**СОДЕРЖАНИЕ**

[Введение 4](#_Toc28214274)

[1 Постановка задачи 6](#_Toc28214275)

[1.1 Формулировка задания 6](#_Toc28214276)

[1.2 Обоснование необходимости автоматизации 7](#_Toc28214277)

[1.3 Определение данных и их представление 8](#_Toc28214278)

[2 Вычислительная система 10](#_Toc28214279)

[2.1 Обоснование выбора языка программирования 10](#_Toc28214280)

[2.2 Обоснование выбора среды разработки 12](#_Toc28214281)

[2.3 Требование к конфигурации вычислительной системы 13](#_Toc28214282)

[3 Проектирование 15](#_Toc28214283)

[3.1 Проектирование интерфейса 15](#_Toc28214284)

[3.2 Инфологическая модель предметной области 16](#_Toc28214285)

[3.3 Логическая модель данных 17](#_Toc28214286)

[3.4 Физическая модель базы данных 19](#_Toc28214287)

[4 Описание программы 22](#_Toc28214288)

[4.1 Физическая структура 22](#_Toc28214289)

[4.2 Логическая структура 22](#_Toc28214290)

[5 Отладка и испытание программы 25](#_Toc28214291)

[5.1 Тестовые примеры 25](#_Toc28214292)

[5.2 Анализ полученных результатов 26](#_Toc28214293)

[6 Описание примененеия программы 32](#_Toc28214294)

[6.1 Назначение и условия применения 32](#_Toc28214295)

[6.2 Описание работы программы 32](#_Toc28214296)

[Заключение 36](#_Toc28214297)

[Список использованных источников 37](#_Toc28214298)

[Приложение А (обязательное) Листинг программы 38](#_Toc28214299)

ВВЕДЕНИЕ

В данном преддипломном проекте необходимо разработать автоматизированное место администратора (АРМ) менеджера салона красоты. Последнее время, все больший интерес к автоматизации явно ощущается со стороны активно развивающихся салонов красоты и маникюрных салонов, желающих минимизировать потери рабочего времени персонала, сэкономить на его численности и поднять уровень сервиса в салонах.

Салоны красоты в первую очередь работают с людьми, а значит, качество обслуживания стоит на одном из первых мест. Сюда входит не только вежливый персонал и качественная работа мастера, но и скорость в том числе. Быстрое оперативное обслуживание – это всегда большой плюс, а обеспечить его поможет автоматизация салона красоты.

Быстрое управление записями, оперативный расчет с клиентами – все это позволяет поднять обслуживание на новый уровень. Автоматизация салонов красоты ускорит работу, а значит, количество обслуживаемых клиентов увеличится, поднимая прибыль. Также стоит отметить, что повысится контроль работы мастеров, учет загруженности мастера станет легче.

Основной целью выполнения преддипломной практики является расширение, углубление знаний по дисциплине «Базы данных и системы управления базами данных», формирование навыков научно-исследовательской деятельности.

Задачи выполнения преддипломной практики:

* систематизация знаний в области программирования на языках PHP и Sql;
* углубление уровня и расширение объема профессионально значимых знаний, умений и навыков разработки программ в средах MySql и Microsoft Visual Code;
* формирование умений и навыков самостоятельной организации научно-исследовательской работы;
* овладение современными методами поиска, обработки и использования информации.

Данная пояснительная записка содержит следующие разделы:

* раздел «Постановка задачи» содержит описание всех функций проекта, входных и выходных данных;
* в разделе «Вычислительная система» приводится описание операционной системы, языка программирования и среды разработки выбранных для реализации проекта;
* раздел «Проектирование» содержит описание внешнего пользовательского интерфейса, описание алгоритма программы и структуры справочной системы;
* в разделе «Описание программы» описываются все библиотечные модули, входящие в состав программы, а также назначение всех функций, входящих в приложение, здесь же приводится интерфейс программы с рисунками пользовательских форм;
* раздел «Отладка и испытание программы» иллюстрирует проверку работы программы на различных тестах и содержит анализ полученных результатов;
* в разделе «Описание применения программы» содержатся сведения о назначении программного средства, классе решаемых задач и ограничениях, накладываемых на область применения.

В «Заключении» описываются использованные методы и средства, полнота раскрытия проблемы и возможность дальнейшей модификации проекта.

1. ПОСТАНОВКА ЗАДАЧИ
   1. Формулировка задания

Необходимо разработать программу «АРМ менеджера салона красоты», позволяющую работать с базой данных салона. В базе данных хранится информация о мастерах, клиентах, материалах, о расходе материалов, об услугах и о записи клиентов.

Предусмотреть добавление, редактирование и удаление записей клиентов. Реализовать возможность редактирования записей в таблицах, содержащих информацию о мастерах, клиентах, материалах и т.д. Также требуется реализовать функции сортировки и фильтрации информации по следующим критериям:

* отсортировать услуги по цене;
* отобразить все услуги, на которые записывался клиент;
* показать мастера и услугу, на которые записался клиент;
* посчитать и предоставить прибыль за любую дату, за любой промежуток времени, за неделю, за месяц.

Для добавления записи клиента необходимо: заполнить информацию о клиенте и услуге (ФИО клиента, контактный телефон, выбрать услугу, дату и время записи, выбрать мастера).

Реализовать возможность:

* формирования отчета о прибыли, за определенное время;
* формирования чека на запись клиента;
* просмотра различной информации;
* запись ведется только на свободные даты/время у свободных мастеров.

Программа должна обеспечить удобную навигацию между информационными окнами. Предусмотреть обработку исключительных ситуаций при вводе информации. Разработать справочную систему по работе с программой.

* 1. Обоснование необходимости автоматизации

Автоматизированное рабочее место - индивидуальный комплекс технических и программных средств, предназначенный для автоматизации профессионального труда специалиста и обеспечивающий подготовку, редактирование, поиск и выдачу на экран и печать необходимых ему документов и данных. Автоматизированное рабочее место обеспечивает оператора всеми средствами, необходимыми для выполнения определенных функций.

Сейчас появилось достаточно много аналогов программ для автоматизации работы салонов, все они похожи друг на друга, хотя и имеют различия. Например, Beaury Pro – вышла на рынок в 2014 году. Система поддерживает интеграцию с 16 различными модулями: Mango Office, SendPulse, Яндекс Диск, DropBox и т.д. В настоящий момент система очень популярна и широко используется в разных странах мира.

Система включает различные модули и приложения. Несмотря на популярность, некоторые пользователи жалуются на достаточно сложный и неинтуитивный интерфейс.

Еще один инструмент автоматизации – BlocknotApp. Система характеризуется наличием большого числа различных модулей, но функционал почти вдвое урезан по сравнению с тем же сервисом Beaty Pro, однако интерфейс у BlocknotApp гораздо проще для освоения.

Большое число модулей, которые подключаются за отдельную плату, довольно непросто администрировать, что приводит к повышению вероятности возникновения сбоев.

Поддержки с помощи чата нет, связь по телефону в режиме 24/7 довольно оперативная, но соединяют с операторами, которые не в курсе всех технических тонкостей системы.

Также есть YCLIENTS – облачная система автоматизации салонного бизнеса, которая позволяет управлять услугами, записью клиентов, принимать оплату и настраивать принтеры для печати чеков. Из небольших недостатков (хотя это скорее гипотетически) — нет поддержки Windows, только iOS и Android. Также, к недостаткам можно отнести онлайн-работу данного сервиса, так как в случае отключения интернета на продолжительное время прервётся система контроля салона, что может понести за собой потерю клиентов.

Преимущества автоматизированнного рабочего места менеджера салона красоты состоит в том, что у него удобный интерфейс, позволяющий управлять записью клиентов, принимать оплату, печатать чеки, делать и просматривать различные запросы. Наглядно видеть загруженость мастеров и запись к ним. А также данный преддипломный проект разработан для Windows.

Менеджер салона красоты владеет информацией об услугах, списке материалов и мастерах, на которых была произведена запись. АРМ позволяет работнику видеть, какие мастера уже свободны, какие еще находятся в работе.

* 1. Определение данных и их представление

Данные – поддающееся многократной интерпретации представление информации в формализованном виде, пригодном для передачи, связи или обработки.

Для долговременного хранения данных обычно используются базы данных. Данные в памяти могут быть организованы в различные виды структур данных таких как массивы, строки, связанные списки или объекты. Структуры данных могут хранить данные различных типов, включая числа, строки и другие структуры данных. Ввод и вывод данных в компьютеры производится через периферийные устройства.

Входными данными для работы программы являются личные данные клиентов, такие как фамилия, контактный телефон. Также в качестве входных данных выступают данные об услуге, дата и время, а так же выбор самого мастера. Данные заносятся в таблицы и хранятся в базе данных.

Выходными данными являются таблицы с отсортированными по выбору пользователя данными, таблица записи клиентов, услуг, материалов, отчеты о прибыли и чеки, загруженность мастера.

1. ВЫЧИСЛИТЕЛЬНАЯ СИСТЕМА
   1. Обоснование выбора языка программирования

Для реализации данного программного продукта был выбран язык объектное ориентированного программирования PHP. Данный язык хорошо подходит для написания различных веб-приложений [1, с 10 – 15].

PHP является одним из наиболее распространённых языков программирования, который позволяет разрабатывать различные виды приложений на платформах ASP.NET и Windows, а также веб-службы XML, баз данных, клиент-серверного ПО и др. Наиболее часто в качестве редактора и компилятора кодов используется среда Visual Code, отличающаяся удобными конструкторами, понятным пользовательским интерфейсом, интегрированным отладчиком и многими другими инструментами, упрощающими создание прикладных программ.

PHP отличается очень богатым синтаксисом, который в то же время является простым и удобным. Особенно легко даётся изучение программистам, которые уже владеют знаниями и навыками работы с языками JSP, Perl и Java. PHP сочетает в себе все преимущества этих языков, при этом не имеет их недостатков. Одним из основных достоинств языка является наличие мощных функций, и предоставление прямого доступа к операционной памяти. PHP пользуется универсальными методами, интеграторами и типами данных, обеспечивающими наивысшую безопасность и производительность разработки программ. Характерной чертой языка является наличие выражений LINQ, с помощью которых можно создавать очень удобные кодовые конструкции со строго типизированными запросами. PHP относится к объектно-ориентированным языкам, который имеет строгую типизацию. Каждая переменная и метод (в том числе входная точка Main), инкапсулируется при определении классов, при этом из родительского класса можно реализовывать неограниченное количество интерфейсов [2, с 25 – 30].

Чтобы взаимодействовать с другими программными обеспечениями типа Windows (объекты COM, собственные библиотеки DLL), язык PHP имеет процесс «Взаимодействия». Они позволяют приложениям, написанным на «пхп», выполнять практически все действия и функции, которые прописаны машинным кодом. Также языком PHP поддерживается набор указателей и «небезопасных» блоков кода в тех случаях, когда критически важно наличие прямого доступа.

SQL – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Является, прежде всего, информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционных базах данных. SQL считается языком программирования, в общем случае (без ряда современных расширений) не является тьюринг-полным, но вместе с тем стандарт языка спецификацией SQL/PSM предусматривает возможность его процедурных расширений [3, с 20 – 30].

Несмотря на наличие диалектов и различий в синтаксисе, в большинстве своём тексты SQL-запросов, содержащие DDL и DML, могут быть достаточно легко перенесены из одной СУБД в другую. Существуют системы, разработчики которых изначально ориентировались на применение по меньшей мере нескольких СУБД. А также еще одним преимуществом языка SQL является наличие стандартов и набора тестов для выявления совместимости и соответствия конкретной реализации SQL общепринятому стандарту только способствует «стабилизации» языка.

С помощью SQL программист описывает только то, какие данные нужно извлечь или модифицировать. То, каким образом это сделать, решает СУБД непосредственно при обработке SQL-запроса. Однако не стоит думать, что это полностью универсальный принцип – программист описывает набор данных для выборки или модификации, однако ему при этом полезно представлять, как СУБД будет разбирать текст его запроса. Чем сложнее сконструирован запрос, тем больше он допускает вариантов написания, различных по скорости выполнения, но одинаковых по итоговому набору данных.

* 1. Обоснование выбора среды разработки

Для преддипломного проекта выбрана среда разработки Visual Code, которая на момент написания данной пояснительной записки является не самой новой версией среды разработки в PHP.

Microsoft Visual Code – линейка продуктов компании Майкрософт, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом.

Visual Code включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и как отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Code позволяет создавать и подключать сторонние дополнения для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода.

Главным преимуществом Visual Code является производительность. Обеспечивает возможность создания разнообразных приложений на основе одного набора навыков [4, с 45 - 56].

Отличия Visual Code от предыдущих версий:

* модульный подход в инсталляции;
* улучшения средств разработки мобильных приложений.

В качестве СУБД выбрана клиентская программа MySQL представляющая собой утилиту командной строки. Эта программа подключается к серверу по сети. Команды, выполняемые сервером, обычно связаны с чтением и записью данных на жестком диске. Обычно MySQL используется в качестве сервера, к которому обращаются локальные или удалённые клиенты, однако в дистрибутив входит библиотека внутреннего сервера, позволяющая включать MySQL в автономные программы.

Гибкость СУБД MySQL обеспечивается поддержкой большого количества типов таблиц: пользователи могут выбрать как таблицы типа MyISAM, поддерживающие полнотекстовый поиск, так и таблицы InnoDB, поддерживающие транзакции на уровне отдельных записей. Более того, СУБД MySQL поставляется со специальным типом таблиц EXAMPLE, демонстрирующим принципы создания новых типов таблиц. Благодаря открытой архитектуре и GPL-лицензированию, в СУБД MySQL постоянно появляются новые типы таблиц.

* 1. Требование к конфигурации вычислительной системы

АРМ разрабатывается для семейства операционных систем Windows. Приоритет отдан десятой версии Windows.

К преимуществам операционной системы Windows можно причислить хорошую производительность, дружелюбный к пользователю интерфейс и поддержку multitouch-ввода. Отдельно стоит выделить удобную панель задач с реализованной функцией предпросмотра окна.

Windows 10 – операционная система для персональных компьютеров и рабочих станций, разработанная корпорацией Microsoft в рамках семейства Windows NT. Windows 10 создана для работы с современными многоядерными процессорами. Все 32-разрядные версии Windows 8 поддерживают до 32 ядер процессора, а 64‑разрядные версии – до 256 ядер.

По состоянию на июль 2017 года Windows 10 имеет долю среди используемых в мире операционных систем для доступа к сети Интернет 35,7 % и занимает первое место в мире по популярности, опередив в апреле предыдущего лидера – Windows 7 [5, с 34 - 35].

Для корректной работы игры компьютер должен удовлетворять системным требованиям Windows 10 [6, с 12 - 15]:

* процессор как минимум 1 ГГц или SoC;
* ОЗУ 1 ГБ (для 32-разрядных систем) или 2 ГБ (для 64-разрядных систем);
* свободного места на жёстком диске 16 ГБ (для 32-разрядных систем) или 20 ГБ (для 64-разрядных систем);
* видеоадаптер DirectX версии не ниже 9 с драйвером WDDM 1.0;
* дисплей 800 x 600.

Для работы с клиентской частью системы требуется компьютер, подключенный по протоколу TCP/IP к сети, в которой находится сервер с установленной CimWebCenter.

АРМ требует минимум 4 ГБ свободного места на жестком диске (рекомендуемый объем 8 ГБ и выше), требования к центральному процессору: Multi Core 64bit x86 CPU, 8 GB RAM.

1. ПРОЕКТИРОВАНИЕ
   1. Проектирование интерфейса

Интерфейс – это комплекс средств, предназначенных для взаимодействия двух систем друг с другом. В качестве таких систем может выступать что угодно, включая людей и искусственный интеллект. Слово «интерфейс» позаимствовано из английского языка: interface означает «место соприкосновения».

В компьютерной и вычислительной технике чаще всего под интерфейсом понимают элементы, обеспечивающие взаимодействие аппаратных и программных средств между собой и с человеком. В электронной коммерции под этим словом подразумеваются методы взаимодействия программного обеспечения с пользователем. Этот вид интерфейса называется человеко-машинным.

При запуске программы появится главное меню – это страница, где пользователь сможет быстро перейти в различные разделы (добавить данные в различные таблицы или просмотреть любую информацию и получить различные сведения, а также просмотреть отчеты или отправить их на печать).

На странице записи клиентов необходимо будет заполнить поля:

* дата и время записи;
* ФИО клиента;
* номер телефона;
* адрес электронной почты;
* выбрать услугу;
* выбрать мастера.

На форме, где расположена вся информация пользователь сможет просматривать таблицы («Клиенты», «Услуги», «Мастера», «Расходные материалы», «Расход материалов», «Запись клиентов»). Также пользователь сможет просматривать отчёты и искать различную информацию.

* 1. Инфологическая модель предметной области

Инфологическая модель отображает реальный мир в некоторые понятные человеку концепции, полностью независимые от параметров среды хранения данных. Существует множество подходов к построению таких моделей: графовые модели, семантические сети, модель «сущность-связь».

Салон дает возможность менеджеру сделать запись клиента. Для того, чтобы записать клента, необходимы следующие данные: фамилия, имя, отчество, контактный телефон, выбранная услуга, дата и время записи, мастер. При записи в салон менеджер должен уточнить личные данные клиента. Салон хранит информацию о занятости мастеров. Базой данных может пользоваться только менеджер, который просматривает базу, вносит изменения, а также заполняет ее.

Спроектируем инфологическую модель системы, предназначенной для хранения информации о записи клиентов. Разработку модели начнем с выделения основных сущностей. Прежде всего, существует сущность «Запись клиента», каждая запись имеет уникальный номер, который является ее ключом, и ряд атрибутов, которые взяты из описания предметной области. Множество экземпляров сущности определяет множество записей, которые находятся в салоне. Для того чтобы отслеживать работу мастеров в салоне, мы должны ввести сущность «Мастер», которая будет содержать номер мастера, ФИО и занятость на определенной промежуток времени. Кроме того, в базе данных содержатся некоторые ограничения, которые способствуют корректной работе программы, например невозможность добавить запись клиента на время, когда мастер будет занят или невозможность добавить запись клиента на прошедшую дату.

* 1. Логическая модель данных

В ходе разработки логической модели базы данных для корректной работы необходимо шесть таблиц. Информация об услугах хранится в таблице 3.1. Таблица хранит основную информацию об услугах. Посредством таблицы об услугах пользователь просматривает такую информацию, как названия услуг, их категорию, время выполнения и стоимость.

Таблица 3.1 – Услуги

|  |  |  |
| --- | --- | --- |
| Наименование | Тип данных | Описание |
| Код услуги | Числовой | Код услуги |
| Название | Короткий текст | Название услуги |
| Категория | Короткий текст | Категория услуги |
| Время выполнения | Время | Время выполнения услуги |
| Стоимость | Числовой | Стоимость услуги |

Информация о мастерах находится в таблице 3.2. Таблица хранит информацию о номере мастера, ФИО мастера, телефон мастера, стаж работы и разряд. Посредством таблицы о мастерах пользователь просматривает информацию о мастерах, которые работают в салоне.

Таблица 3.2 – Мастера

|  |  |  |
| --- | --- | --- |
| Наименование | Тип данных | Описание |
| Номер мастера | Числовой | Номер мастера |
| ФИО | Короткий текст | ФИО мастера |
| Телефон | Короткий текст | Телефон мастера |
| Стаж | Числовой | Стаж работы мастера |
| Разряд | Числовой | Разряд мастера |

Информация о клиентах находится в таблице 3.3. Таблица хранит информацию о номере клиента, ФИО клиента, его контактный номер телефона и почта. Посредством таблицы о клиентах пользователь просматривает информацию о клиентах, ранее посещавших салон.

Таблица 3.3 – Клиенты

|  |  |  |
| --- | --- | --- |
| Наименование | Тип данных | Описание |
| Код записи | Числовой | Код записи |
| ФИО клиента | Короткий текст | ФИО клиента |
| Телефон | Короткий текст | Телефон клиента |
| Почта | Короткий текст | Адрес электронной почты клиента |

Информация о расходных материалах находится в таблице 3.4. Таблица хранит информацию о названии материала, о количестве данного материала на складе или в запасах салона, о производителе и единицы измерения данного материала.

Таблица 3.4 – Расходные материалы

|  |  |  |
| --- | --- | --- |
| Наименование | Тип данных | Описание |
| Код записи | Числовой | Код записи |
| Название материала | Короткий текст | Название материала |
| Количество | Числовой | Количество материала |
| Производитель | Короткий текст | Производитель материала |
| Единицы измерения | Короткий текст | Единицы измерения материала |

Информация о расходе материалов на услугу находится в таблице 3.5. Таблица хранит информацию о номере услуги, номере материала, количестве материала на услугу.

Таблица 3.5 – Расход материалов

|  |  |  |
| --- | --- | --- |
| Наименование | Тип данных | Описание |
| Код записи | Числовой | Код записи |
| Код услуги | Числовой | Код услуги |
| Код материала | Числовой | Код материала |
| Расход | Числовой | Количество материала на 1 услугу |

Информация о записи клиентов находится в таблице 3.6. Таблица хранит информацию о номере записи, дате и времени, номере клиента, номере услуги, номере мастера. Посредством таблицы о записи пользователь просматривает информацию о записи клиента.

Таблица 3.6 – Запись

|  |  |  |
| --- | --- | --- |
| Наименование | Тип данных | Описание |
| Код записи | Числовой | Код записи |
| Дата | Дата | Дата записи |
| Время | Время | Время записи |
| Код клиента | Числовой | Код клиента |
| Код услуги | Числовой | Код услуги |
| Код мастера | Числовой | Код мастера |

* 1. Физическая модель базы данных

Построение физической модели базы данных производилось на основе логической модели. Схема физической модели представлена на рисунке 3.1.

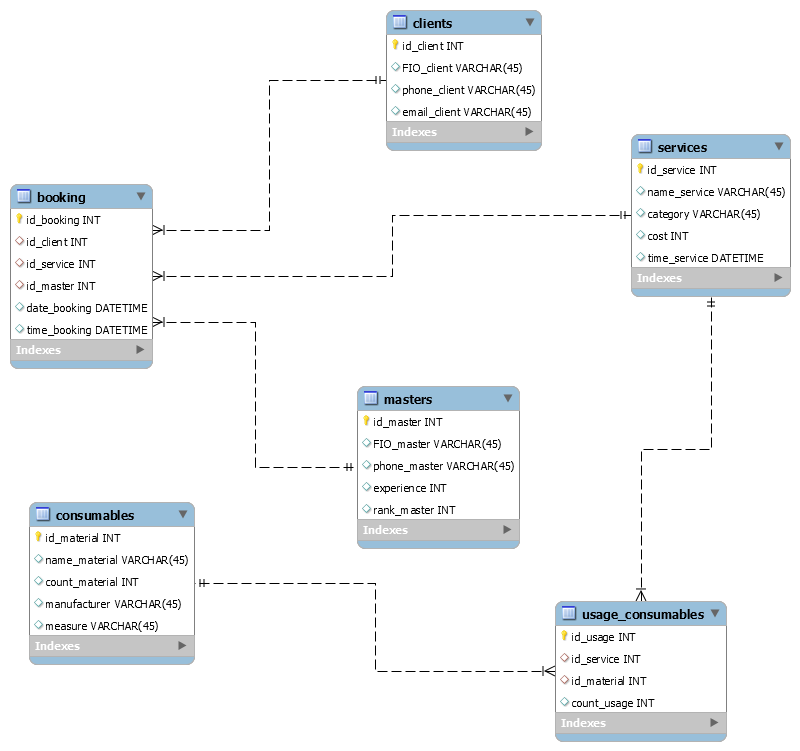


Рисунок 3.1 – Физическая модель базы данных

Код создания таблицы services: create table Services (id\_service int primary key auto\_increment, name\_service varchar(40), category varchar(20), cost int, time\_service time).

Код создания таблицы clients: create table Clients (id\_client int primary key auto\_increment, FIO\_client varchar(70), phone\_client varchar(15), email\_client varchar(50)).

Код создания таблицы masters: create table Masters (id\_master int primary key auto\_increment, FIO\_master varchar(40), phone\_master varchar(15), experience int, rank\_master int).

Код создания таблицы booking: create table Booking (id\_booking int primary key auto\_increment, date\_booking date, time\_booking time, id\_client int, id\_service int, id\_master int, foreign key (id\_client) references clients(id\_client) on delete cascade on update cascade, foreign key (id\_service) references services(id\_service) on delete cascade on update cascade, foreign key (id\_master) references masters(id\_master) on delete cascade on update cascade).

Код создания таблицы consumables: create table Consumables (id\_material int primary key auto\_increment, name\_material varchar(30), count\_material int, manufacturer varchar(30), measure varchar(5)).

Код создания таблицы usage\_consumables: create table UsageConsumables (id\_usage int primary key auto\_increment, id\_service int, id\_material int, count\_usage int, foreign key (id\_service) references services(id\_service) on delete cascade on update cascade, foreign key (id\_material) references consumables(id\_material) on delete cascade on update cascade).

1. ОПИСАНИЕ ПРОГРАММЫ
   1. Физическая структура

Автоматизированное рабочее место менеджера салона красоты содержит три страницы.

1. TablesView предназначена для просмотра всех таблиц базы данных, а так же для перехода на другие страницы.
2. TablesAdd предназначена для добавления и удаления записей во все таблицы базы данных, а так же для редактирования количества материалов на складе.
3. SelectTables предназначена для выборки информации из базы данных и просмотра отчета.
   1. Логическая структура

Процедуры и функции TablesIndex:

* customer\_search\_view – показывает таблицу «Клиенты»;
* visit\_search\_view – показывает таблицу «Запись клиентов»;
* employee\_search\_view – показывает таблицу «Мастера»;
* consumptionmaterial\_search\_view – показывает таблицу «Расходные материалы»;
* material\_search\_view – показывает таблицу «Расход материалов»;
* service\_search\_view – показывает таблицу «Услуги»;
* query\_add\_customer – открывает страницу query\_add и показывает панель с компонентами для добавления записей в таблицу «Запись клиентов»;
* query\_add\_ employee – открывает форму query\_add и показывает панель с компонентами для добавления записей в таблицу «Мастера»;
* query\_add\_ material – открывает форму query\_add и показывает панель с компонентами для добавления записей в таблицу «Расход материалов»;
* query\_add\_consumptionmaterial – открывает форму query\_add и показывает панель с компонентами для добавления записей в таблицу «Расходные материалы»;
* index\_view – открывает страницу index.php;
* delete\_and\_add\_visit\_customer – показывает или скрывает кнопки добавления, кроме записи клиентов;
* add\_service\_view\_and\_search – открывает страницу AddQuery и показывает панель с компонентами для добавления записей в таблицу «Услуги».

Процедуры и функции AddQuery:

* add\_service\_tables\_query\_and\_delete – добавляет запись в таблицу «Услуги»;
* add\_visit\_and\_delete\_visit – добавляет запись в таблицу «Запись клиентов»;
* add\_employee\_and\_delete\_employee – добавляет запись в таблицу «Мастера»;
* add\_customer\_and\_delete\_customer – добавляет запись в таблицу «Клиенты»;
* edit\_material\_and\_add – скрывает или показывает компоненты для добавления или редактирования материалов;
* add\_consumptionmaterial – добавляет запись в таблицу «Расходные материалы»;
* Update\_master – добавляет запись в таблицу или обновлять информацию «Расходные материалы»;
* edit\_consumptionmaterial – позволяет редактировать запись в таблице «Расходные материалы»;
* all\_delete\_tables– позволяет удалить записи из любой таблицы базы данных;
* add\_query\_consuptionmaterial– добавляет запись в таблицу «Расход материалов».

Процедуры и функции SelectTables:

* material\_group\_service– показывает материалы, необходимые для выполнения выбранной услуги;
* text\_for\_third\_query – показывает информацию о прибыли за текущий месяц;
* text\_for\_third\_query\_two – показывает информацию о прибыли за текущий год;
* text\_for\_third\_query\_user – показывает информацию о прибыли за выбранный год;
* query\_recorded\_today\_or\_year – показывает всех клиентов, записанных к определенному мастеру;
* search\_view\_customer\_and\_single\_customer – показывает базу клиентов.

1. ОТЛАДКА И ИСПЫТАНИЕ ПРОГРАММЫ
   1. Тестовые примеры

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, имеющий две различные цели:

* продемонстрировать разработчикам и заказчикам, что программа соответствует требованиям;
* выявить ситуации, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации.

Существующие на сегодня методы тестирования программного обеспечения не позволяют однозначно и полностью выявить все дефекты и установить корректность функционирования анализируемой программы, поэтому все существующие методы тестирования действуют в рамках формального процесса проверки исследуемого или разрабатываемого программного обеспечения. Такой процесс формальной проверки может доказать, что дефекты отсутствуют с точки зрения используемого метода. Существует множество подходов к решению задачи тестирования и верификации программного обеспечения, но эффективное тестирование сложных программных продуктов – это процесс в высшей степени творческий, не сводящийся к следованию строгим и чётким процедурам или созданию таковых.

Для данного проекта будет проведено модульное / юнит-тестирование для формы отображения в форме вывода таблиц на экран.

Также для данного проекта будет проведено модульное / юнит-тестирование для формы добавления и удаления:

1. Сохранение изменений для таблицы «Расходные материалы».
2. Проверка ввода повторяющихся значений.
3. Просмотр списка материалов, необходимых для выполнения заданной услуги.
4. Просмотр информации о прибыли за текущий месяц.
5. Просмотр информации о прибыли за текущий год.
6. Отображение статистики загруженности мастеров.
7. Создание базы клиентов.

В данном преддипломном проекте будет проведено дымовое тестирование:

1. Добавление записи клиента.
2. Добавление услуги.
   1. Анализ полученных результатов

Результат теста модульного / юнит тестирования формы отображения таблиц.

Тест 1 – проверка корректности отображения таблиц. Таблицы корректно отображаются без указания индексов записей. Результат изображен на рисунке 5.1.

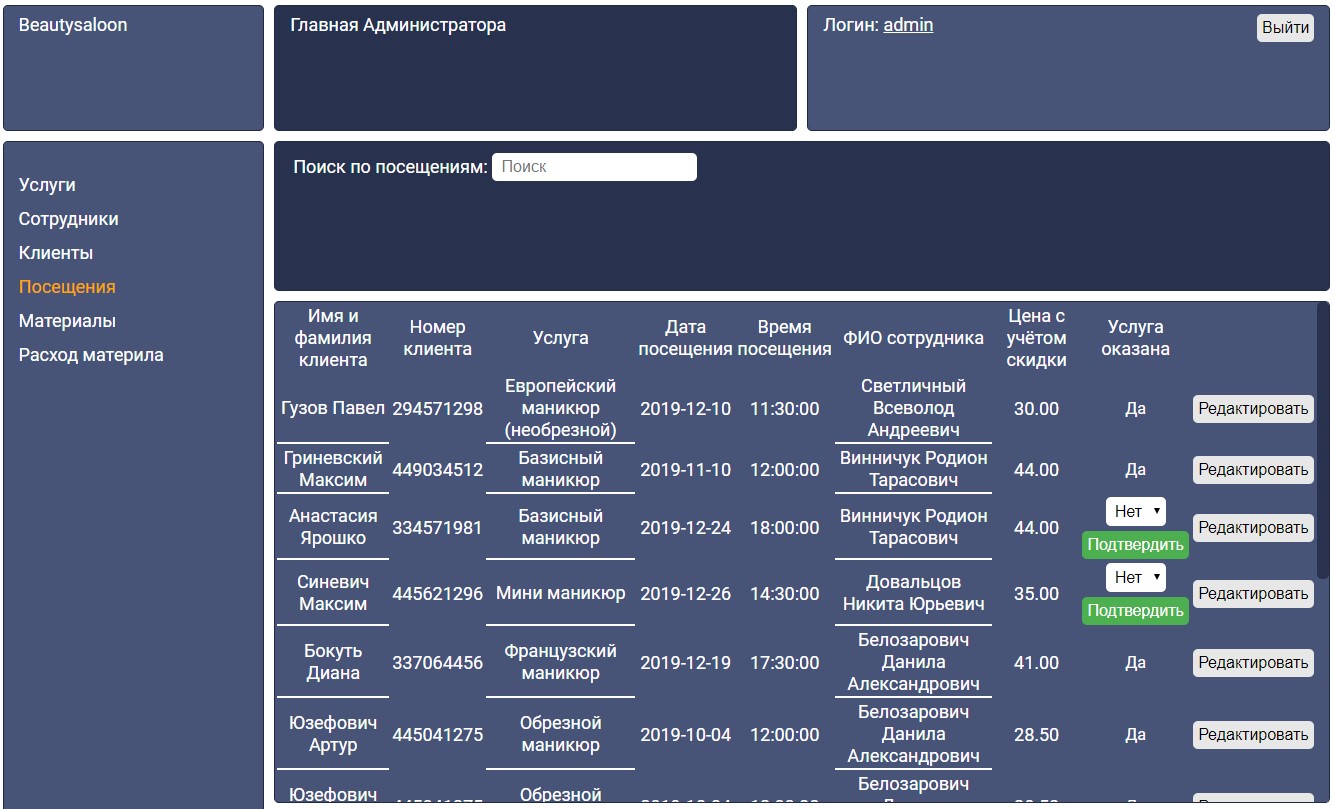


Рисунок 5.1 – Отображение информации о записи клиентов

Результаты тестов модульного / юнит тестирования для страницы добавления и удаления записей.

Тест 1 – сохранение изменений для таблицы «Расходные материалы». Пользователь выбирает определенный материал и меняет количество материала на складе. Результат показан на рисунке 5.2.

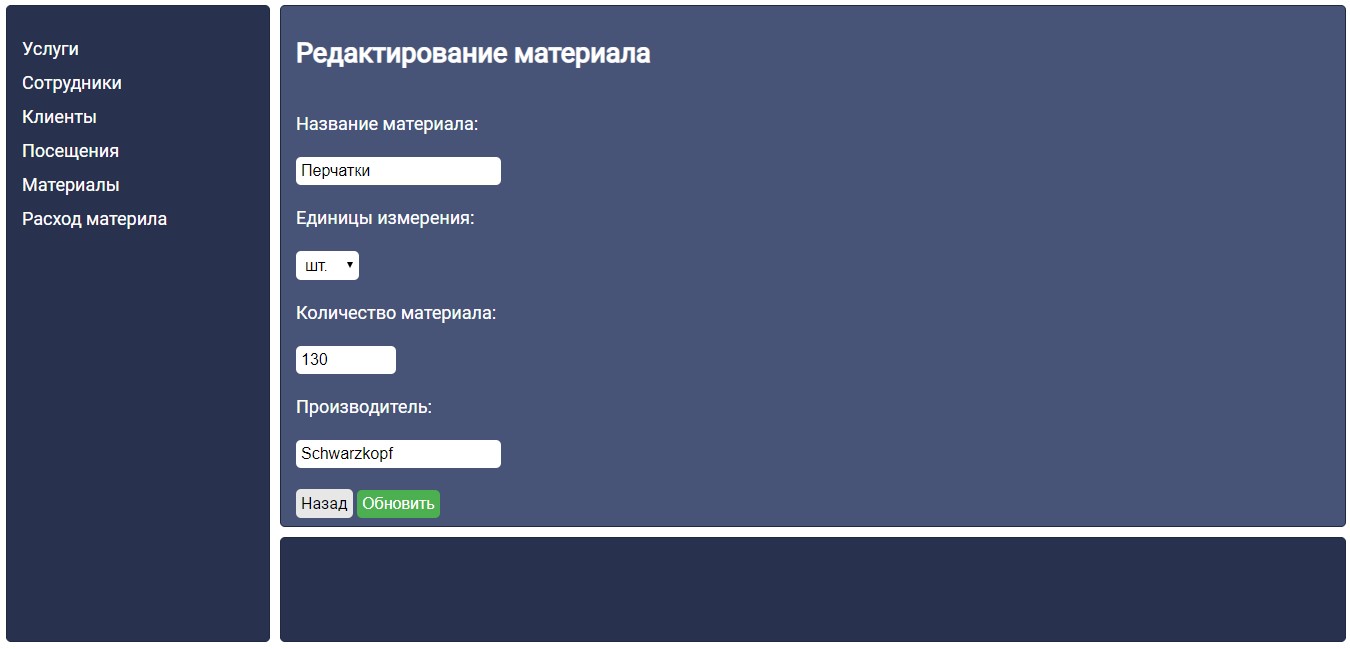


Рисунок 5.2 – Сохранение изменений для раздела «Расходные материалы»

Тест 2 – просмотр списка материалов, необходимых для выполнения заданной услуги. Пользователь выбирает услугу из выпадающего списка. Результат показан на рисунке 5.3.

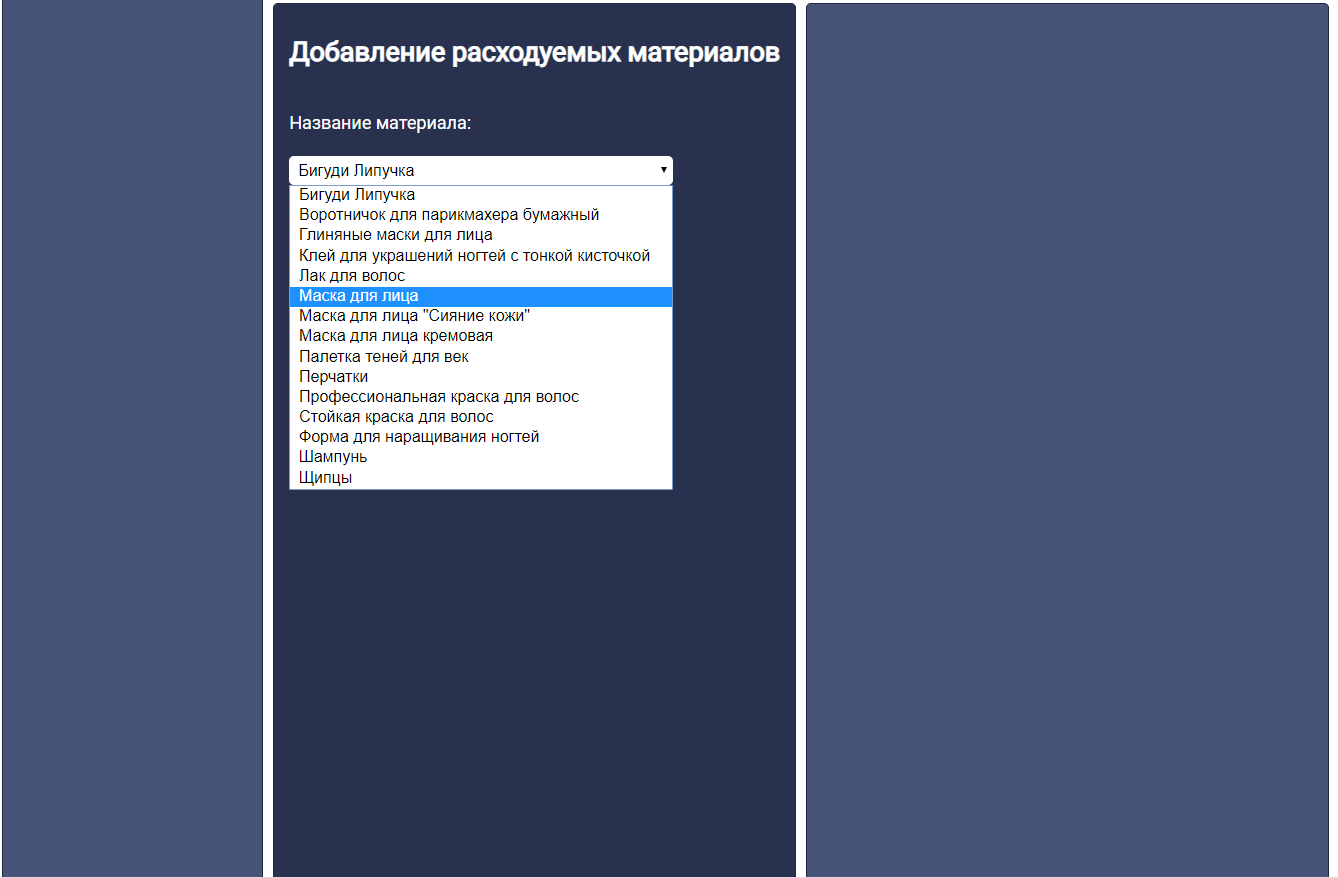


Рисунок 5.3 – Просмотр списка материалов для данной услуги

Тест 3 – просмотр информации о прибыли за текущий месяц. При нажатии кнопки «Прибыль за текущий месяц» пользователь получит информацию о клиентах, записанных на текущий месяц и о прибыли. Результат показан на рисунке 5.4.

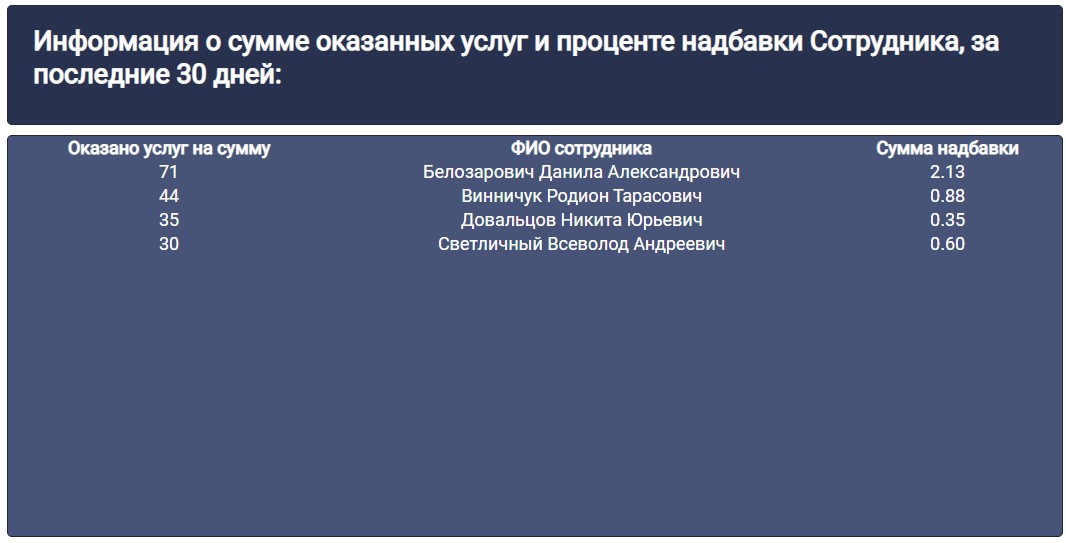


Рисунок 5.4 – Просмотр информации о прибыли за текущий месяц

Тест 4 – просмотр информации о прибыли за текущий год. При нажатии кнопки «Прибыль за текущий год» пользователь получит информацию о клиентах, записанных на текущий год и о прибыли. Результат теста показан на рисунке 5.5.

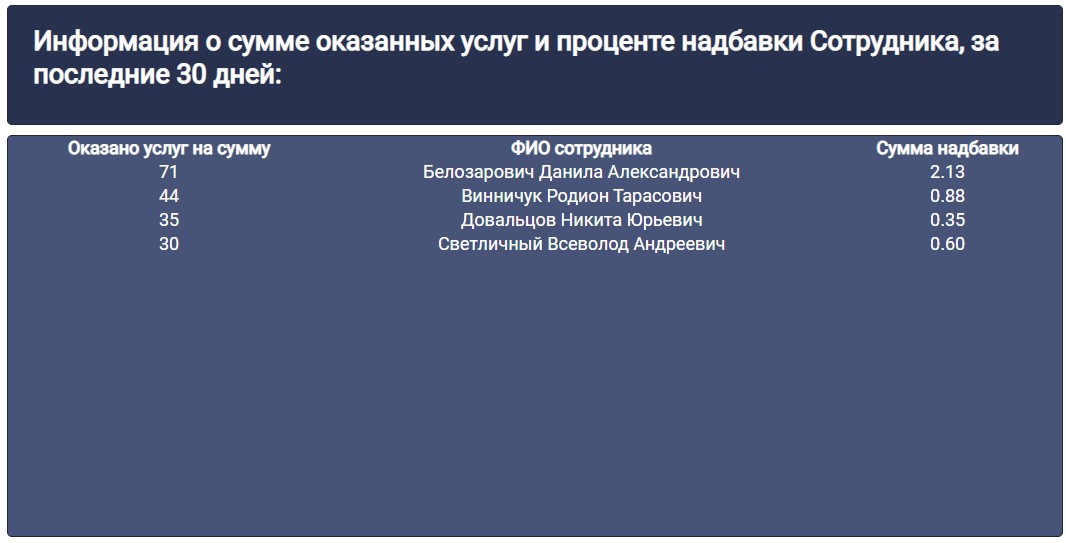
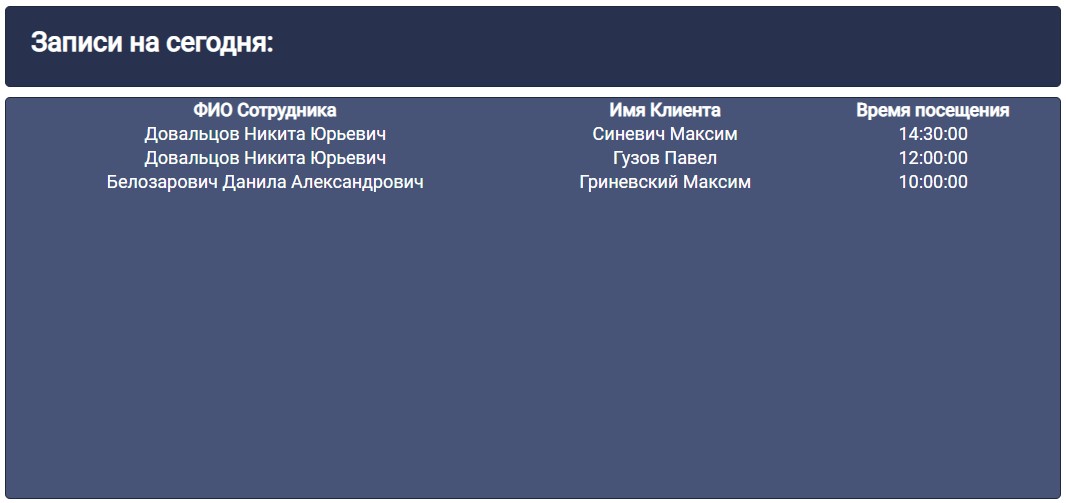


Рисунок 5.5 – Просмотр информации о прибыли за текущий год

Тест 5 – отображение статистики загруженности мастеров. Статистика загруженности мастеров показывается в виде диаграммы при входе на страницу SelectTables. Результат теста показан на рисунке 5.6.

Рисунок 5.6 – Статистика загруженности мастеров

Тест 6 – создание базы клиентов. База клиентов создается в странице отчетов и отображается на форме после нажатия на кнопку «База клиентов». Результат выполнения теста показан на рисунке 5.7.

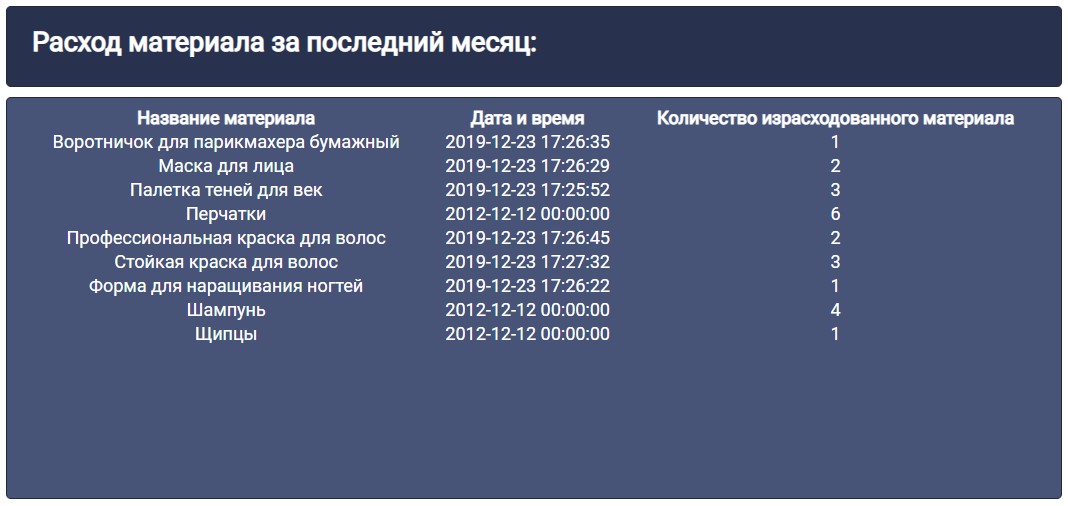


Рисунок 5.7 – Создание базы клиентов

Результаты дымового тестирования программы.

Тест 1 – добавление записи клиентаы. После заполнения всех полей на странице записи клиента, запись добавится в таблицы «Клиенты» и «Запись клиента». Результаты теста показаны на рисунках 5.8, 5.9 и 5.10.

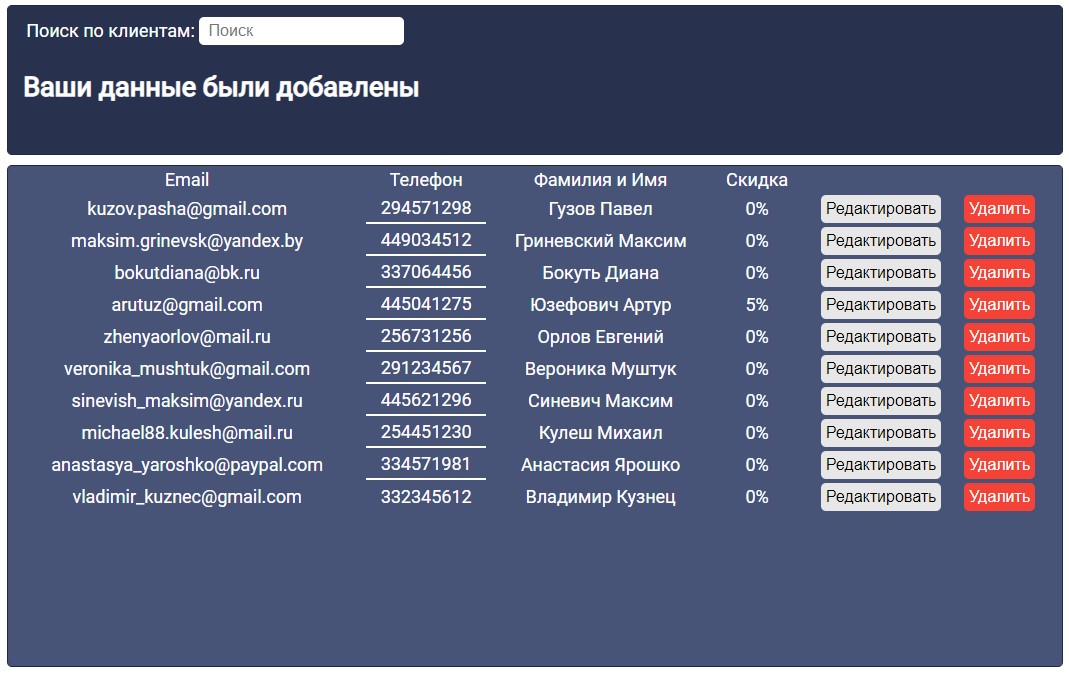


Рисунок 5.8 – Добавление клиента

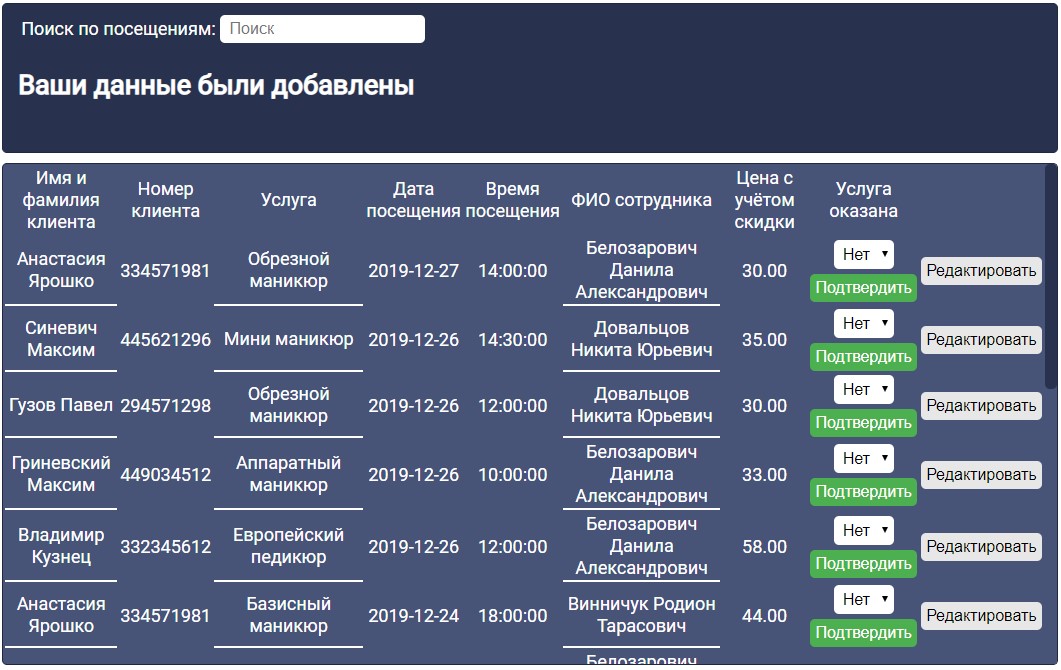


Рисунок 5.9 – Запись клиента

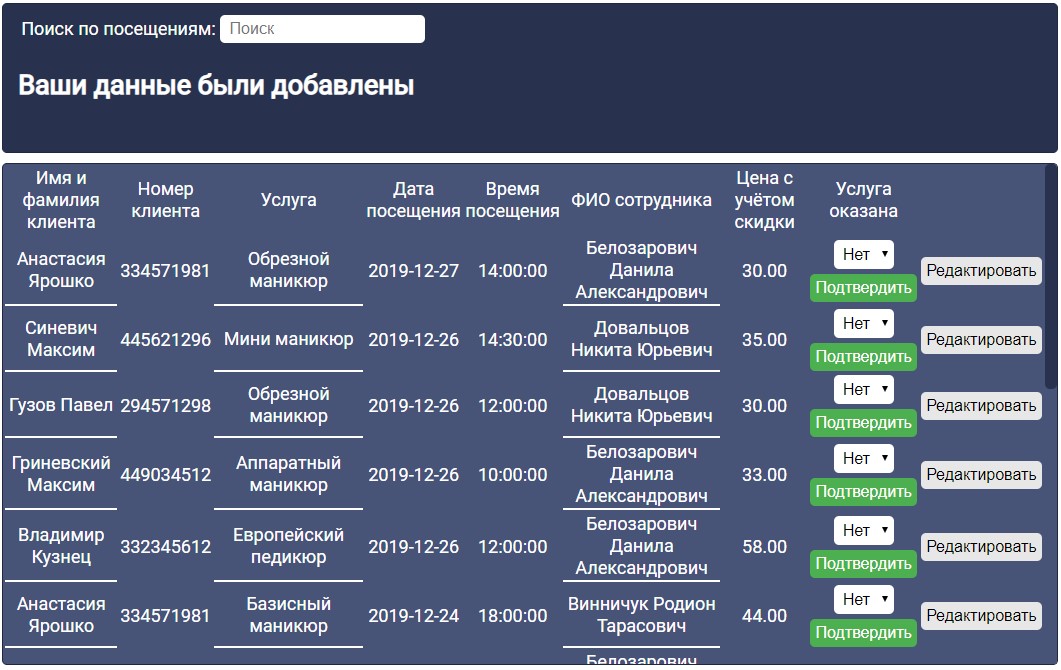


Рисунок 5.10 – Клиент записан

Тест 2 – добавление услуги. После заполнения всех полей на странице добавления услуг появится новая запись. Результаты теста показаны на рисунке 5.11.

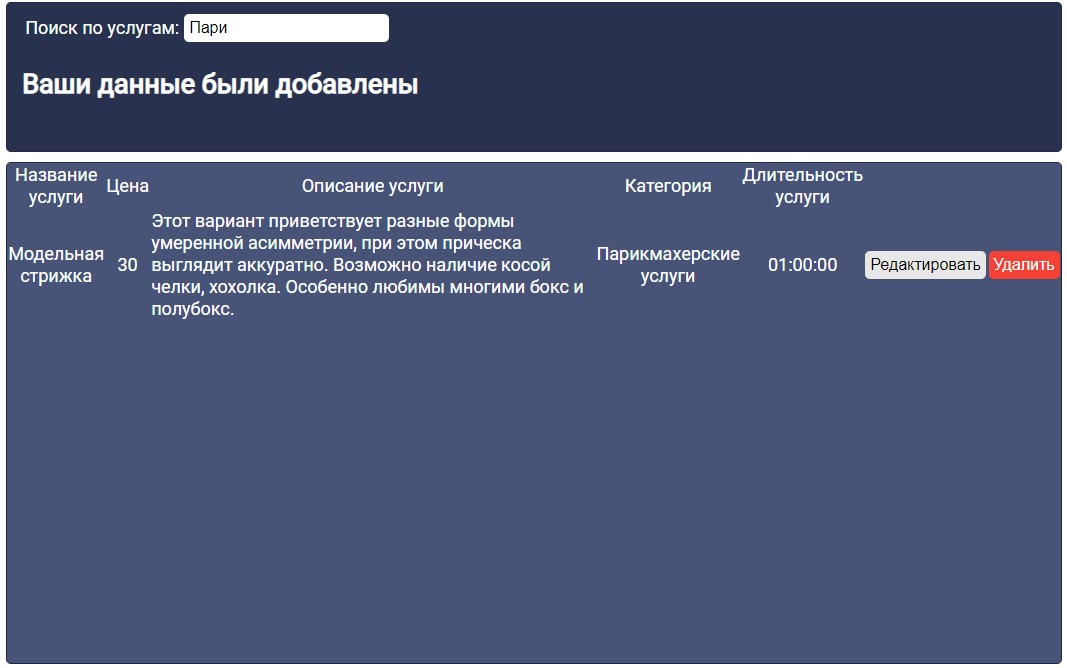


Рисунок 5.10 – Добавление услуги

1. ОПИСАНИЕ ПРИМЕНЕНИЯ ПРОГРАММЫ
   1. Назначение и условия применения

Автоматизированное рабочее место менеджера салона красоты предназначено для улучшения и большей эффективности работы менеджера салона красоты. Позволяет лучше отслеживать загруженность мастеров, видеть прибыль, вносить и корректировать запись клиентов.

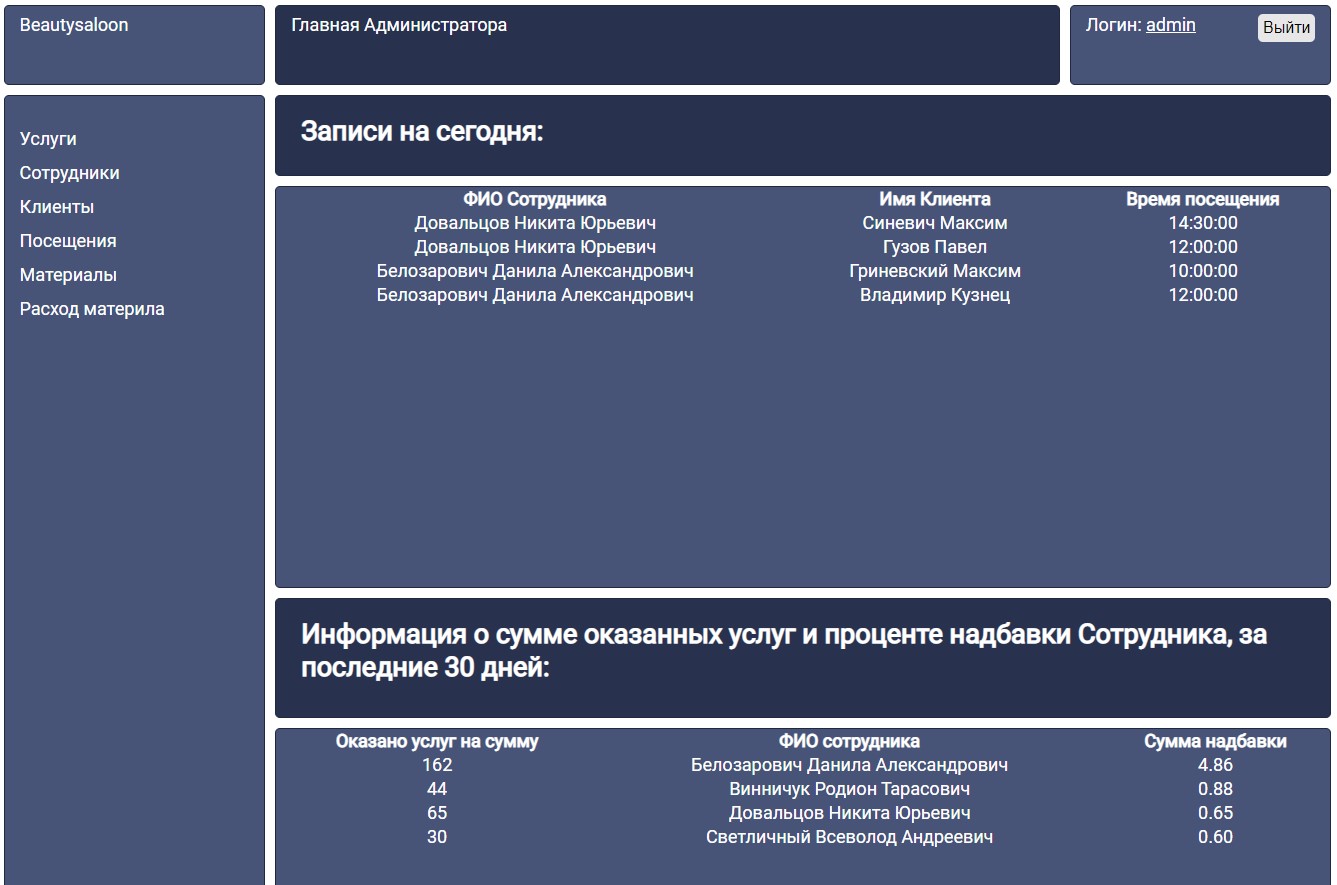
Для установки и запуска АРМ менеджера салона красоты необходимо запустить файл «index.php».

В качестве целевой системы для разработки преддипломного проекта выбрана Windows 10. Данная версия операционной системы семейства Windows сочетает в себе скорость и надежность работы, к тому же Windows 10 на сегодняшний момент является самой популярной операционной системой, что позволит расширить потенциальный круг применения программы.

Системные требования:

* операционная система Windows 7 и выше;
* процессор 1.8 ГГц;
* 256 МБ оперативной памяти;
* клавиатура, мышь.
  1. Описание работы программы

При запуске программы появляется страница, которая приветствует пользователя, а также даёт возможность отобразить информацию из базы данных или перейти на страницы добавления или выборки информации. Данная страница показана на рисунке 6.1.

 Рисунок 6.1 – Вход в программу

Для того, чтобы посмотреть информацию из разделов «Клиенты» или «Записи» или «Мастера» или «Расходные материалы» или «Расход материалов» или «Услуги» необходимо нажать соответствующие кнопки, как показано на рисунке 6.2.

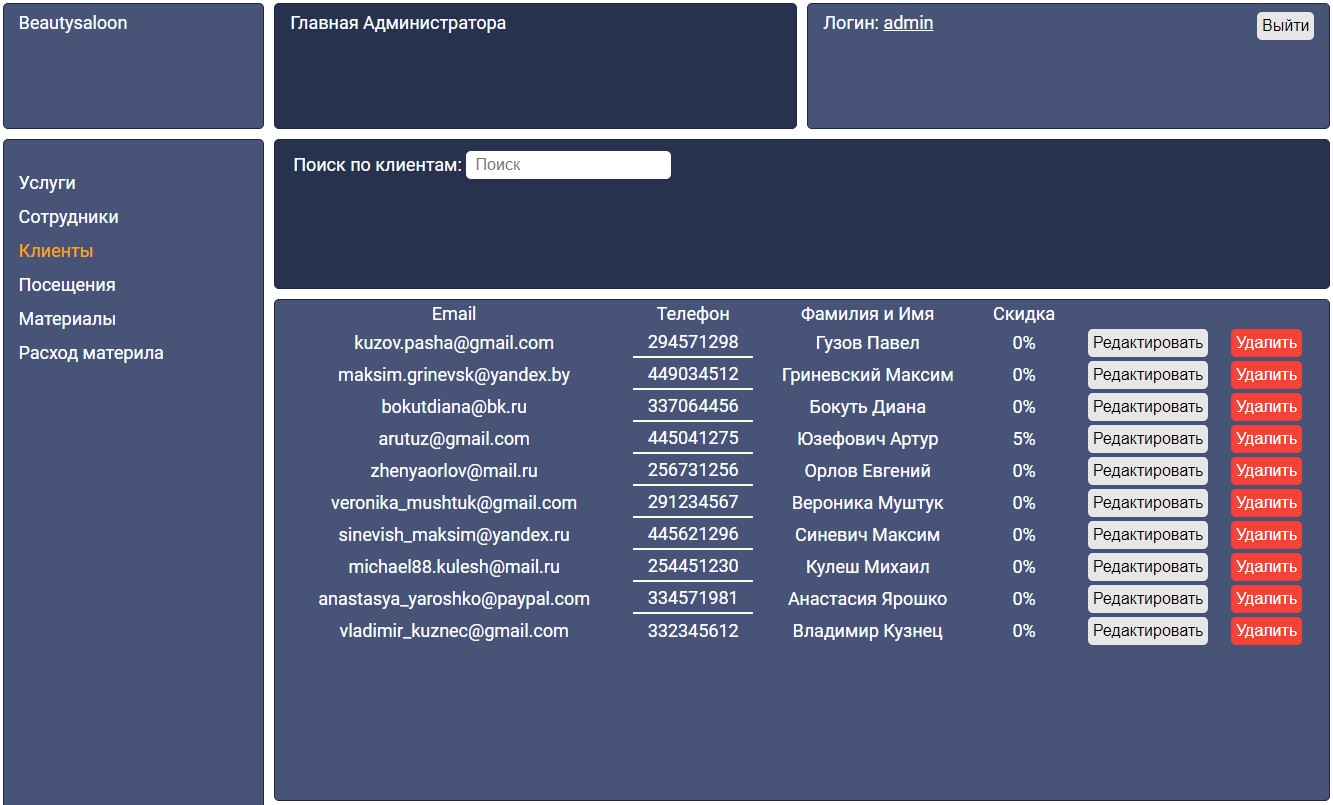


Рисунок 6.2 – Страницы отображения информации из базы данных

Для записи клиента необходимо нажать на кнопку «Запись клиента», как показано на рисунке 6.3.

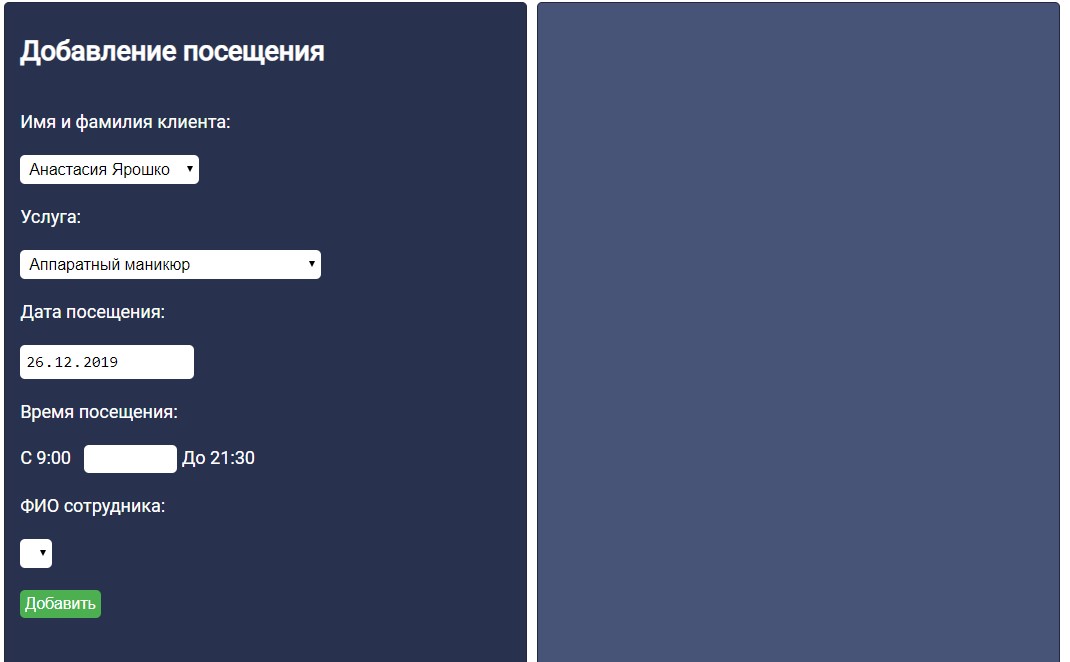


Рисунок 6.3 – Страница добавления записи клиента

Для того, чтобы добавить информацию в другие разделы, необходимо нажать на кнопку «Показать дополнительные кнопки добавления».

Для того, чтобы просмотреть информацию о загруженности мастеров или о прибыли за определенный промежуток времени, необходимо нажать на кнопку «Выборка», как показано на рисунке 6.5.

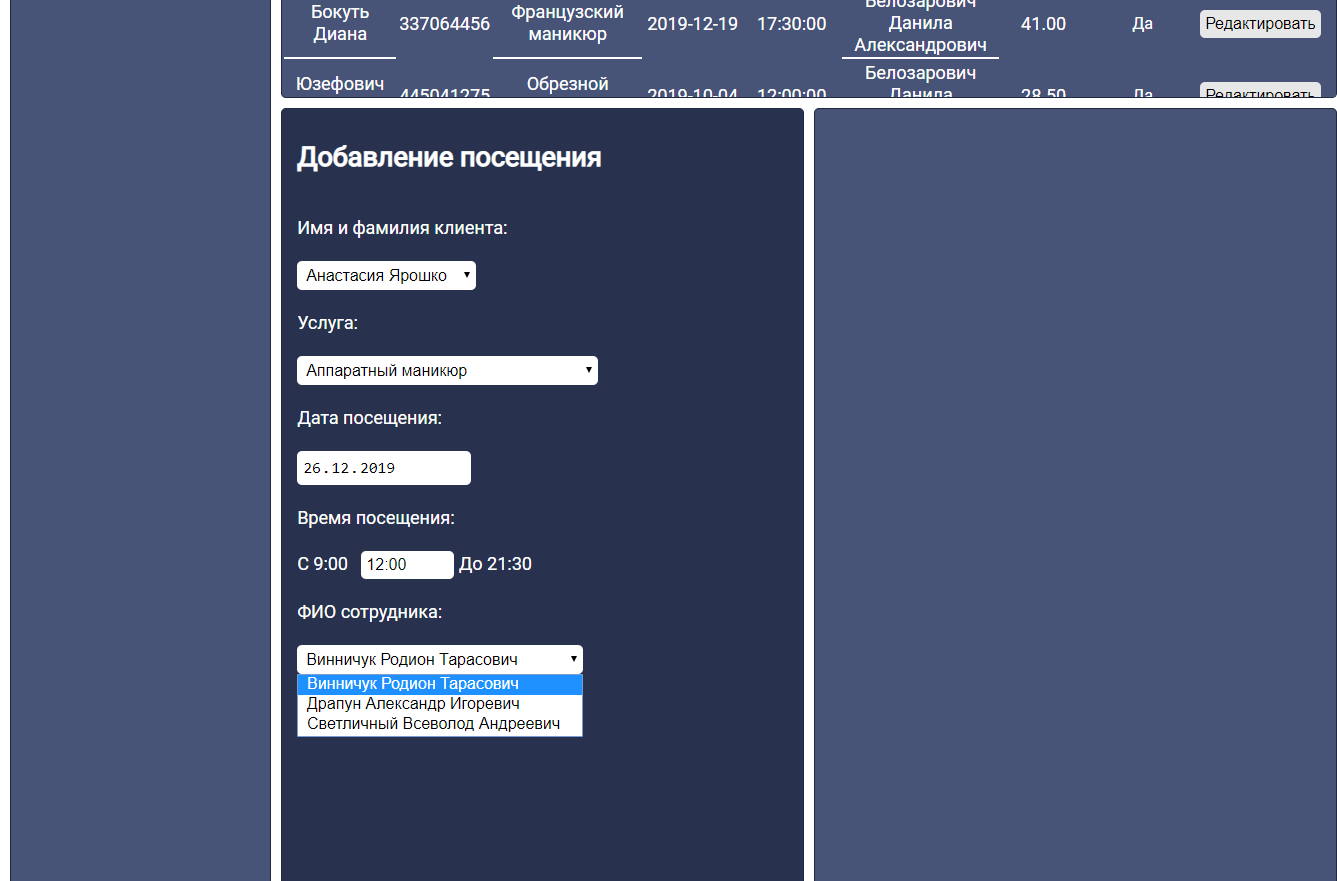


Рисунок 6.5 – Страница выборки

Чтобы просмотреть базу клиентов салона, необходимо нажать на кнопку «База клиентов», как показано на рисунке 6.6.

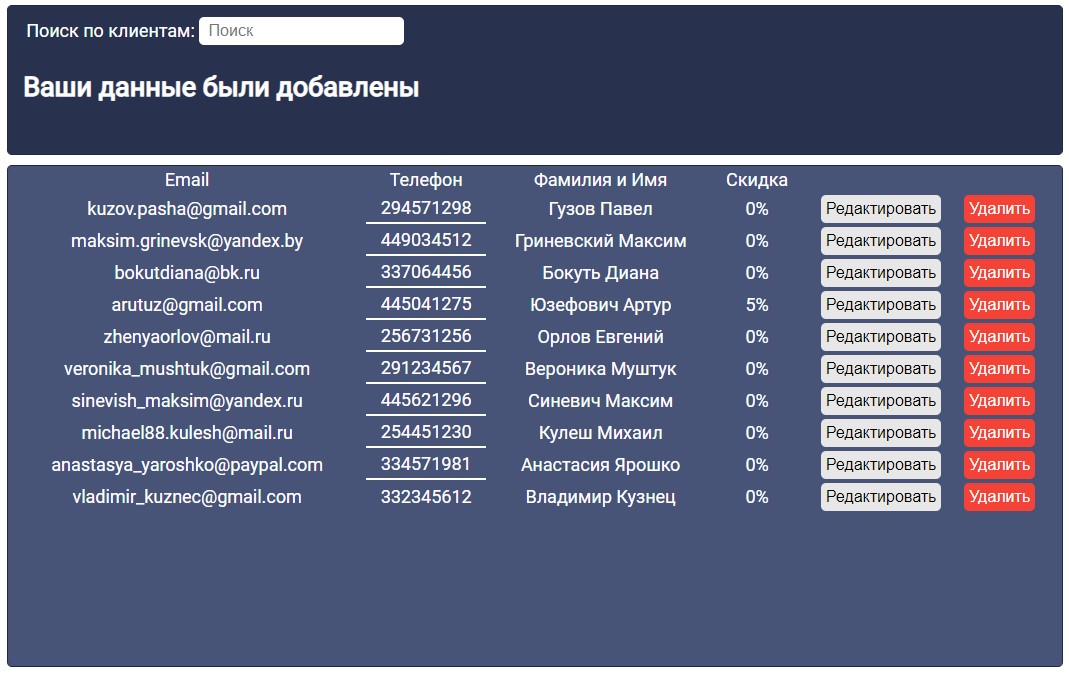


Рисунок 6.6 – Просмотр базы клиентов

ЗАКЛЮЧЕНИЕ

В проекте необходимо было реализовать веб-приложение АРМ менеджера салона красоты, где пользователь может записывать клиентов, добавлять услуги, просматривать необходимую информацию.

Для пользователя предусмотрены следующие функции:

* запись клиентов;
* добавление услуг;
* добавление мастеров;
* просмотр информации об услугах, о мастерах, о клиентах, о расходных материалах, о расходе материалов на одну услугу и о записи клиентов;
* просмотр отчетов о прибыли за любой промежуток времени;
* корректировка записей в материалах;
* просмотр услуг и материалов, необходимых для их выполнения;
* просмотр информации о загруженности мастеров;
* просмотр базы клиентов.

Программа была разработана в среде разработки Visual Code 2017, с использованием технологии ASP.NET.

При разработке интерфейса был использован графический редактор Adobe Photoshop. В нем были разработаны кнопки и фон программы.

Имеются возможности для дальнейшей модификации проекта такие как улучшение интерфейса, добавление возможности использования программы через MySql Cloud Servise.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Виктор Гольцман MySQL 5.0 2009. – Питер, – 320c.
2. Гультяев А. Разработка справочных систем. – Питер, 2004. – 420с.
3. Корнилов Е.Н. Программирование шахмат и других логических игр. – [Москва, 2005. – 272 стр.](http://whychess.ru/560rpogrammirovanie-shahmat-i-logicheskih_igr.html)
4. Леонтьев В. Windows 10: новейший самоучитель для компьютеров и планшетов. – Москва: Издательство «Э», 2017. – 2017. – 416с.
5. Либерти Д. Язык программирования PHP - Санкт-Петербург. – 2003: Символ-Плюс. – С. 26. – ISBN 5-93286-038-3 – 688 с.
6. Макки А. Введение в .NET 4.0 и Visual Code 2017 для профессионалов. – «Вильямс», 2010. – ISBN 978-5-8459-1639-6 – 416с.
7. Синицын С. В. Налютин Н. Ю. Верификация программного обеспечения. - М. БИНОМ, 2008. – ISBN 978-5-94774-825-3 – 368 с.
8. Шилдт Г. Полный справочник по PHP = PHP: The Complete Reference. ­- М.: Издательский дом «Вильямс», 2004. – С. 26 - 27. - ISBN 5-8459-0563-X. – 752 с.
9. Microsoft. Характеристики и системные требования Windows 10. – М. БИНОМ, 2008. – ISBN 978-3-92721-965-3 – 473с.

Приложение А

(обязательное)

Листинг программы

<?php

session\_start();

require\_once "start.php";

if(isset($\_POST["button\_exit"])){

session\_destroy();

header('Location: /admin/auth.php');

exit;

}

$servername = "localhost";

$username = "root";

$password = "";

$database = "beautysaloon";

$connect = mysqli\_connect($servername, $username, $password, $database);

$output = '';

$query = '';

?>

<!DOCTYPE html>

<html lang="ru">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<link rel="stylesheet" href="/admin/style.css">

<title>Админ панель</title>

</head>

<body>

<div class="grid\_main\_admin">

<?php

require\_once "block/asidepanel.php";

require\_once "block/header.php";

?>

<article class="text\_for\_first\_query">

<h2 style="margin:10px;">Записи на сегодня:</h2>

</article>

<article class="query\_recorded\_today container\_table\_view">

<?php

$query = "SELECT Employee.full\_name, Customer.first\_and\_last\_name, Visit.time\_of\_visit

FROM Employee INNER JOIN Customer INNER JOIN Visit

ON Employee.id\_employee = Visit.id\_employee

AND Customer.id\_customer = Visit.id\_customer WHERE Visit.date\_of\_visit = Date(Now())";

$result = mysqli\_query($connect, $query);

if(mysqli\_num\_rows($result) > 0)

{

$output .= '

<table id="myTable" class="query\_recorder\_today\_table query\_table">

<thead>

<tr id="point\_cursor">

<th onclick="sortTable(0)">ФИО Сотрудника</th>

<th onclick="sortTable(1)">Имя Клиента</th>

<th onclick="sortTable(2)">Время посещения</th>

</tr>

</thead>

<tbody>

';

while($row = mysqli\_fetch\_array($result))

{

$output .= '

<tr>

<td>'.$row["full\_name"].'</td>

<td>'.$row["first\_and\_last\_name"].'</td>

<td>'.$row["time\_of\_visit"].'</td>

</tr>

';

}

$output .= ' </tbody></table>';

echo $output;

}

else

{

echo '<h2 style="margin:20px;">На сегодня записей нет</h2>';

}

?>

</article>

<article class="text\_for\_second\_query">

<h2 style="margin:10px;">Информация о сумме оказанных услуг и проценте надбавки Сотрудника, за последние 30 дней:</h2>

</article>

<article class="query\_rendered\_service container\_table\_view">

<?php

$query = "SELECT Sum(Service.price) as SumPrice, Employee.full\_name, round(Sum(Service.price) \* Employee.bonus\_percentage / 100,2) as Allowance FROM

Customer INNER JOIN Service INNER JOIN Employee INNER JOIN Visit

ON Customer.id\_customer = Visit.id\_customer AND Service.id\_service = Visit.id\_service AND Employee.id\_employee = Visit.id\_employee

WHERE TO\_DAYS(NOW()) - TO\_DAYS(Visit.date\_of\_visit) <= 30 group by Employee.full\_name";

$result = mysqli\_query($connect, $query);

$output ='';

if(mysqli\_num\_rows($result) > 0)

{

$output .= '

<table id="myTable" class="query\_recorder\_today\_table query\_table">

<thead>

<tr id="point\_cursor">

<th onclick="sortTable(0)">Оказано услуг на сумму</th>

<th onclick="sortTable(1)">ФИО сотрудника</th>

<th onclick="sortTable(2)">Сумма надбавки</th>

</tr>

</thead>

<tbody>

';

while($row = mysqli\_fetch\_array($result))

{

$output .= '

<tr>

<td>'.$row["SumPrice"].'</td>

<td>'.$row["full\_name"].'</td>

<td>'.$row["Allowance"].'</td>

</tr>

';

}

$output .= ' </tbody></table>';

echo $output;

}

else

{

echo '<h2 style="margin:10px;">Нету информации</h2>';

}

?>

</article>

<article class="text\_for\_third\_query">

<h2 style="margin:10px;">Расход материала за последний месяц:</h2>

</article>

<article class="query\_consumption\_material">

<?php

$query = "SELECT Material.name\_material, Consumptionmaterial.date\_consumption , sum(Consumptionmaterial.consumed\_quantity) as quantity

from material inner join Consumptionmaterial on

Consumptionmaterial.id\_material = Material.id\_material where month(Consumptionmaterial.date\_consumption)=month(now()) group by Material.name\_material";

$result = mysqli\_query($connect, $query);

$output ='';

if(mysqli\_num\_rows($result) > 0)

{

$output .= '

<table id="myTable" class="query\_recorder\_today\_table query\_table">

<thead>

<tr id="point\_cursor">

<th onclick="sortTable(0)">Название материала</th>

<th onclick="sortTable(1)">Дата и время</th>

<th onclick="sortTable(2)">Количество израсходованного материала</th>

</tr>

</thead>

<tbody>

';

while($row = mysqli\_fetch\_array($result))

{

$output .= '

<tr>

<td>'.$row["name\_material"].'</td>

<td>'.$row["date\_consumption"].'</td>

<td>'.$row["quantity"].'</td>

</tr>

';

}

$output .= ' </tbody></table>';

echo $output;

//mysqli\_close($connect);

}

else

{

echo '<h2 style="margin:10px;">Нету информации</h2>';

}

?>

</article>

<style>

.container\_table\_view {

height: 400px;

padding: 0px;

overflow: auto;

}

.query\_recorder\_today\_table>tbody>tr>td:nth-child(1) {

text-align: center;

}

.query\_recorder\_today\_table>tbody>tr>td:nth-child(2) {

text-align: center;

}

.query\_recorder\_today\_table>tbody>tr>td:nth-child(3) {

text-align: center;

}

</style>

<?php

require\_once "block/asidetable.php";

require\_once "block/footer.php";

?>

</div>

</body>

</html>

<?

session\_start();

require\_once "start.php";

$connect = mysqli\_connect("localhost", "root", "", "beautysaloon");

if(isset($\_POST['add\_service']))

{

    $name\_service = mysqli\_real\_escape\_string($connect, trim($\_POST['name\_service']));

    $price = mysqli\_real\_escape\_string($connect, trim($\_POST['price']));

    $description\_service = mysqli\_real\_escape\_string($connect, trim($\_POST['description\_service']));

    $category = mysqli\_real\_escape\_string($connect, trim($\_POST['category']));

    $duration\_service = mysqli\_real\_escape\_string($connect, trim($\_POST['duration\_service']));

    $query\_name\_service = "SELECT \* FROM service WHERE name\_service = '$name\_service'";

    $result\_name\_service = mysqli\_query($connect, $query\_name\_service);

    if(mysqli\_num\_rows($result\_name\_service) == 0) {

        $query ="INSERT INTO service (name\_service, price, description\_service, category, duration\_service)

                 VALUES ( '$name\_service', '$price', '$description\_service', '$category', '$duration\_service')";

        $result = mysqli\_query($connect, $query);

        if($result)

        {

            $\_SESSION['success'] = "Ваши данные были добавлены";

            header('Location: /admin/service/service.php');

        }

        else

        {

            $\_SESSION['status'] = "Ваши данные не были добавлены";

            header('Location: /admin/service/service.php');

        }

    }

    else{

        $\_SESSION['success'] = "Услуга с таким названием уже существует";

        header('Location: /admin/service/service.php');

    }

}

<?php

session\_start();

require\_once "../start.php";

    $connect = mysqli\_connect("localhost", "root", "", "beautysaloon");

    $output = '';

    $query = '';

    error\_reporting(0);

    if(isset($\_POST["time"]) && isset($\_POST["date"]))

    {

        $time = mysqli\_real\_escape\_string($connect, $\_POST["time"]);

        $date = mysqli\_real\_escape\_string($connect, $\_POST["date"]);

        $query = "SELECT full\_name

        from Employee

        where full\_name not in

        (

            (select full\_name from Employee inner join Visit

                on Employee.id\_employee = Visit.id\_employee

                where  Visit.date\_of\_visit = '$date' and Visit.time\_of\_visit = '$time'

                group by full\_name))

        group by full\_name";

        $result = mysqli\_query($connect, $query);

        while ($row = mysqli\_fetch\_row($result)) {

            echo "<option>$row[0]</option>";

        }

    }