**Министерство образования и науки РФ**

**Санкт-Петербургский политехнический университет Петра Великого**

**Институт компьютерных наук и технологий**

**Высшая школа программной инженерии**

**К У Р С О В А Я Р А Б О Т А**

**Графический интерфейс пользователя БД «Автосервис»**

по дисциплине «Системы управления базами данных»

|  |  |  |
| --- | --- | --- |
| Выполнил  студент гр. в3530904/70321 | <*подпись*> | С.А. Сафронов |
| Руководитель  ст. преподаватель | <*подпись*> | О.В. Александрова |

« » 2023г.

Санкт-Петербург

2023

Содержание

[Задание. 3](#_Toc152427759)

[Пользователи базы данных. 4](#_Toc152427760)

[Работа интерфейса 4](#_Toc152427761)

[Основные Функции и Элементы Интерфейса 4](#_Toc152427762)

[Роль в Общем Приложении 4](#_Toc152427763)

[Визуальное Оформление и Удобство Использования 5](#_Toc152427764)

[Основные Функциональные Особенности 5](#_Toc152427765)

[Техническая Реализация 6](#_Toc152427766)

[Пользовательский Интерфейс 6](#_Toc152427767)

[AddServiceView: Добавление Новой Услуги 6](#_Toc152427768)

[EditServiceView: Редактирование Существующей Услуги 7](#_Toc152427769)

[CarListView: Просмотр и Управление Списком Автомобилей 8](#_Toc152427770)

[AddCarView: Добавление Нового Автомобиля 9](#_Toc152427771)

[EditCarView: Редактирование Существующего Автомобиля 9](#_Toc152427772)

[MasterListView: Просмотр и Управление Списком Мастеров 10](#_Toc152427773)

[AddMasterView: Добавление Нового Мастера 11](#_Toc152427774)

[EditMasterView: Редактирование Существующего Мастера 11](#_Toc152427775)

[Стек Используемых Технологий 12](#_Toc152427776)

[**Вывод** 12](#_Toc152427777)

[Список используемой литературы 13](#_Toc152427778)

# Задание.

В рамках данного цикла лабораторных работ необходимо автоматизировать работу автосервиса. Для этого в рамках базы данных PostgreSQL необходимо создать объекты в схеме вашего пользователя (логин и пароль пользователя для доступа к базе данных студент должен получить у преподавателя) и написать клиентское приложение на базе компонентов ADO.NET или JDBC.

Автосервис специализируется на ремонте и техническом обслуживании автомобилей. При этом цена услуг для отечественных и зарубежных отличается. В сервисе есть ограниченный круг клиентов для которых производятся работы. Каждый автомобиль клиента характеризуется государственным номером, цветом и маркой. Периодически для автомобиля мастер выполняет работу, данные о которой заносятся в журнал работ. Одну услугу оказывает один мастер.

**База данных должна удовлетворять следующим требованиям:**

1. Контроль целостности данных, используя механизм связей
2. Операции модификации групп данных и данных в связанных таблицах должны быть выполнены в рамках транзакций.
3. Логика работы приложения должна контролироваться триггерами. В частности:
   * Не позволяет добавить автомобиль с уже существующим номером
   * Не позволяет добавить мастера, если их уже больше 10
   * Не позволяет дать работу мастеру, если он в этот день уже выполнил больше одной работы
4. Все операции вычисления различных показателей (из требований к клиентскому приложению) должны реализовываться хранимыми процедурами.

**Требования к клиентскому приложению:**

1. Необходимо реализовать интерфейсы для ввода, модификации и удаления справочников:
   * Мастеров;
   * Автомобилей;
   * Услуг.
2. В главном окне приложения должен быть реализован интерфейс, позволяющий оператору назначать работы из перечня услуг мастерам автосервиса для обслуживания имеющейся базы клиентов с возможностью указания даты проведения работы.
3. Необходимо реализовать возможность просмотра оператором следующих показателей:
   * Общая стоимость обслуживания отечественных и импортных автомобилей (с возможностью фильтрации по датам оказания услуги).
   * Пять мастеров, которые в заданном месяце выполнили наибольшее число работ для разных автомобилей.

# **Пользователи базы данных.**

Было создано два пользователя наделенными соответствующими правами доступа. Пароли пользователей были захешированы.

а) Пользователь admin. Выданы все права доступа. Имеет права на создание, изменение, удаление мастеров и автомобилей, а также доступ к статистике.

б) Пользователь user. Имеет права только на просмотр таблиц БД и создание работ.

# **Работа интерфейса**

Экран "Назначения работ" (NaznachenieRabotView) в приложении AutoServiceERP-Frontend представляет собой важный интерфейс, позволяющий оператору назначать работы мастерам автосервиса. Он включен в основную структуру MainTabView, которая управляет навигацией по основным функциям приложения через вкладки. Экран "Назначения работ" доступен через вкладку с иконкой календаря и меткой "Назначения".

### Основные Функции и Элементы Интерфейса

1. **Выбор Мастера, Автомобиля и Услуги:**
   * Экран предоставляет выпадающие списки (или аналогичные элементы управления), позволяющие выбрать мастера, автомобиль и услугу из заранее загруженных данных.
   * Данные о мастерах, автомобилях и услугах загружаются из соответствующих сетевых менеджеров (MasterNetworkManager, CarNetworkManager, ServiceNetworkManager).
2. **Выбор Даты:**
   * Пользователи могут выбрать дату выполнения работы с помощью компонента DatePicker, который обеспечивает удобный и интуитивно понятный интерфейс для выбора дат.
3. **Кнопка Назначения Работы:**
   * После выбора всех необходимых параметров, оператор может назначить работу, нажав кнопку "Назначить работу".
   * Нажатие этой кнопки инициирует процесс создания новой записи о работе, которая включает выбранные параметры мастера, автомобиля, услуги и дату.
4. **Отправка Данных на Сервер:**
   * Созданная запись работы отправляется на сервер с помощью WorkNetworkManager. Это обеспечивает интеграцию с серверной частью приложения и позволяет управлять назначениями работ в рамках общей информационной системы автосервиса.

### Роль в Общем Приложении

Экран "Назначения работ" является ключевым элементом в приложении AutoServiceERP-Frontend, позволяя операторам эффективно управлять рабочими процессами автосервиса. Он взаимодействует с другими компонентами приложения, такими как "Справочники" и "Статистика", обеспечивая централизованное и удобное управление всеми аспектами работы сервиса.

### Визуальное Оформление и Удобство Использования

Экран разработан с учетом принципов удобства использования и интуитивно понятного интерфейса, что обеспечивает легкость в навигации и эффективность при выполнении задач по назначению работ. Элементы управления и дизайн интерфейса соответствуют общей эстетике и стилю приложения AutoServiceERP-Frontend.

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

В рамках разработки приложения AutoServiceERP-Frontend я создал ServiceListView, экран, предназначенный для управления списком услуг автосервиса. Этот экран является частью модуля "Справочники" и предоставляет функционал для просмотра, добавления, редактирования и удаления услуг.

### Основные Функциональные Особенности

1. **Отображение Списка Услуг:**
   * List: Используется для отображения всех услуг, загруженных из ServiceNetworkManager. Каждый элемент списка представляет собой услугу с указанием её названия.
   * NavigationLink: Позволяет перейти к экрану редактирования (EditServiceView) для выбранной услуги.
2. **Добавление и Редактирование Услуг:**
   * Кнопка "Добавить" (Image(systemName: "plus")): Открывает модальное окно AddServiceView, позволяя добавить новую услугу.
   * EditServiceView: Предоставляет форму для редактирования существующей услуги. Изменения сохраняются в базе данных посредством ServiceNetworkManager.
3. **Удаление Услуг:**
   * Свайп влево по элементу списка активирует функцию удаления. При подтверждении действия выбранная услуга удаляется из списка и из базы данных через ServiceNetworkManager.
4. **Обновление Списка Услуг:**
   * Кнопка "Обновить": Позволяет обновить список услуг, загружая актуальные данные из ServiceNetworkManager.

### Техническая Реализация

* **Загрузка Данных:** Использование ServiceNetworkManager для асинхронной загрузки данных об услугах. При успешном получении данных список services обновляется.
* **Менеджмент Состояний:** Использование @State для управления состоянием списка услуг и отслеживания открытия модальных окон.
* **Обработка Пользовательских Действий:** Реализация логики добавления, редактирования и удаления услуг, включая обработку ошибок и подтверждение действий пользователя.

### Пользовательский Интерфейс

Пользовательский интерфейс ServiceListView организован таким образом, чтобы обеспечить простоту и интуитивную понятность в использовании. Визуальный дизайн и расположение элементов управления соответствуют общему стилю приложения, обеспечивая целостность пользовательского опыта.

Этот экран является ключевым для управления услугами в рамках приложения AutoServiceERP-Frontend, позволяя быстро и эффективно управлять аспектами, связанными с услугами автосервиса.

В рамках разработки приложения AutoServiceERP-Frontend я создал два важных экрана для управления услугами автосервиса: AddServiceView и EditServiceView. Эти экраны обеспечивают функциональность для добавления новых услуг и редактирования существующих, что является ключевым аспектом управления операциями в автосервисе.

### AddServiceView: Добавление Новой Услуги

**Основные Функции:**

* Предоставляет форму для ввода названия услуги, стоимости для отечественных и импортных автомобилей.
* Поля для стоимости оснащены клавиатурой для ввода чисел, упрощая ввод цен.
* Кнопка "Добавить" отправляет введенные данные на сервер для создания новой услуги.

**Процесс Работы:**

* Пользователь вводит название услуги и её стоимость в соответствующие поля.
* При нажатии на кнопку "Добавить", данные проверяются на корректность, преобразуются в объект Service и отправляются в ServiceNetworkManager.
* После успешного добавления услуги, она отображается в списке услуг, и экран закрывается.

**Техническая Реализация:**

* Использует @Binding для связывания с массивом услуг в родительском представлении, что позволяет обновить список услуг после добавления новой.
* Обработка ошибок предусмотрена для сценариев, когда стоимость услуги введена некорректно или при возникновении ошибок при добавлении услуги на сервер.

### EditServiceView: Редактирование Существующей Услуги

**Основные Функции:**

* Предоставляет форму для редактирования выбранной услуги, включая название и стоимость.
* Аналогично AddServiceView, предусмотрены поля для ввода стоимости и названия услуги.

**Процесс Работы:**

* Загружает текущие данные выбранной услуги в поля формы.
* Пользователь может изменить любую информацию о услуге.
* После нажатия на кнопку "Обновить", обновленные данные отправляются на сервер.

**Техническая Реализация:**

* Использует @State для локального управления состоянием редактируемой услуги.
* Функция updateService в ServiceNetworkManager используется для обновления данных услуги на сервере.
* После успешного обновления вызывается onServiceUpdated, что позволяет обновить список услуг в родительском представлении.

Оба эти экрана являются частью модуля "Услуги" в приложении AutoServiceERP-Frontend и вносят значительный вклад в удобство управления услугами автосервиса, предоставляя пользователю интуитивно понятные и эффективные инструменты для работы с данными услуг.

A screen shot of a phone

Description automatically generatedA screen shot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

В рамках проекта AutoServiceERP-Frontend мной были разработаны три ключевых экрана, относящихся к управлению автомобилями: CarListView, AddCarView и EditCarView. Эти экраны играют важную роль в обеспечении удобства управления информацией об автомобилях в автосервисе.

### CarListView: Просмотр и Управление Списком Автомобилей

**Основные Функции:**

* Предоставляет список всех автомобилей, загруженных из CarNetworkManager. Каждый элемент списка содержит номер автомобиля.
* Использование NavigationLink для перехода к экрану редактирования (EditCarView) выбранного автомобиля.
* Кнопки "Обновить" и "Добавить" для загрузки актуального списка автомобилей и добавления нового автомобиля соответственно.

**Процесс Работы:**

* При нажатии на кнопку "Обновить", список автомобилей обновляется, загружая данные из CarNetworkManager.
* Для добавления нового автомобиля используется модальное окно AddCarView.

**Удаление Автомобилей:**

* Функционал удаления автомобиля реализован через свайп по элементу списка с последующим подтверждением удаления.

### AddCarView: Добавление Нового Автомобиля

**Основные Функции:**

* Позволяет ввести данные нового автомобиля, включая номер, цвет, марку и статус (импортный или нет).
* Кнопка "Добавить" для отправки данных нового автомобиля на сервер через CarNetworkManager.

**Процесс Работы:**

* После ввода данных и нажатия кнопки "Добавить", новый автомобиль добавляется в список и сохраняется в базе данных.

### EditCarView: Редактирование Существующего Автомобиля

**Основные Функции:**

* Форма для редактирования выбранного автомобиля, включая его номер, цвет, марку и статус.
* Кнопка "Обновить" для сохранения изменений в данных автомобиля.

**Процесс Работы:**

* При нажатии на кнопку "Обновить", обновленные данные автомобиля отправляются на сервер. После успешного обновления вызывается функция onCarUpdated для обновления списка автомобилей в родительском представлении.

Эти экраны обеспечивают полный цикл управления данными автомобилей в AutoServiceERP-Frontend, от просмотра и добавления до редактирования и удаления, предоставляя пользователям удобный и интуитивно понятный интерфейс.

A screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generatedA screenshot of a phone

Description automatically generated

В рамках проекта AutoServiceERP-Frontend я разработал три основных экрана для управления информацией о мастерах автосервиса: MasterListView, AddMasterView и EditMasterView. Эти экраны обеспечивают полный функционал для работы с данными мастеров, включая просмотр, добавление, редактирование и удаление.

### MasterListView: Просмотр и Управление Списком Мастеров

**Основные Функции:**

* Представление списка мастеров, полученных из MasterNetworkManager. Каждый элемент списка содержит имя мастера.
* Возможность перехода к экрану редактирования (EditMasterView) для выбранного мастера через NavigationLink.
* Кнопки "Обновить" и "Добавить" для загрузки актуального списка мастеров и добавления нового мастера соответственно.

**Процесс Работы:**

* При нажатии на кнопку "Обновить", список мастеров обновляется, загружая данные из MasterNetworkManager.
* Для добавления нового мастера используется модальное окно AddMasterView.

**Удаление Мастеров:**

* Функционал удаления мастера реализован через свайп по элементу списка с последующим подтверждением удаления.

### AddMasterView: Добавление Нового Мастера

**Основные Функции:**

* Позволяет ввести имя нового мастера.
* Кнопка "Добавить" для отправки данных нового мастера на сервер через MasterNetworkManager.

**Процесс Работы:**

* После ввода имени и нажатия кнопки "Добавить", новый мастер добавляется в список и сохраняется в базе данных.

### EditMasterView: Редактирование Существующего Мастера

**Основные Функции:**

* Форма для редактирования выбранного мастера, включая его имя.
* Кнопка "Обновить" для сохранения изменений в данных мастера.

**Процесс Работы:**

* При нажатии на кнопку "Обновить", обновленные данные мастера отправляются на сервер. После успешного обновления вызывается функция onMasterUpdated для обновления списка мастеров в родительском представлении.

Эти экраны играют ключевую роль в управлении информацией о мастерах в AutoServiceERP-Frontend, обеспечивая удобный и эффективный интерфейс для администрирования данных о персонале автосервиса.

A screenshot of a phone

Description automatically generatedA black rectangular frame with a white screen

Description automatically generatedA screen shot of a phone

Description automatically generated

В ходе разработки приложения AutoServiceERP-Frontend был реализован широкий спектр функциональностей, направленных на управление и оптимизацию процессов в автосервисе. Этот проект включает в себя такие ключевые аспекты, как управление данными о мастерах, автомобилях и услугах, а также назначение работ и аналитику. Разработка охватывает как серверную, так и клиентскую части, с использованием современных технологий и практик программирования.

### Стек Используемых Технологий

1. **Spring Boot (start.spring.io)**: Используется для быстрого создания и настройки серверной части приложения, облегчая конфигурацию и развертывание.
2. **PostgreSQL (pgadmin4, Docker Postgres Image)**: Выбор PostgreSQL в качестве базы данных обеспечивает надежное и масштабируемое хранение данных. PgAdmin4 и Docker-образ Postgres используются для управления и локального развертывания базы данных соответственно.
3. **Testcontainers и JUnit 5**: Обеспечивают надежное тестирование серверной части с использованием контейнеризированных сред тестирования и расширенных возможностей JUnit для unit и integration тестов.
4. **Postman**: Используется для тестирования и дебаггинга RESTful API, разработанного на основе лучших практик (restfulapi.net).
5. **Spring Framework, Hibernate, Jakarta EE**: Являются основой для создания серверной части, предоставляя мощные инструменты для работы с REST API, ORM и другими аспектами enterprise-разработки.
6. **Gradle**: Служит для автоматизации сборки и управления зависимостями проекта.
7. **Apache Tomcat**: Используется как основной веб-контейнер для развертывания и запуска серверного приложения.

### **Вывод**

Реализация AutoServiceERP-Frontend является примером комплексного подхода к разработке программного обеспечения для бизнеса. Основываясь на современных технологиях и лучших практиках, проект успешно сочетает в себе функциональность, удобство использования и надежность. Интеграция между клиентскими и серверными частями приложения обеспечивает гладкое взаимодействие всех компонентов системы, в то время как использование передовых инструментов и фреймворков гарантирует высокое качество и расширяемость проекта. Этот проект становится значительным вкладом в управление и автоматизацию процессов в сфере автосервисов, демонстрируя эффективность современных технологий в решении конкретных бизнес-задач.

# **Список используемой литературы**

1. Новиков Б.А., Горшкова Е.А. Основы технологии баз данных. / ДМК Пресс, 2019. – 241 с.
2. Джуба С., Волкова А. Изучаем PostgresSQL 10 / ДМК Пресс, 2019. – 402 c.