**ПРОЕКТНАЯ ДОКУМЕНТАЦИЯ**

**Общее описание программы:**

Программа представляет собой настольное приложение с графическим интерфейсом. Цель приложения — автоматизацию и упрощение работы фотосалона, включая такие действия, как просмотр, добавление, редактирование и удаление записей.

**Окна программы:**

Приложение состоит из нескольких окон, каждое из которых выполняет свою задачу:

* Регистрация (Registration\_Window) – позволяет новым пользователям создать учетную запись в приложении. В этом окне пользователи вводят свои личные данные, такие как имя, адрес электронной почты и пароль. Регистрация необходима для получения доступа к функционалу приложения и позволяет сохранять информацию о пользователе;
* Авторизация (Authorization\_Window) – позволяет зарегистрированным пользователям входить в свою учетную запись. Пользователи вводят свои данные (почту и пароль) для доступа к основным функциям приложения. Это окно обеспечивает безопасность и защиту личной информации пользователей. Так же это окно служит для входа в систему под аккаунтом администратора;
* Главное окно для пользователя (Main\_User) – центральное окно для взаимодействия пользователя с приложением. Здесь пользователь может видеть свою учетную запись, доступные услуги, а также переходить к другим функциям приложения, таким как оформление заказа или история заказов. Это окно служит основным интерфейсом для пользователя;
* Оформление заказа (AddOrder) – позволяет пользователям заказать фото на печать. В этом окне пользователи могут выбирать количество копий, необходимые материалы и оборудование, адрес салона, сотрудника, который будет выполнять заказ, указывать дату и время, а также пользователь может посмотреть какая стоимость будет у заказа. Это окно упрощает процесс заказа и делает его более удобным;
* Редактирование учетной записи пользователя (EditUser) – позволяет пользователям изменять свою личную информацию, такую как имя, адрес электронной почты и пароль. Это важно для поддержания актуальности данных пользователя и обеспечения безопасности учетной записи;
* Главное окно для администратора (Main\_Admin) – центральное окно для администраторов, где они могут управлять всеми аспектами приложения. Это окно предоставляет доступ к функционалу, необходимому для управления пользователями, заказами, сотрудниками и ресурсами фотосалона;
* Добавление, изменение, удаление сотрудников (AddEmployee) – позволяет администраторам управлять списком сотрудников фотосалона. В этом окне администраторы могут добавлять новых сотрудников, редактировать информацию существующих или удалять их из базы данных. Это важно для поддержания актуальности информации о персонале;
* Добавление, изменение, удаление оборудования (AddEquipment) – позволяет администраторам управлять оборудованием, используемым в фотосалоне. В этом окне они могут добавлять новое оборудование, вносить изменения в характеристики существующего или удалять устаревшее. Это помогает поддерживать базу данных оборудования в актуальном состоянии;
* Добавление, изменение, удаление материалов (AddMaterials) – позволяет администраторам управлять материалами, необходимыми для выполнения услуг. Это может включать фотобумагу, краски и другие расходные материалы. Окно позволяет администраторам следить за запасами и обновлять информацию о материалах;
* Добавление, изменение, удаление салонов печати (AddPrintShops) – позволяет администраторам управлять списком салонов печати, с которыми работает фотосалон. В этом окне можно добавлять новые салоны, редактировать информацию о существующих или удалять неактуальные. Это важно для организации работы с партнерами и обеспечения качества услуг;

**Модули:**

* BtnRes – задает стили элементам программы, такие как скругление углов элементов, цвет фона, цвет и толщина обводки, анимации появления;
* Scrollbar – задает стили элементам прокрутки окон программы;
* Add\_itm – предназначен для добавления нового элемента сотрудников, оборудования, материалов, салонов печати;
* EmployeeAdminItem – предназначен для отображения элементов таблицы «Сотрудники» из базы данных для аккаунта администратора. Этот элемент предоставляет возможность перейти на страницы для добавления, изменения и удаления сотрудников;
* EquipmentAdminItem – предназначен для отображения элементов таблицы «Оборудование» из базы данных для аккаунта администратора. Этот элемент предоставляет возможность перейти на страницы для добавления, изменения и удаления оборудования;
* MaterialAdminItem – предназначен для отображения элементов таблицы «Материалы» из базы данных для аккаунта администратора. Этот элемент предоставляет возможность перейти на страницы для добавления, изменения и удаления материалов;
* PrintShopsAdminItem – предназначен для отображения элементов таблицы «Салоны печати» из базы данных для аккаунта администратора. Этот элемент предоставляет возможность перейти на страницы для добавления, изменения и удаления салонов печати;
* OrderAdminItem – предназначен для отображения элементов таблицы «Заказы» из базы данных для аккаунта администратора. Этот элемент предоставляет возможность удаления заказов пользователей;
* PaymentsAdminItem – предназначен для отображения элементов таблицы «Платежи» из базы данных для аккаунта администратора. Этот элемент предоставляет возможность удаления платежей пользователей;
* UserAdminItem – предназначен для отображения элементов таблицы «Клиенты» из базы данных для аккаунта администратора. Этот элемент предоставляет возможность удаления аккаунтов пользователей;
* OrderUserItem – предназначен для отображения элементов таблицы «Заказы» из базы данных для аккаунта пользователя. Этот компонент позволяет пользователю, который авторизовался под своими учетными данными, добавлять и редактировать свои заказы;
* PaymentsUserItem предназначен для отображения элементов таблицы «Платежи» из базы данных, связанных с аккаунтом пользователя. Этот компонент позволяет авторизованным пользователям просматривать свои оплаченные заказы, а также выбирать способ оплаты для выполнения неоплаченных транзакций;
* UserUserItem – предназначен для отображения элементов таблицы «Клиенты» из базы данных для аккаунта пользователя. Этот компонент позволяет авторизованным пользователям просматривать свой аккаунт и изменять данные.

**Классы:**

* Connection – необходим для подключения программы к базе данных. Он включает методы для открытия и закрытия подключения, а также обеспечивает быстрый и эффективный доступ к данным из базы данных:

1. **Открытие подключения** (Open): Метод Open создает новое подключение к базе данных с использованием строки подключения, определенной в методе. Он открывает соединение с базой данных, что позволяет выполнять последующие операции, такие как запросы или команды.
2. **Закрытие подключения** (CloseConnection): Метод CloseConnection закрывает переданное подключение к базе данных. Он также очищает пул подключений, что помогает управлять ресурсами и предотвращает утечки памяти.
3. **Выполнение запроса** (Query): Метод Query принимает SQL-запрос и подключение к базе данных. Он создает объект SqlCommand, который выполняет указанный запрос. Возвращает объект SqlDataReader, который позволяет читать результаты запроса построчно.

* Clients предназначен для представления клиентов фотосалона и управления их данными. Вот основные функции и цели этого класса:

1. **Хранение информации о клиентах**: класс содержит свойства, такие как Id, FIO, Email и Password, которые представляют собой идентификатор клиента, его полное имя, адрес электронной почты и пароль соответственно. Эти данные могут использоваться для аутентификации клиентов и хранения их информации в базе данных.
2. **Получение всех клиентов**: метод AllClients отвечает за извлечение списка всех клиентов из базы данных. Он устанавливает соединение с базой данных, выполняет SQL-запрос для получения всех записей из таблицы Clients и создает список объектов Clients, который затем возвращается. Этот метод может использоваться для отображения списка клиентов в интерфейсе пользователя или для дальнейшей обработки данных.
3. **Аутентификация клиентов**: метод GetClientByFIOAndEmail служит для проверки учетных данных клиента. Он принимает адрес электронной почты и пароль, выполняет запрос к базе данных, чтобы найти клиента с указанными данными, и возвращает объект Clients, если клиент найден. Если нет, метод возвращает null. Это полезно для реализации функций входа в систему.

* Класс Employee предназначен для представления сотрудников фотосалона и управления их данными. Вот основные функции и цели этого класса:

1. **Хранение информации о сотрудниках:** класс содержит свойства, такие как Id, FIO, Email и Post, которые представляют собой идентификатор сотрудника, его полное имя, адрес электронной почты и должность соответственно. Эти данные могут использоваться для управления сотрудниками и хранения их информации в базе данных.
2. **Получение всех сотрудников:** метод AllEmployee отвечает за извлечение списка всех сотрудников из базы данных. Он устанавливает соединение с базой данных, выполняет SQL-запрос для получения всех записей из таблицы Employee и создает список объектов Employee, который затем возвращается. Этот метод может использоваться для отображения списка сотрудников в интерфейсе пользователя или для дальнейшей обработки данных.
3. **Создание экземпляра Employee:** конструктор класса позволяет создавать объекты Employee с заданными параметрами. Это удобно для создания новых сотрудников и их добавления в систему, а также для работы с данными, полученными из базы данных.

* Класс Equipment предназначен для представления оборудования фотосалона и управления его данными. Вот основные функции и цели этого класса:

1. **Хранение информации об оборудовании**: класс содержит свойства, такие как Id, EquipmentName и CostPerUnit, которые представляют собой идентификатор оборудования, его наименование и стоимость за единицу соответственно. Эти данные могут использоваться для учета оборудования и определения его стоимости при аренде или продаже.
2. **Получение всего оборудования**: метод AllEquipment отвечает за извлечение списка всего оборудования из базы данных. Он устанавливает соединение с базой данных, выполняет SQL-запрос для получения всех записей из таблицы Equipment и создает список объектов Equipment, который затем возвращается. Этот метод может использоваться для отображения списка оборудования в интерфейсе пользователя или для дальнейшего анализа данных.
3. **Управление данными об оборудовании**: класс может быть расширен методами для добавления, обновления и удаления данных об оборудовании. Например, можно создать метод AddEquipment для добавления нового оборудования в базу данных, метод UpdateEquipment для обновления информации об уже существующем оборудовании и метод DeleteEquipment для удаления оборудования из базы. Эти функции обеспечивают возможность полноценного управления учетной записью оборудования в фотосалоне.

* Класс Materials предназначен для представления материалов, используемых в фотосалоне, и управления их данными. Вот основные функции и цели этого класса:

1. **Хранение информации о материалах:** класс содержит свойства, такие как Id, MaterialName, MaterialRest и CostPerUnit, которые представляют собой идентификатор материала, его название, остаток материала на складе и стоимость за единицу соответственно. Эти данные могут использоваться для управления запасами материалов и расчета стоимости услуг.
2. **Получение всех материалов:** метод AllMaterials отвечает за извлечение списка всех материалов из базы данных. Он устанавливает соединение с базой данных, выполняет SQL-запрос для получения всех записей из таблицы Materials и создает список объектов Materials, который затем возвращается. Этот метод может использоваться для отображения списка доступных материалов в интерфейсе пользователя или для дальнейшей обработки данных, например, для анализа использования материалов.
3. **Обновление остатка материалов:** в будущем можно добавить метод, который будет отвечать за обновление остатка материалов на складе. Этот метод сможет принимать идентификатор материала и новое значение остатка, устанавливая новое значение в базе данных. Это полезно для управления запасами и контроля за использованием материалов в процессе работы фотосалона.

* Класс Orders предназначен для представления заказов фотосалона и управления их данными. Вот основные функции и цели этого класса:
  1. **Хранение информации о заказах:** класс содержит свойства, такие как Id, Image, NumberCopies, OrderDate, IdClient, FIOEmployee, MaterialName, EquipmentName, AddressSalon и FullCost, которые представляют собой уникальный идентификатор заказа, изображение, количество копий, дату заказа, идентификатор клиента, полное имя сотрудника, название материала, название оборудования, адрес салона и общую стоимость заказа соответственно. Эти данные могут использоваться для учета заказов и их обработки.
  2. **Получение всех заказов:** метод AllOrders отвечает за извлечение списка всех заказов из базы данных. Он устанавливает соединение с базой данных, выполняет SQL-запрос для получения всех записей из таблицы Orders и создает список объектов Orders, который затем возвращается. Этот метод может использоваться для отображения списка заказов в интерфейсе пользователя или для дальнейшей обработки данных.
  3. **Получение заказов по клиенту:** метод GetOrdersByClient служит для получения всех заказов, связанных с определенным клиентом. Он принимает идентификатор клиента, выполняет запрос к базе данных, чтобы найти все заказы с указанным идентификатором, и возвращает список объектов Orders, соответствующих этому клиенту. Это полезно для анализа истории заказов конкретного клиента или для отображения информации о его заказах.
* Класс Payments предназначен для представления платежей, произведенных клиентами фотосалона, и управления их данными. Вот основные функции и цели этого класса:

1. **Хранение информации о платежах:** класс содержит свойства, такие как Id, IdOrder, FullCost, PaymentDate, PaymentStatus, PaymentMethod и IdClient. Эти свойства представляют собой идентификатор платежа, идентификатор заказа, общую стоимость, дату платежа, статус платежа, способ платежа и идентификатор клиента соответственно. Эти данные могут использоваться для отслеживания платежей и генерации отчетов.
2. **Получение всех платежей:** метод AllPayments отвечает за извлечение списка всех платежей из базы данных. Он устанавливает соединение с базой данных, выполняет SQL-запрос для получения всех записей из таблицы Payments и создает список объектов Payments, который затем возвращается. Этот метод может использоваться для отображения списка всех платежей в интерфейсе пользователя или для дальнейшей обработки данных, например, для создания отчетов по всем платежам.
3. **Получение платежей по клиенту:** метод GetPaymentsByClient служит для извлечения всех платежей, связанных с конкретным клиентом. Он принимает идентификатор клиента, выполняет SQL-запрос с условием по этому идентификатору и возвращает список объектов Payments, соответствующих указанному клиенту. Это полезно для анализа платежной активности клиентов и предоставления информации о их транзакциях.

* Класс PrintShops предназначен для представления фотосалонов и управления их данными. Вот основные функции и цели этого класса:

1. **Хранение информации о фотосалонах**: класс содержит свойства, такие как Id, AddressSalon и OpeningHours, которые представляют собой идентификатор фотосалона, его адрес и часы работы соответственно. Эти данные могут быть использованы для отображения информации о салонах в интерфейсе пользователя или для дальнейшей обработки.
2. **Получение всех фотосалонов**: метод AllPrintShops отвечает за извлечение списка всех фотосалонов из базы данных. Он устанавливает соединение с базой данных, выполняет SQL-запрос для получения всех записей из таблицы PrintShops и создает список объектов PrintShops, который затем возвращается. Этот метод может использоваться для отображения списка фотосалонов в интерфейсе пользователя или для дальнейшей обработки данных.
3. **Управление данными о фотосалонах**: класс может быть расширен методами для добавления, обновления и удаления фотосалонов. Например, можно реализовать метод, который будет добавлять новый фотосалон в базу данных, принимая необходимые параметры (адрес, часы работы) и выполняя соответствующий SQL-запрос. Это позволит администраторам фотосалона легко управлять информацией о доступных салонах.