

Конкуренты: Breakpoint SIA, Organization for Transformative Works.

* 1. Основание для разработки приложения

Основанием для разработки приложения является Договор № 00000001 от 25.09.2024 между Исполнителем и Заказчиком.

* 1. Назначение приложения

Назначением мобильного приложения «Склад» является организация доступа к ресурсам и услугам Заказчика, поддержание её положительного и современного имиджа, а также создание сообщества единомышленников, где пользователи могут обмениваться идеями и находить вдохновение.

Актуальность разработки приложения:

* приложение может предоставить пользователям инструменты для публикации работ;
* приложение может организовать комфортный доступ к обширному репозиторию работ.
  1. Целевая аудитория
  2. Пол: женский, мужской;
  3. Возраст: от 16 лет;
  4. Занятость: школьники, студенты, заинтересованные в фанатском творчестве;
  5. Цели и задачи приложения
  6. Тип приложения: мобильное приложение.
  7. Цели приложения:
  + Организация доступа к репозиторию работ;
  + Организация возможности добавления, изменения и удаления новых работ.
  1. Целевые действия пользователей приложения:
  + Авторизация пользователей;
  + Добавление/изменение/удаление текстовых работ;
  + Поиск и просмотр работ;
  + Добавление/удаление понравившихся работ и их просмотр;
  + Просмотр и изменение данных пользователя в профиле.
  1. Участники и роли

Таблица 1 – Участники и их роль в проекте

|  |  |
| --- | --- |
| Участники | Роли |
| М.А. Трянина | Разработка дизайна и работа с графическим контентом. Осуществление тестирования приложения. |
| Разработка и администрирование базы данных. Разработка функционала приложения и интерфейсов. |

* 1. Стадии разработки приложения

Таблица 2 – Стадии разработки проекта

|  |  |  |  |
| --- | --- | --- | --- |
| Стадии | Этапы | Ответственность | Сроки |
| Постановка целей и задач | Разработка технического задания | Трянина М.А. | 25.09.2024 – 25.09.2024 |
| Выбора паттерна проектирования |
| Проектирование требований к серверной части приложения |
| Проектирование требований к мобильному приложению |
| Выбор для реализации ПО и обоснование выбора |
| Разработка макета приложения | Выбор цветовой палитры | 09.09.2024 – 09.09.2024 |
| Выбор шрифта для заголовков и основного текста |
| Разработка макета каждого экрана разрабатываемого мобильного приложения |
| Разработка серверной части приложения | Проектирование серверной части | 23.10.2024 – 30.10.2024 |
| Разработка серверной части |
| Тестирование серверной части приложения | Разработка документации для тестирования сервера приложения | 13.11.2024 – 20.11.2024 |
| Unit-тесты |
| Разработка мобильного приложения | Подключение шрифтов и настройка ранее выбранных цветов | 20.11.2024  –  18.12.2024 |
| Верстка макета |
| Подключение библиотек и интернета |
| Подключение к серверной части приложения |
| Разработка вспомогательных методов, и методов для обработки данных с сервера |
| Тестирование мобильного приложения | Разработка документации для тестирования мобильного приложения | 18.12.2024 – 25.12.2024 |
| Unit-тесты |
| Мониторинг работоспособности и решение возникающих проблем. |
| Сопровождение и поддержка | Исправление ошибок и внесение изменений по запросам пользователей. | С 25.12.2024 |
| Модернизация, обновление и масштабирование системы. |
| Мониторинг работоспособности и решение возникающих проблем. |

* 1. Ресурсы

1. Человеческие ресурсы: разработчики и пользователи;
2. Инфраструктурные ресурсы: тестовая база данных;
3. Программные ресурсы: ОС Windows 10, инструменты разработки Visual Studio и Android Studio, графический онлайн-редактор Figma, система управления версиями Git, система хранения проектов Gogs, SaaS Supabase;
4. Информационные ресурсы: документация по языкам программирования.
   1. Нефункциональные требования
   2. Требования к персоналу

Технических специалистов узкого профиля для поддержания приложения не требуется, необходимыми и достаточными навыками будет владение общими понятиями работы с персональным компьютером на уровне базового пользователя и выше.

* 1. Требования к разграничению доступа

В соответствии с правами доступа можно выделить 1 категорию. пользователей сайта: Пользователи – доступ ко всей общедоступной информации в приложении, просмотр, редактирование и удаление.

* 1. Предварительная структура приложения

Основная страница приложения должна представлять собой страницу с новостями и событиями платформы, на которую выполняется переход после авторизации пользователя в системе.

Должна быть предусмотрена возможность перехода на следующие страницы с главной:

* Популярные работы;
* Читатели и авторы;
* Поиск работ;
* Профиль пользователя.
  1. Элементы макета приложения

Структура страниц приложения должна состоять из следующих элементов:

* «Навигационные элементы» – в данном блоке должны располагаться ссылки на соответствующие функциональные страницы, на которые можно перейти;
* «Шапка» – горизонтальный блок, расположенный наверху приложения – в данном блоке представлены возможности перехода на упомянутые выше страницы;
* «Основная часть» – часть страницы, на которой представлен функционал либо информация, соответствующие данному блоку.
  1. Требования к оформлению и дизайну
  2. Логотип: должен быть отображен;
  3. Предпочтительные цвета: оттенки зеленого, чёрного и коричневого;
  4. Возможные предпочтительные шрифты: Roboto, дополнительные.
  5. Количество вариантов дизайна

Основная тема приложения – светлая, дополнительно по требованию Заказчика может быть разработана темная тема.

* 1. Информационное наполнение приложения

Требуется. Представление информации, хранящейся в базе данных.

* 1. Функциональные требования
  2. Примерный набор модулей
  3. Регистрация и аутентификация пользователей;
  4. Добавление/изменение/удаление работ;
  5. Поиск и фильтрация работ;
  6. Управление личным кабинетом пользователя.
  7. Дополнительно
  8. Дополнительные пожелания

Дополнительные пожелания диктуются Заказчиком в процессе разработки.

* 1. Выбор паттерна проектирования

Основой дальнейшего проектирования и разработки является выбор паттерна.

1. The Model—View—Controller (MVC) Pattern – шаблон, в которое приложение делится на на три компонента (Model – бизнес-логика, View – видимая область, Controller – основная логика).

Схематичное построение данного шаблона можно увидеть на рисунке 1.

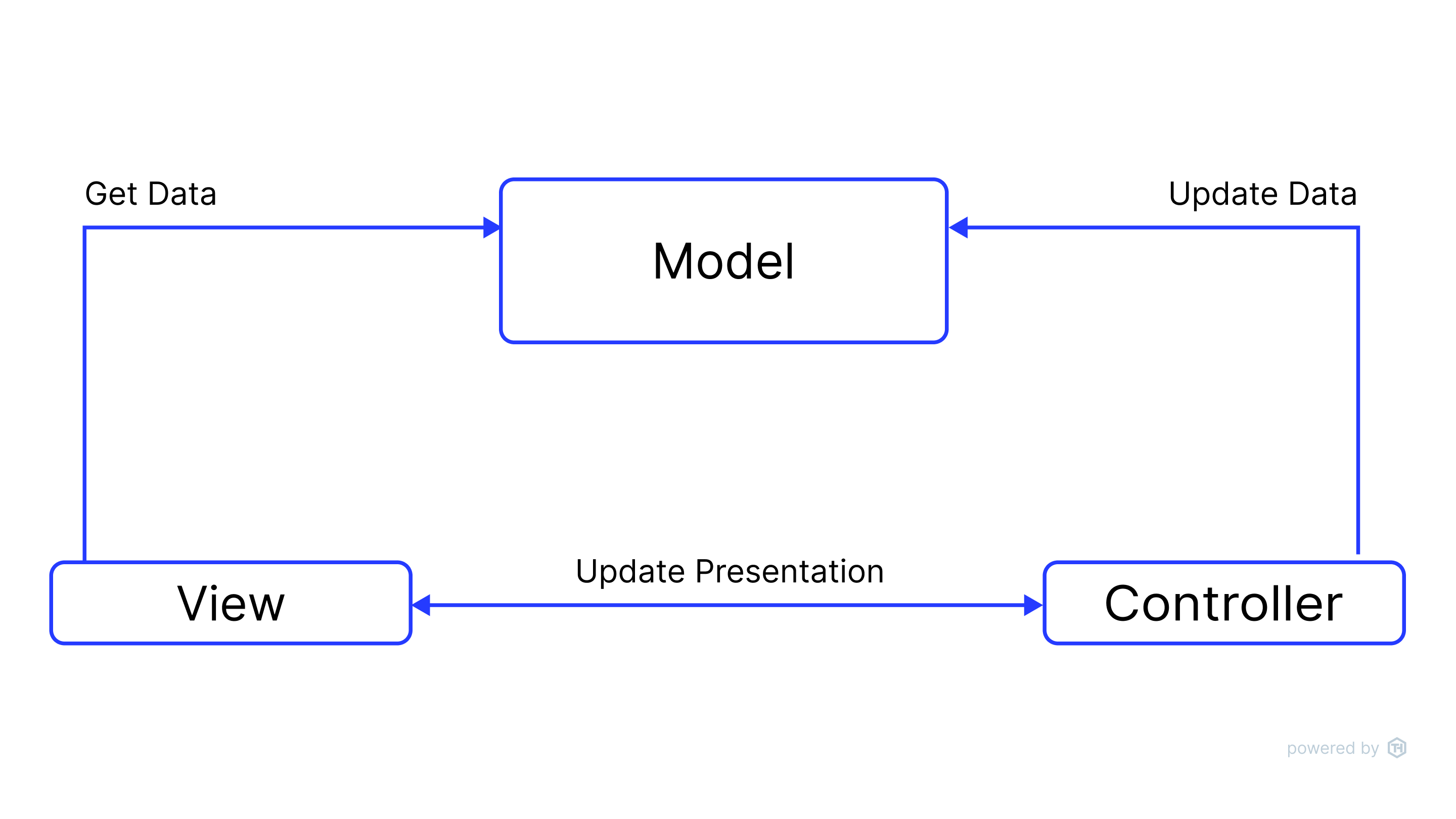


Рисунок 1 – MVC

Данный паттерн подлежит применению из-за реализуемости в нем бизнес-логики, поддержки асинхронных методов, изолированности всей модели от модификаций и в целом довольного быстрого процесса разработки. Однако, MVC может быть достаточно захламленным из-за большого количества кода в контроллере, а также не подвержен модульному тестированию.

1. The Model—View—Presenter (MVP) Pattern – паттерн, изолирующий логику представления от интерфейса для упрощения тестирования и поддержки. Также состоит включает в себя три компонента (Model – бизнес-логика, View – пользовательский интерфейс, Presenter – представление данных из модели).

Схематичное построение данного шаблона можно увидеть на рисунке 2.

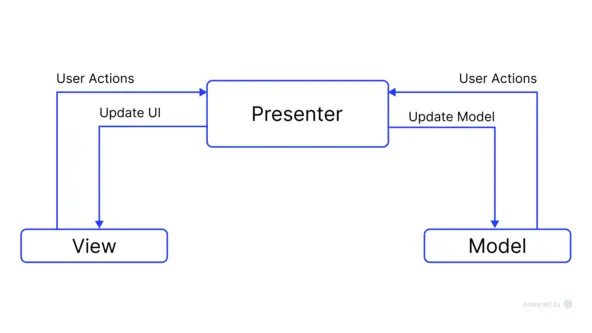


Рисунок 2 – MVP

Среди преимуществ данного шаблона: делает View абстрактной, так, чтобы ее можно было легко поменять; переиспользует View и Presenter, имеет сравнительно более читабельный и удобный в сопровождении код, а также удобен и прост в тестировании. Среди недостатков: имеет тесную связь между View и Presenter, реализует огромное количество интерфейсов между слоями, но также имеет довольно избыточный размер кода.

1. The Model—View—ViewModel (MVVM) Pattern – паттерн, реализующий двустороннюю привязку данных между View и ViewModel для упрощения взаимодействия. Точно так же состоит из 3 компонентов: Model – бизнес-логика, View – пользовательский интерфейс, ViewModel – большая часть логики пользовательского интерфейса. Изменения в ViewModel автоматически отражаются в View, что упрощает управление состоянием.

Схематичное построение данного шаблона можно увидеть на рисунке 3.

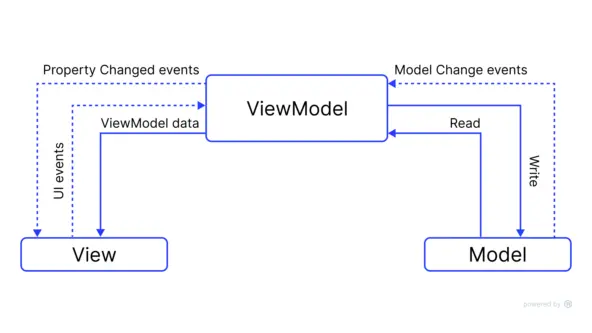


Рисунок 3 – MVVM

Сочетает в себе многие преимущества, в том числе разделение логики приложения и представления, гибкость и переиспользование, прямое связывание данных, а также упрощение тестирования. Среди недостатков – опять же, лишь большой объем кода.

Итак, ввиду некоторых особенностей, включая тематику приложения, язык и среду разработки, а также вышеописанных преимуществ и недостатков в дальнейшем приложение будет основано на паттерне MVVM.

* 1. Проектирование требований к серверной части приложения

Для организации хранения данных приложения необходимо спроектировать основные сущности будущей базы данных. Так как на физическом уровне сущности соответствует таблица, то перечислим основные таблицы БД.

В реляционную модель проектированной БД будут входить следующие таблицы (сущности): данные о ролях (Roles), данные о сотрудниках (Employees), данные о проектах (Projects), смежная таблица для описания связи между сотрудниками и проектами (ProjectsAndEmployees), данные о торговых сетях (Trade Networks), данные о торговых точках (Outlets), смежная таблица для описания связи между торговыми точками и проектами (ProjectsAndOutlets), данные о продуктах (Products), данные о продуктах, содержащихся в визитах (ProductReports), данные о визитах (Visits) и данные о том, принят ли визит (Acceptance).

Назначения всех таблиц (сущностей), имеющихся в базе данных, можно увидеть в таблице 3.

Таблица 3 – Список сущностей

|  |  |  |
| --- | --- | --- |
| № | Название | Данные |
| 1 | Сотрудники | Данные о сотрудниках. |
| 2 | Роли | Перечень ролей сотрудников. |
| 3 | Проекты | Данные о проектах. |
| 4 | Проекты и сотрудники | Смежная таблица для описания связи между сотрудниками и проектами. |
| 5 | Торговые сети | Данные о торговых сетях. |
| 6 | Торговые точки | Данные о торговых точках. |
| 7 | Проекты и торговые точки | Смежная таблица для описания связи между торговыми точками и проектами. |
| 8 | Продукты | Данные о продуктах. |
| 9 | Отчёты по продуктам | Данные о продуктах, содержащихся в визитах. |
| 10 | Визиты | Данные о визитах. |
| 11 | Принятие | Перечень статусов принятия визитов. |

Данные таблицы связывают определённые отношения, которые отражены в таблице 4.

Таблица 4 – Список связей

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название связи | Сущности, участвующие в связи | Назначение |
| 1 | 1:М | Сотрудник-Роль | У одного сотрудника может быть одна роль. Одна роль может принадлежать нескольким сотрудникам. |
| 2 | М:М | Сотрудник-Проект | У одного сотрудника может быть много проектов. У одного проекта может быть много сотрудников. |
| 3 | 1:М | Торговая сеть-Торговая точка | У одной торговой сети может быть много торговых точек. Торговая точка может принадлежать одной торговой сети. |
| 4 | М:М | Торговая точка-Проект | У одного проекта может быть много торговых точек. У одной торговой точки может быть много проектов. |
| 5 | 1:М | Проект-Продукт | У одного проекта может быть много продуктов. |
| 6 | 1:М | Продукт-Отчёт по продуктам | В одном отчёте по продуктам может быть много продуктов. |
| 7 | 1:М | Визит-Отчёт по продуктам | В одном визите может быть много отчётов по продуктам. |
| 8 | 1:М | Визит-Принятие | Один статус принятия может быть у нескольких визитов. |

Помимо вышеперечисленного, стоит отметить, что база данных будет развернута в облачном инструментальном сервисе Supabase, имеющем собственные механизмы авторизации и хранения медиаданных.

Так, данные для входа хранятся в таблице с пользователями в соответствующей схеме «auth», с которой и будет производиться взаимодействие в дальнейшем.

Помимо этого, Supabase предоставляет хранилище для разного рода медиаданных, которые имеют зависимости в базе данных.

* + 1. Описание предметной области

Отдел координации мерчандайзингового агентства Acosta занимается учетом и контролем за работой мерчандайзеров множества проектов на торговых точках различных торговых сетей.

Таким образом, предметная область в данном случае характеризуется следующими сущностями:

Сотрудники – таблица, хранящая в себе личную, контактную и учетную информацию обо всех сотрудниках, так или иначе контролируемых агентством Acosta. Дополнительно для каждого из сотрудников имеется информация о проектах, за которыми тот закреплен.

Проекты – поставляемая от одного заказчика продукция, количество и качество расстановки которой на полках торговых точек контролируется мерчандайзерами. У каждого проекта есть соответственно название и плановое количество визитов в неделю, требуемое заказчиком, а также конкретный перечень возможных продуктов.

Торговые точки представляют собой собственно магазины, в которые поставляется продукция по проектам. О каждой торговой точке хранится информация о ее адресе и местоположении (координатах), а также торговой сети, которой принадлежит данная торговая точка.

Визиты совершаются на конкретную торговую точку по конкретному проекту. О них известна дата, время, имеется комментарий мерчандайзера, оценка принятия и отчет по продуктам, в котором для каждого из них хранится его цена, цена по карте, актуальный и виртуальный остатки.

* + 1. Описание основных сущностей предметной области

В результате проведенного анализа предметной области базы данных «мерчендайзинговое агенство "Acosta"» легко перечислить основные сущности этой БД. Так как на физическом уровне сущности соответствует таблица, то просто перечислим основные таблицы БД.

В реляционную модель проектированной БД будут входить следующие таблицы (сущности): данные о ролях (Roles), данные о сотрудниках (Employees), данные о проектах (Projects), смежная таблица для описания связи между сотрудниками и проектами (ProjectsAndEmployees), данные о торговых сетях (Trade Networks), данные о торговых точках (Outlets), смежная таблица для описания связи между торговыми точками и проектами (ProjectsAndOutlets), данные о продуктах (Products), данные о продуктах, содержащихся в визитах (ProductReports), данные о визитах (Visits) и данные о том, принят ли визит (Acceptance).

Назначения всех таблиц (сущностей), имеющихся в базе данных, можно увидеть в таблице 3.

Таблица 3 – Список сущностей

|  |  |  |
| --- | --- | --- |
| № | Название | Данные |
| 1 | Сотрудники | Данные о сотрудниках. |
| 2 | Роли | Перечень ролей сотрудников. |
| 3 | Проекты | Данные о проектах. |
| 4 | Проекты и сотрудники | Смежная таблица для описания связи между сотрудниками и проектами. |
| 5 | Торговые сети | Данные о торговых сетях. |
| 6 | Торговые точки | Данные о торговых точках. |
| 7 | Проекты и торговые точки | Смежная таблица для описания связи между торговыми точками и проектами. |
| 8 | Продукты | Данные о продуктах. |
| 9 | Отчёты по продуктам | Данные о продуктах, содержащихся в визитах. |
| 10 | Визиты | Данные о визитах. |
| 11 | Принятие | Перечень статусов принятия визитов. |

Данные таблицы связывают определённые отношения, которые отражены в таблице 4.

Таблица 4 – Список связей

|  |  |  |  |
| --- | --- | --- | --- |
| № | Название связи | Сущности, участвующие в связи | Назначение |
| 1 | 1:М | Сотрудник-Роль | У одного сотрудника может быть одна роль. Одна роль может принадлежать нескольким сотрудникам. |
| 2 | М:М | Сотрудник-Проект | У одного сотрудника может быть много проектов. У одного проекта может быть много сотрудников. |
| 3 | 1:М | Торговая сеть-Торговая точка | У одной торговой сети может быть много торговых точек. Торговая точка может принадлежать одной торговой сети. |
| 4 | М:М | Торговая точка-Проект | У одного проекта может быть много торговых точек. У одной торговой точки может быть много проектов. |
| 5 | 1:М | Проект-Продукт | У одного проекта может быть много продуктов. |
| 6 | 1:М | Продукт-Отчёт по продуктам | В одном отчёте по продуктам может быть много продуктов. |
| 7 | 1:М | Визит-Отчёт по продуктам | В одном визите может быть много отчётов по продуктам. |
| 8 | 1:М | Визит-Принятие | Один статус принятия может быть у нескольких визитов. |

* + 1. Создание даталогической модели базы данных

Даталогическая модель отражается графически в виде схемы базы данных, где указываются имена сущностей, их атрибуты и связи между сущностями.

Схема связей представлена на рисунке 11.

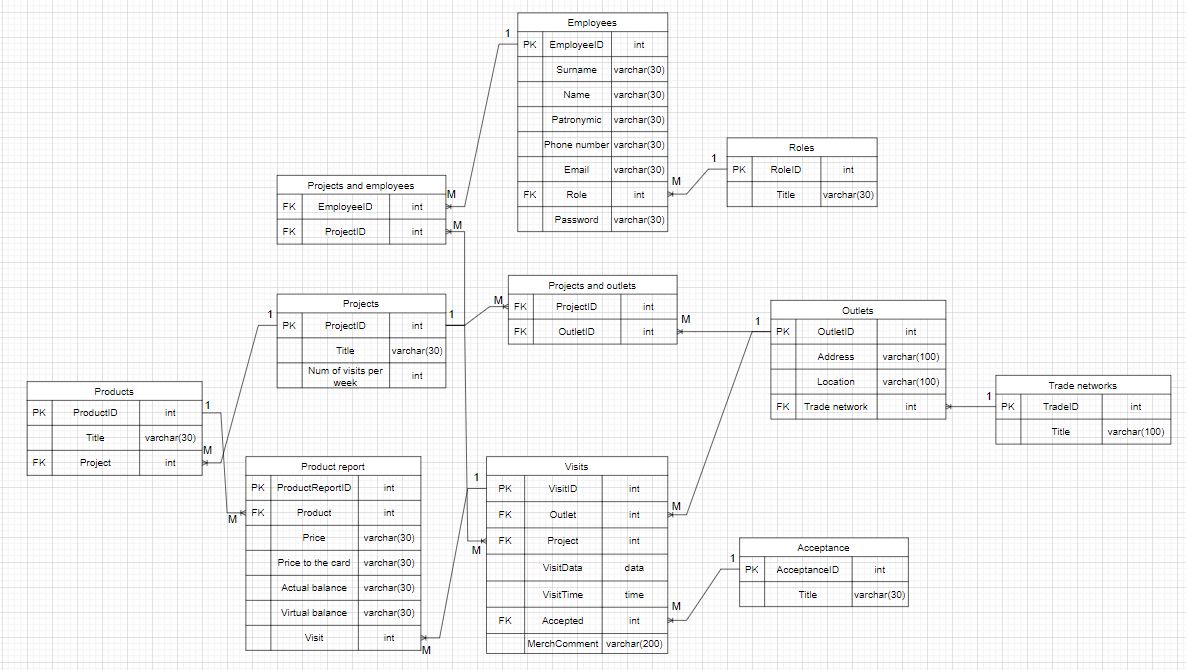


Рисунок 11 – Даталогическая модель базы данных

* + 1. Создание таблиц и связей

Для создания таблиц будет использоваться инструмент администрирования базы данных DBeaver с развернутой там БД PostgreSQL.

Листинги запросов на создание и заполнение таблиц можно увидеть в приложении Г соответственно в таблицах Г1-Г2. Результат создания таблиц представлен на рисунке 12, полученная схема БД – на рисунке 13.

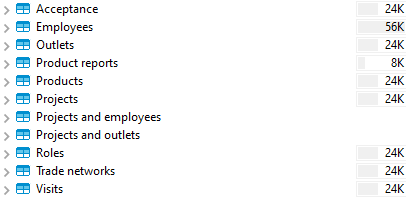


Рисунок 12 – Результат создания таблиц

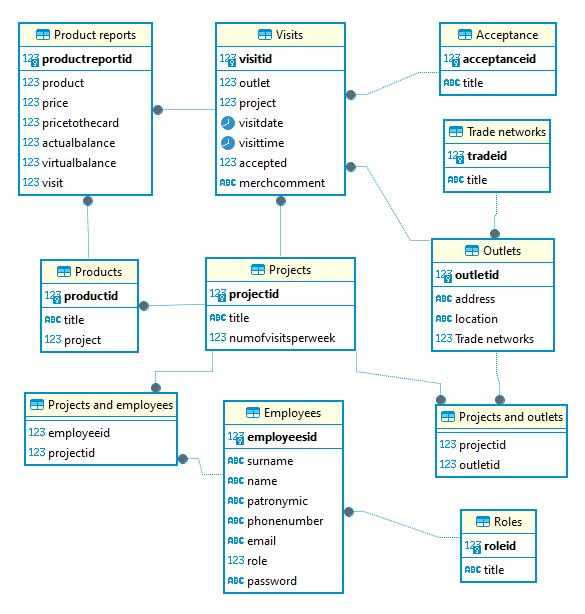


Рисунок 13 – Схема БД

* + 1. Подключение базы данных к проекту

Для того, чтобы подключить базу данных к проекту в первую очередь нужно установить два NuGet пакета: Npgsql.EntityFrameworkCore.PostgreSQL и Microsoft.EntityFrameworkCore.Tools. Подключенные пакеты представлены на рисунке 14.



Рисунок 14 – Пакеты, необходимы для работы с БД

Далее следует открыть командную строку и перейти в папку с проектом. После этого нужно подключится к БД с помощью команды, которая представлена на рисунке 15.

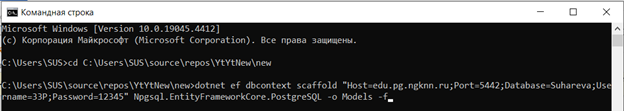


Рисунок 15 – Запрос на подключение

После этого в папке Models в проекте должны отобразиться классы, соответствующие таблицам базы данных.

* + 1. Вывод оператора в строке приветствия

В первую очередь для работы с базой данных нужно создать объект основного класса. В нашем случае, экземпляр будет создан в файле «MainWindowViewModel.cs». Это продемонстрировано на рисунке 16.

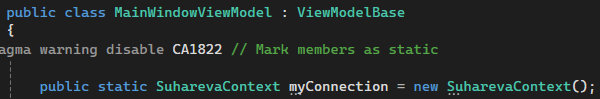


Рисунок 16 – Объект класса для работы с БД

Затем следует создать объект класса ViewModel, с которым мы будем работать, и свойство, благодаря которому мы сможем к нему обратиться в дальнейшем. В нашем случае это объект «personalAccountVM». Результат представлен на рисунке 17.

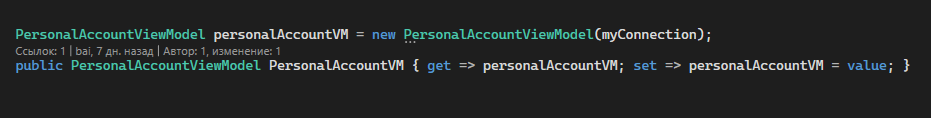


Рисунок 17 – Создание объекта класса ViewModel

Затем, в этом же классе, нам необходимо «запомнить» идентификатор текущего пользователя, чтобы затем вывести его личные данные в строку приветствия. Для этого создадим отдельную переменную, которой затем присвоим значение идентификатора в методе, что запускается при загрузке страницы персонального аккаунта, где и будет располагаться самая первая приветственная строка. После этого вызываем конструктор ранее созданного объекта (через его свойство), куда передаем текущее подключение к базе данных и идентификатор пользователя. Это можно увидеть на рисунке 18.

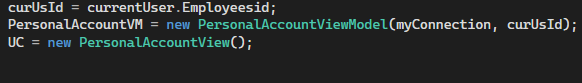


Рисунок 18 – Получение идентификатора текущего пользователя

Далее, перейдя в «PersonalAccountViewModel.cs», уже в том самом конструкторе присваиваем новое значение (необходимое нам) для фамилии текущего пользователя, после чего в файле «PersonalAccountView.axaml» для TextBlock строки приветствия устанавливаем свойство «Text = “{Binding …}”», где вызываем через свойство в основном классе измененную фамилию текущего пользователя. Отображено это на рисунке 19.



Рисунок 19 – Биндинг данных

* + 1. Вывод списка проектов и визитов

Несколько иным способом происходит формирование страниц со списком проектов и визитов.

Первым делом в главном классе программы создаются листы типа соответствующей модели (таблицы базы данных), где будут находиться все имеющиеся в ней данные. Это можно увидеть на рисунке 20.

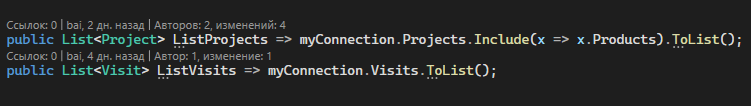


Рисунок 20 – Листы с данными

Затем данные из листов необходимо привязать к соответствующим визуальным элементам на странице. Так как заранее неизвестно количество записей для вывода, будет применяться метод вывода данных в шаблон, а конкретно конструкции «ListBox» и «DataTemplate», внутри которых располагаются обыкновенные визуальные элементы с тем же свойством «Text = “{Binding …}”». В этом случае в свойстве вызываются необходимые измерения (столбцы) листа (Navigation в случае, если измерение является вторичным ключом), а источником данных («ItemSource») для «ListBox»-а будут являться соответствующие листы с данными. Результат написания можно рассмотреть на рисунках 21-22.

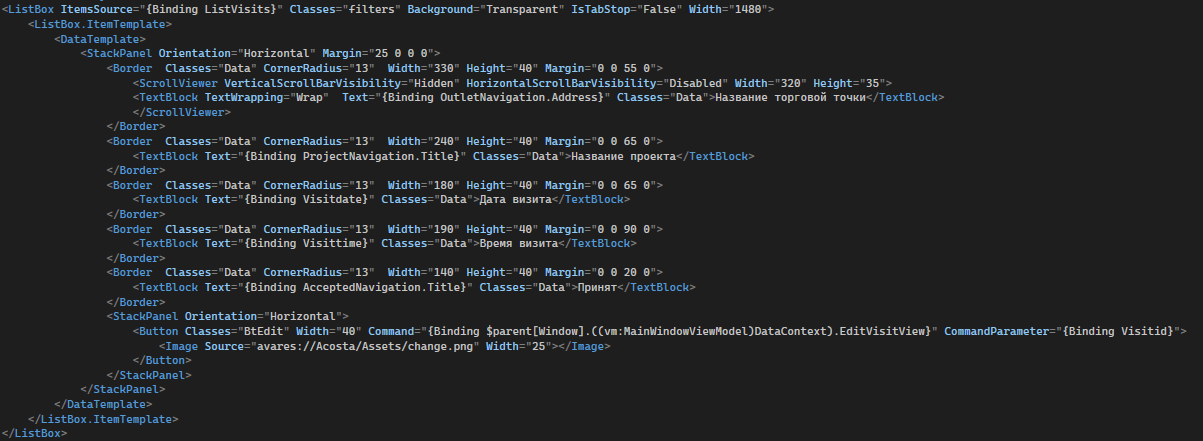


Рисунок 21 – Вывод списка визитов

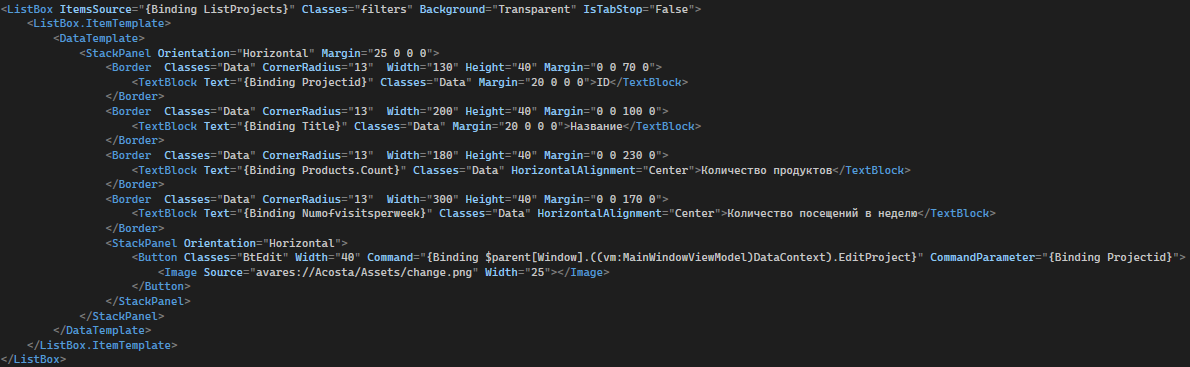


Рисунок 22 – Вывод списка проектов

* + 1. Создание проектов и визитов

Для организации создания проектов либо визитов (т.е. добавления их в базу данных), необходимо также первым делом создать соответствующий объект класса и свойство для обращения к нему, после чего создать все в том же главном классе метод, подтверждающий внесенные изменения (добавляющий новый объект в базу данных). Это возможно благодаря встроенному методу «SaveChanges». Отображение метода можно увидеть на рисунке 23.

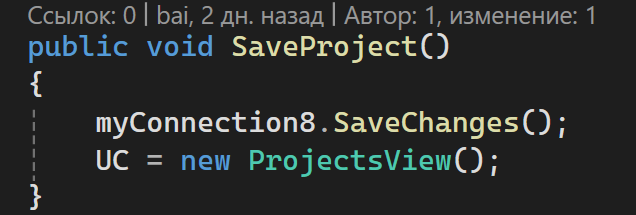


Рисунок 23 – Метод сохранения нового проекта

Сохранение визитов происходит идентичным методом.

Далее, необходимо внести некоторые инструкции в класс добавления проекта.

Так, необходимо создать объекты подключения к базе и соответствующей модели (таблицы), а также свойства для последнего для обращения к нему. Это необходимо, чтобы в дальнейшем записать данные в базу посредством создания конструктора класса, в котором будут находиться инструкции по созданию новой записи в соответствующей таблице, а также добавлению этой новой записи в соединение (соответственно, для визитов порядок действий будет аналогичным, помимо преобразований даты и времени, а также листов данных для списков, представленные на рисунке 25). Итоговый набор инструкций класса представлен на рисунке 24.

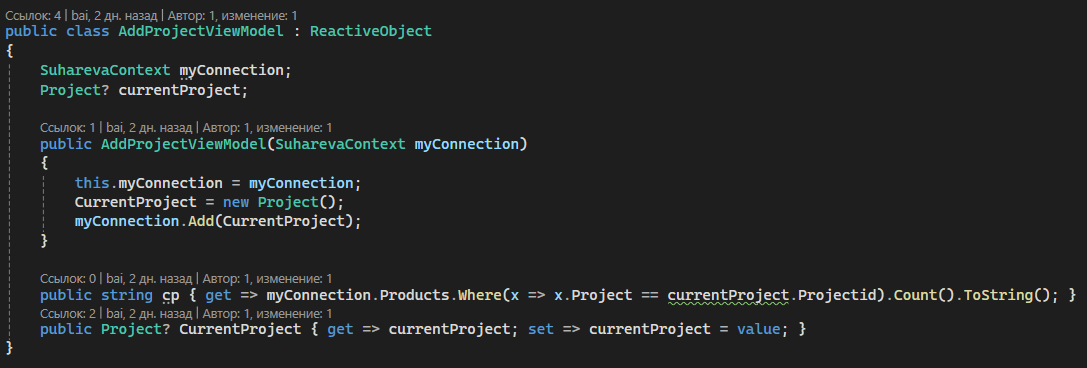


Рисунок 24 – Класс добавления нового проекта

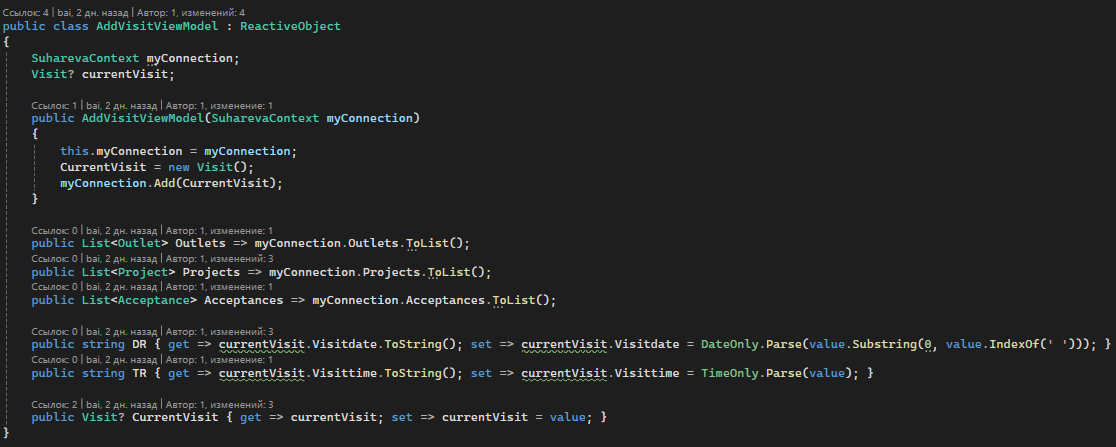


Рисунок 25 – Класс добавления нового визита

В последнюю очередь необходимо привязать созданный ранее метод, а также все поля новой записи в таблице к разметке. Для этого (для обыкновенных полей ввода) используется все то же свойство «Text = “{Binding …}”». Вызов же команды сохранения по кнопке представлен на рисунке 26.



Рисунок 26 – Кнопка сохранения проекта

В случае добавления нового визита порядок действий будет аналогичным, однако стоит упомянуть об исключительном элементе пользовательского управления, что появляется там – «ComboBox» с наполнением в виде данных из БД. Так, пример реализации одного из них представлен на рисунке 27.

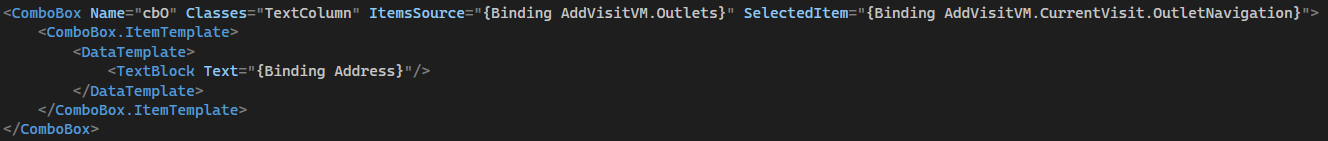


Рисунок 27 – Список торговых точек при создании визита

* + 1. Изменение пароля, проектов и визитов

При изменении какого-либо проекта или визита схема действий очень схожа с тем, что описано выше.

Во-первых, создаем нужные нам объекты классов и методы. Так, в этом случае это будут методы по изменению и сохранению изменений в конкретном проекте (визите). Смысл первого из них в том, чтобы запустить изменения определенной выбранной пользователем записи по ее идентификатору, второго же – закрепить изменения в БД. Так, оба метода можно увидеть на рисунке 28.

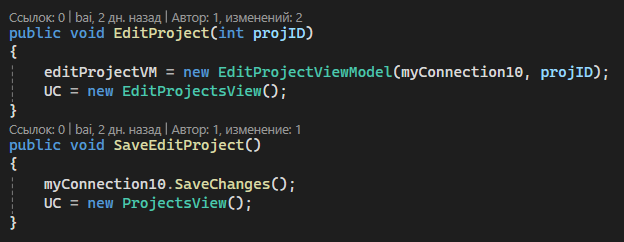


Рисунок 28 – Методы по изменению проекта

Во-вторых, в самом классе по изменению проекта (визита), соответственно, нет нужды создавать новую запись и добавлять ее в таблицу. Вместо это необходимо «вытянуть» из модели только нужный клиенту кортеж и присвоить его ячейкам новые значения. Так, полученный в конечном итоге класс для изменения проекта отображен на рисунке 29.

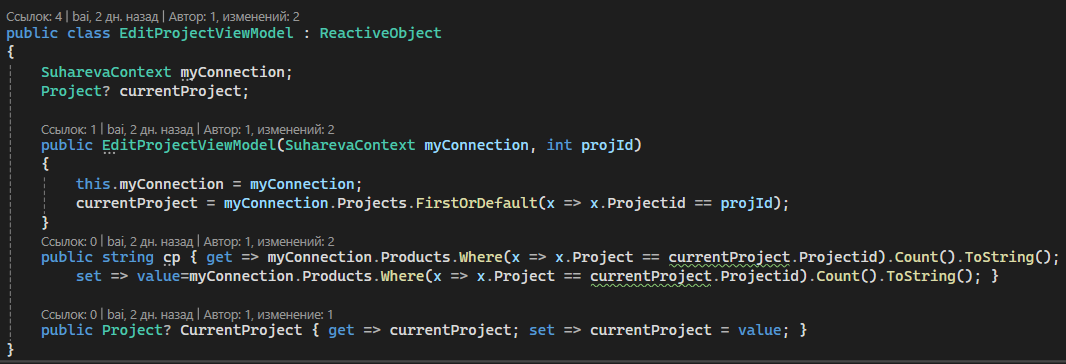


Рисунок 29 – Класс изменения проектов

В-третьих, есть некоторый нюанс при вызове описанных выше методов по кнопке. Так, увидеть, как это было реализовано, можно на рисунке 30.



Рисунок 30 – Кнопка сохранения изменений в проекте

С результатом же выполнения абсолютно всех модулей – готовым приложением, можно ознакомиться по ссылке[[1]](#footnote-1).

ЗАКЛЮЧЕНИЕ

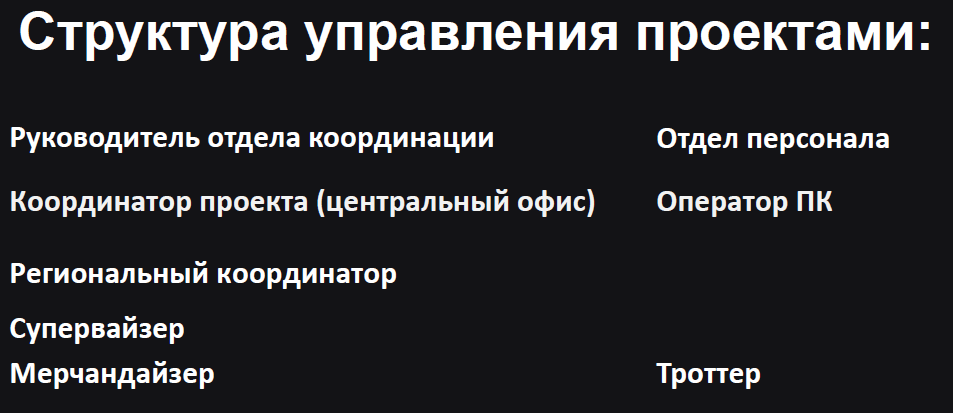
В результате прохождения производственной практики все поставленные цели и задачи были достигнуты и выполнены на высоком уровне. В ходе практики была успешно освоена интеграция различных программных модулей с целью создания единого, функционального и эффективного приложения, что свидетельствует о значительном росте навыков и компетенций в области информационных технологий. Каждый этап практики был отмечен заметным улучшением навыков работы с различными программами, что позволило развить системное мышление и способность к созданию сложных систем. Были успешно преодолены сложности, возникшие в процессе интеграции модулей, а также обеспечено надежное взаимодействие между ними. Благодаря прохождению учебной практики был значительно расширен профессиональный арсенал, приобретены ценные навыки и компетенции, необходимые для успешной карьеры в сфере информационных технологий. Была приобретена уверенность в своих силах и готовность к решению сложных задач, что будет способствовать успешному профессиональному развитию в будущем.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ И ИНФОРМАЦИОННЫХ РЕСУРСОВ

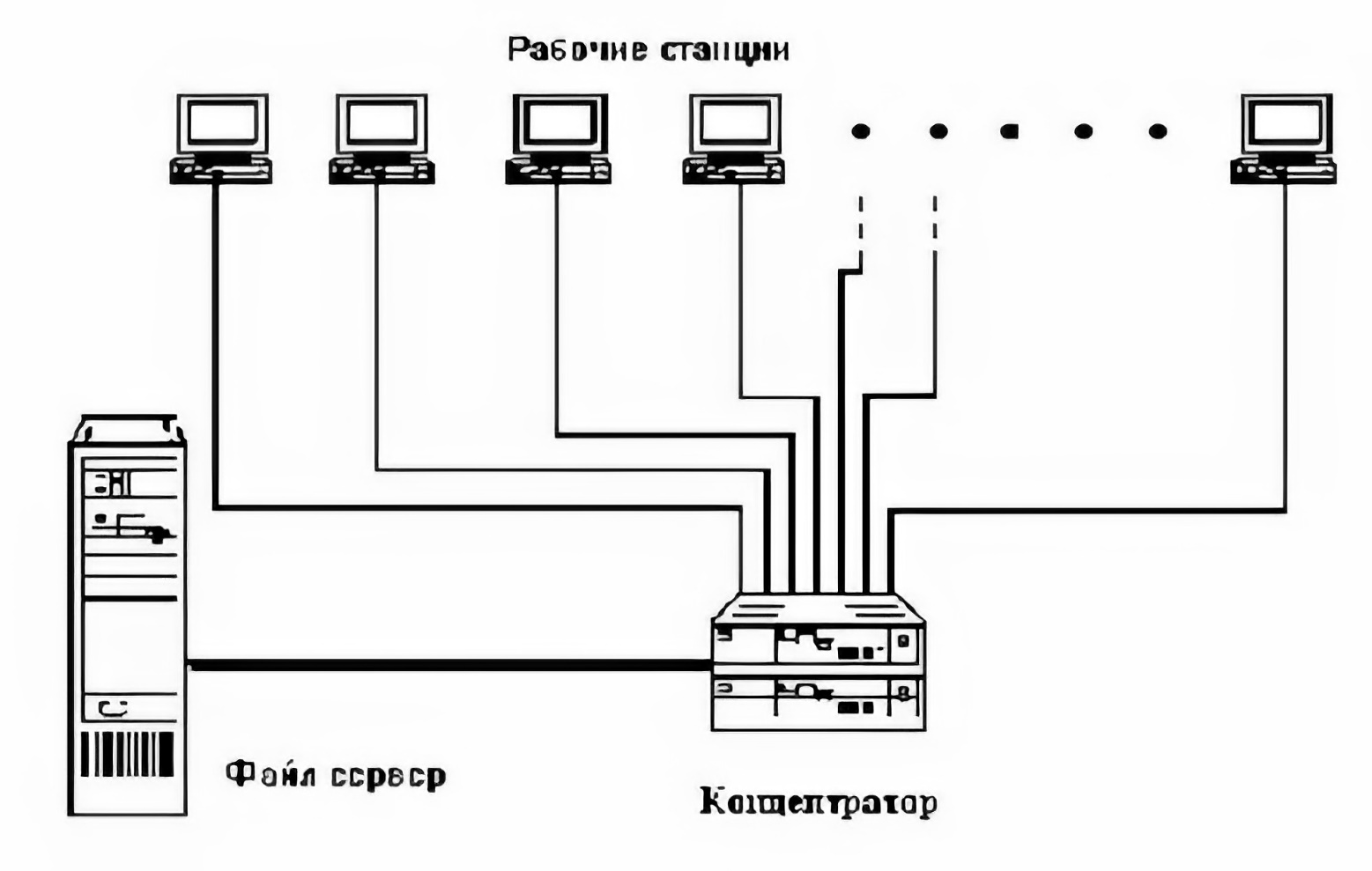
1. Галиаскаров, Э. Г.  Анализ и проектирование систем с использованием UML: учебное пособие для вузов / Э. Г. Галиаскаров, А. С. Воробьев. — Москва: Издательство Юрайт, 2024. — 125 с. — (Высшее образование). — ISBN 978-5-534-14903-6. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/544559> (дата обращения: 02.07.2024).
2. Черткова, Е. А.  Программная инженерия. Визуальное моделирование программных систем: учебник для среднего профессионального образования / Е. А. Черткова. — 3-е изд., испр. и доп. — Москва: Издательство Юрайт, 2024. — 146 с. — (Профессиональное образование). — ISBN 978-5-534-18094-7. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/539955> (дата обращения: 02.07.2024).
3. Советов, Б. Я.  Базы данных: учебник для вузов / Б. Я. Советов, В. В. Цехановский, В. Д. Чертовской. — 4-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2024. — 403 с. — (Высшее образование). — ISBN 978-5-534-18479-2. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/535113> (дата обращения: 02.07.2024).
4. Зализняк, В. Е.  Введение в математическое моделирование: учебное пособие для вузов / В. Е. Зализняк, О. А. Золотов. — Москва: Издательство Юрайт, 2024. — 133 с. — (Высшее образование). — ISBN 978-5-534-12249-7. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/542735> (дата обращения: 02.07.2024).
5. Дубина, И. Н.  Основы математического моделирования социально-экономических процессов: учебник и практикум для вузов / И. Н. Дубина. — Москва: Издательство Юрайт, 2024. — 349 с. — (Высшее образование). — ISBN 978-5-534-00501-1. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/536868> (дата обращения: 02.07.2024).
6. Рейзлин, В. И.  Математическое моделирование: учебное пособие для вузов / В. И. Рейзлин. — 2-е изд., перераб. и доп. — Москва: Издательство Юрайт, 2024. — 126 с. — (Высшее образование). — ISBN 978-5-534-08475-7. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: <https://urait.ru/bcode/537305> (дата обращения: 02.07.2024).
7. Маркин, А. В.  Программирование на SQL: учебное пособие для среднего профессионального образования / А. В. Маркин. — Москва: Издательство Юрайт, 2023. — 435 с. (дата обращения 02.07.2024).
8. Официальная страница документации СУБД «PostgreSQL» - URL: <https://www.postgresql.org/docs/> (дата обращения 02.07.2024).
9. Среда для разработки Microsoft Visual Studio – URL: <https://visualstudio.microsoft.com/ru/> (дата обращения 02.07.2024);
10. Документация по системе контроля версий Git – URL: <https://git-scm.com/doc> (дата обращения 02.07.2024);
11. Документация по языку программирования C# - URL: <https://learn.microsoft.com/ru-ru/dotnet/csharp/> (дата обращения 02.07.2024);
12. Build with Avalonia // Avalonia URL: https://docs.avaloniaui.net/ (дата обращения: 02.07.2024);
13. Avalonia UI // Avalonia URL: https://github.com/AvaloniaUI/Avalonia (дата обращения: 02.07.2024).

ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ А

Рисунок А1 – Организационная структура предприятия

ПРИЛОЖЕНИЕ Б

Рисунок Б1 –Схема звездообразной топологии

(на рисунке представлены рабочие станции, файл сервер и сетевой концентратор)

ПРИЛОЖЕНИЕ В



Рисунок В1 – Цветовая схема приложения



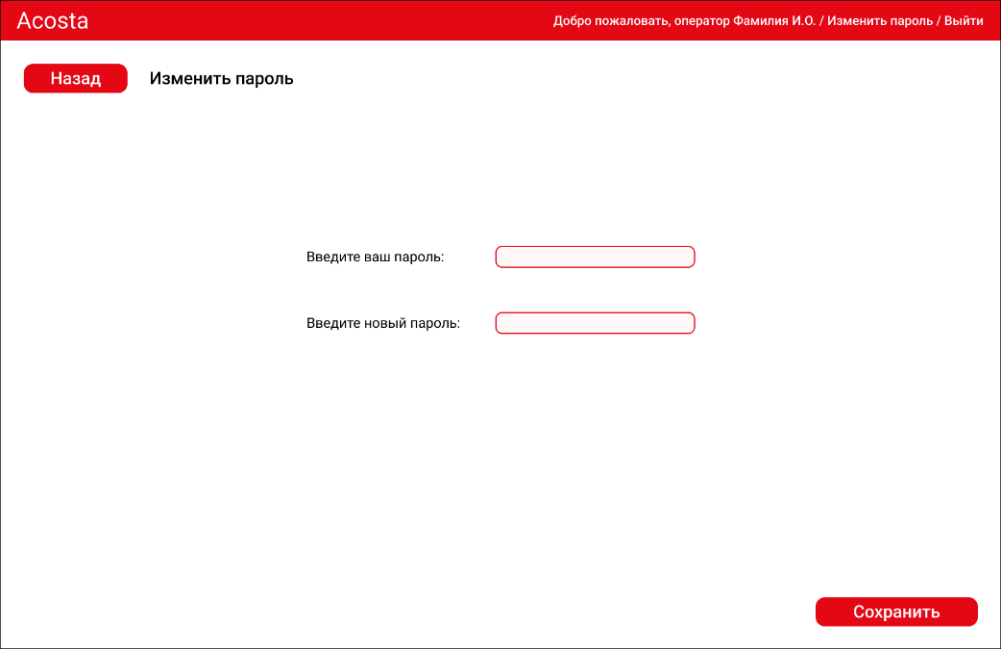
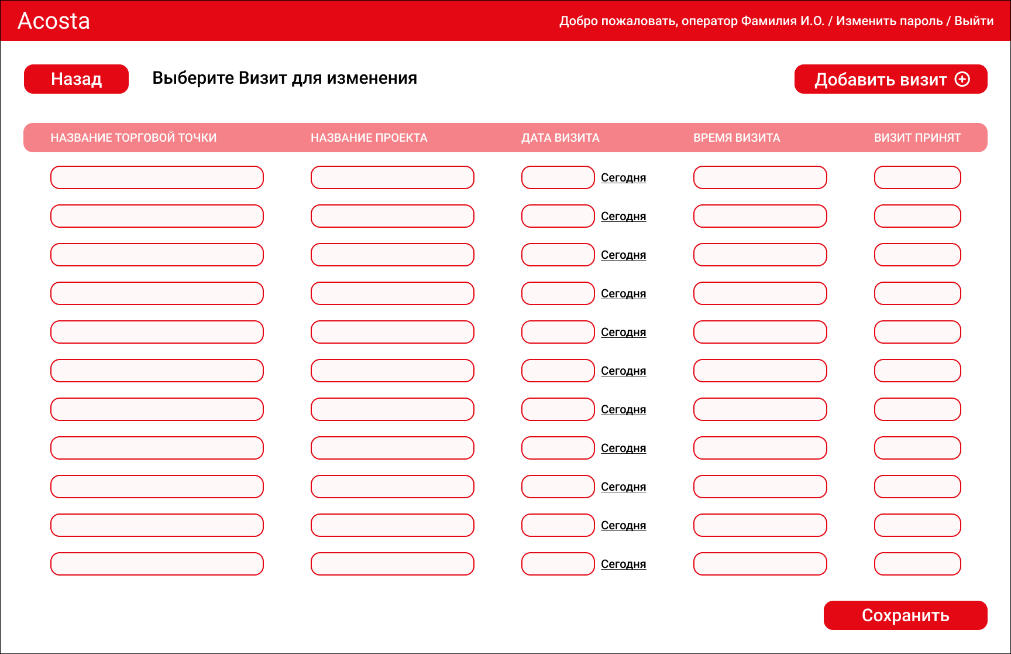
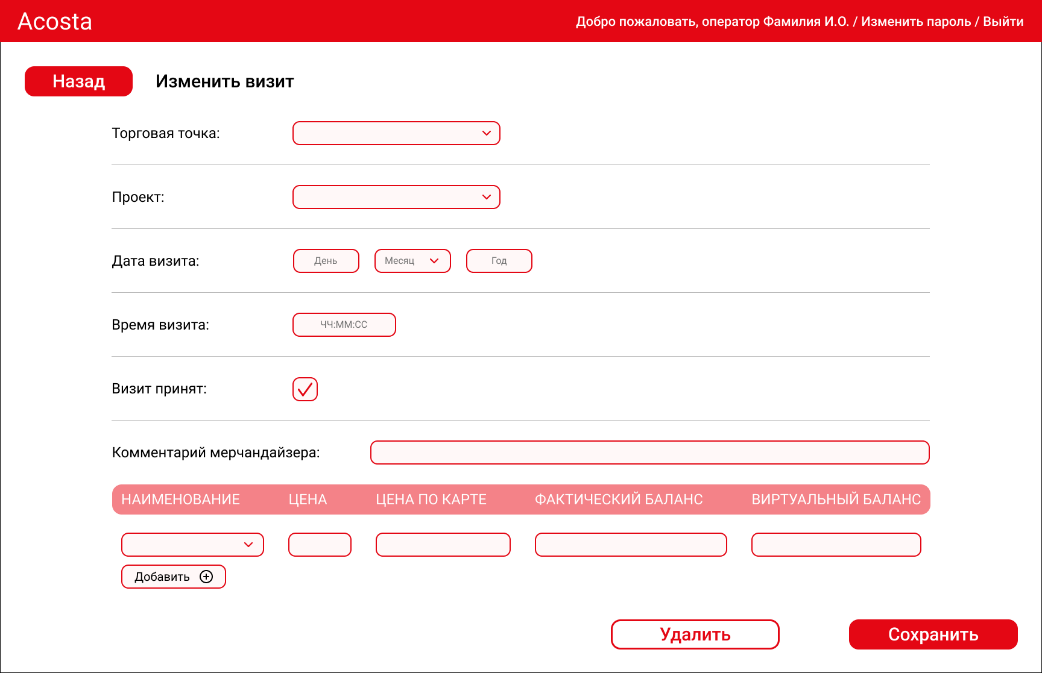
Рисунок В2 – Форма авторизации пользователя

Рисунок В3 – Форма редактирования профиля пользователя

Рисунок В4 – Страница со списком всех визитов компании

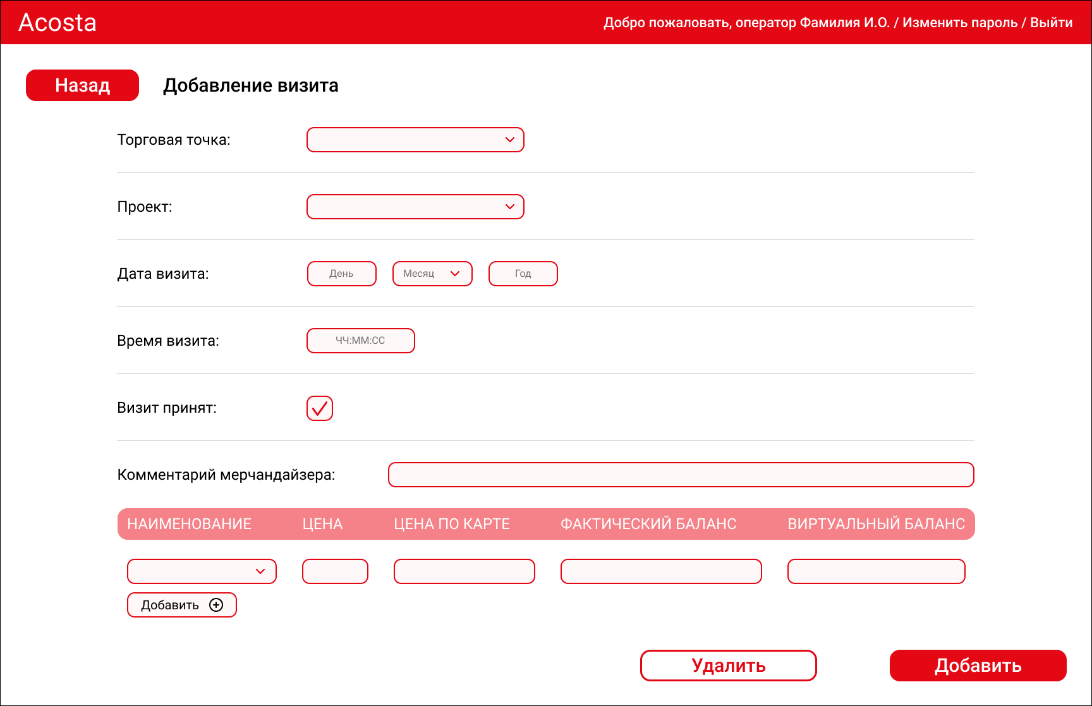
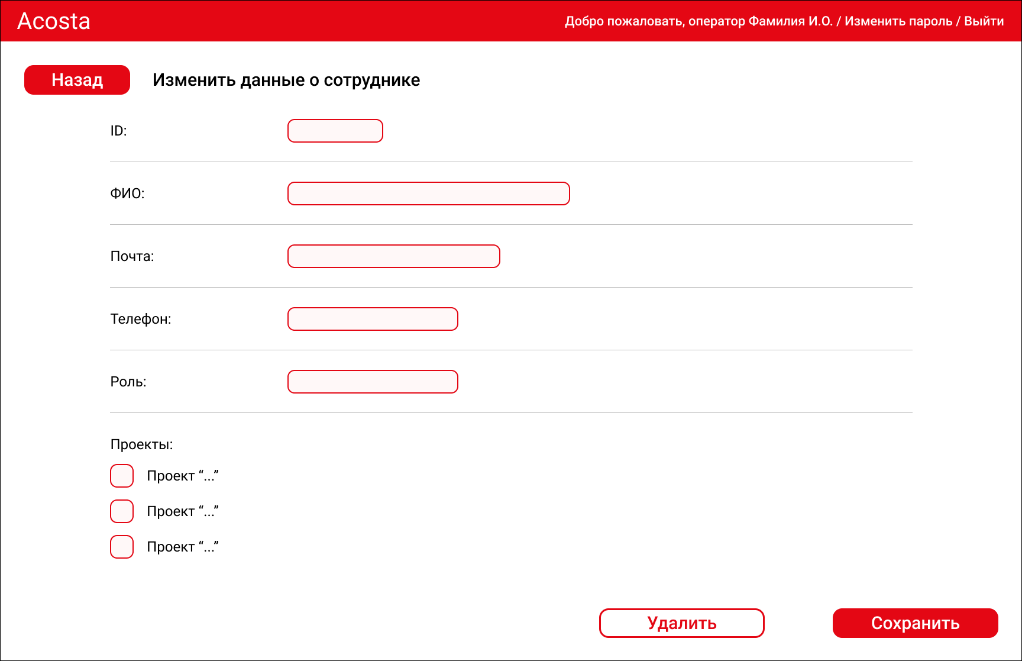
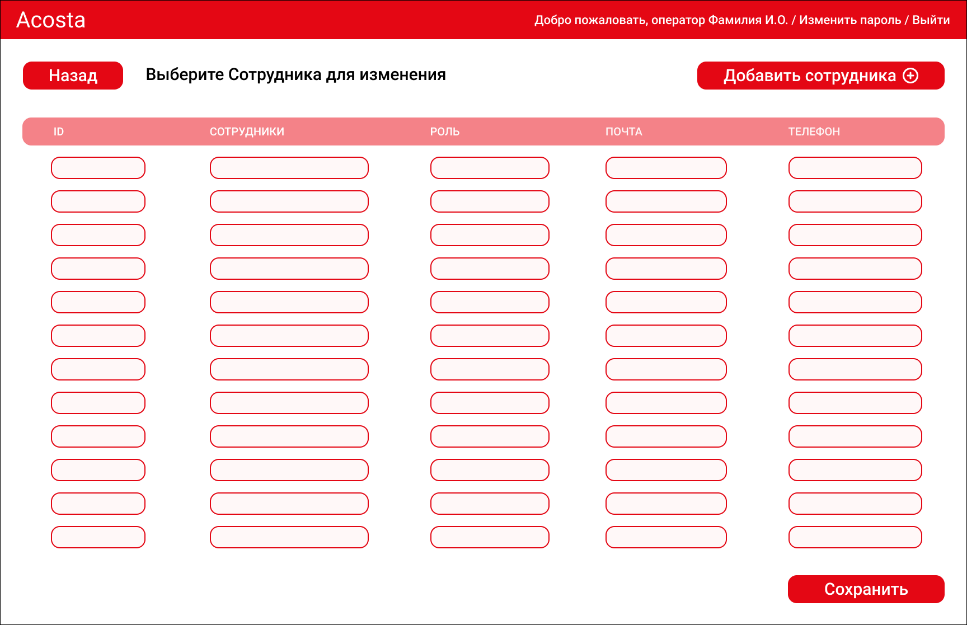
Рисунок В5 – Форма редактирования визита

Рисунок В6 – Форма добавления визита

Рисунок В7 – Страница со списком всех сотрудников компании

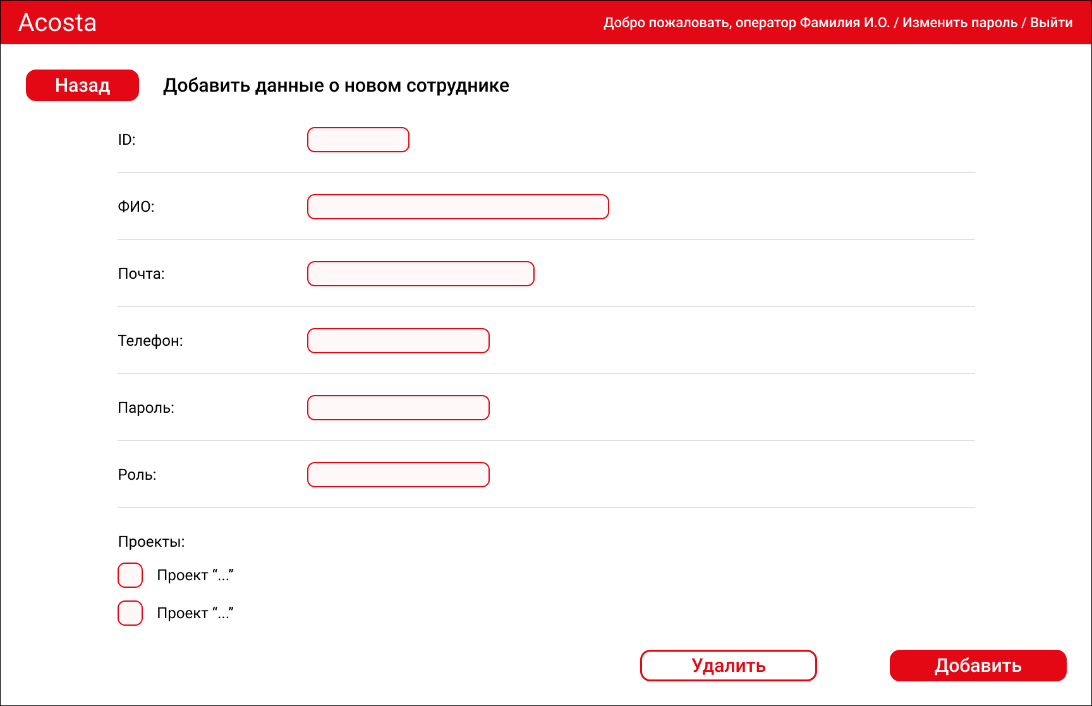
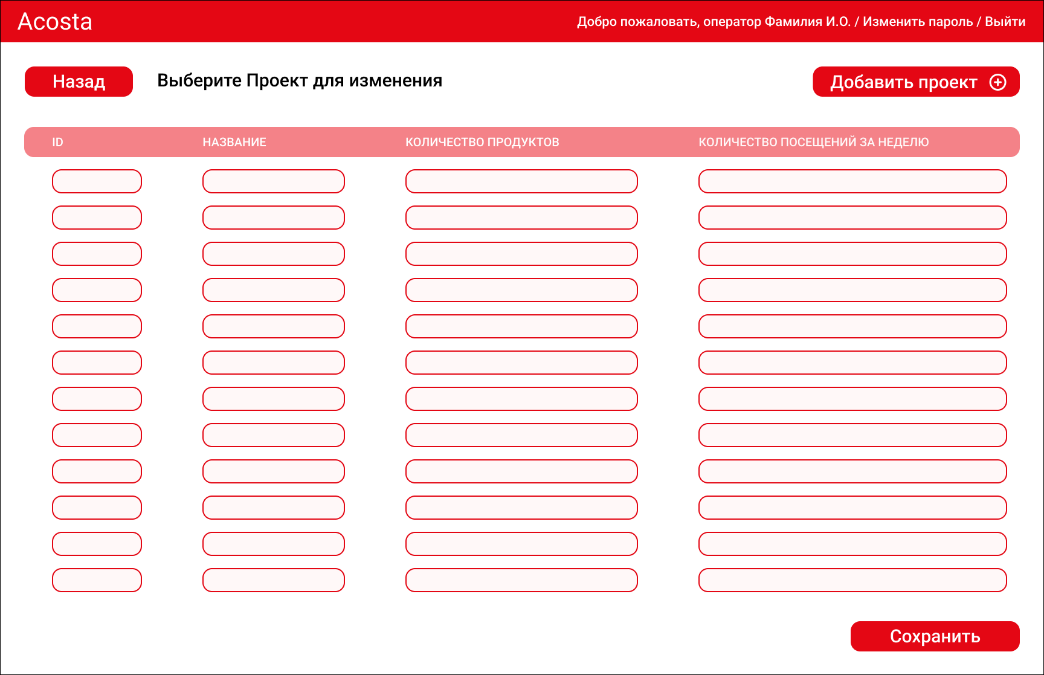
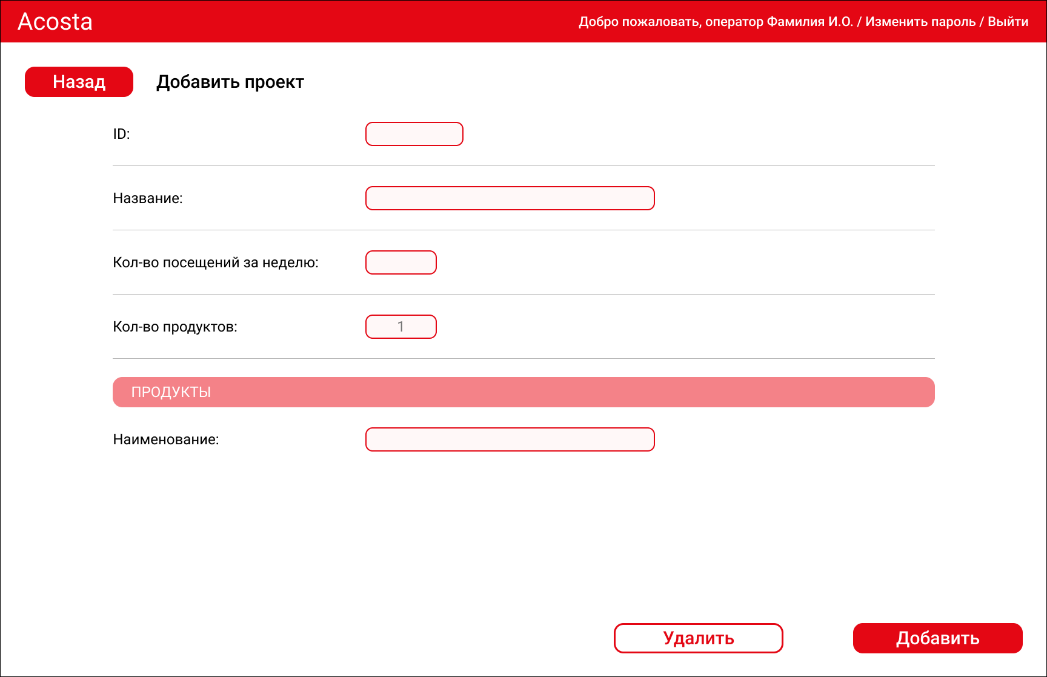
Рисунок В8 – Форма редактирования сотрудника

Рисунок В9 – Форма регистрации сотрудника

Рисунок В10 – Страница со списком всех проектов компании

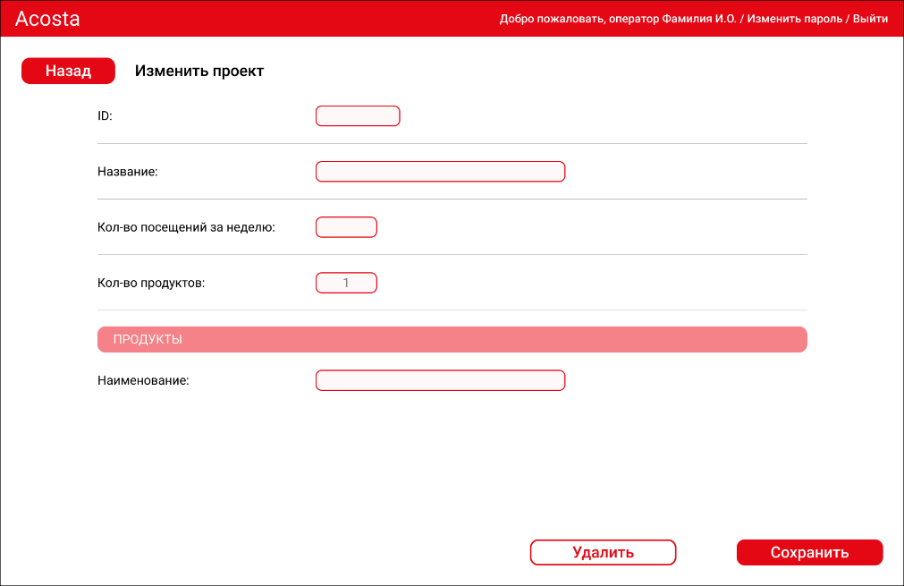
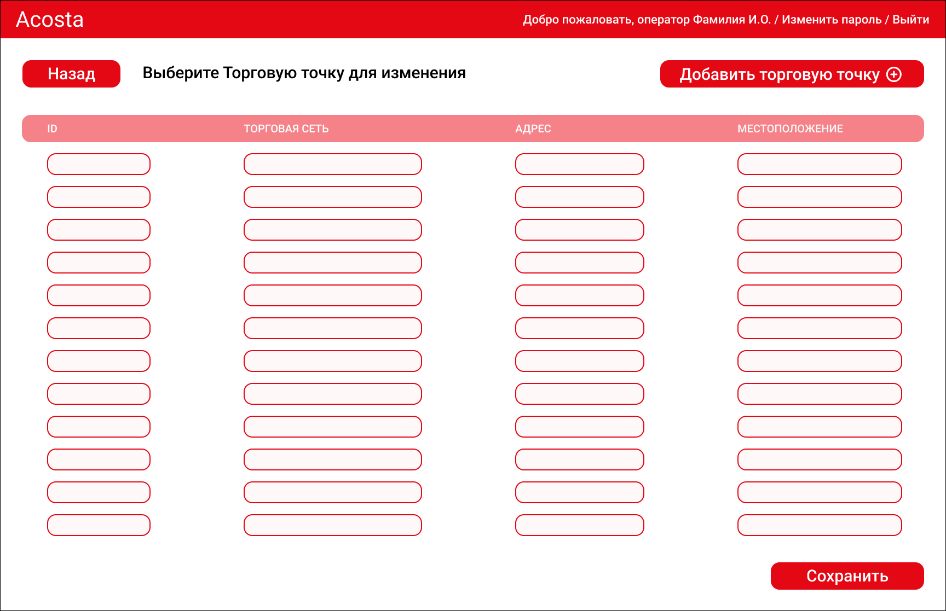
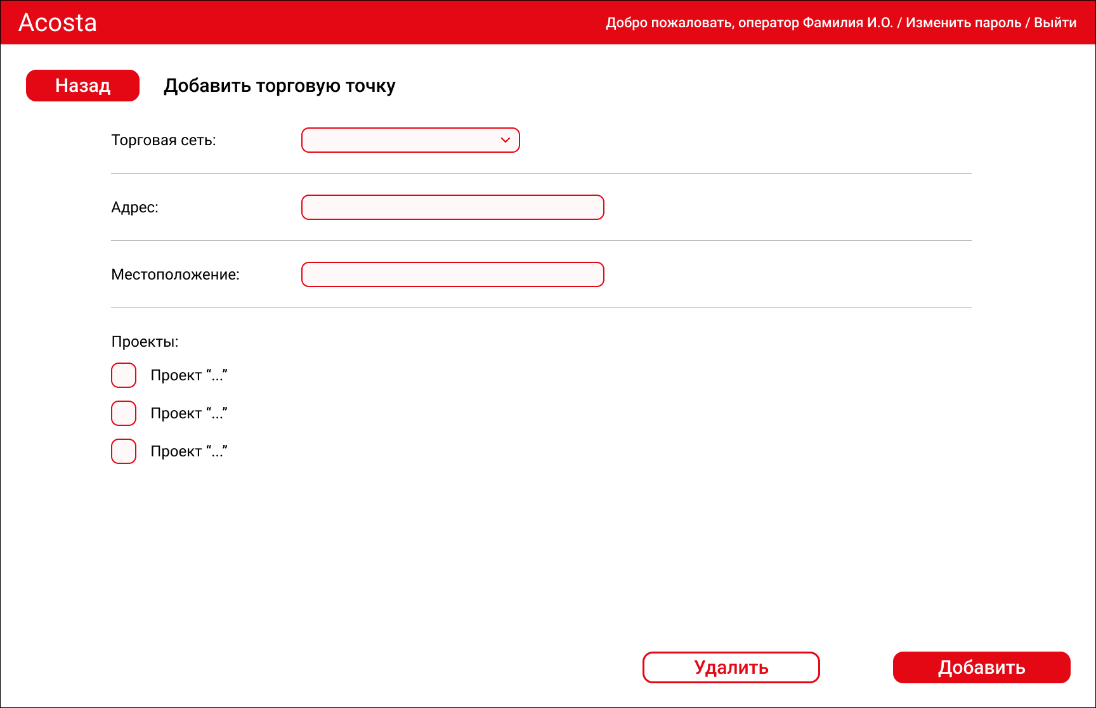
Рисунок В11 – Форма добавления проекта

Рисунок В12 – Форма редактирования проекта

Рисунок В13 – Страница со списком всех торговых точек компании

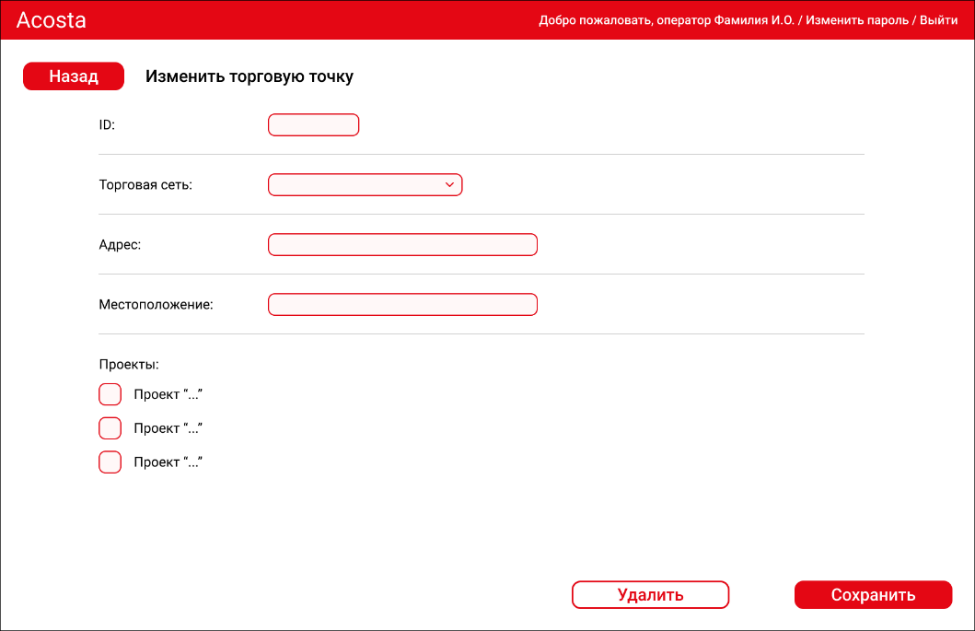
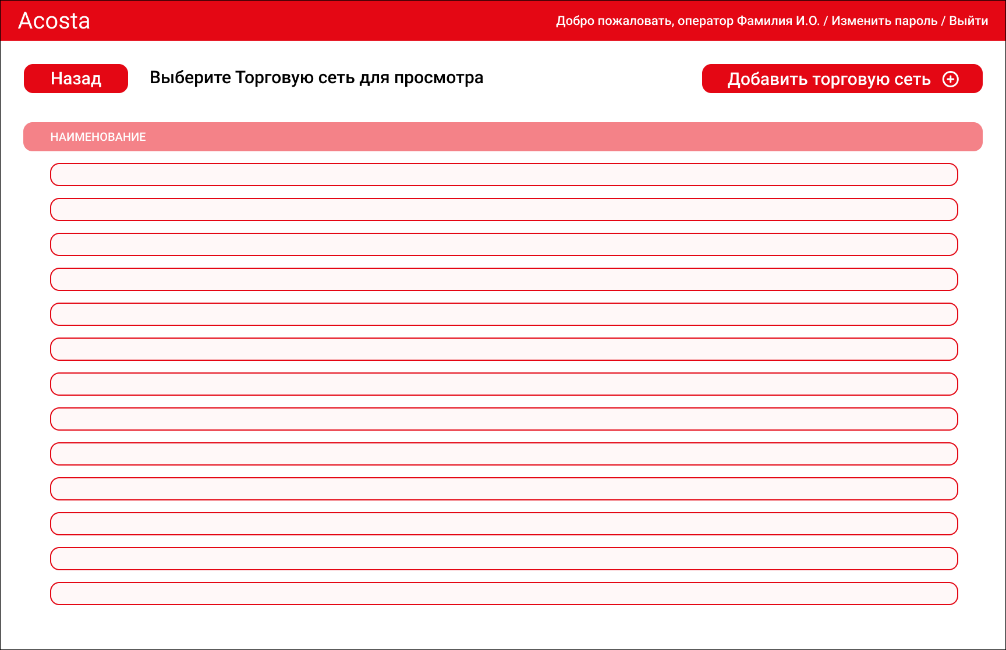
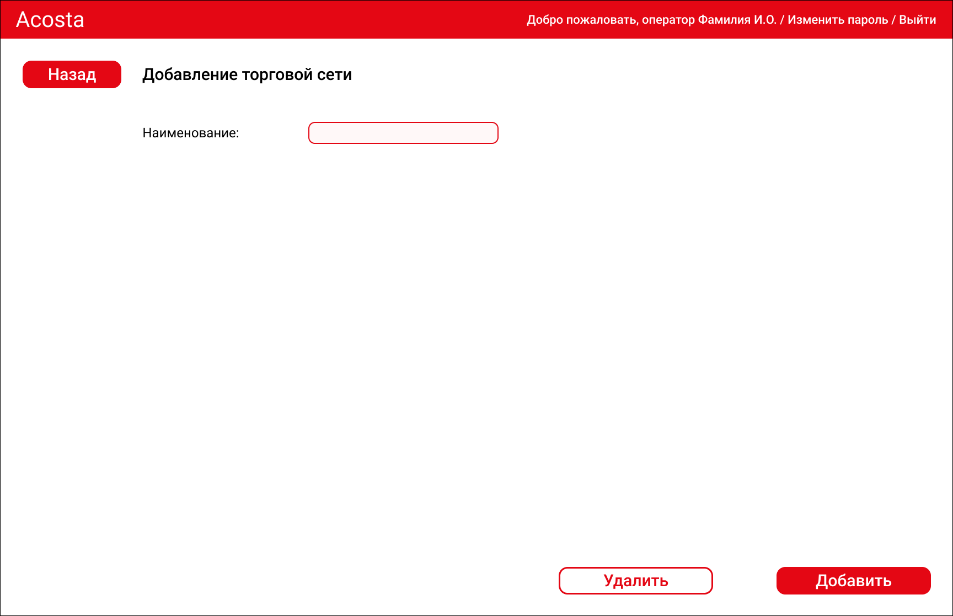
Рисунок В14 – Форма добавления торговой точки

Рисунок В15 – Форма редактирования торговой точки

Рисунок В16 – Страница со списком всех торговых сетей компании

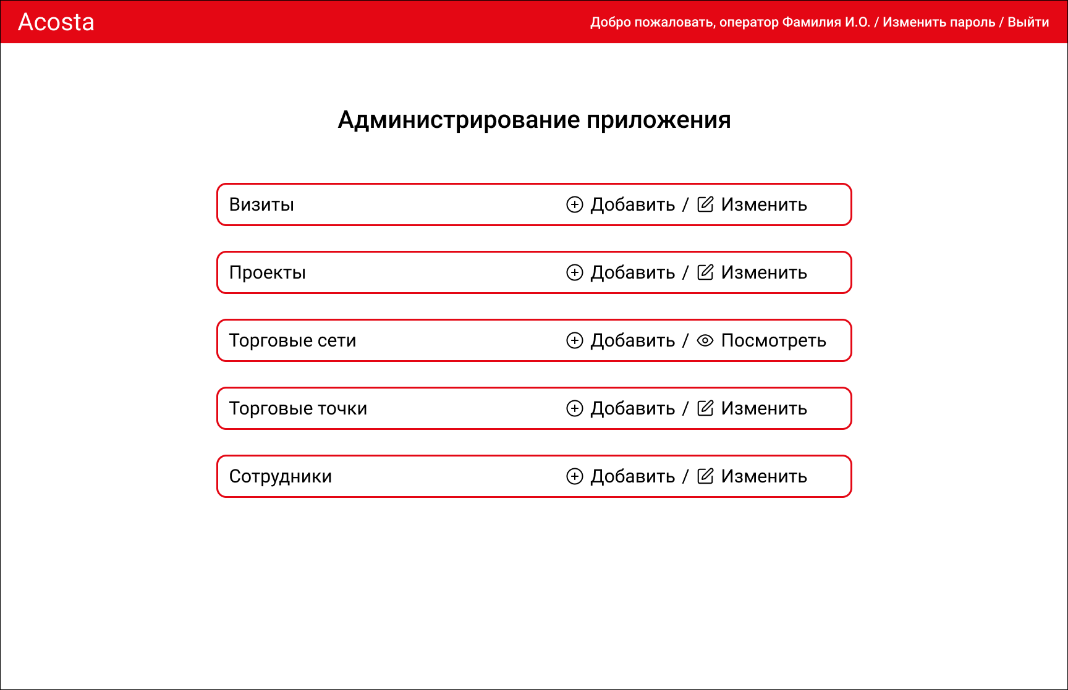
Рисунок В17 – Форма добавления торговой сети

Рисунок В18 – Страница личного кабинета, с переходом на другие разделы приложения

ПРИЛОЖЕНИЕ Г

Таблица Г1 – Код создания таблиц

|  |
| --- |
| **create** **table** **"Roles"**(  RoleID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Title **varchar**(30) **not** **null**  );  **create** **table** **"Trade networks"**(  TradeID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Title **varchar**(30) **not** **null**  );  **create** **table** **"Acceptance"**(  AcceptanceID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Title **varchar**(30) **not** **null**  );  **create** **table** **"Employees"**(  EmployeesID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Surname **varchar**(30) **not** **null**,  **"name"** **varchar**(30) **not** **null**,  Patronymic **varchar**(30) **not** **null**,  PhoneNumber **varchar**(30) **not** **null**,  Email **varchar**(30) **not** **null**,  **"role"** **int** **not** **null**,  **"password"** **varchar**(30) **not** **null**  );  **create** **table** **"Outlets"**(  OutletID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Address **varchar**(100) **not** **null**,  **"location"** **varchar**(100) **not** **null**,  **"Trade networks"** **int** **not** **null**  );  **create** **table** **"Projects"**(  ProjectID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Title **varchar**(30) **not** **null**,  NumOfVisitsPerWeek **int** **not** **null**  );  **create** **table** **"Products"**(  ProductID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Title **varchar**(30) **not** **null**,  Project **int** **not** **null**  );  **create** **table** **"Visits"**(  VisitID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**,  Outlet **int** **not** **null**,  Project **int** **not** **null**,  VisitDate **date** **not** **null**,  VisitTime **time** **not** **null**,  Accepted **int** **not** **null**,  MerchComment **varchar**(200) **not** **null**  );  **create** **table** **"Product reports"**(  ProductReportID **int** **not** **null** **generated** **always** **as** **identity** **primary** **key**, |

Продолжение таблицы Г1

|  |
| --- |
| Product **int** **not** **null**,  Price **int** **not** **null**,  PriceToTheCard **int** **not** **null**,  ActualBalance **int** **not** **null**,  VirtualBalance **int** **not** **null**,  Visit **int** **not** **null**  );  **create** **table** **"Projects and outlets"**(  ProjectID **int** **not** **null**,  OutletID **int** **not** **null**  );  **create** **table** **"Projects and employees"**(  EmployeeID **int** **not** **null**,  OutletID **int** **not** **null**  );  **alter** **table** **"Employees"**  **add** **foreign** **key** (**Role**) **references** **"Roles"**(roleiD)  **alter** **table** **"Outlets"**  **add** **foreign** **key** (**"Trade networks"**) **references** **"Trade networks"**(tradeiD)  **alter** **table** **"Visits"**  **add** **foreign** **key** (accepted) **references** **"Acceptance"**(acceptanceid)  **alter** **table** **"Products"**  **add** **foreign** **key** (project) **references** **"Projects"**(projectid)  **alter** **table** **"Product reports"**  **add** **foreign** **key** (product) **references** **"Products"**(productid)  **alter** **table** **"Product reports"**  **add** **foreign** **key** (visit) **references** **"Visits"**(visitid)  **alter** **table** **"Projects and outlets"**  **add** **foreign** **key** (projectid) **references** **"Projects"**(projectid)  **alter** **table** **"Projects and outlets"**  **add** **foreign** **key** (outletid) **references** **"Outlets"**(outletid)  **alter** **table** **"Projects and employees"**  **rename** **column** outletid **to** projectid  **alter** **table** **"Projects and employees"**  **add** **foreign** **key** (projectid) **references** **"Projects"**(projectid)  **alter** **table** **"Projects and employees"**  **add** **foreign** **key** (employeeid) **references** **"Employees"**(employeesid)  **alter** **table** **"Visits"**  **add** **foreign** **key** (project) **references** **"Projects"**(projectid)  **alter** **table** **"Visits"**  **add** **foreign** **key** (outlet) **references** **"Outlets"**(outletid) |

Таблица Г2 – Код заполнения таблиц

|  |
| --- |
| **insert** **into** "Roles" (title) **values**  (**'Оператор'**),  (**'Агент'**),  (**'Супервайзер'**),  (**'Мерчендайзер'**);  **insert** **into** "Acceptance" (title) **values**  (**'Принят'**),  (**'Не принят'**);  **insert** **into** "Trade networks" (title) **values**  (**'Пятерочка'**),  (**'Магнит'**),  (**'SPAR'**),  (**'ВкусВилл'**),  (**'Перекресток'**),  (**'Дикси'**),  (**'Ашан'**),  (**'Красное&Белое'**),  (**'МЕТRО'**),  (**'ОКЕЙ'**);  **insert** **into** "Employees" (surname, "name", patronymic, phonenumber, email, "role", "password") **values**  (**'Сухарева'**, **'Анжелика'**, **'Кирилловна'**, **'89200213577'**, **'asuhareva356@gmail.com'**, 1, **'123'**),  (**'Трянина'**, **'Марина'**, **'Александровна'**, **'89302903399'**, **'ardaismine@gmail.com'**, 1, **'456'**),  (**'Гончарова'**, **'Надежда'**, **'Евгеньевна'**, **'89625062815'**, **'bts152@gmail.com'**, 2, **'1715'**),  (**'Шохирева'**, **'Екатерина'**, **'Николаевна'**, **'89960023356'**, **'eshohireva2005@gmail.com'**, 3, **'1806'**),  (**'Веселова'**, **'Анна'**, **'Федоровна'**, **'89308172572'**, **'imnotginger@gmail.com'**, 4, **'2024'**);  **insert** **into** "Projects" (title , numofvisitsperweek) **values**  (**'RITTER SPORT'**, 2),  (**'Активиа'**, 3),  (**'Балахнинский хлеб'**, 7),  (**'Арзамасский хлеб'**, 7),  (**'Останкино'**, 4);  **insert** **into** "Products" (title , project) **values**  (**'RITTER SPORT ВАФЛЯ И КАКАО МУСС'**, 1),  (**'Актибио Черника Злаки Лен'**, 2),  (**'Балахнинские пончики'**, 3),  (**'Рулет Ветчина Сыр'**, 4),  (**'Шашлык свиной охлажденный'**, 5);  **insert** **into** "Outlets" (address, "location", "Trade networks") **values**  (**'ул. Горького, 34А, Балахна'**, **'56.520454, 43.548214'**, 3),  (**'ул. Спутника, 44, Нижний Новгород'**, **'56.237277, 43.827069'**, 1),  (**'проспект Ленина, 113, Нижний Новгород'**, **'56.249807, 43.876485'**, 10),  (**'улица 50 лет ВЛКСМ, 63, Ковернино'**, **'57.124760, 43.813711'**, 2),  (**'улица Максимова, 14, Бор'**, **'56.336912, 44.107388'**, 8)  **insert** **into** "Visits" (outlet, project, visitdate, visittime, accepted, merchcomment) **values**  (1, 3, **'2024-03-09'**, **'12:38:12'**, 1, **'Всё в норме.'**),  (4, 5, **'2024-05-27'**, **'17:35:07'**, 1, **'Всё в норме.'**), |

Продолжение таблицы Г2

|  |
| --- |
| (2, 1, **'2024-04-15'**, **'09:44:31'**, 2, **'Виртуальный остаток.'**),  (3, 2, **'2024-03-21'**, **'10:29:01'**, 2, **'Всё в норме.'**),  (5, 4, **'2024-06-10'**, **'21:53:24'**, 1, **'Виртуальный остаток.'**) |

1. http://gogs.ngknn.local:3000/Acosta/Frontend.git [↑](#footnote-ref-1)