**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

|  |  |
| --- | --- |
| КР.09.02.07.22.201.08 ПЗ |  |

ИНФОРМАЦИОННАЯ СИСТЕМА

«СЕРВИСНЫЙ ЦЕНТР ПО РЕМОНТУ БЫТОВОЙ ТЕХНИКИ»

|  |  |  |
| --- | --- | --- |
| Председатель ВЦК: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (К.А. Дивонин) |

Иркутск 2022

**Содержание**

[ВВЕДЕНИЕ 3](#_Toc121248092)

[1. Описание предметной области ИС 4](#_Toc121248093)

[2. Анализ инструментальных средств разработки. 6](#_Toc121248094)

[3. Техническое задание 16](#_Toc121248095)

[4. Проектирование ИС 17](#_Toc121248096)

[4.1. Структурная схема ИС 17](#_Toc121248097)

[4.2 Функциональная схема ИС 19](#_Toc121248098)

[4.3 Проектирование базы данных 22](#_Toc121248099)

[4.4. Проектирование интерфейса 27](#_Toc121248100)

[5. Разработка ИС 30](#_Toc121248101)

[5.1. Разработка интерфейса ИС 30](#_Toc121248102)

[6. Технологическая документация ИС 40](#_Toc121248103)

[6.1 Руководство пользователя ИС 40](#_Toc121248104)

[ЗАКЛЮЧЕНИЕ 42](#_Toc121248105)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 43](#_Toc121248106)

[Приложение А – Техническое задание 44](#_Toc121248107)

[Приложение Б – Листинг roles.php 49](#_Toc121248108)

ВВЕДЕНИЕ

Сервисный центр — организация, занимающаяся оказанием услуг по сервисной поддержке и обслуживанию техники, оборудования и другой продукции. Сервисных центров в России становится всё больше из-за увеличения количества различной бытовой техники и её количества. Главная задача такой организации - устранить дисфункцию аппарата и не допустить новых, обеспечить все необходимое для того, чтобы аппарат выполнял свою функцию у владельца как можно дольше. Предоставляет гарантийный и послегарантийный ремонт электрических устройств, обеспечивает поддержку пользователей.

В современных сервисных центрах такие функции как прием заявок, регистрация заказов, прослеживание истории заказов должны быть автоматизированы для облегчения работы сотрудников, а также для удобства заказчиков.

Целью курсовой работы является создание информационной системы «Сервисный центр по ремонту бытовой техники».

Для достижения конечного результата, а именно создание информационной системы (далее - ИС) «Сервисный центр по ремонту бытовой техники», необходимо решить следующие задачи:

* описать предметную область;
* провести анализ инструментальных средств разработки;
* составить техническое задание;
* выполнить проектирование ИС «Сервисный центр»;
* разработать ИС «Сервисный центр по ремонту бытовой техники»;
* разработать интерфейс ИС;
* разработать базу данных ИС;
* написать руководство пользователя.

1. Описание предметной области ИС

Предметной областью курсовой работы является сервисный центр по ремонту бытовой техники.

Сервисный центр — организация, занимающаяся оказанием услуг по сервисной поддержке и обслуживанию техники, оборудования и другой продукции. Он имеет штат работников и пользователей. В первую очередь администратор составляет список оказываемых услуг.

Сервисный центр специализируется на ремонте бытовой техники различных производителей, а также предоставляет услуги по ремонту бытовой техники, находящейся на гарантии, услуги по послегарантийному ремонту и диагностику неисправностей.

Доставка техники в сервисный центр осуществляется клиентом. Каждому клиенту при приеме техники в ремонт выдается акт приема-передачи товара, в котором описывается поломка, указывается наименование компьютера, его серийный номер, дата сдачи компьютера в ремонт. Акт подписывается заказчиком и исполнителем. В случае если бытовая техника находится на гарантии, клиент обязан предъявить гарантийный талон.

Ремонт техники начинается с диагностических работ, определяется характер неисправностей, после которых мастер перезванивает клиенту и оговаривает срок, а также стоимость необходимых работ и запчастей. В случае если клиента не устраивают какие-либо моменты, он может отказаться от проведения ремонта. Только после получения согласия клиента мастер производит ремонт техники.

Ремонт техники выполняется высококвалифицированными специалистами. Отслеживать процесс ремонта можно, позвонив в сервисный центр. После окончания ремонтных работ мастер или менеджер перезванивают клиенту и сообщают об окончании ремонтных работ и уточняют дату, когда клиент может забрать технику.

При получении компьютера из ремонта клиент предъявляет акт приема-передачи товара, после чего оплачивает все оговоренные с мастером работы. На каждое отремонтированное изделие дается гарантийная поддержка.

Для информационного обслуживания сервисного центра создается база данных, которая содержит следующие данные:

1. Сведения о технике, которая была сдана в ремонт.

2. Сведения о причине поломки.

3. Сведения о мастере, который проводил диагностику поломки и ремонт техники.

4. Сведения о проделанных, определенным мастером, работах, с указанием времени начала и конца ремонта.

5. Стоимость услуг для каждого клиента.

База данных строится с учетом следующих особенностей:

1. Каждый клиент может принести в ремонт несколько единиц техники.

2. Каждая техника может иметь несколько поломок.

3. В сервисном центре работают сотрудники, специализирующиеся на нескольких видах работ, поэтому один мастер может принять несколько заказов.

4. Ремонт техники, находящейся на гарантии, проводится бесплатно.

С базой данных могут работать администраторы, для которых доступны следующие задачи:

1. Добавление, редактирование и удаление сведений о технике, которая была сдана в ремонт.

2. Добавление, редактирование и удаление сведений о сотрудниках сервисного центра.

3. Добавление, редактирование и удаление сведений об услугах, предоставляемых сервисным центром.

4. Изменение цен на услуги, предоставляемые сервисным центром.

2. Анализ инструментальных средств разработки.

Инструменты разработки программного продукта определяют будущий результат.

Проектировать структуру веб-приложения удобно через MySQL Workbench и Draw.io, а его дизайн – через онлайн-сервис Figma. Веб-приложение будет состоять из двух частей – клиентская и серверная. Для реализации клиентской части отлично подойдут следующие инструменты: HTML5, CSS3 и JS. Серверная часть будет действовать на PHP со стандартной базой данных MySQL(MariaDB).

MySQL Workbench — инструмент для визуального проектирования баз данных, интегрирующий проектирование, моделирование, создание и эксплуатацию БД в единое бесшовное окружение для системы баз данных MySQL. В проекте используется для создания ER-диаграммы БД.

Draw.io — это удобное бесплатное онлайн-приложение для создания диаграмм для рабочих процессов, BPM, организационных, сетевых диаграмм, блок-схемм (флоучарты), UML и принципиальных электросхем. В проекте используется для создания прототипа страниц.

Figma – бесплатный удобный онлайн-сервис для дизайнеров, веб-разработчиков и маркетологов. Он предназначен для создания прототипов сайтов или приложений, иллюстраций и векторной графики. В редакторе можно настроить совместную работу, вносить и обсуждать правки, причём как в браузере, так и через приложение на компьютере. Популярен, в частности, для разработки прототипа и дизайна сайта или приложения. В проекте используется для создания наглядного ожидаемого дизайна проекта, а также уникальных SVG-элементов в едином стиле.

HTML – язык разметки гипертекста. Язык разметки дает браузеру необходимые инструкции о том, как отображать тексты и другие элементы страницы на мониторе. Язык HTML интерпретируется браузерами и отображается в виде документа, в удобной для человека форме.

CSS – каскадные таблицы стилей, которые используются для определения стилей (правил) оформления документов — включая дизайн, вёрстку и вариации макета для различных устройств и размеров экрана.

JavaScript – это полноценный динамический язык программирования, который применяется к HTML-документу и может обеспечить динамическую интерактивность. JavaScript является объектно-ориентированным языком, но используемое в языке прототипирование обуславливает отличия в работе с объектами по сравнению с традиционными класс-ориентированными языками. Кроме того, JavaScript имеет ряд свойств, присущих функциональным языкам — функции как объекты первого класса, объекты как списки, карринг, анонимные функции, замыкания — что придаёт языку дополнительную гибкость.

Веб-приложение будет содержать в себе информацию – её необходимо хранить, изменять, структурировать и использовать. Это реализуется благодаря базе данных. Были рассмотрены следующие варианты реализации СУБД:

1. MySQL.

2. SQLite.

3. PostgreSQL.

MySQL – свободная реляционная система хранения и управления базами данных. Разработку и поддержку MySQL осуществляет корпорация Oracle, получившая права на торговую марку вместе с поглощённой Sun Microsystems, которая ранее приобрела шведскую компанию MySQL AB. Продукт распространяется как под GNU General Public License, так и под собственной коммерческой лицензией. Помимо этого, разработчики создают функциональность по заказу лицензионных пользователей. Именно благодаря такому заказу почти в самых ранних версиях появился механизм репликации.

На сегодняшний день является самой популярной серверной базы данных (далее – БД), за счёт своей простоты, скорости работы и внушительного функционала. Поддерживаются такие основные движки MyISAM, InnoDB, MEMORY, Berkeley DB. Реализация всех новых возможностей стандарта SQL отсутствует в пользу простоты использования.

PostgreSQL – свободная объектно-реляционная система хранения и управления базами данных. Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS, Solaris/OpenSolaris, Tru64, QNX, а также для Microsoft Windows.

Работает только на одном движке – Storage Engine. Все таблицы представлены в виде объектов, они могут наследоваться, а все действия с таблицами выполняются с помощью объектно-ориентированных функций. Обладает открытым исходным кодом, разрабатывается командой энтузиастов, при этом старается максимально соответствовать стандарту SQL. Реализует все самые новые стандарты, что приводит к ущербу простоты, из-за чего PostgreSQL очень сложный и уступает в популярности MySQL.

SQLite — компактная встраиваемая СУБД с исходным кодом. В 2005 году проект получил награду Google-O’Reilly Open Source Awards. SQLite поддерживает динамическое типизирование данных. Возможные типы значений: INTEGER, REAL, TEXT и BLOB. Также поддерживается специальное значение NULL. Размеры значений типа TEXT и BLOB не ограничены ничем, кроме константы SQLITE\_MAX\_LENGTH в исходниках sqlite, равной миллиарду.

SQLite напрямую хранит информацию в одном файле, что облегчает его копирование. Большая популярность в мобильной разработке и небольших автономных приложениях, поскольку она занимает меньше места на дисковом пространстве, имеет высокую скорость работы и не требует в отличии от MySQL не требует наличие сервера для запуска. Минусы: ограничения на запись, всего 5 типов данных, отсутствие встроенного механизма аутентификации.

Для наглядности сравнения вариантов реализации базы данных была составлена таблица 1.

Таблица 1 – Сравнение средств реализации базы данных

|  |  |  |  |
| --- | --- | --- | --- |
| Название БД | MySQL | SQLite | PostgreSQL |
| Большое кол-во типов данных | + | - | + |
| Популярность | + | + | - |
| Отказоустойчивость | - | - | + |
| Не требует удаленного сервера | - | + | - |
| Простота использования | - | + | - |
| Портативность | - | + | - |

Таким образом, в качестве базы данных для будущего продукта была выбрана MySQL, так как она знакома мне, у неё большое количество типов данных.

Для взаимосвязи баз данных и северной части продукта необходимо использовать серверный язык. Для реализации этого были рассмотрены два языка программирования – Python и Php.

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным – всё является объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации. Сам же язык известен как интерпретируемый и используется в том числе для написания скриптов.

Недостатками языка являются использование языка в серверной разработке, только благодаря фреймворку, а также зачастую более низкая скорость работы и более высокое потребление памяти написанных на нём программ по сравнению с аналогичным кодом, написанным на компилируемых языках, таких как Си или C++. Основные архитектурные черты — динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений, высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

PHP — скриптовый язык общего назначения, интенсивно применяемый для разработки веб-приложений. В настоящее время поддерживается подавляющим большинством хостинг-провайдеров и является одним из лидеров среди языков, применяющихся для создания динамических веб-сайтов. Действует, как и самостоятельно, так и с фреймворками.

Язык и его интерпретатор (Zend Engine) разрабатываются группой энтузиастов в рамках проекта с открытым кодом. Проект распространяется под собственной лицензией, несовместимой с GNU GPL.

Язык автоматически поддерживает HTTP Cookies в соответствии со стандартами Netscape. Это позволяет проводить установку и чтение небольших сегментов данных на стороне клиента. Работа с Cookies организована посредством сеансов (сессий).

Для наглядности сравнения языков программирования была составлена таблица 2.

Таблица 2 – Сравнение языков программирования для разработки программного продукта

|  |  |  |
| --- | --- | --- |
| Название языка программирования | Php | Python |
| Наличие библиотек | + | + |
| Инструменты для работы с БД | + | + |
| Объектно-ориентированные возможности | + | + |
| Лёгкий понятный синтаксис | + | - |
| Более активное сообщество | + | - |
| Более лёгкая простая модульность | + | - |

Таким образом, Php будет более лучшим вариантом, так как он имеет большое количество библиотек и имеет инструменты для работы с БД.

Для разработки программного продукта рассмотрены следующие инструментальные средства разработки программных продуктов:

1. Aptana Studio.

2. PhpStorm.

3. Atom.

Aptana Studio — кроссплатформенная, свободная, интегрированная среда разработки приложений (IDE) c открытым исходным кодом для создания динамических веб-приложений. Включает в себя такие функции, как автодополнение набираемых конструкций на лету для кода JavaScript, HTML, и CSS, выделение цветом JavaScript, HTML и CSS кода, предупреждения об ошибках и возможность для настройки и расширения пользовательского интерфейса.

С помощью дополнительных плагинов Aptana Studio также поддерживает разработку для следующих языков и платформ: Ruby on Rails, PHP, Python, Adobe Air.

Помимо самостоятельных версий для Windows NT, Mac OS и Linux, Aptana доступна в качестве плагина для платформы Eclipse.

Последняя версия Aptana Studio включает в себя сервер Jaxer, позволяющий выполнять JavaScript на стороне сервера. А также интегрирована с сервисом Aptana Cloud, который предлагает развёртывание и хостинг веб-приложений на серверах cloud computing компании Aptana с возможностью масштабируемости ресурсов по требованию. PhpStorm — коммерческая кросс-платформенная интегрированная среда разработки для PHP. Разрабатывается компанией JetBrains на основе платформы IntelliJ IDEA.

PhpStorm представляет собой интеллектуальный редактор для PHP, HTML и JavaScript с возможностями анализа кода на лету, предотвращения ошибок в коде и автоматизированными средствами рефакторинга для PHP и JavaScript. Автодополнение кода в PhpStorm поддерживает спецификацию PHP 5.3, 5.4, 5.5, 5.6, 7.0, 7.1, 7.2, 7.4 и 8.0. Имеется полноценный SQL-редактор с возможностью редактирования полученных результатов запросов.

PhpStorm разработан на основе платформы IntelliJ IDEA, написанной на Java. Пользователи могут расширить функциональность среды разработки за счет установки плагинов, разработанных для платформы IntelliJ, или написав собственные плагины.

Atom – это многофункциональный текстовый редактор от разработчиков GitHub. Он поддерживает огромное количество различных расширений, благодаря которым его можно сравнить с настоящей средой разработки. Другая особенность – платформа Electron, на которой держится Atom. Она включает в себя Node.js и движок от Chrome – такой инструментарий позволяет разрабатывать приложения для рабочего стола на основе веб-технологий.

Atom позволяет синхронизировать проект с репозиторием GitHub, находить и устанавливать необходимые компоненты под требуемые задачи, поскольку у Atom открытый исходный код, и много разработчиков уже внесли вклад в развитие программы. Сравнение IDE для разработки программного продукта наглядно представлено в таблице 3.

Таблица 3 – Сравнение IDE для разработки программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| Название IDE | PhpStorm | Aptana Studio | Atom |
| Распространяется бесплатно | -  (только студенческая  лицензия) | + | + |
| Автоматическое сохранение | + | + | + |
| Подсказки по коду | + | + | + |
| Интеграция с системой контроля версия (GIT) | + | + | + |
| Возможность расширения функционала библиотеками | + | + | + |
| Заточен под Php-разработку | + | - | - |
| Поддержка CSS/HTML/JS | + | + | + |
| Комфортное использование на слабых ПК | + | + | + |

Таким образом, после рассмотрения вариантов средств разработок, было принято решение использовать PhpStorm.

Основные преимущества PHPStorm:

* PhpStorm идеально подходит для работы с Symfony, Drupal, WordPress, Zend Framework, Laravel, Magento, Joomla!, CakePHP, Yii и другими фреймворками.
* PhpStorm глубоко анализирует структуру кода, поддерживая все возможности языка PHP как в новых, так и в legacy-проектах.
* В PhpStorm вы можете работать с самыми современными технологиями: HTML 5, CSS, Sass, Less, Stylus, CoffeeScript, TypeScript, Emmet и JavaScript. При этом будут доступны рефакторинг, отладка и юнит-тестирование. Благодаря функции Live Edit все изменения можно тут же посмотреть в браузере.
* Рутинные операции удобно выполнять прямо в PhpStorm. IDE интегрирована с системами контроля версий, поддерживает удаленное развертывание, базы данных и SQL, инструменты командной строки, Docker, Composer, REST-клиент и многие другие инструменты.
* PhpStorm включает в себя всю функциональность WebStorm, а также полноценную поддержку PHP, баз данных и SQL.
* PhpStorm позаботится о качестве вашего кода с помощью сотен инспекций, которые проверяют код на лету и анализируют весь проект целиком. Поддержка PHPDoc, code (re)arranger, инструмент форматирования, быстрые исправления и другие возможности помогают разработчикам писать аккуратный код, который легко поддерживать.
* PhpStorm безопасно преобразует ваш код с помощью надежных рефакторингов переименования, перемещения и удаления, извлечения методов, введения переменных, перемещения элементов вверх/вниз, изменения сигнатуры и других. Рефакторинги, учитывающие особенности конкретного языка помогут применить изменения по всему проекту за пару кликов. При этом любое преобразование можно отменить.
* Графический отладчик PhpStorm не требует дополнительной настройки. Он очень наглядно визуализирует, что происходит в вашем приложении на каждом этапе отладки. Отладчик работает с Xdebug и Zend Debugger и может использоваться как локально, так и удаленно. IDE также поддерживает модульное тестирование с PHPUnit, BDD с Behat и интегрируется с профилировщиком.

Таким образом, для создания программного продукта было решено использовать средства:

1. Для создания структурных схем, контекстной и диаграмм декомпозиции использовались CASE-средства – Draw.io.
2. Для наглядного составления структуры базы данных использовался инструмент для визуального проектирования баз данных – MySQL Workbench.
3. Для разработки дизайна web-приложения использовался онлайн-сервис для разработки дизайна и прототипа сайта или приложения – Figma.
4. На этапе разработки программного продукта использовались инструменты и среды для разработки: Php, фреймворк Bootstrap, CSS3, JavaScript, а также редактор кода PhpStorm.
5. Для структурирования, чтения, изменения и удаления информации использовалась база данных MySQL.

Основные преимущества веб-технологий:

1. Доступность – пользователь может пользоваться web-приложением из любой точки мира при доступе в интернет.
2. Переносимость – приложение легко перенести с одного места хранения на другое.
3. Простота интерфейса – пользоваться web-приложением интуитивно-понятно.
4. Секретность и безопасность – доступ к данным ограничен и защищён несколькими способами.
5. Масштабность – предоставления доступа большому количеству пользователей.
6. Простота обслуживания – удобное добавление новых функций в web-приложение.

3. Техническое задание

Техническое задание (далее – ТЗ) — документ или несколько документов, определяющих цель, структуру, свойства и методы какого-либо проекта, и исключающие двусмысленное толкование различными исполнителями.

ТЗ устанавливает основное назначение разрабатываемого объекта, его технические характеристики, показатели качества и технико-экономические требования, предписание по выполнению необходимых стадий создания документации (конструкторской, технологической, программной и т. д.) и её состав, а также специальные требования.

ТЗ представлено в Приложении А.

4. Проектирование ИС

4.1. Структурная схема ИС

Проектирование информационной системы началось с построения диаграммы вариантов использования. На рисунке 1 представлена диаграмма прецедентов Uses CASE. Она содержит 3 актёра, которые могут выполнять суммарно 16 функций, часть из которых может делать несколько актёров, а часть – только определённый актёр.

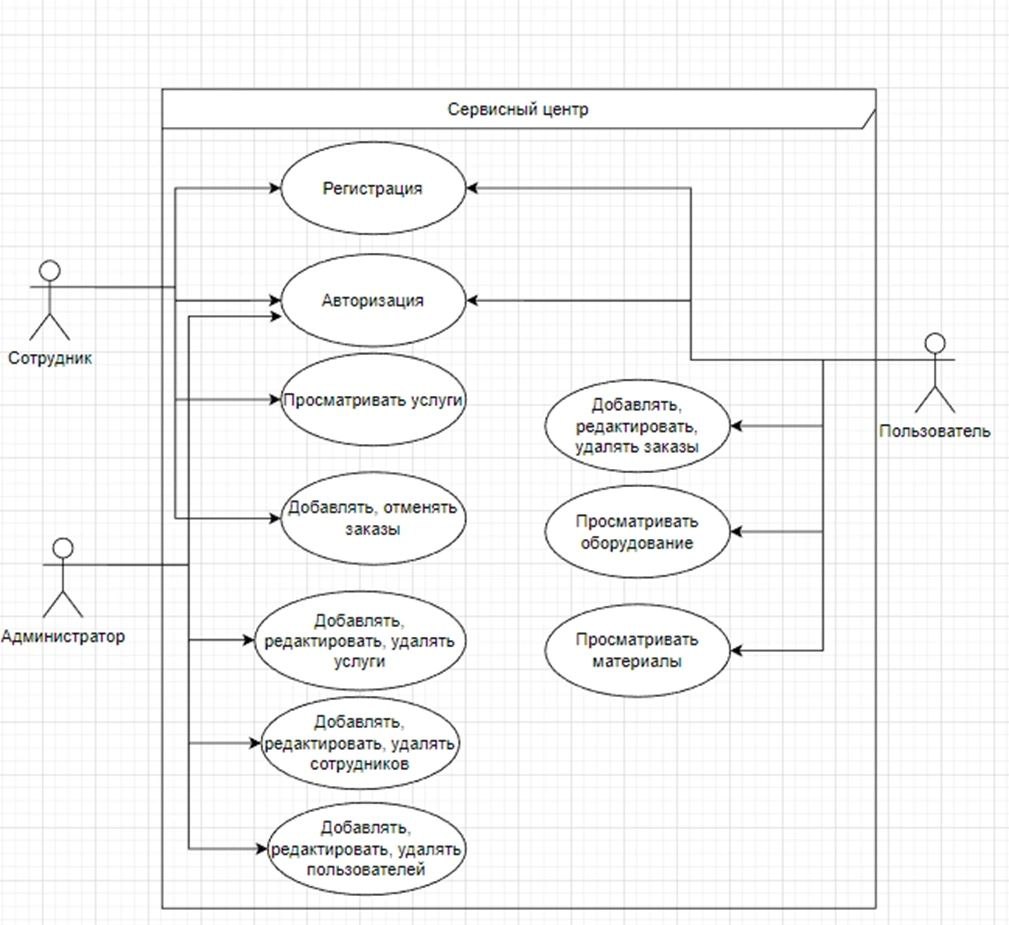


Рисунок 1 – Диаграмма прецедентов

На рисунке 2 представлена диаграмма деятельности. Она содержит 3 объекта. В начале Клиент приносит технику в сервисный центр, далее производится диагностика и выявляется подлежит ли ремонту техника. Если подлежит, то регистрируется заказ и далее производится ремонт. Если техника не подлежит ремонту или Клиент отказывается платить установленную цену, то техника отправляется в утиль или возвращается Клиенту.

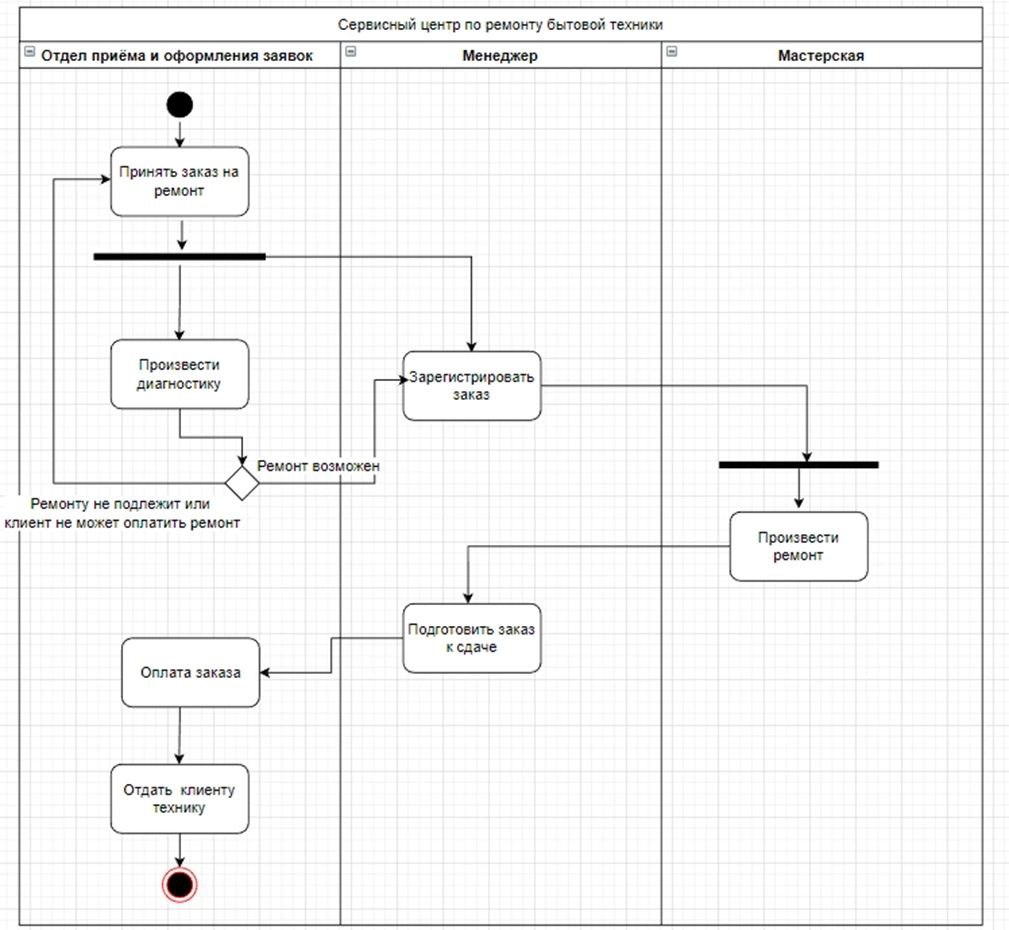


Рисунок 2 – Диаграмма деятельности

На рисунке 3 представлена диаграмма компонентов. В общей сложности 7 элементов, которые служат для регистрации, авторизации и выхода из своей учётной записи, просмотра услуг и создания заказов.

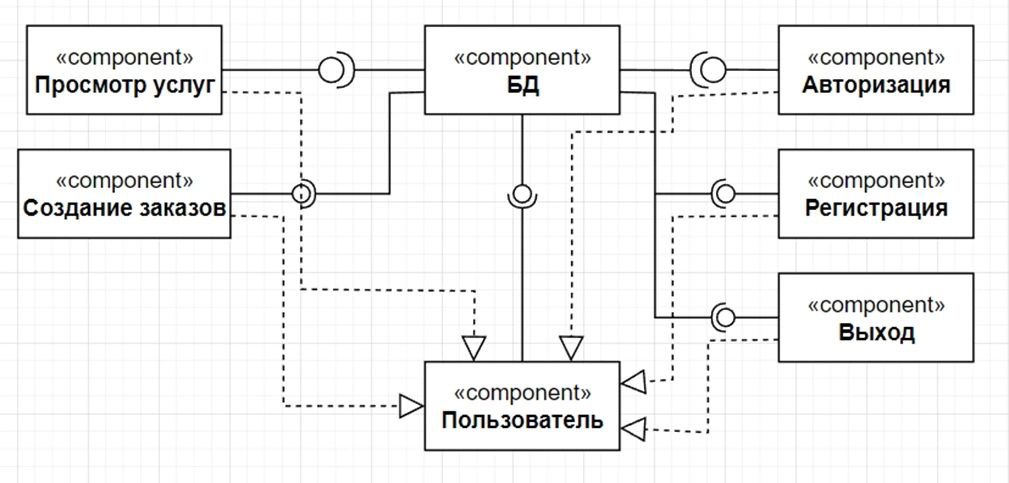


Рисунок 3 – Диаграмма компонентов

На рисунке 4 представлена диаграмма развёртывания. Она показывает, что, чтобы пользоваться программным продуктом, необходим web-сервер, на котором размещаются ИС и БД, а также для администратора, сотрудника и просто пользователя необходимо каждому по ПК с браузером и выходом в интернет.

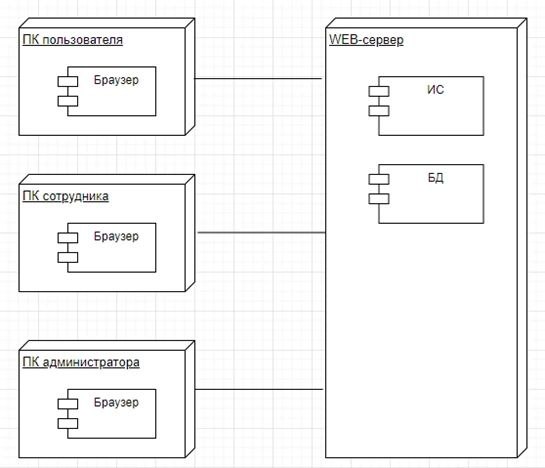


Рисунок 4 – Диаграмма развёртывания

Вывод по построению всех диаграмм – они построены. Теперь представить и разработать ИС проще (вывод в разработке).

4.2 Функциональная схема ИС

На рисунке 5 представлена контекстная диаграмма, отображающая деятельность ИС.

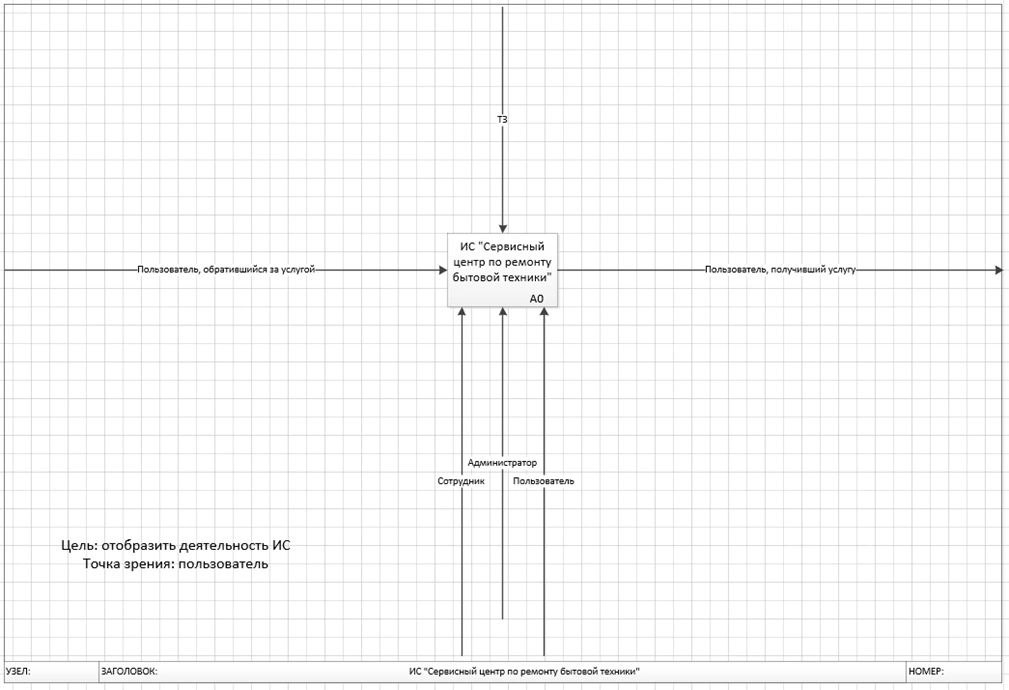


Рисунок 5 – Контекстная диаграмма IDEF0

На рисунке 6 представлена диаграмма декомпозиций (А1), отображающая деятельность ИС более подробно предыдущей.

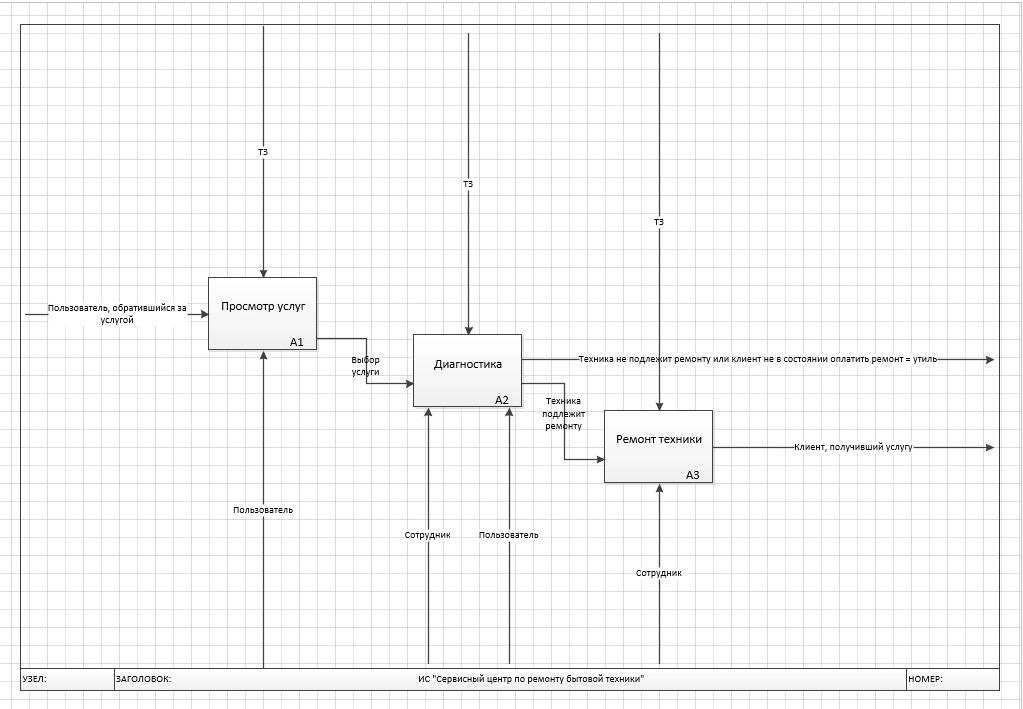


Рисунок 6 – Диаграмма декомпозиций А1

На рисунке 7 представлена диаграмма классов. Она содержит 15 классов, среди них база данных, html-страница, информационная сеть «Сервисный центр по ремонту бытовой техники», пользователь, просто пользователь, сотрудник, администратор, услуги.



Рисунок 7 – Диаграмма классов

На рисунке 8 представлена диаграмма потоков данных. В центре всего пользователь, который может регистрироваться, просматривать услуги, добавлять, редактировать и удалять заказы. Для соответствующих действий в ИС предусмотрены соответствующие БД.

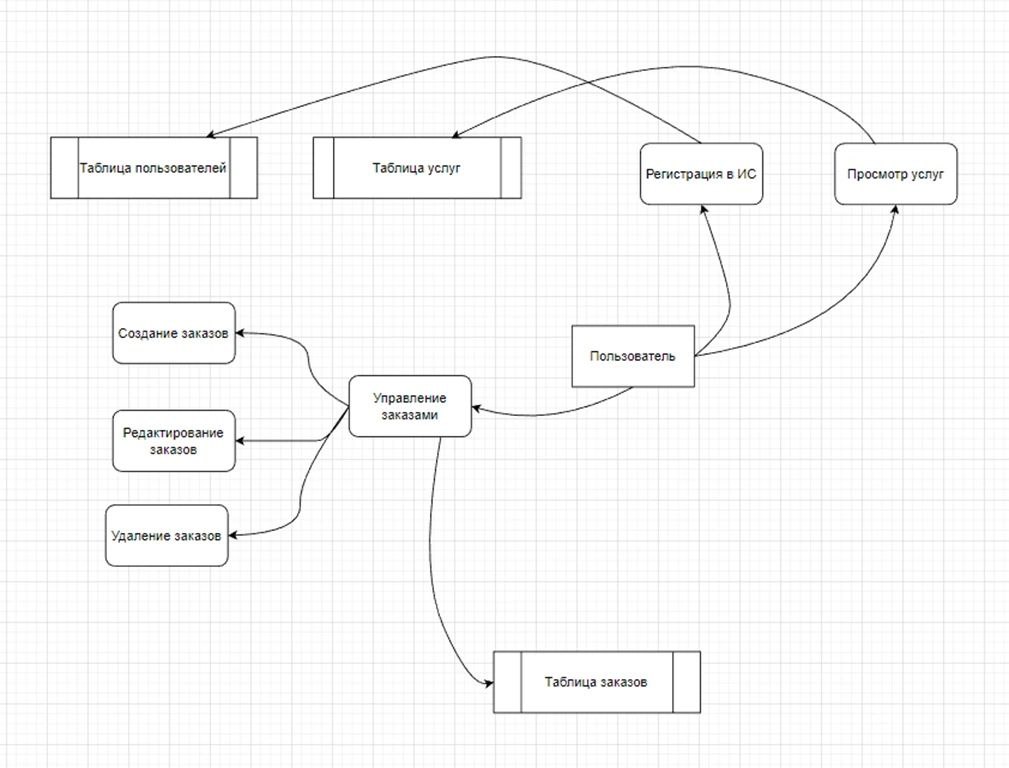


Рисунок 8 – Диаграмма потоков данных

4.3 Проектирование базы данных

Проектирование базы данных начинается с концептуального проектирование базы данных.

Концептуальное проектирование – построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных.

На рисунке 9 представлена инфологическая модель базы данных.

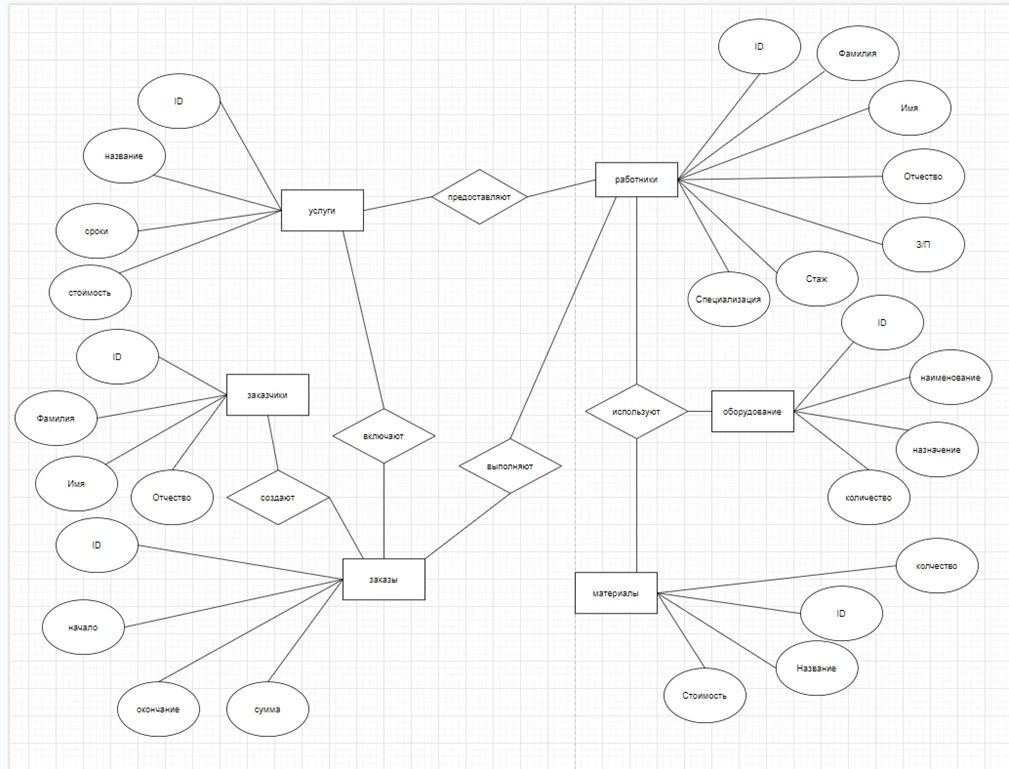


Рисунок 9 – Инфологическая модель базы данных

На инфологической модели базы данных выше схематично отображены сущности системы, их атрибуты и связи между ними. Так, в прямоугольник отображены сущности, такие как: Пользователь, Сотрудники, Оборудование, Материалы, Заказы и Заказчики. В овалах отображены атрибуты сущностей, например: ФИО, ID и другие. Робами изображены связи между сущностями, например сотрудник просматривает материалы.

Далее происходит преобразование концептуальной модели в логическую модель, по формальным правилам. Таким образом, логическое (даталогическое) проектирование — создание схемы базы данных на основе конкретной модели данных, например, реляционной модели данных.

На даталогической модели базы данных (рисунок 10) отображены сущности приложения, а также первичные и внешние ключи, связывающие сущности между собой. Так, сущность «Заказы» содержит в качестве внешнего ключа поле «id\_Сотрудника», которое содержит первичный ключ сотрудника, выполняющего этот заказ.

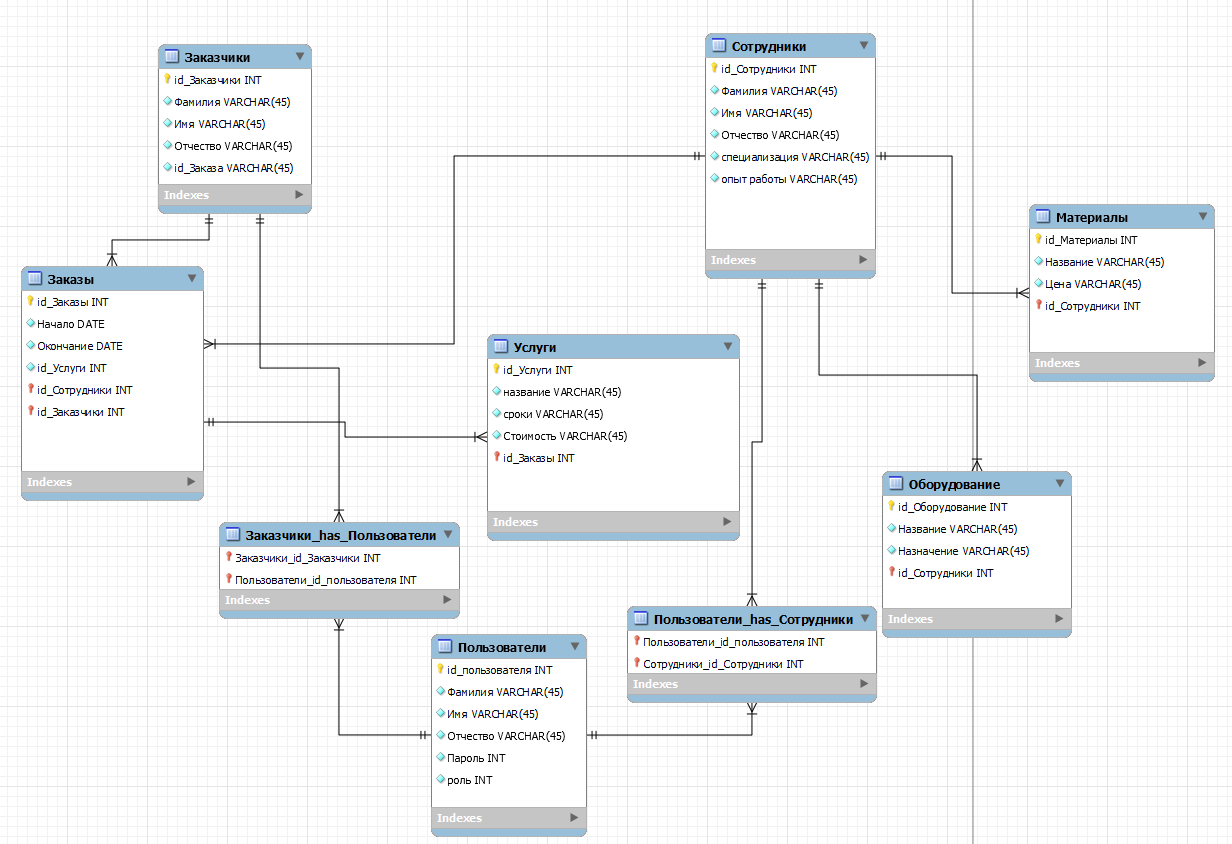


Рисунок 10 – Даталогическая модель базы данных

На рисунке 11 представлена ER-диаграмма базы данных. Она содержит 11 таблиц для полного функционирования и качественной сортировки информации.

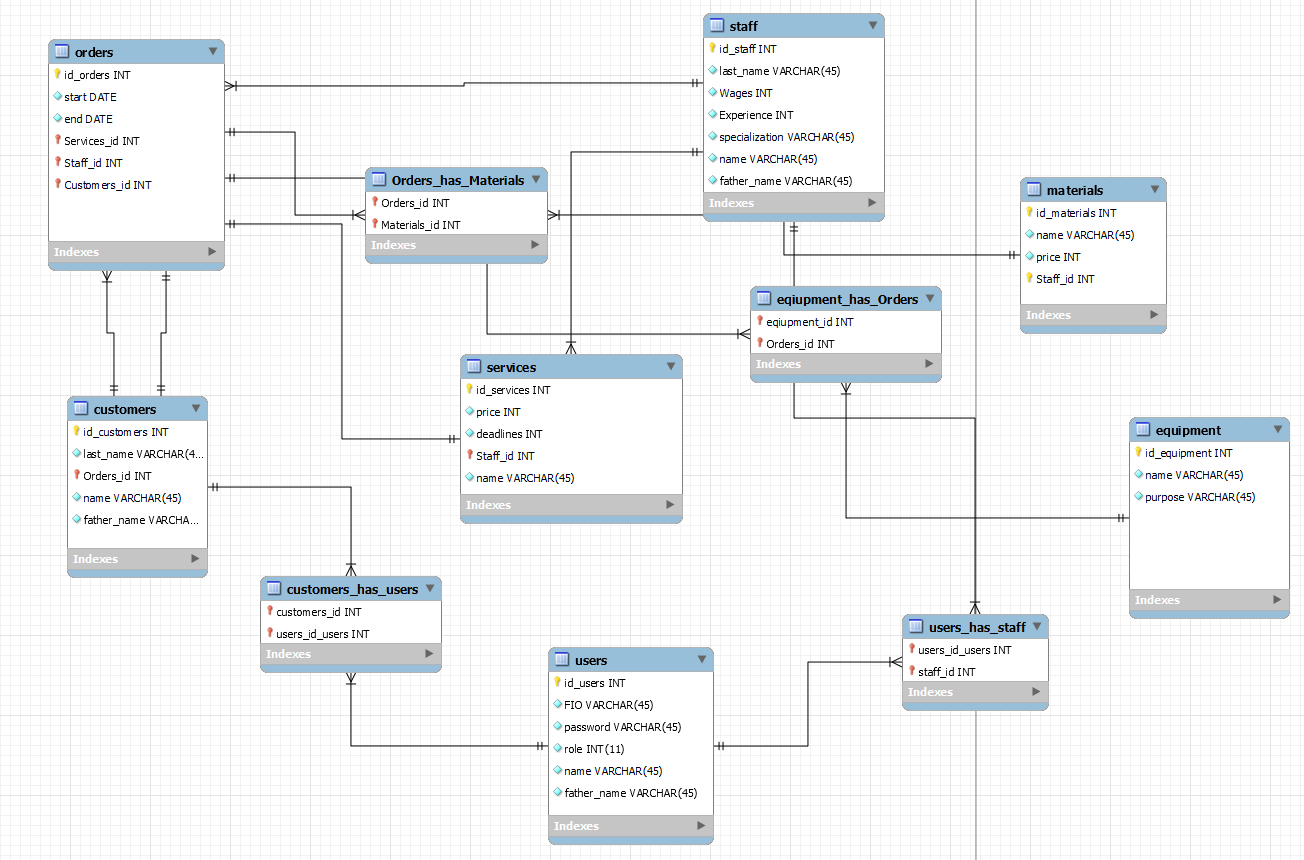


Рисунок 11 – ER-модель базы данных

Перечень таблиц представлены в таблицах 5-15.

Таблица 5 – Таблица Service

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| id\_service | int | Id услуги |
| name | Varchar(45) | Название услуги |
| price | int | Цена услуги |
| deadlines | Varchar(45) | Сроки |
| Staff\_id | int | Id сотрудника |

Таблица 6 – Таблица Customers

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_customer | int | Id клиента |
| Last\_name | Varchar(45) | Фамилия |
| Name | Varchar(45) | Имя |
| Father\_name | Varchar(45) | Отчество |
| Orders\_id | int | Id заказа |

Таблица 7 – Таблица Orders

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_order | int | Id заказа |
| Start | Date | Начало |
| End | Date | Конец |
| Service\_id | int | Id услуги |
| Staff\_id | int | Id сотрудника |
| Customer\_id | int | Id клиента |

Таблица 8 – Таблица Materials

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_material | int | Id материала |
| Name | Varchar(45) | Название |
| Price | int | Цена |
| Staff\_id | int | Id сотрудника |

Таблица 9 – Таблица Equipment

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_equipment | int | Id оборудования |
| Name | Varchar(45) | Название |
| purpose | Varchar(45) | Назначение |

Таблица 10 – Таблица user

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_user | int | Id пользователя |
| Last\_name | Varchar(45) | Фамилия |
| Name | Varchar(45) | Имя |
| Father\_name | Varchar(45) | Отчество |
| Password | int | Пароль |
| role | int | Роль |

Таблица 11 – Таблица Staff

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| Id\_staff | int | Id сотрудника |
| Last\_name | Varchar(45) | Фамилия |
| Name | Varchar(45) | Имя |
| Father\_name | Varchar(45) | Отчество |
| Experience | Varchar(45) | Опыт |
| Wages | Varchar(45) | ЗП |
| Specialization | Varchar(45) | Специализация |

Таблица 12 – Таблица customers\_has\_users

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| customer\_id | Int | Id клиента |
| user\_id\_user | Int | Id пользователя |

Таблица 13 – Таблица users\_has\_staff

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| user\_id\_user | int | Id пользователя |
| Staff\_id | int | Id сотрудника |

Таблица 14 – Таблица orders\_has\_materials

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| material\_id | int | Id материала |
| orders\_id | int | Id заказа |

Таблица 15 – Таблица equipment­\_has\_orders

|  |  |  |
| --- | --- | --- |
| Таблица | Тип данных | Описание |
| equipment\_ id | int | Id оборудования |
| orders\_id | int | Id заказа |

БД приведена к 1 начальное форме (далее – НФ), т.к. все поля, принимаемые больше одного значения, декомпозированы. Также БД имеет 2 и 3 НФ, т.к. каждый не ключевой атрибут приведен к неприводимости, и каждый не ключевой атрибут не транзитивно (непосредственно) зависит от первичного ключа.

4.4. Проектирование интерфейса

Для разработки пользовательского интерфейса был выбран инструмент Figma – браузерный инструмент для создания интерфейсов.

В результате проектирование интерфейса будущей информационной системы были спроектированы прототипы трёх страниц: главная страница (рисунок 12), регистрация (рисунок 13), авторизация (рисунок 14). Благодаря созданию прототипов, разработка непосредственно программного продукта будет значительно облегчена, за счёт наглядных примеров будущих страниц ИС.

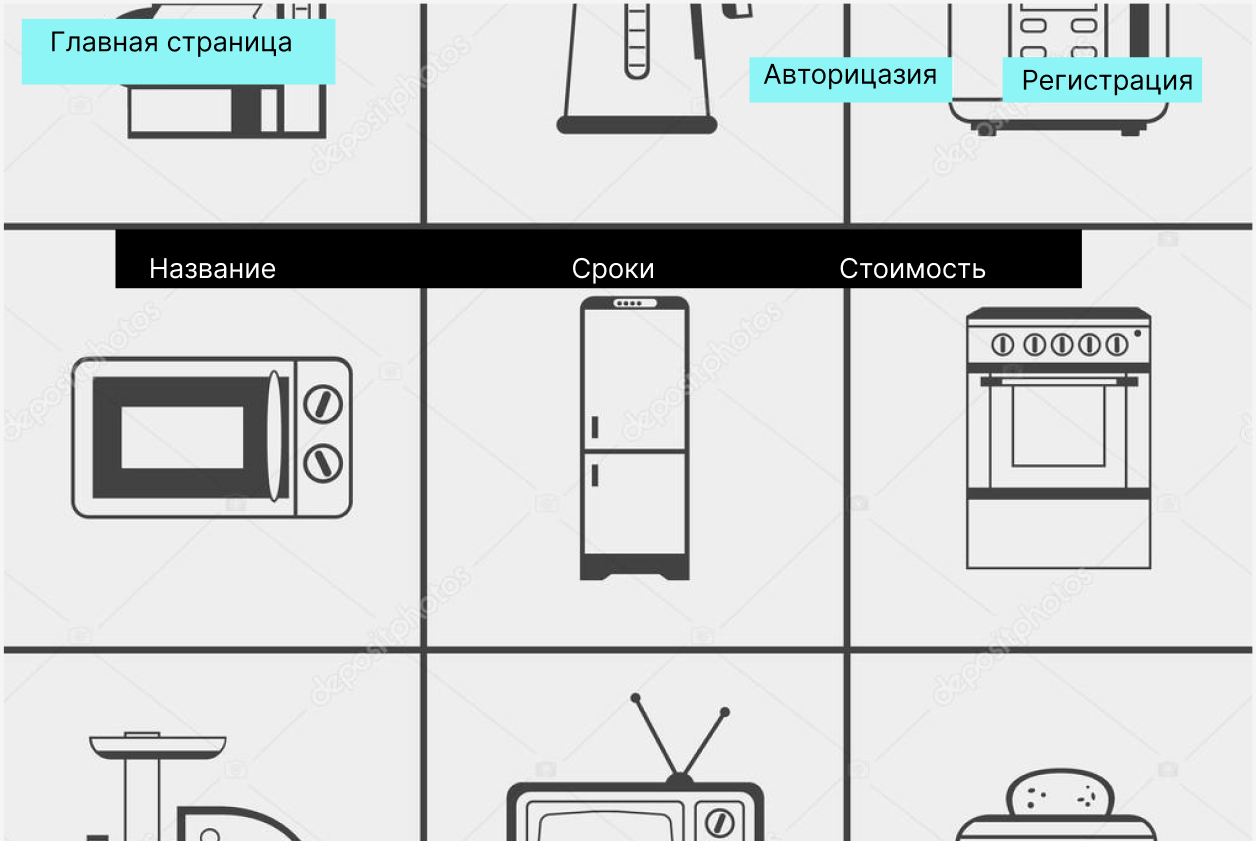


Рисунок 12 – Главная страница

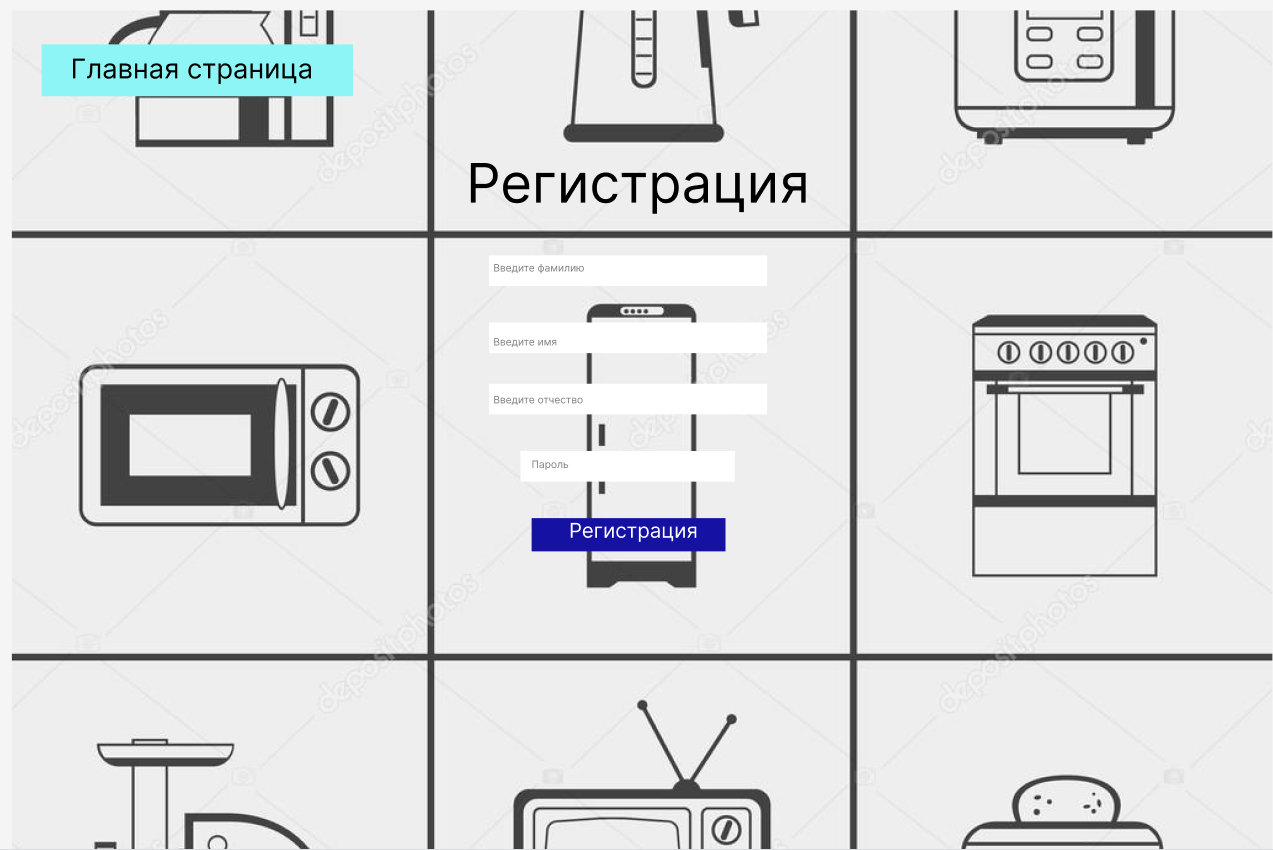


Рисунок 13 – Регистрация



Рисунок 14 – Авторизация

5. Разработка ИС

5.1. Разработка интерфейса ИС

В информационной системе «Сервисный центр по ремонту бытовой техники» были поставлены и выполнены следующие задачи:

− авторизация;

− регистрация;

− создание заказов;

− просмотр текущих клиентов;

− удаление заказов.

− редактирование заказов.

− создание сотрудников;

− удаление сотрудников;

− редактирование сотрудников.

Программный продукт прост в освоении, т.к. имеет интуитивный интерфейс, в котором разобраться не составит труда даже начинающему пользователю ИС.

Для серверной операционной системы, на которой будет располагаться программный код ИС, выдвигаются следующие минимальные требования:

* процессор: Intel Pentium 4 или более поздней версии;
* ОЗУ 512 Мб DDR3;
* 15 ГБ на HDD;
* поддержка БД MySQL;
* операционная система Windows;
* доступ к сети Интернет.

Требования к клиентской части ИС, необходимые для пользования информационной системой:

* процессор Intel Pentium 4 или более поздней версии;
* ОЗУ 2 ГБ.
* доступ к сети Интернет;
* HDD 1ГБ;
* Браузер;
* встроенный видеоадаптер;
* клавиатура, мышь;
* операционная система: Windows.

Общий размер дисковой памяти, занимаемой информационной системой, составляет 100 МБ (рисунок 15).



Рисунок 15 – Объём дисковой памяти ИС

Объем потребляемой ОЗУ составляет 71680К (что примерно равно 70Мб) на одну вкладку в браузере Google Chrome (рисунок 16).



Рисунок 16 – Потребляемая ОЗУ в Google Chrome

На рисунках 17 и 18 показан html код главной страницы. На ней находится лого, которое играет роль кнопки «Главная страница», кнопка «Авторизация» и кнопка «Новости» форма подключена к файлу header.php( в нем подключены стили css) и Services.php(в нем связь с таблицей services) к которому подключена форма и из которого осуществляется вывод данных из бд.



Рисунок 17 – 1 часть HTML кода главной страницы

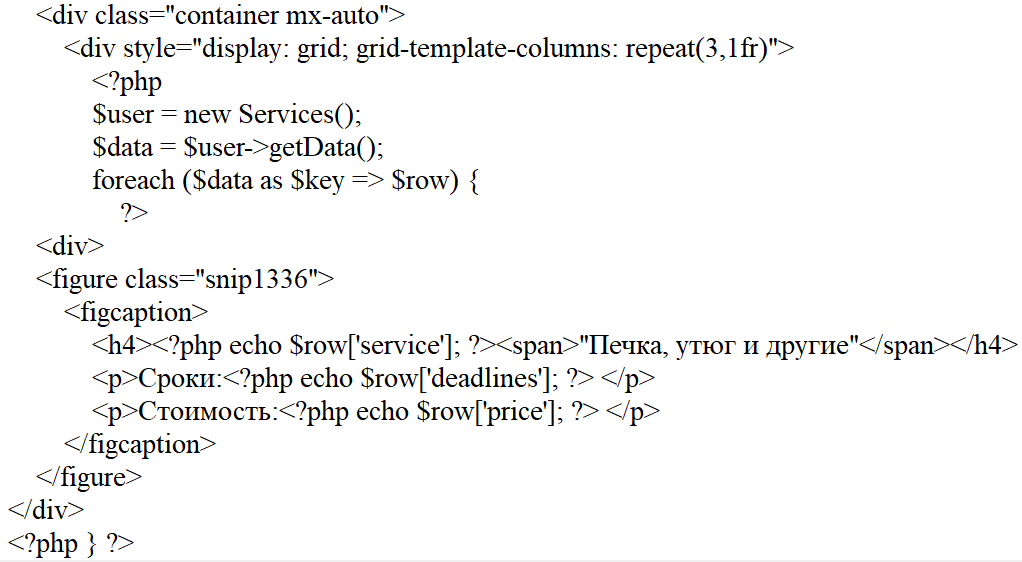


Рисунок 18 – 2 часть HTML кода главной страницы

На рисунке 19 изображён результат создания главной страницы.

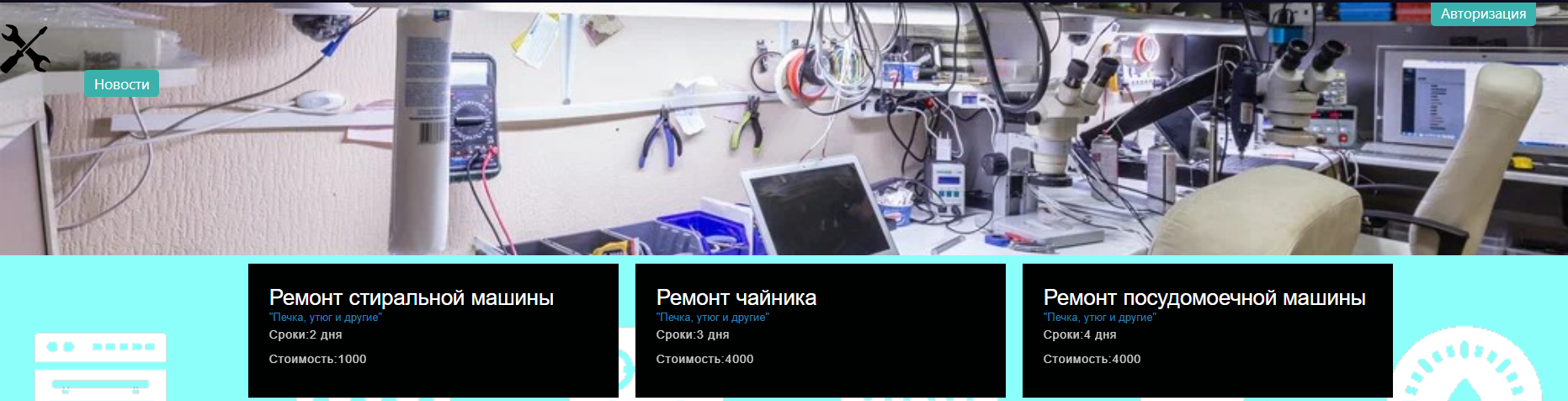


Рисунок 19 – Результат HTML кода главной страницы

На рисунках 20 и 21 изображен html код страницы авторизации.

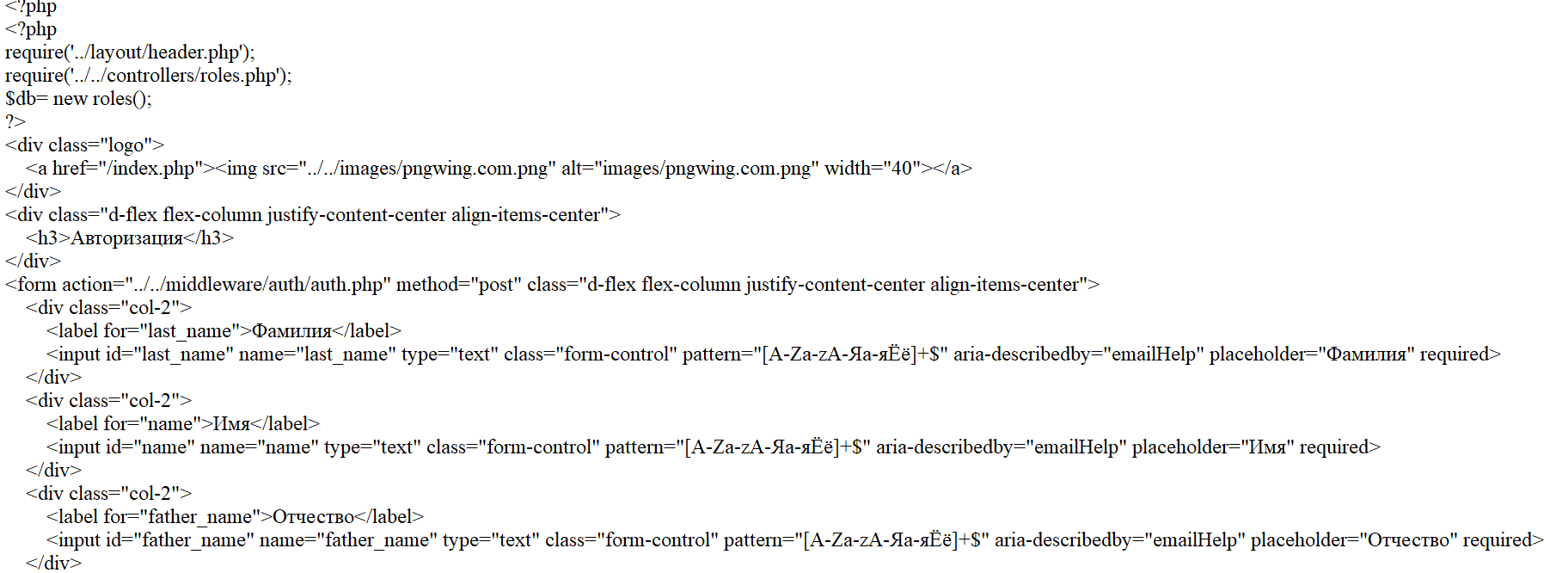


Рисунок 20 – 1 часть HTML кода страницы авторизации

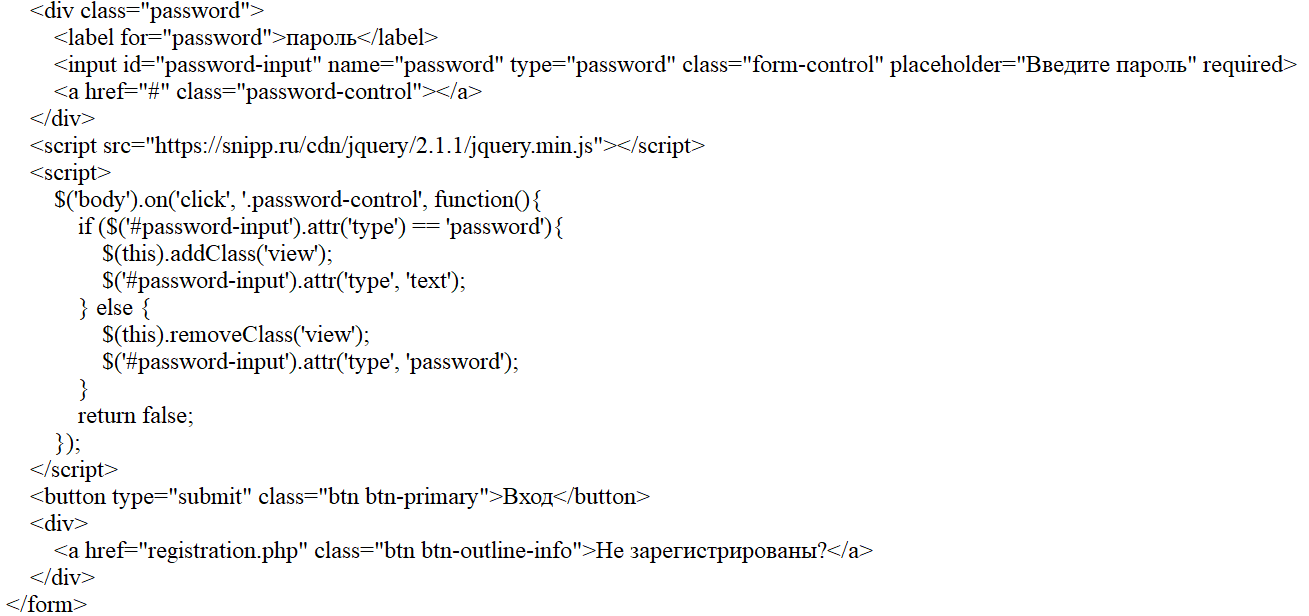


Рисунок 21 – 2 часть HTML кода страницы авторизации

На рисунке 22 изображён результат создания страницы авторизации.



Рисунок 21 – результат HTML кода страницы авторизации

На рисунках 23 и 24 изображен html код страницы регистрации.

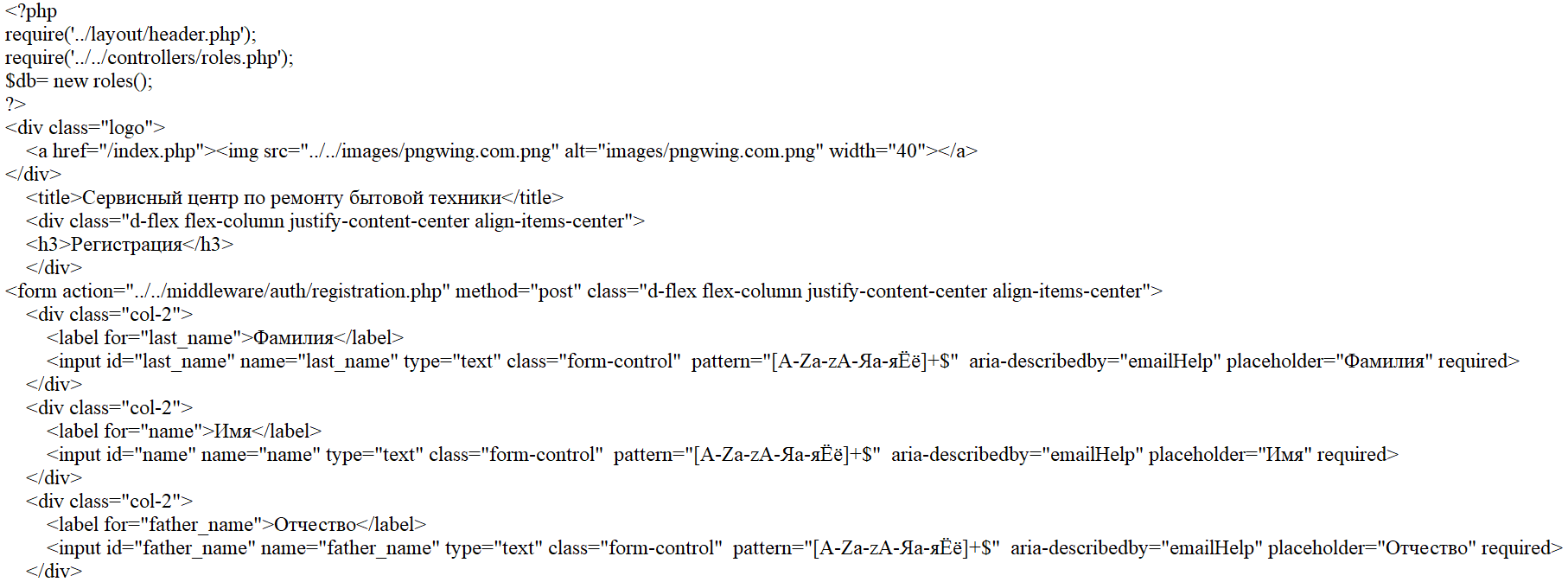


Рисунок 23 – 1 часть HTML кода страницы регистрации

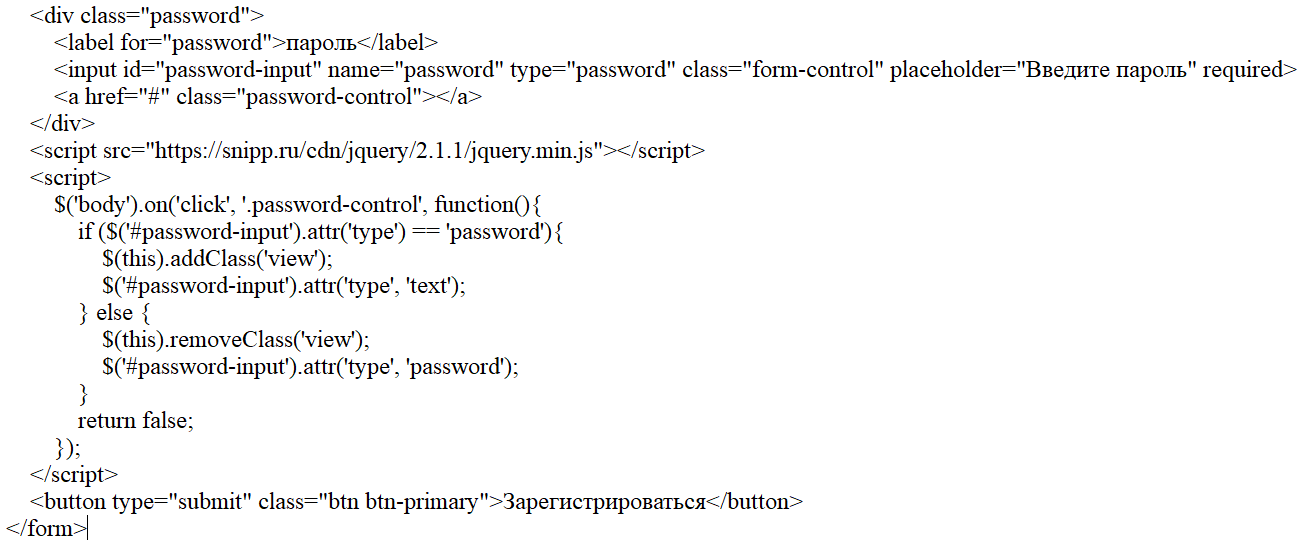


Рисунок 24 – 2 часть HTML кода страницы регистрации

На рисунке 25 изображён результат создания страницы регистрации.

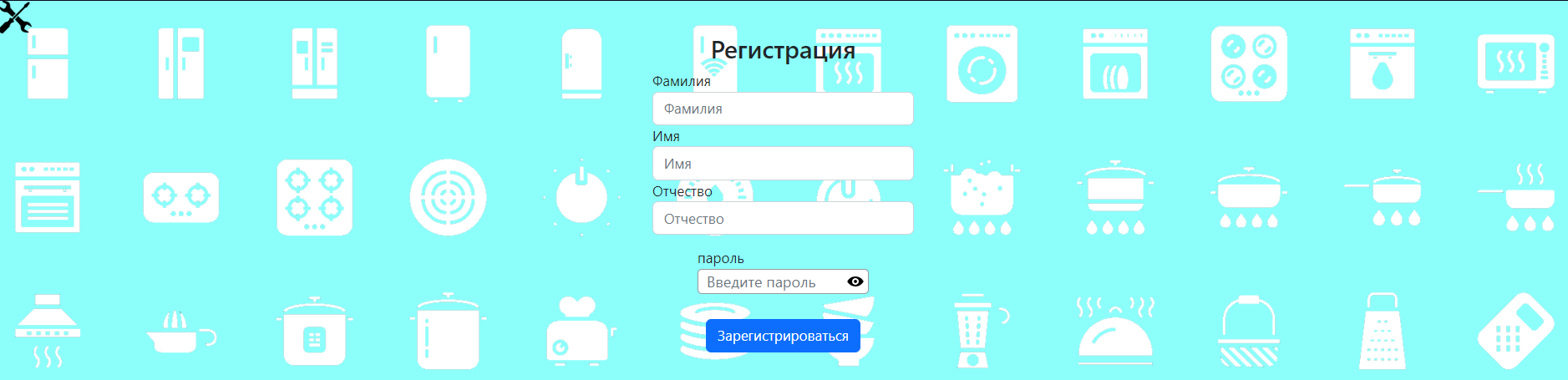


Рисунок 25 – результат HTML кода страницы авторизации

На рисунке 26 изображен html код страницы новостей.

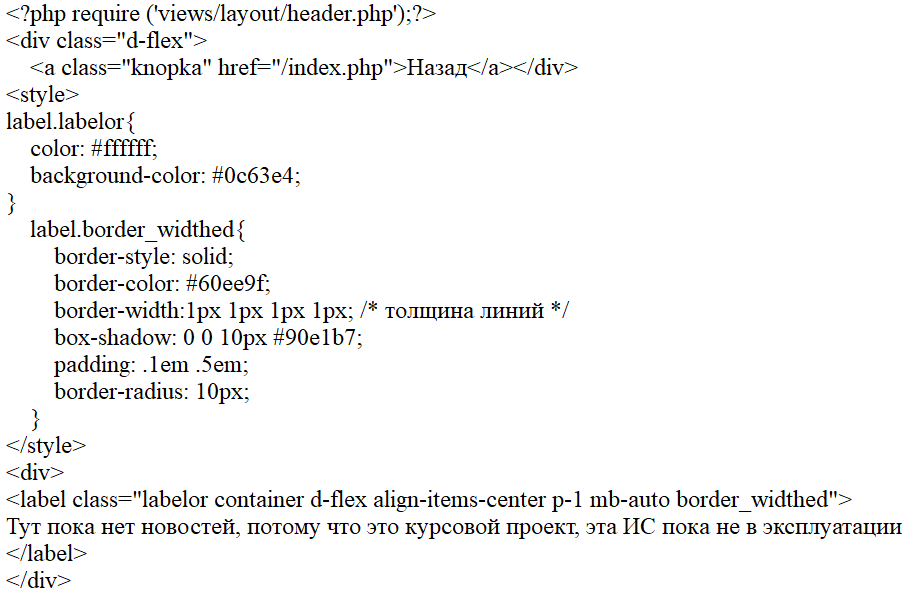


Рисунок 26 – HTML код страницы новости

На рисунке 27 изображён результат создания страницы новости.

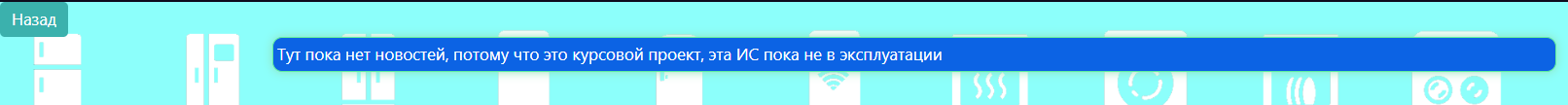


Рисунок 27 – результат HTML кода страницы новости

Так же было создано 3 личных кабинета (далее – ЛК) общего вида: для администратора, для сотрудника и для пользователя.

На картинках 28 и 29 изображён html ЛК администратора.

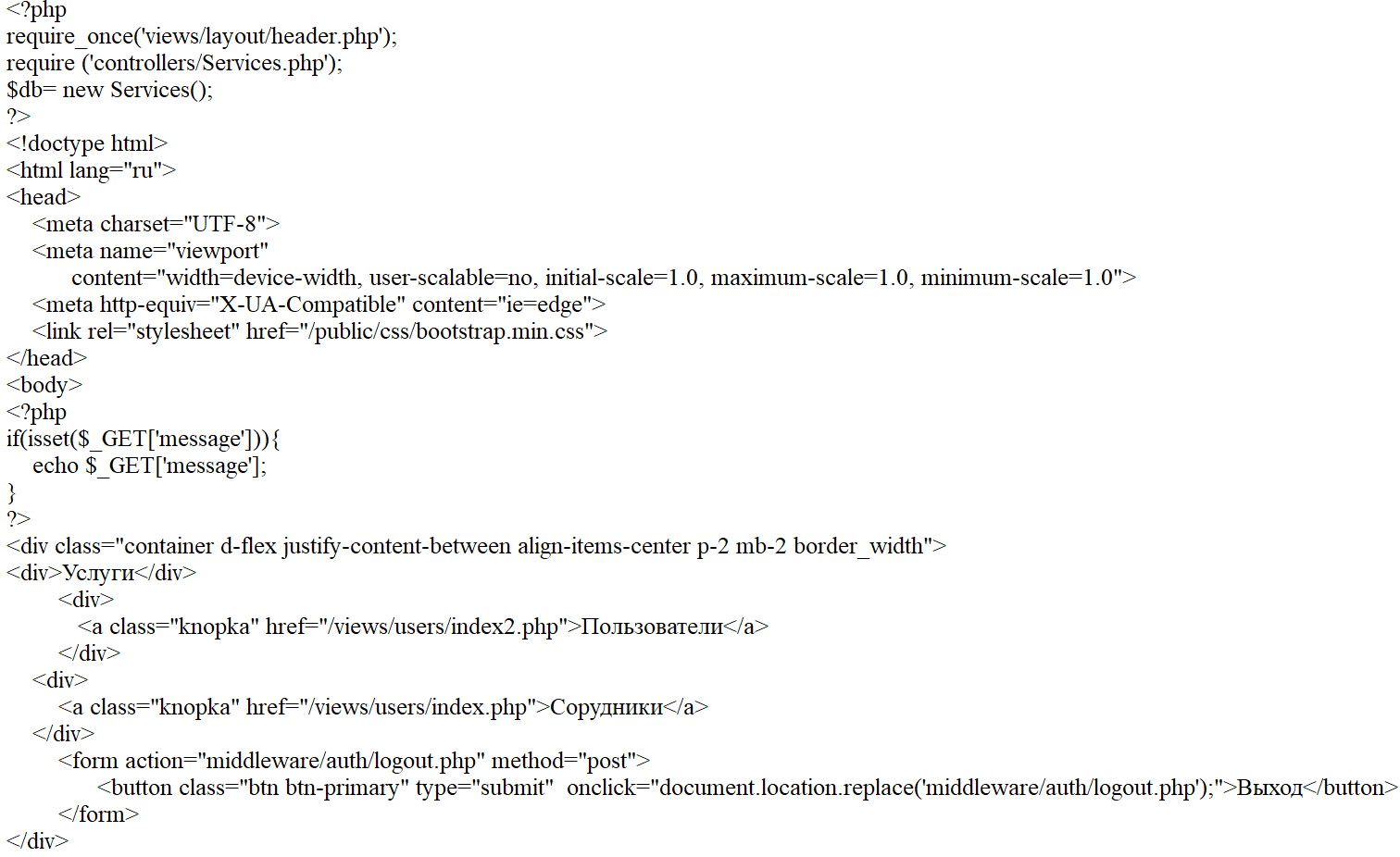


Рисунок 28 – 1 часть HTML кода ЛК администратора

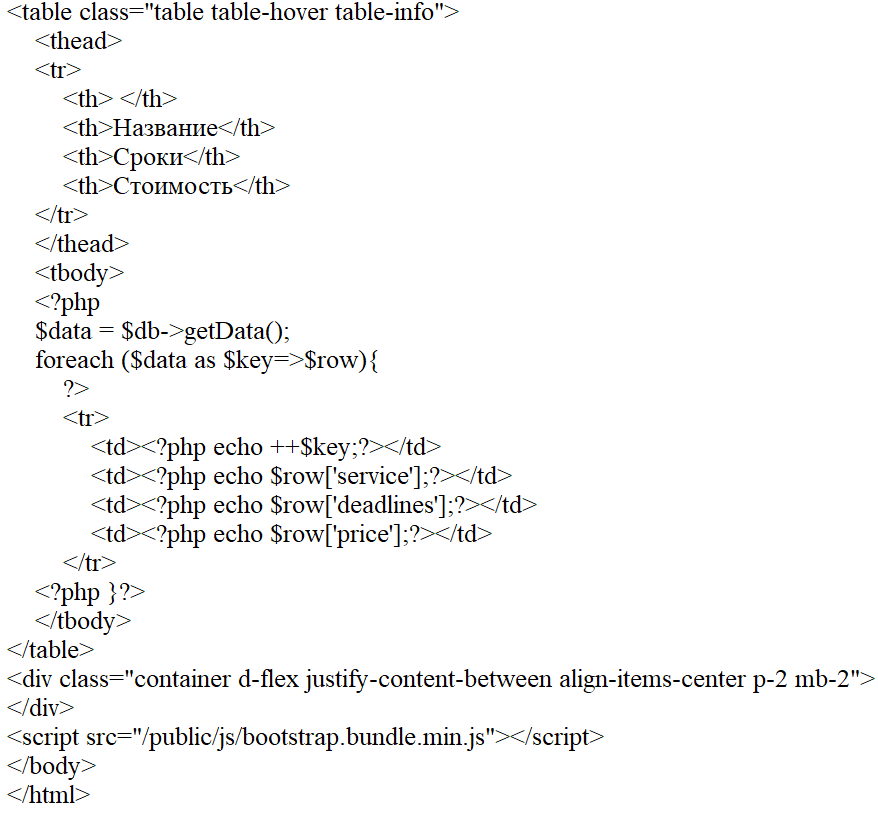


Рисунок 29 – 2 часть HTML кода ЛК администратора

На рисунке 30 изображён результат создания ЛК администратора.

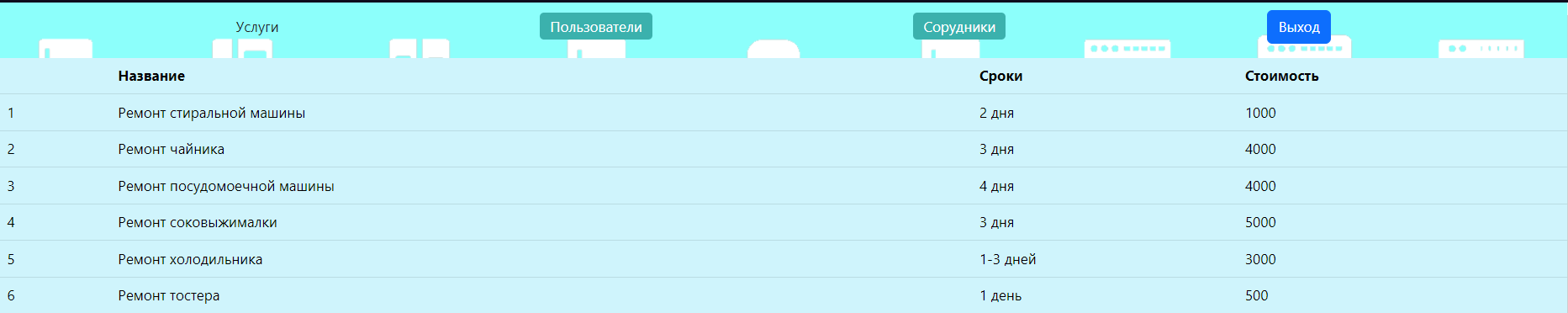


Рисунок 30 – результат HTML кода ЛК администратора

На картинках 31 и 32 изображён html ЛК сотрудника.

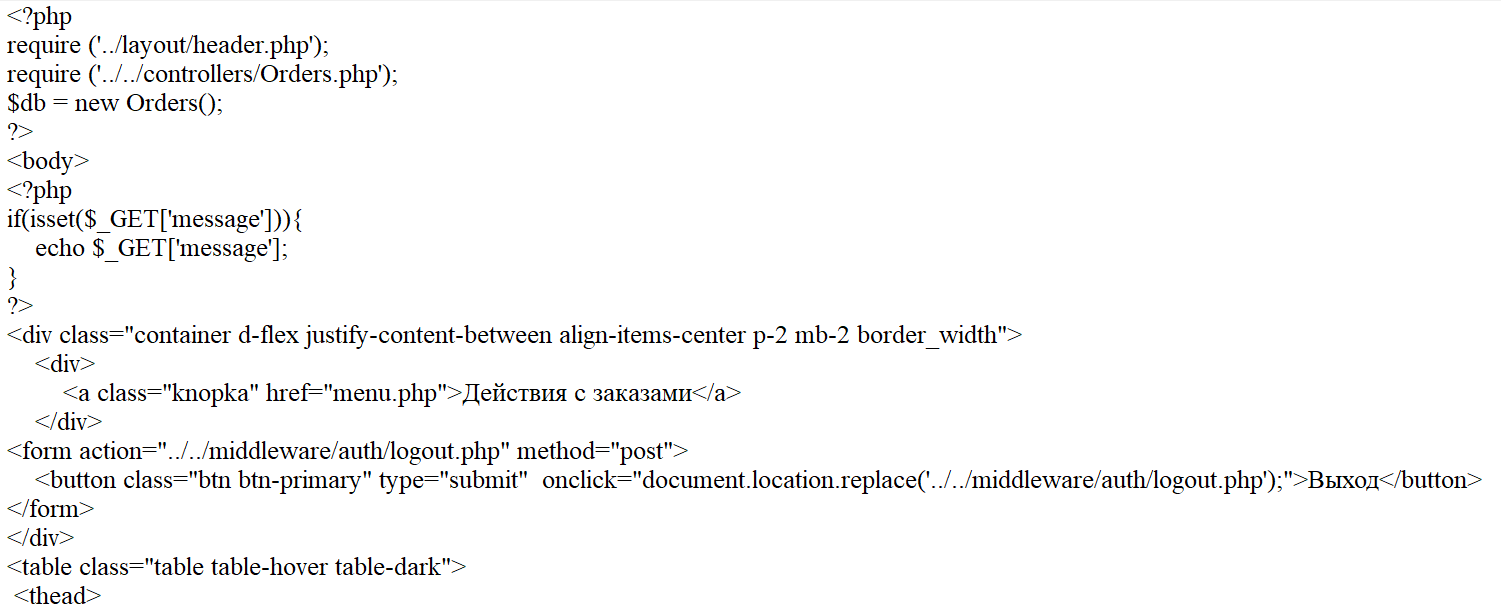


Рисунок 30 – 1 часть HTML кода ЛК сотрудника

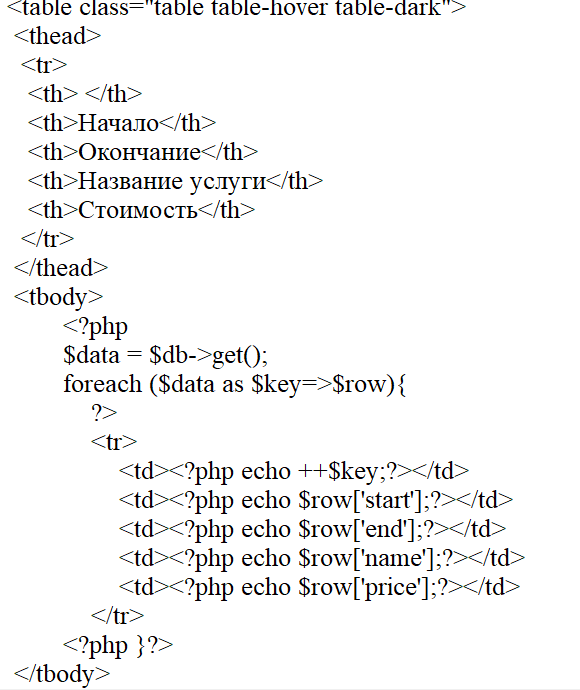


Рисунок 31 – 2 часть HTML кода ЛК сотрудника

На рисунке 32 изображён результат создания ЛК администратора.

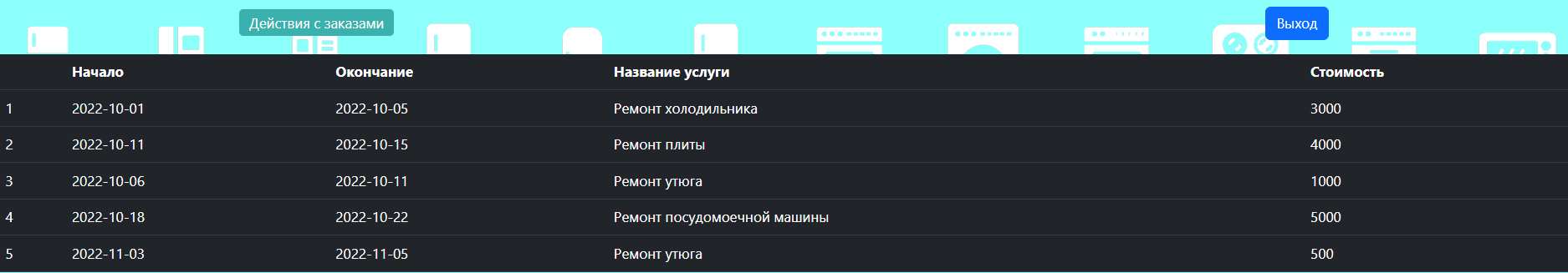


Рисунок 32 – результат HTML кода ЛК сотрудника

На картинке 33 изображён html ЛК пользователя.

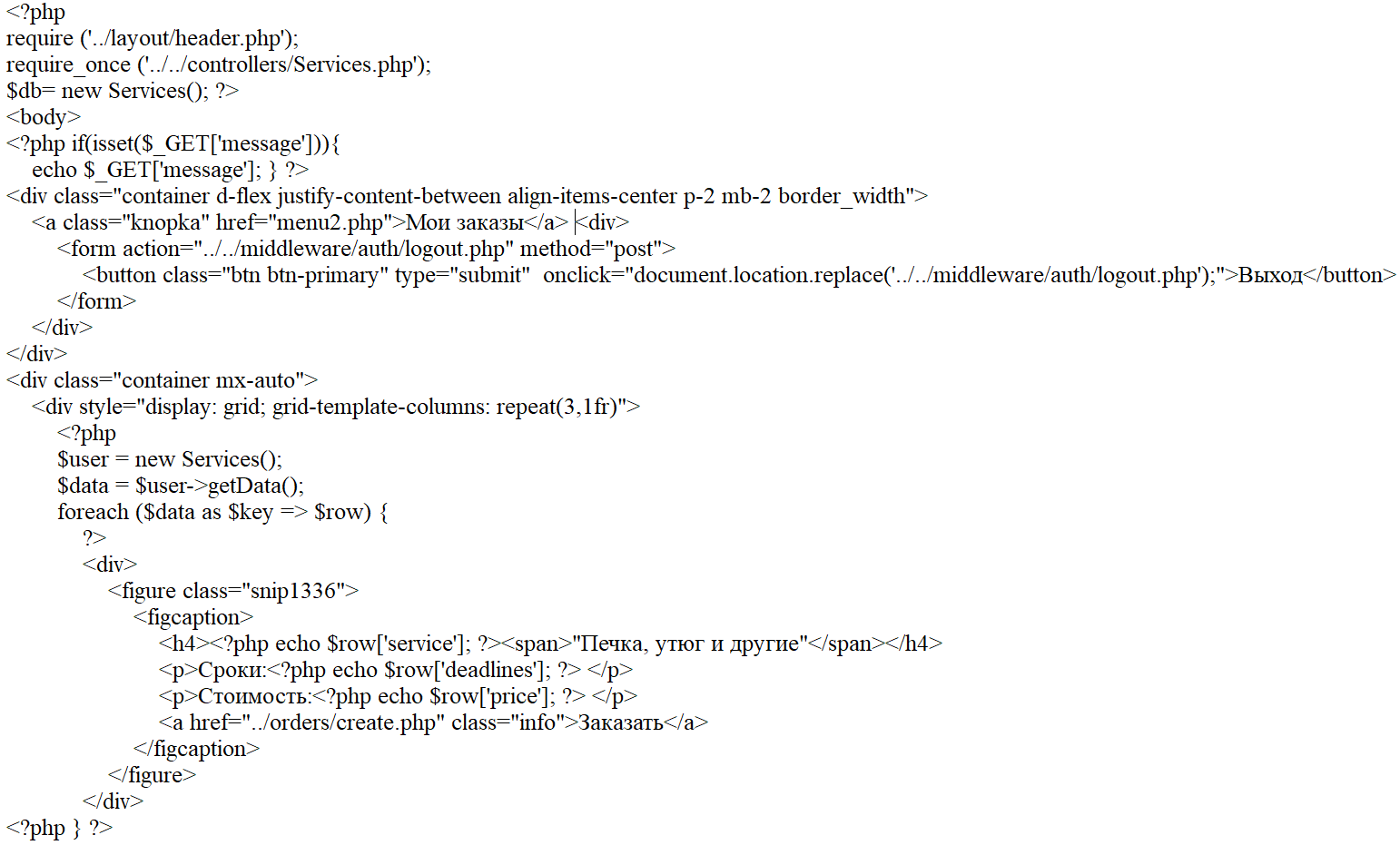


Рисунок 33 – HTML кода ЛК пользователя

На рисунке 34 изображён результат создания ЛК пользователя.

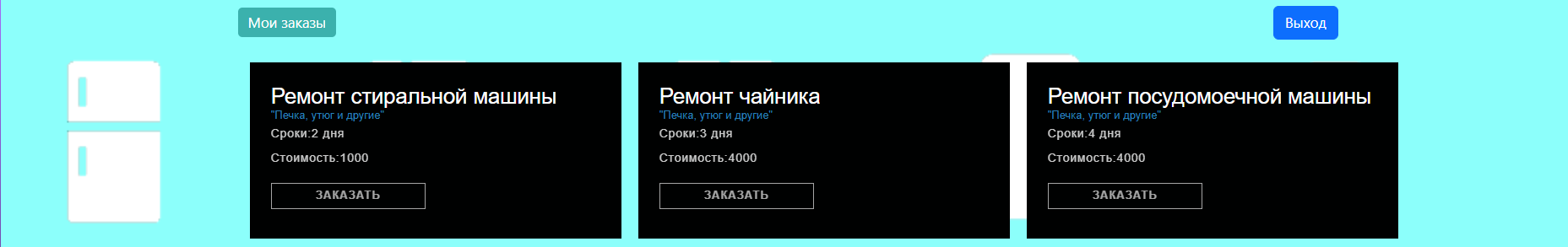


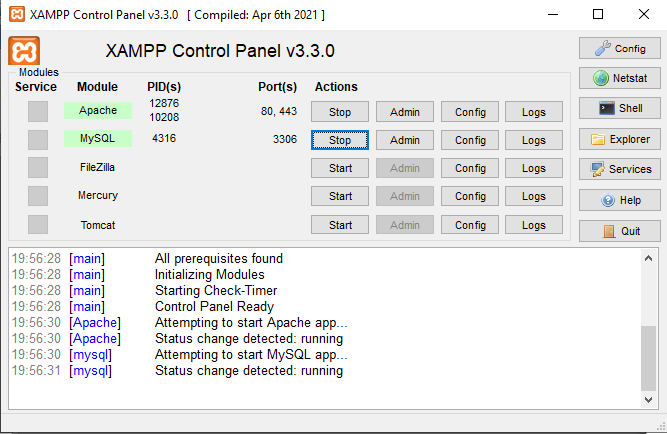
Рисунок 34 – результат HTML кода ЛК пользователя

6. Технологическая документация ИС

6.1 Руководство пользователя ИС

Для того чтобы запустить ИС для начала нужно запустить файл start.bat и копируем оттуда адрес сайта и вставляем в адресную строку браузера. Так же нужно запустить XAMPP и запустить Apache и MySQL.





В данном руководстве пользователя рассмотрен пример Личный кабинет (Далее – ЛК) пользователя. На рисунке 35 можно видеть список услуг, предоставляемых сервисным центром, а так же 2 кнопки: «Выход» и «Мои заказы» для просмотра заказов.

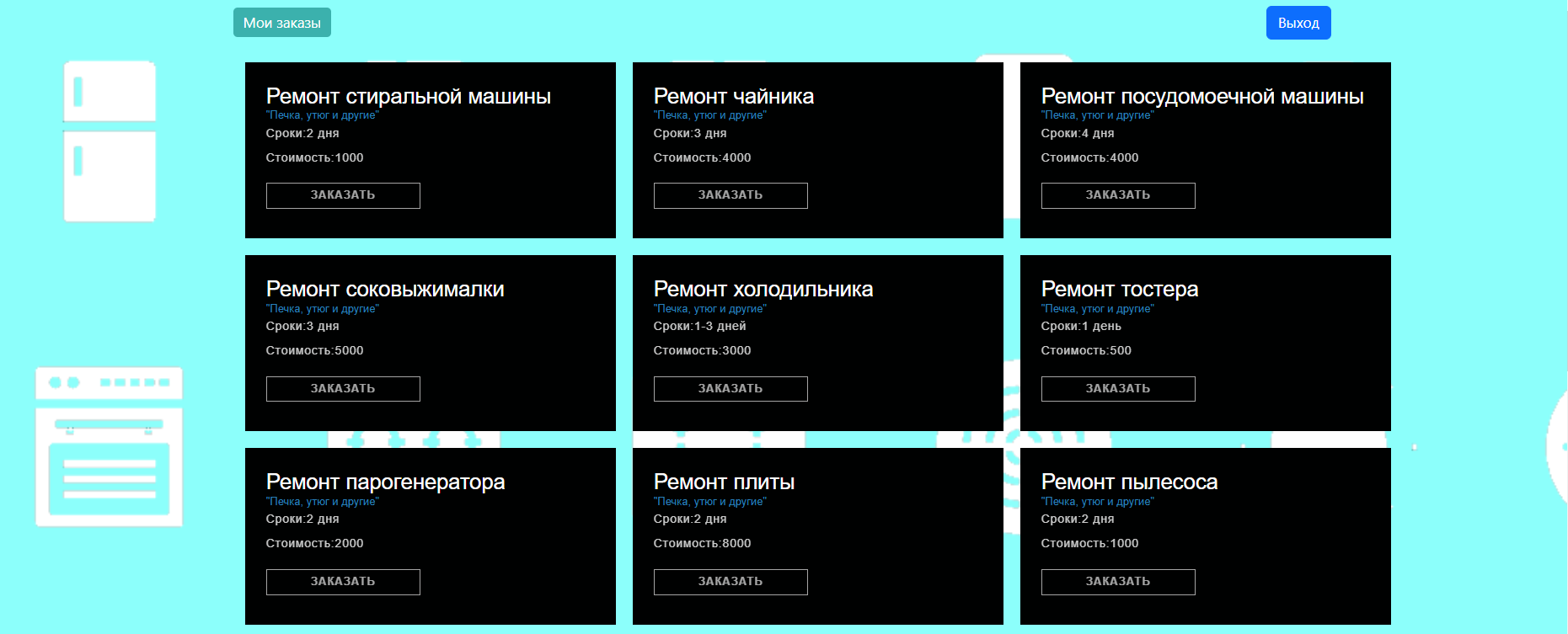


Рисунок 35 – ЛК пользователя

При нажатии кнопки заказать на карточке услуги можно перейти на форму создания заказа (рисунок 36) и из выплывающего списка выбрать услугу, которая требуется пользователю.



Рисунок 36 – Создание заказа

Далее представлена можно сказать «корзина» пользователя, где он может отслеживать свои заказы (рисунок 37).

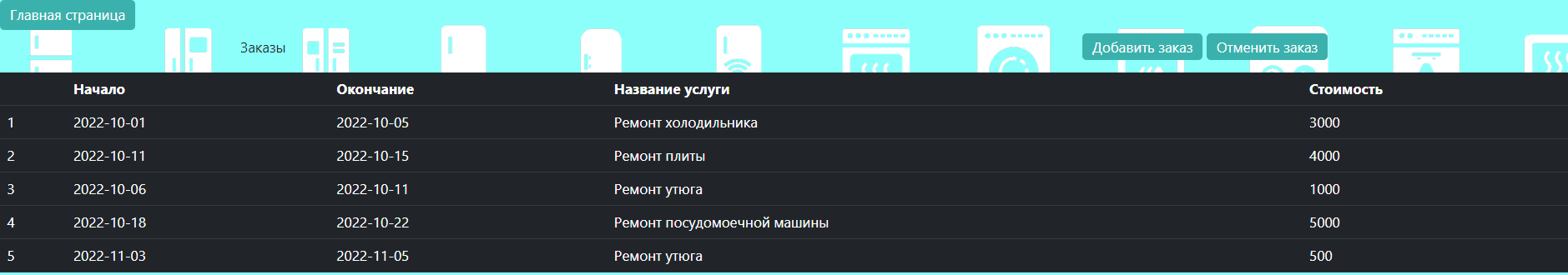


Рисунок 37 – Меню корзины

Ниже приведена форма для отмены заказа (рисунок 38).

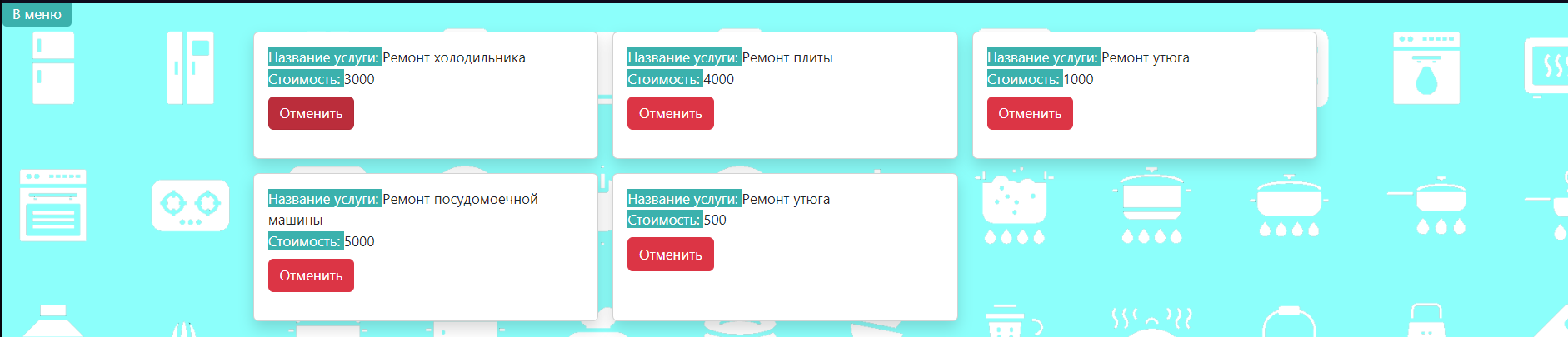


Рисунок 38 – Отмена заказа

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной курсовой работы была разработана информационная система «Сервисный центр по ремонту бытовой техники» – место, где люди могут выбрать ремонт своей бытовой техники. Был определен и реализован следующий функционал информационной системы:

− авторизация;

− регистрация;

− создание заказов;

− просмотр текущих клиентов;

− удаление заказов.

− редактирование заказов.

− создание сотрудников;

− удаление сотрудников.

− редактирование сотрудников.

Были рассмотрены возможные реализации ИС с использованием разных технологий и языков программирования, но в результате анализа инструментальных средств разработки выбор остановился на языке программирования PHP.

Был разработан браузерный программный продукт, с интуитивным и современным интерфейсом и широким функционалом.

В дальнейшем ИС может развиваться путём расширения функционала и совершенствования интерфейса, тем самым набирая большее сообщество пользователей, а также и актуальность программного продукта.

Все поставленные цели и задачи курсового проекта были успешно выполнены.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

* 1. Dmosk – Примеры SQL-запросов в MariaDB (MySQL) – URL: <https://www.dmosk.ru/miniinstruktions.php?mini=sql-mysql> (дата обращения: 11.10.2022). – Текст: электронный.
  2. FrontEnd Resource – 107 Beautiful CSS Cards examples to improve your UI – URL: https://frontendresource.com/css-cards/(дата обращения: 21.11.2022). – Текст: электронный.
  3. HTML5 BOOK – Основы CSS – URL: <https://html5book.ru/osnovy-css/> (дата обращения:03.10.2022). – Текст: электронный.
  4. Htmlbook – Самоучитель HTML4 – URL: <http://htmlbook.ru/samhtml> (дата обращения: 01.10.2022). – Текст: электронный.
  5. PHP – Объекты данных PHP – URL: <https://www.php.net/manual/ru/book.pdo.php> (дата обращения: 29.09.2022). – Текст: электронный.
  6. studwood – Анализ предметной области – URL:[АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ, Модель предметной области - Проектирование объектов базы данных "Ремонт компьютеров" (studwood.net)](https://studwood.net/1707220/informatika/analiz_predmetnoy_oblasti?ysclid=l89xcxi3y8378848729) (дата обращения: 20.09.2022). – Текст: электронный.
  7. Википедия – Сервисный центр – URL:[Сервисный центр — Википедия (wikipedia.org)](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B8%D1%81%D0%BD%D1%8B%D0%B9_%D1%86%D0%B5%D0%BD%D1%82%D1%80) (дата обращения: 20.09.2022). – Текст: электронный.
  8. sql academy – Интерактивный учебник по SQL – URL: https://sql-academy.org/ru/guide(дата обращения: 10.11.2022). – Текст: электронный.
  9. old.code – Работа с сессиями PHP – URL: http://old.code.mu/books/php/auth/rabota-s-sessiyami-php.html(дата обращения: 10.11.2022). – Текст: электронный.
  10. Stackoverflow – Реализовываем роли доступа на PHP сайте – URL: https://ru.stackoverflow.com/questions/231834/Реализовываем-роли-доступа-на-php-сайте(дата обращения: 14.11.2022). – Текст: электронный.

**Приложение А** ­– Техническое задание

**Министерство образования Иркутской области**

Государственное бюджетное профессиональное

образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

ИНФОРМАЦИОННАЯ СИСТЕМА

«СЕРВИСНЫЙ ЦЕНТР ПО РЕМОНТУ БЫТОВОЙ ТЕХНИКИ»

|  |  |  |
| --- | --- | --- |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (К.А. Дивонин) |

Иркутск 2022

**1 Общие сведения**

Наименование работы: информационная система «Сервисный центр по ремонту бытовой техники».

Исполнитель: студент иркутского авиационного техникума, группы ИС-20-1, Дивонин К.А.

Разработка информационной системы проходит в рамках курсовой работы.

Сроки разработки информационной системы с 15.09.2022 по 06.12.2022 года.

**2 Цели и назначение создания автоматизированной системы**

Целью курсовой работы является создание информационной системы «Сервисный центр по ремонту бытовой техники», которое включает в себя возможности создания заказов и ведение других бизнес-процессов.

В данной информационной реализуется следующий функционал:

− авторизация;

− регистрация;

− создание заказов;

− просмотр текущих клиентов;

− удаление записей.

**3 Характеристика объекта автоматизации**

Данная информационная система разрабатывается для облегчения ведения бизнеса компании по ремонту и обслуживанию бытовой техники. При помощи сайта будет проще создавать заказ.

**4 Требования к системе в целом**

**4.1 Требования к структуре и функционированию сайта**

Функции информационная система:

1. Окно «Авторизации»:
   1. авторизация пользователей.
2. Окно «Регистрация»:
   1. регистрация пользователей.
3. Окно «Главная страница»:
   1. Информация о заказах;
4. Окно «Услуги»:
   1. Таблица «Услуги»:
      1. добавление наименований;
      2. удаление наименований;
      3. редактирование данных.
   2. Окно «Пользователи»:
      1. добавление пользователей;
      2. удаление пользователей;
      3. редактирование данных пользователей.
   3. Окно «Сотрудники»:
      1. добавление сотрудников;
      2. удаление сотрудников;
      3. редактирование данных.

**4.2 Требования к надежности**

Для обеспечения надежности необходимо проверять корректность получаемых данных и реализовать валидность полей. Входные данные поступают в виде значений c клавиатуры. Эти значения отображаются в отдельных полях таблицы.

**4.3 Требования к безопасности**

Для обеспечения безопасности в информационной системе, необходимо реализовать разграничение прав доступа.

**4.4 Требования к эксплуатации, техническому обслуживанию, ремонту и**

**хранению компонентов системы**

Минимальные системные требования для сервера:

1. Процессор: Intel Pentium 4 2.0Ghz / AMD XP 2200+;
2. Оперативная память: 512 Мб;
3. Жёсткий диск: 150мб;
4. Операционная система: Windows 7/8/10.
5. Версия MySQL 5.0 и выше;

Минимальные системные требования для рабочей станции:

1. Процессор: Intel Pentium 4 2.0Ghz / AMD XP 2200+;
2. Оперативная память: 512 Мб;
3. Жёсткий диск: 150мб;
4. Операционная система: Windows 7/8/10.

**5 Состав и содержание работ по созданию системы**

В таблице 16 представлены плановые сроки начала и окончания работы по созданию.

Таблица 16 – Этапы выполнения разработки

|  |  |  |
| --- | --- | --- |
| Наименование этапов разработки | Срок | |
| Начало выполнения | Окончание выполнения |
| Предпроектное исследование предметной области (постановка цели, задач, описание области применения, исследование предметной области) | 15.09.22 | 21.09.22 |
| Разработка технического задания (выбор архитектуры программного обеспечения, выбор типа пользовательского интерфейса, выбор языка и среды программирования) | 22.09.22 | 26.09.22 |
| Проектирование программного обеспечения (разработка структурной и функциональной схемы ПО, проектирование базы данных (инфологическое, ER-модель, физическая модель) | 27.09.22 | 25.10.22 |
| Разработка (программирование) и отладка программного продукта | 26.10.22 | 28.11.22 |
| Составление программной документации (написание руководства пользователя) | 29.11.22 | 06.12.22 |

Приложение Б – Листинг roles.php

<?php  
require('db.php');  
class roles extends DB  
{  
 public function login($request)  
 {  
 $req = json\_decode($request);  
 $last\_name = $req->last\_name;  
 $name = $req->name;  
 $father\_name = $req->father\_name;  
 $password = $req->password;  
 $connect = $this->connect();  
 $sql = $connect->prepare('SELECT \* from users where last\_name=:last\_name and name=:name and father\_name=:father\_name and password=:pass');  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 'pass' => $password,  
 ]);  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 session\_start();  
 $\_SESSION['user'] = (object)[  
 'last\_name' => $data->last\_name,  
 'name' => $data->name,  
 'father\_name' => $data->father\_name,  
 'role' => $data->role  
 ];  
 }  
 }  
  
 public function registration($request)  
 {  
 $req = json\_decode($request);  
 $last\_name = $req->last\_name;  
 $name = $req->name;  
 $father\_name = $req->father\_name;  
 $password = $req->password;  
 $connect = $this->connect();  
 $sql = $connect->prepare('SELECT \* from users where last\_name=:last\_name and name=:name and father\_name=:father\_name and password=:pass');  
 $sql->execute(array(  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 "pass" => $password,  
 ));  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 return json\_encode([  
 'message' => "Такой пользователь существует"  
 ]);  
 }  
 $req = json\_decode($request);  
 $last\_name = $req->last\_name;  
 $name = $req->name;  
 $father\_name = $req->father\_name;  
 $connect = $this->connect();  
 $sql = $connect->prepare('SELECT \* from customers where last\_name=:last\_name and name=:name and father\_name=:father\_name');  
 $sql->execute(array(  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 ));  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 return json\_encode([  
 'message' => "Такой пользователь существует"  
 ]);  
 }  
 $sql = $connect->prepare("INSERT INTO users(last\_name,name,father\_name,password,role) values (:last\_name,:name,:father\_name,:pass,:role)");  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 "pass" => $password,  
 "role" => 3  
 ]);  
 $sql = $connect->prepare("INSERT INTO customers (last\_name,name,father\_name) values (:last\_name,:name,:father\_name)");  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 ]);  
 $sql = $connect->prepare('SELECT \* from users where last\_name=:last\_name and name=:name and father\_name=:father\_name and password=:pass');  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 "pass" => $password,  
 ]);  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 session\_start();  
 $\_SESSION['user'] = (object)[  
 'last\_name' => $data->last\_name,  
 'name' => $data->name,  
 'father\_name' => $data->father\_name,  
 'role' => $data->role  
 ];  
 return json\_encode([  
 'message' => 'Пользователь добавлен'  
 ]);  
 }  
 $sql = $connect->prepare('SELECT \* from customers where last\_name=:last\_name and name=:name and father\_name=:father\_name');  
 $sql->execute([  
 'last\_name' => $last\_name,  
 'name' => $name,  
 'father\_name' => $father\_name,  
 ]);  
 $data = $sql->fetch(PDO::*FETCH\_OBJ*);  
 if ($data) {  
 session\_start();  
 $\_SESSION['user'] = (object)[  
 'last\_name' => $data->last\_name,  
 'name' => $data->name,  
 'father\_name' => $data->father\_name,  
 ];  
 return json\_encode([  
 'message' => 'Пользователь добавлен'  
 ]);  
 } }  
}