

Оглавление

1.	Техническое задание	5
1.1	Наименование системы	5
1.2	Основания для разработки	5
1.3	Исполнитель	5
1.4	Назначение и цель разработки системы	5
1.5	Содержание работы	5
1.5.1	Задачи, подлежащие решению	5
1.5.2	Требования к архитектуре АСОИ	6
1.5.3	Требования к составу программных компонентов.....	6
1.5.4	Требования к прикладным программам	7
1.5.5	Требования к временным характеристикам	7
1.5.6	Требования к составу технических средств	7
1.6	Этапы разработки.....	7
1.7	Техническая документация, предъявляемая по окончании работы	8
1.8	Дополнительные условия	8
2.	Научно-исследовательская часть.....	9
2.1	Постановка задачи	9
2.2	Перечень задач, подлежащих решению в процессе разработки	10
2.3	Анализ существующих аналогов.....	11
2.4	Обоснование выбора языка, среды и платформы разработки.....	13
2.5	Выводы.....	16
3.	Проектно-конструкторская часть	17
3.1	Разработка структуры приложения	17
3.2	Разработка алгоритмов обработки информации.....	18
3.3	Логическая схема базы данных	20
3.4	Описание физической модели базы данных	20
3.5	Разработка архитектуры приложения	21
3.6	Разработка систем передачи информации.....	23
3.7	Разработка интерфейса взаимодействия пользователя с системой	26

3.7.1	Интерфейс сервера.....	26
3.7.2	Интерфейс клиента	26
4.	Проектно-Технологическая часть	27
4.1	Тестирование и отладка приложения.....	27
4.2	Разработка руководства пользователя	28
4.4	Системные требования	31
	Заключение	33
	Список использованных источников.....	34

1. ТЕХНИЧЕСКОЕ ЗАДАНИЕ

1.1 Наименование системы

Настоящее Техническое задание определяет требования и порядок создания клиент-серверного приложения для определения местоположения сотрудников.

1.2 Основания для разработки

Основанием для разработки приложения для определения местоположения сотрудников является задание в соответствии с рабочей программой дисциплины «Сетевые технологии в автоматизированных системах обработки информации и управления».

1.3 Исполнитель

Исполнителем проекта является студент Калужского филиала МГТУ им. Н. Э. Баумана, факультета ИУК «Информатики и управления», кафедры ИУК5 «Системы обработки информации», группы ИУК5-61Б, Антюхов Алексей Александрович.

1.4 Назначение и цель разработки системы

Разрабатываемое приложение предназначено для хранения и обновления данных о местоположении персонала.

Целью создания системы является закрепление полученных навыков работы с программными интерфейсами для обеспечения обмена данными между процессами.

1.5 Содержание работы

1.1.1 Задачи, подлежащие решению

В ходе реализации программного продукта необходимо разработать приложение-клиент, которое будет подключаться к серверу с удаленных компьютеров, разработать приложение-сервер, предназначенное для

параллельной обработки запросов и работающее в ОС Windows, реализовать взаимодействие двух частей приложения с использованием механизма сокетов.

1.1.2 Требования к архитектуре АСОИ

Архитектура системы должна быть достаточно гибкой, т.е. допускать относительно простое, без коренных структурных изменений, наращивания функциональности в соответствии с расширением задач её применения.

Должны быть обеспечены безопасность функционирования системы, надежная защита данных от ошибок.

Должен быть обеспечен эргономичный доступ пользователей к сервисам и результатам функционирования АСОИ на основе современных пользовательских интерфейсов.

1.1.3 Требования к составу программных компонентов

Программный продукт должен представлять собой протестированное клиент-серверное приложение, размещенное на удаленном сервере.

Для передачи данных между сервером и клиентом будет использоваться технология Windows Sockets API (далее Winsock).

Будут использоваться следующие библиотеки:

Winsock 2.0 – для функционирования Winsock

MySQLClient – для функционирования БД

Для представления визуальной части приложения будут использоваться стандартные аппаратные компоненты среды разработки Microsoft Visual Studio – Windows Forms (далее Winforms).

Каждое окно и поведение включённых в него элементов будет располагаться в отдельном файле с соответствующим названием. Для серверной части приложений клиента и сервера будет выделено по одному

файлу. Также клиент и сервер будут содержать один главный файл и функцией `main()`, в котором будут прописаны начальные инструкции приложений.

1.1.4 Требования к прикладным программам

Для работы БД необходимо приложение с поддержкой технологии MySQL. В моём случае будет использоваться МАР.

1.1.5 Требования к временным характеристикам

Система должна обеспечивать возможность одновременной работы 10 пользователей при следующих характеристиках времени отклика системы:

- подключение к серверу – не более 5 сек;
- получение ответа от сервера по запросу – не более 10 сек
- время, необходимое для функционирования протокола, передачи данных без разрыва соединения по истечению таймаута – не более 2 минут

1.1.6 Требования к составу технических средств

Для построения программного продукта необходимы следующие аппаратно-технические и программные средства:

- Персональный компьютер (ПЭВМ)
- В качестве серверной и клиентской частей допускается использование ПК, удовлетворяющего мин требованиям для ОС MS Windows 10
- Наличие сетевого оборудования для функционирования локальной ВС

1.6 Этапы разработки

На стадии разработки технического задания должен быть выполнен этап разработки, согласования и утверждения настоящего технического задания.

На исследовательской стадии должен быть выполнен обзор существующих аналогов программному продукту; выбор программного обеспечения, библиотек для создания продукта.

На стадии проектирования компонентов программного продукта выполняется проектирование архитектуры системы; состава и методов взаимодействия компонентов; алгоритмов обработки и представления данных.

На стадии реализации производится разработка и тестирование спроектированной программы, ввод ее в эксплуатацию, а также оформление технической документации.

1.7 Техническая документация, предъявляемая по окончании работы

Должны быть разработаны следующие программные документы:

1. Расчетно-пояснительная записка:

- техническое задание;
- научно-исследовательская часть;
- проектирований компонентов программного продукта;
- проектно-технологическая часть.

2. Графическая часть.

1.8 Дополнительные условия

Дополнительных условий не предъявляется.

2. НАУЧНО–ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

2.1 Постановка задачи

Предприятие — самостоятельный, организационно-обособленный хозяйствующий субъект с правами юридического лица, который производит и сбывает товары, выполняет работы, оказывает услуги. Часто в качестве частичных синонимов термина «предприятие» выступают понятия «фирма», «компания», «корпорация». Распространёнными примерами предприятий являются завод, фабрика, фермерское хозяйство, артель, производственный кооператив и другие.

Офис или контора — помещение, здание, комплекс зданий, в котором работают служащие предприятия (фирмы). В офисе (конторе) принимают клиентов, хранят и обрабатывают документы, архивы и тому подобное.

Главный офис — офис, в котором находится руководство предприятия, место пребывания центрального аппарата компании или предприятия, где размещаются топ-менеджмент компании или предприятия, дирекция, секретариат и другие важные административные подразделения.

Фабрика— промышленное предприятие, основанное на применении машин, характеризующееся крупномасштабным производством. Как правило, состоит из одного или нескольких расположенных неподалёку друг от друга строений, в которых располагаются цеха, сформированные по функции или по виду выпускаемой продукции, а также складские и офисные помещения.

Фабрики имеют большие площади, на них может работать несколько сотен рабочих одновременно.

Территория фабрики может быть разделена на зоны, в которых сотрудники выполняют различную работу, которые служат для определённых отличных друг от друга целей. Например, зона нарезки, зона сушки, зона упаковки, зона хранения.

Каждый работник имеет определённую должность и, соответственно, определённые обязанности и зону работы. Например, стеклодув и художник-декоратор ёлочных игрушек.

За таким большим количеством персонала нужно следить, чтобы эффективность работы не падала ниже определённой нормы, чтобы сотрудники не входили в места, в которых им не положено находиться. Видеонаблюдение может не дать полной картины происходящего на предприятии (слепые зоны), особенно в случае экстренной ситуации, например, из-за плохой видимости. Назначение на должность людей, контролирующую работу других сотрудников, может быть слишком затратно, также имеет место человеческий фактор, например, плохой сон или подкуп\коррупция. Использование же систем контроля местоположения сотрудников позволяет узнать местоположение каждого отдельного сотрудника предприятия в реальном времени, историю его перемещений, его время, проведённое эффективно, за работой.

Главной задачей курсовой работы является проектирование и создание веб-приложения с использованием базы данных для реализации возможностей обновления местоположения сотрудников и просмотра статистики по перемещениям в выбранной форме.

2.2 Перечень задач, подлежащих решению в процессе разработки

Для выполнения поставленной задачи необходимо определить оптимальную архитектуру для построения веб-приложения и средства реализации программного продукта, соответствующего данной архитектуре. Также необходимо исследовать существующие аналоги приложений для кинотеатров, разработать концептуальные и физические модели проектирования базы данных для приложения, разработать базу данных для

хранения необходимой информации, спроектировать и разработать визуальную часть приложения и провести её тестирование.

2.3 Анализ существующих аналогов

Разрабатываемая программа не является уникальной и имеет аналоги. Я не могу получить полный доступ к конкретным программам, которые используют в реальных предприятиях. Поэтому я возьму в качестве аналогов рекламные предложения подобных систем.

1) «GPSHome» - Сервис для контроля сотрудников в офисе внутри закрытых помещений.

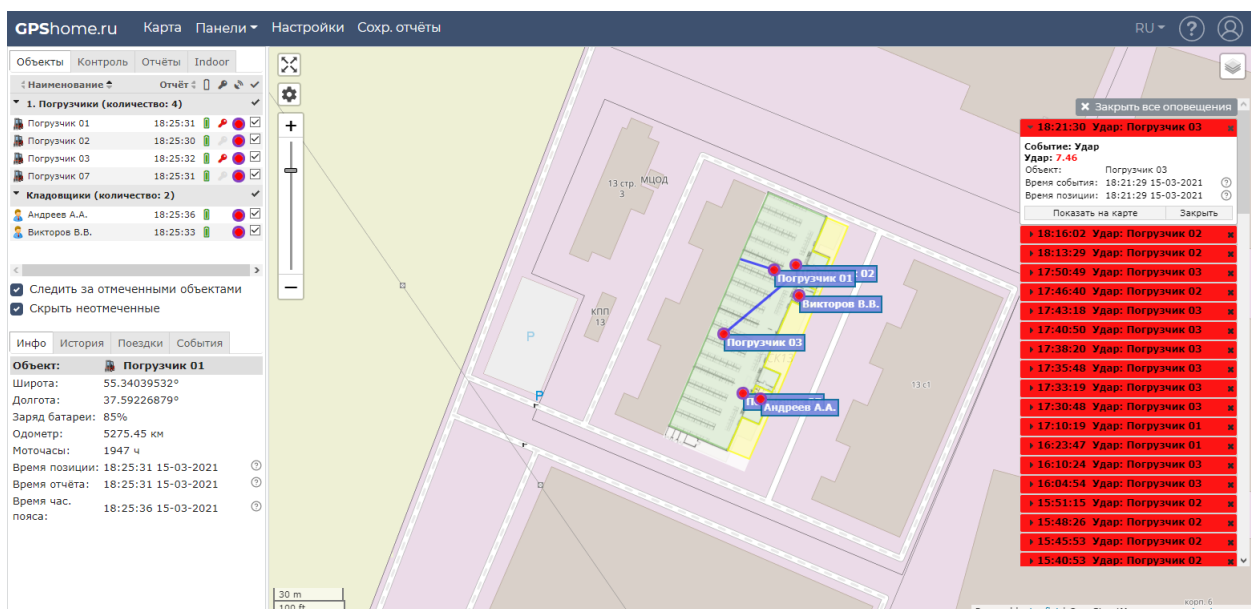


Рисунок 1 – демо-версия личного кабинета сервиса «GPSHome»

На сайте относительно подробно описан весь функционал сервиса.

Сервис GPSHome.ru предлагает решение для отслеживания сотрудников компании. Можно контролировать местоположение каждого сотрудника в режиме онлайн, получать статистику и визуализацию по всем перемещениям сотрудников за любой выбранный период. Можно формировать широкий перечень отчетов о фактическом нахождении всех сотрудников. Можно контролировать приход и уход сотрудников в офис и из офиса. Контролировать время нахождения на рабочем месте. Если в компании

существует ограничения для сотрудников на посещение ими определенных помещений, то сервис позволяет собирать статистику по посещению этих помещений, а также подавать сигнал тревоги при нахождении неавторизованных сотрудников в этих помещениях.

2) Система RealTrac

Модульная система мониторинга персонала для зданий и офисов.

ПРИНЦИП РАБОТЫ И КОМПОНЕНТЫ

Позиционирование персонала на промышленном предприятии может осуществляться по трем направлениям:

ЗОНАЛЬНОЕ ПОЗИЦИОНИРОВАНИЕ



Определение местоположения рабочего или техники с точностью до метра или геозоны.

- Точность: +/- 20 метров
- Дальность радиообнаружения: до 100 метров
- Конфигурация зоны: от 20 метров до зоны*

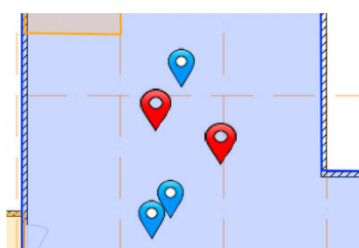
* - возможна настройка зон по требованию Заказчика

- Канал обратной связи: BLE или другой протокол передачи данных*

* - Под требования Заказчика

- Ограничения: Работа только в зоне радиовидимости точек доступа
- Затраты на внедрение: низкие

ГЛОБАЛЬНОЕ ПОЗИЦИОНИРОВАНИЕ



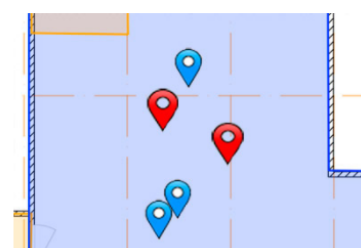
Определение местоположения рабочего или техники с точностью до метра.

- Точность: +/- 1-5 метров
- Дальность радиообнаружения: не ограничена
- Конфигурация зоны: от 10 м2
- Канал обратной связи: Wi-Fi, LoRa или другой протокол передачи данных*

* - Под требования Заказчика

- Ограничения: Работа в условиях видимости минимум 4х спутников
- Затраты на внедрение: низкие

ГИБРИДНОЕ ПОЗИЦИОНИРОВАНИЕ



Объединяет в себе преимущества зонального и глобального позиционирования.

- Точность: +/- 1-5 (20) метров
- Дальность обнаружения: не ограничена
- Конфигурация зоны: от 10 м2
- Канал обратной связи: Wi-Fi, LoRa или другой протокол передачи данных*

* - Под требования Заказчика

- Ограничения: В помещении работает зональное позиционирование. На открытой территории работает глобальное позиционирование
- Затраты на внедрение: средние

Рисунок 2 – направления позиционирования персонала системы RealTrac «Производство»

В отличии от прошлой системы, эта ориентирована на большие производственные предприятия, а не на офисные помещения и здания.

Решаемые задачи:

- Контроль местоположения рабочих на территории предприятия
- Контроль уровней доступа рабочих
- Контроль рабочего времени, времени, проведенного в геозоне
- Аварийное оповещение персонала в случае ЧП
- Контроль эвакуации персонала

2.4 Обоснование выбора языка, среды и платформы разработки

В качестве основного языка разработки был выбран C# и разработка с использованием Winforms в Microsoft Visual Studio.

Основные языки программирования для разработки десктопных приложений для MS Windows – C++, C#, Java. Далее проведу небольшое сравнение.

C# в сравнении с C++ и Java имеет поддержку аппаратных средств для работы с графическим интерфейсом внутри основной среды разработки MS Visual Studio. В случае с C++ и Java необходимо дополнительное ПО в виде библиотек, фреймворков или сред разработки с необходимым функционалом.

C++ является наиболее производительным языком программирования из представленных. Но в данном случае приложение не будет сильно загружать систему, поэтому жертва скоростью и удобством программирования на C# и Java в угоду производительности на C++ не целесообразна. В то же время на MS Windows язык программирования C# будет более производительным, чем Java, так как разрабатывался специально под Windows.

Остальные отличия касаются функционала и его реализации в конкретном языке, поэтому затрагивать их не имеет смысла.

Среди сред разработки под Windows выделяются MS Visual Studio и MS Visual Studio Code. Они были разработаны компанией Microsoft специально под Windows и являются наиболее удобными, производительными, с богатым функционалом.

VS Code поддерживает множество языков программирования (можно расширить установкой соответствующих плагинов и библиотек) и содержит в себе среду их выполнения, по сути являясь усовершенствованным редактором кода. Это больше подходит для поддержки готового кода, тестирования, веб-разработки.

VS «из коробки» также поддерживает все основные языки программирования за исключением Java (можно расширить установкой соответствующих плагинов и библиотек), но в то же время имеет в разы больший функционал в сравнении с VS Code. Например, создание и работа с БД, конструктор графического интерфейса, фреймворк. VS имеет аппаратную поддержку таких технологий как WinForms и WinSock. Именно эти технологии я собираюсь использовать.

VS Code может работать с WinSock, но не может полноценно работать с WinForms, так как может только скомпилировать код. Мне же нужна возможность удобно редактировать графический интерфейс моего приложения. Соответственно мой выбор – MS Visual Studio.

Подробнее про WinSock. Это технология поддерживается только ОС Windows и является частью среды разработки VS. .NET имеет аппаратную поддержку данной технологии в лице класса `System.Net.Sockets.Socket`, поэтому для работы достаточно прописать `using System.Net.Sockets;`

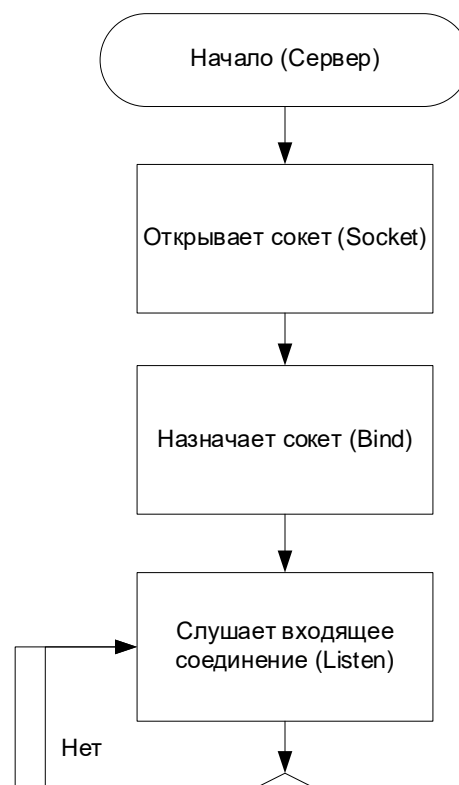


Рисунок 3 – Блок-схема алгоритма работы серверного сокета (часть 1)

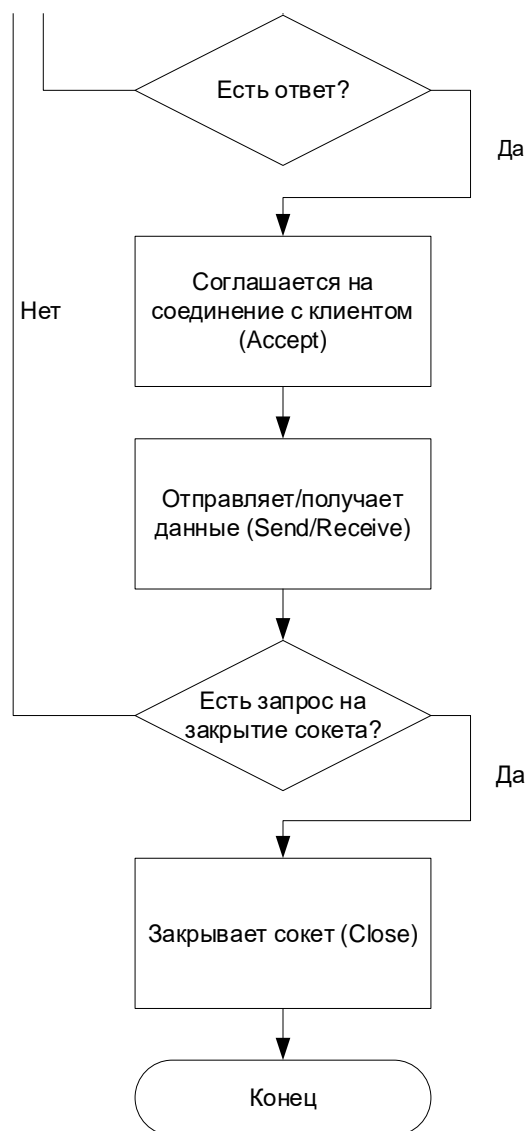


Рисунок 4 – Блок-схема алгоритма работы серверного сокета (часть 2)

Основными ОС для ПК на сегодняшний день являются MS Windows, Mac OS и Linux OS. Проведём сравнение.

Сразу же отклоню Mac OS, так как она поставляется только с техникой компании Apple, которая имеет цену намного выше, чем аналогичная под управлением другой ОС. В данном случае переплата в несколько раз за излишнюю производительность и отсутствие возможности дальнейшего апгрейда нецелесообразна. Linux бесплатна, базовая версия Windows стоит 200\$ в официальном магазине компании Microsoft.

Linux имеет много версий и вариаций, но подавляющее большинство из них имеют большее быстродействие, чем Windows, при одинаковых

комплектующих. Но в то же время ОС на базе Linux из-за их разнообразия может не иметь нужных драйверов для установленных комплектующих, когда под Windows драйвера будут всегда. Т.е. мы жертвуем некоторой производительностью ради удобства установки и работы с ОС.

Также стоит отметить, что Windows является самой популярной в мире ОС. Вероятно, не человека, который бы был знаком с техникой, но не пользовался Windows. Linux же из трёх представленных ОС является наименее популярной и установлена на менее чем 5% ПК по всему миру. Чтобы полноценно пользоваться Linux нужно время на адаптацию, в случае с Windows это не требуется.

2.5 Выводы

Таким образом, исходя из требований к реализуемой системе, рассмотрения возможностей наиболее подходящих инструментов, вариантов разработки и последующего их сравнения было решено использовать следующие решения:

Основной язык программирования — C#. Для разработки приложения была выбрана интегрированная среда разработки Microsoft Visual Studio. В качестве базы данных — MySQL.

3. ПРОЕКТНО-КОНСТРУКТОРСКАЯ ЧАСТЬ

3.1 Разработка структуры приложения

Приложение состоит из трёх частей. Из клиента, написанного с использованием Winforms, сервера и реляционной базы данных MySQL. То есть я использую «трёхуровневую архитектуру».

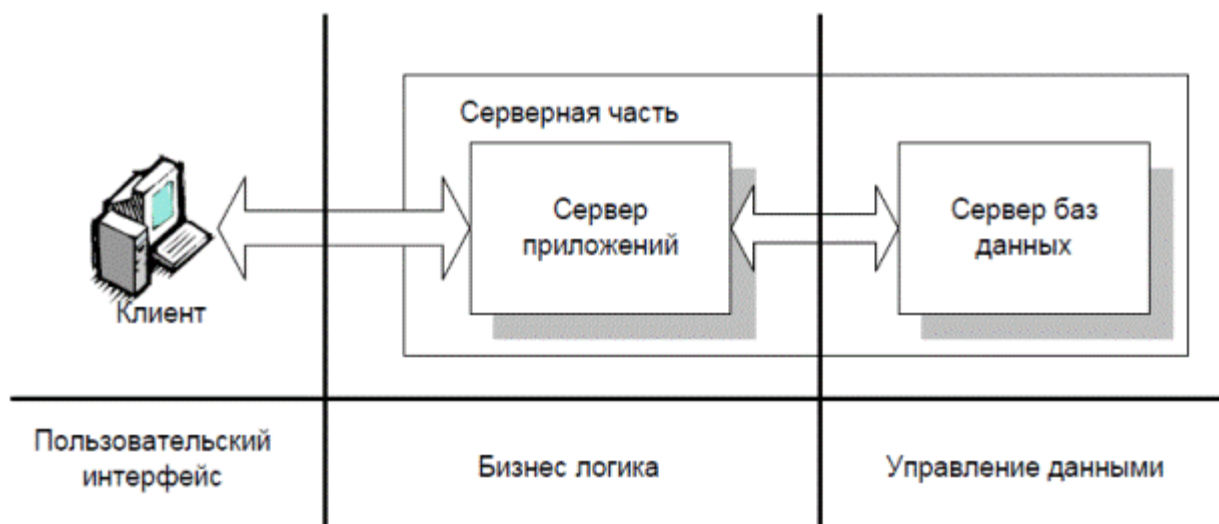


Рисунок 5 - Макет структуры приложения

Пользователь через интерфейс приложения-клиента выбирает действие, которое он хочет произвести. Клиент через сокет отправляет серверу сообщение с операцией, которую требуется выполнить, и входные данные для составления запроса к БД.

Вся работа по подготовке запросов происходит на сервере. Полученные от клиента входные данные он подставляет в макет нужного запроса и отправляет его БД.

БД хранит данные и отвечает на приходящие от сервера запросы.

Сначала необходимо разработать БД и сервер и наладить между ними подключение. Так можно выявить основные проблемы в процессе получения данных от БД. Если обмен данными между сервером и БД происходит стабильно и ожидаемым образом, значит ошибок в этой части структуры нет.

Далее необходимо разработать клиент и его взаимодействие с сервером. Так как сервер исправно получает данные от БД, нужно создать интерфейс для передачи этих данных от сервера на клиент через сокет.

Приложение для взаимодействия с базой данных написано на языке C# при помощи Windows Forms. Windows Forms позволяет разрабатывать интеллектуальные клиенты. Интеллектуальный клиент — это приложение с полнофункциональным графическим интерфейсом, простое в развертывании, способное работать при наличии или отсутствии подключения к Интернету и использующее более безопасный доступ к ресурсам на локальном компьютере по сравнению с традиционными приложениями Windows.

3.2 Разработка алгоритмов обработки информации

Все функции приложения активируются после нажатия соответствующих кнопок.

При помощи функций коннектора `mySqlConnection.Open()`; и `mySqlConnection.Close()`; либо при помощи созданного класса DB, в который встроены эти функции, открывается и закрывается доступ к базе данных.

После открытия доступа используя, `MySqlCommand` и `MySqlDataAdapter`, программа получает доступ к базе данных. Считывает их от туда таблицы, добавляет или изменяет строки в таблицах, согласно написанному на SQL коду.

Пример команды:

```
MySqlCommand command = new MySqlCommand("SELECT * FROM  
`users` WHERE `login` = @ln AND `password` = @pswd",  
db.getConnection());
```

Каждая кнопка на клиенте при нажатии запускает проверку соединения с сервером. При отсутствии такового, производится повторное подключение.

Для обмена данными, полученными от запроса к БД необходима сериализация. Для передачи данных между клиентом и сервером используется протокол TSP.

Поскольку TSP передает поток байтов данных, будет необходима сериализация необходимых данных пакетов при отправке и их десериализация у получателей.

Каждая кнопка на клиенте при нажатии запускает проверку соединения с сервером. При отсутствии такового, производится повторное подключение. Каждое сообщение сериализуется у отправителя, десериализуется и обрабатывается у отправителя, и, в случае сервера, отправляется ответ клиенту по такому же механизму.

На сервере происходит обработка запроса от клиента. В соответствии с полученными входными данными составляется запрос к БД. После получения таблицы с ответом, данные таблицы считываются в переменную, которая далее сериализуется и отправляется клиенту.

3.3 Логическая схема базы данных

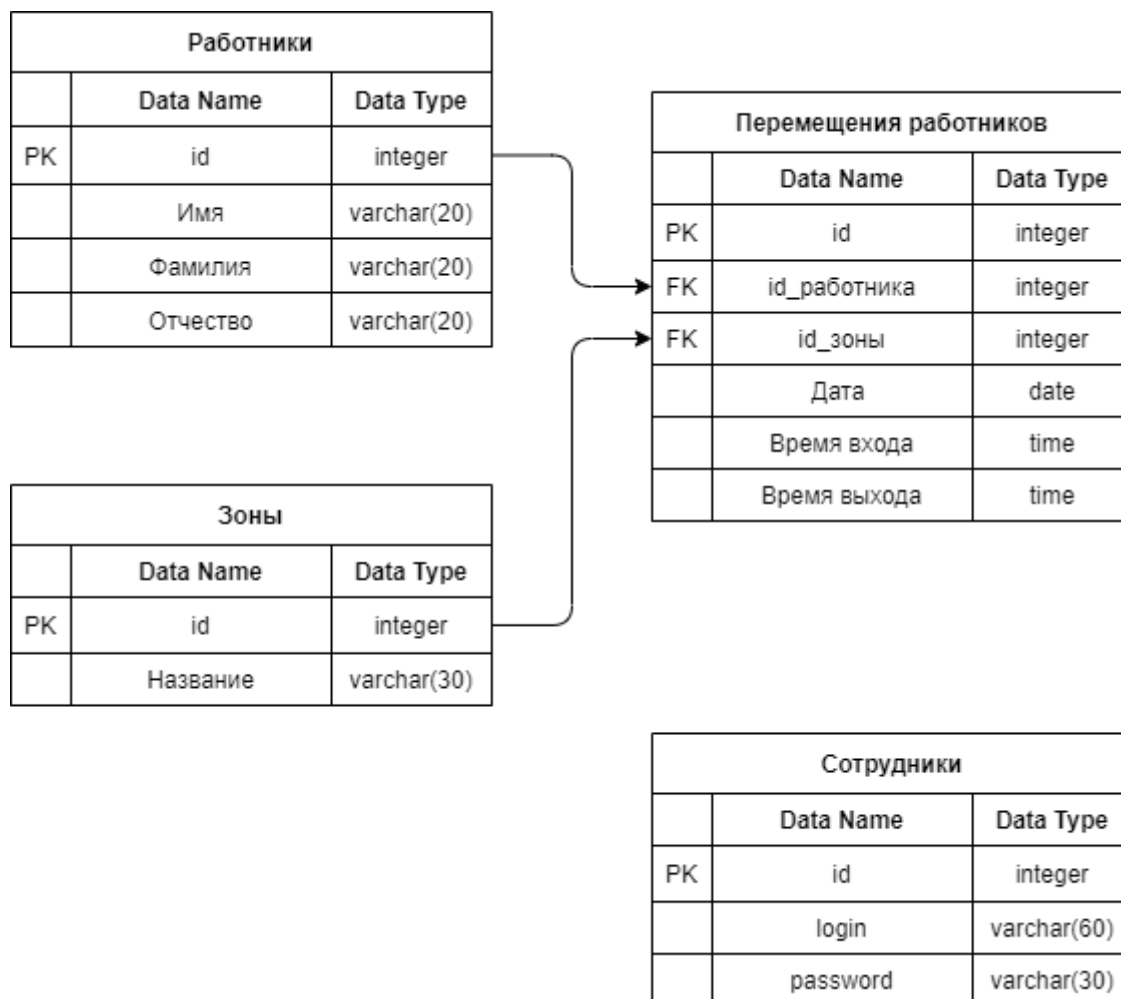


Рисунок 6 – Схема логической модели данных

3.4 Описание физической модели базы данных

Опишем сущности, использованные в базе данных.

`users` - Сущность для сотрудников, работающих с приложением. Она имеет 3 поля.

- ID(Первичный ключ) тип данных bigint(20) (serial)
- Логин тип данных varchar(60)
- Пароль тип данных varchar(30)

`members` - Сущность работника. В ней присутствуют 4 поля

- ID(Первичный ключ) тип данных bigint(20) (serial)
- Имя тип данных varchar(20)
- Фамилия тип данных varchar(20)
- Отчество тип данных varchar(20)

`rooms` - Сущность работника. В ней присутствуют 2 поля

- ID(Первичный ключ) тип данных bigint(20) (serial)
- Название тип данных varchar(30)

`st` - Сущность перемещения работников. Имеет 6 полей

- | | |
|-------------------------------|--------------------------------|
| • ID (Первичный ключ) | тип данных bigint(20) (serial) |
| • id работника (Внешний ключ) | тип данных int(11) |
| • id зоны (Внешний ключ) | тип данных int(11) |
| • Дата входа/выхода | тип данных date |
| • Время входа | тип данных time |
| • Время выхода | тип данных time |

На данный момент есть понимание, что было бы целесообразнее использовать для даты и времени входа/выхода 2 поля типа datetime и уже из них по отдельности извлекать дату и время.

3.5 Разработка архитектуры приложения

В приложения используется архитектура клиент-сервер-БД. В данном случае один клиент, один центральный сервер, на котором размещена логика программы, и БД, в которой хранятся данные.

Характеристика клиент-сервер-БД описывает отношения взаимодействующих программ в приложении. Серверный компонент предоставляет функцию или услугу одному или нескольким клиентам, которые инициируют запросы на такие услуги.

Вообще говоря, служба — это абстракция компьютерных ресурсов, и клиенту не нужно беспокоиться о том, как сервер работает при выполнении запроса и доставке ответа. Клиенту нужно только понять ответ, основанный на известном протоколе приложения, то есть содержание и форматирование данных для запрашиваемой услуги.

Клиенты и серверы обмениваются сообщениями в шаблоне запрос-ответ. Клиент отправляет запрос, а сервер возвращает ответ. Этот обмен сообщениями является примером межпроцессного взаимодействия. Для взаимодействия компьютеры должны иметь общий язык, и они должны следовать правилам, чтобы и клиент, и сервер знали, чего ожидать. Язык и правила общения определены в протоколе связи. Все протоколы клиент-серверной модели работают на уровне приложений. Протокол прикладного уровня определяет основные шаблоны диалога.

Приложение разбивается на две части, которые могут выполняться в разных узлах сети, - клиентскую и серверную части. Прикладная программа или конечный пользователь взаимодействуют с клиентской частью системы, которая обеспечивает надсетевой интерфейс. Клиентская часть системы при потребности обращается по сети к серверной части. Интерфейс серверной части определен и фиксирован. Поэтому возможно создание новых клиентских частей существующей системы.

Разделен код программы клиентского и серверного приложения. Это главное преимущество архитектуры. Выбрана локальная сеть. Поэтому мы имеем следующие плюсы:

- к клиентским рабочим станциям выдвигают низкие запросы;

- преимущественно все вычислительные операции выполняются на серверах;
- гибкая система;
- реально повысить защиту локальной сети

3.6 Разработка систем передачи информации

Для взаимодействия между сервером и клиентами будет использовать потоковые сокеты (на основе TCP) - сокеты с установленным соединением на основе протокола TCP, передают поток байтов, который может быть двунаправленным - т. е. приложение может и получать, и отправлять данные.

Сокет состоит из IP-адреса и порта. Для работы с сокетами используется библиотеку `System.Net.Sockets`.

Функции, которые мы будем использовать в нашем приложении для работы с сокетами показаны в таблице 1.

Используемая функция	Назначение функции
<code>TcpListener(IPAddress, Int32)</code>	Инициализирует новый экземпляр класса <code>TcpListener</code> , который выполняет ожидание входящих попыток подключения для заданных локального IP-адреса и номера локального порта.
<code>AcceptSocket()</code>	Принимает ожидающий запрос на подключение.

AcceptTcpClient()	Принимает ожидающий запрос на подключение.
Start()	Запускает ожидание входящих запросов на подключение.
Stop()	Закрывает слушатель
TcpClient()	Инициализирует новый экземпляр класса TcpClient.
Close()	Удаляет данный экземпляр TcpClient и запрашивает закрытие базового подключения TCP.
Connect(IPAddress, Int32)	Подключает клиента к удаленному TCP-узлу, используя указанный IP-адрес и номер порта.

GetStream()	Возвращает объект NetworkStream, используемый для отправки и получения данных.
NetworkStream(Socket)	Создает новый экземпляр класса NetworkStream для указанного объекта Socket.
Close()	Закрывает текущий поток и отключает все ресурсы (например, сокеты и файловые дескрипторы), связанные с текущим потоком. Вместо вызова данного метода, убедитесь в том, что поток надлежащим образом ликвидирован.
Serialize(Stream, Object)	Сериализует объект или граф объектов с заданным корнем в предоставляемом потоке.
Deserialize(Stream)	Десериализует данные в предоставленный поток и воспроизводит граф объектов.

Работа с сокетом во многом схожа с работой с файловым объектом. Принцип - открыли соединение - считали данные - закрыли соединение.

Слушающий сокет прослушивает соединения от клиентов. Когда клиент подключается, сервер вызывает `AcceptTcpClient()`, чтобы принять подключение.

Клиент вызывает `TcpClient()`, чтобы установить соединение с сервером.

Для отправки данных используется `Serialize(Stream, Object)`, который автоматически записывает в поток данные для передачи другой стороне.

3.7 Разработка интерфейса взаимодействия пользователя с системой

3.7.1 Интерфейс сервера

Т.к. главная задача сервера – постоянное функционирование и ведение логов работы, сервер сделан в виде консольного приложения. В окне будут выводиться логи о происходящих событиях и тексты ошибок, если таковые возникнут.

3.7.2 Интерфейс клиента

У пользователей, в отличие от сервера, интерфейс должен быть более сложным.

В окне авторизации пользователь должен ввести логин и пароль и нажать кнопку ОК.

В главном окне появляются два сценария использования: внести данные в БД и получить данные из БД в нужной пользователю форме.

Все данные для совершения действия пользователь выбирает из соответствующего всплывающего списка.

Внесение и получение данных осуществляется по нажатию кнопок «Отправить» и «Получить».

4. ПРОЕКТНО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

4.1 Тестирование и отладка приложения

Для проверки отказоустойчивости приложения будет выполнено нагрузочное тестирование сервера. Для этого будет запущено некоторое количество клиентов. Так как сервер в рамках курсовой сервер рассчитан на одного клиента, все клиенты, запущенные после первого, являются не смогут взаимодействовать с сервером.

При попытке авторизации они зависнут при запущенном первом клиенте и вылетят при закрытом первом клиенте.

Сервер и первый запущенный клиент при этом работают в штатном режиме.

Тест-кейс №2 Проверка авторизации пользователя

Шаги:

1. В поле “login” вводим логин
2. В поле “password” вводим пароль
3. Нажимаем кнопку “ОК”

Действие пользователя	Ожидаемый результат
Пользователь ввел логин, которого нет в БД	В консоли сервера сначала выводится сообщение о попытке авторизации, после о неудачной авторизации. У клиента выводится сообщение о неправильном логине или пароле.
Пользователь ввел пароль, который не соответствует	В консоли сервера сначала выводится сообщение о попытке

введённому логину, который есть в БД	авторизации, после о неудачной авторизации. У клиента выводится сообщение о неправильном введённом логине или пароле.
Пользователь оставил поле логина пустым	В поле логина появится красная надпись «login», сигнализирующая о его отсутствии
Пользователь оставил поле пароля пустым	В поле пароля появится красная надпись «password», сигнализирующая о его отсутствии
Пользователь ввёл логин и пароль, которые являются заведомо правильными	В консоли сервера сначала выводится сообщение о попытке авторизации, после об удачной авторизации. У клиента закроется окно авторизации и откроется основное диалоговое окно.

4.2 Разработка руководства пользователя

Для того, чтобы обычный пользователь начал пользоваться приложением, ему необходимо:

- 1) Запустить приложение MAMP.exe , с помощью двойного клика по файлу и дождаться запуска БД

- 2) запустить файл Server.exe, с помощью двойного клика по файлу
- 3) запустить файл WindowsFormsApp1.exe, с помощью двойного клика по файлу

Далее в открывшемся диалоговом окне необходимо ввести логин и пароль и нажать кнопку «ОК» (Рис. 6). После этого откроется главное окно (Рис. 7)

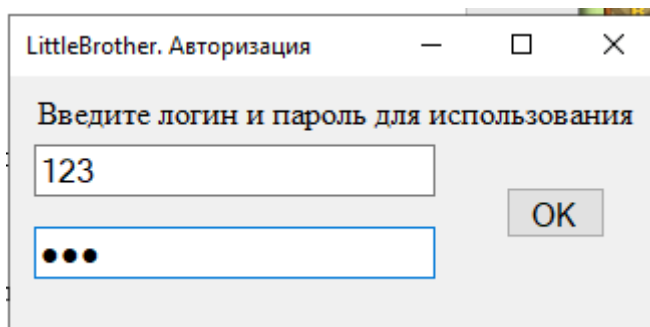


Рисунок 6 — Ввод логина и пароля

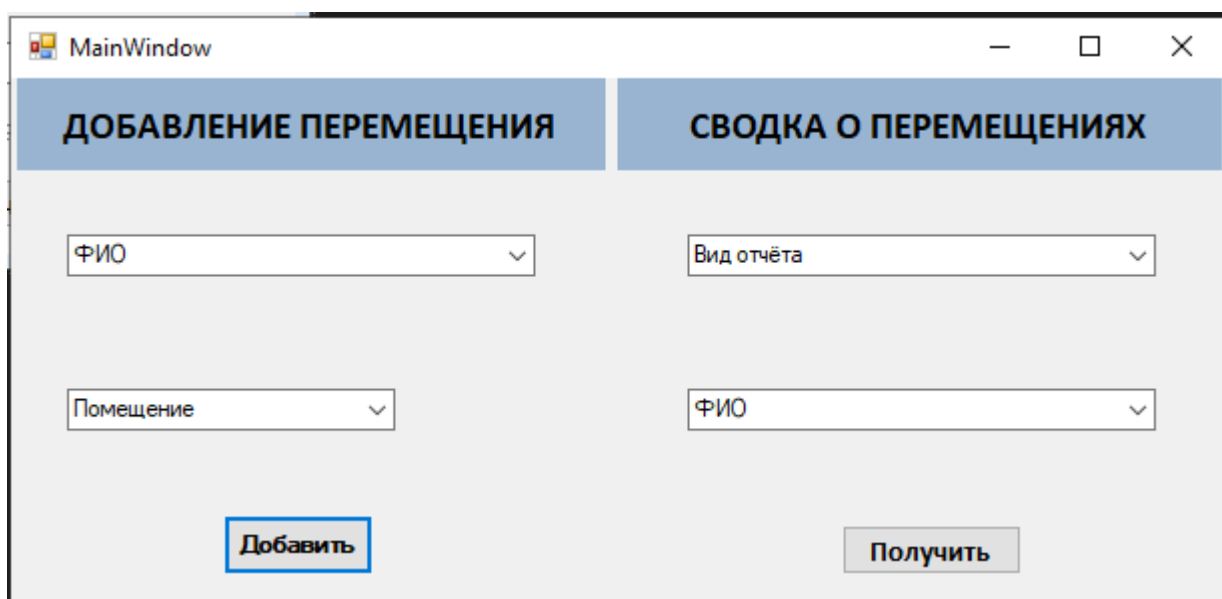
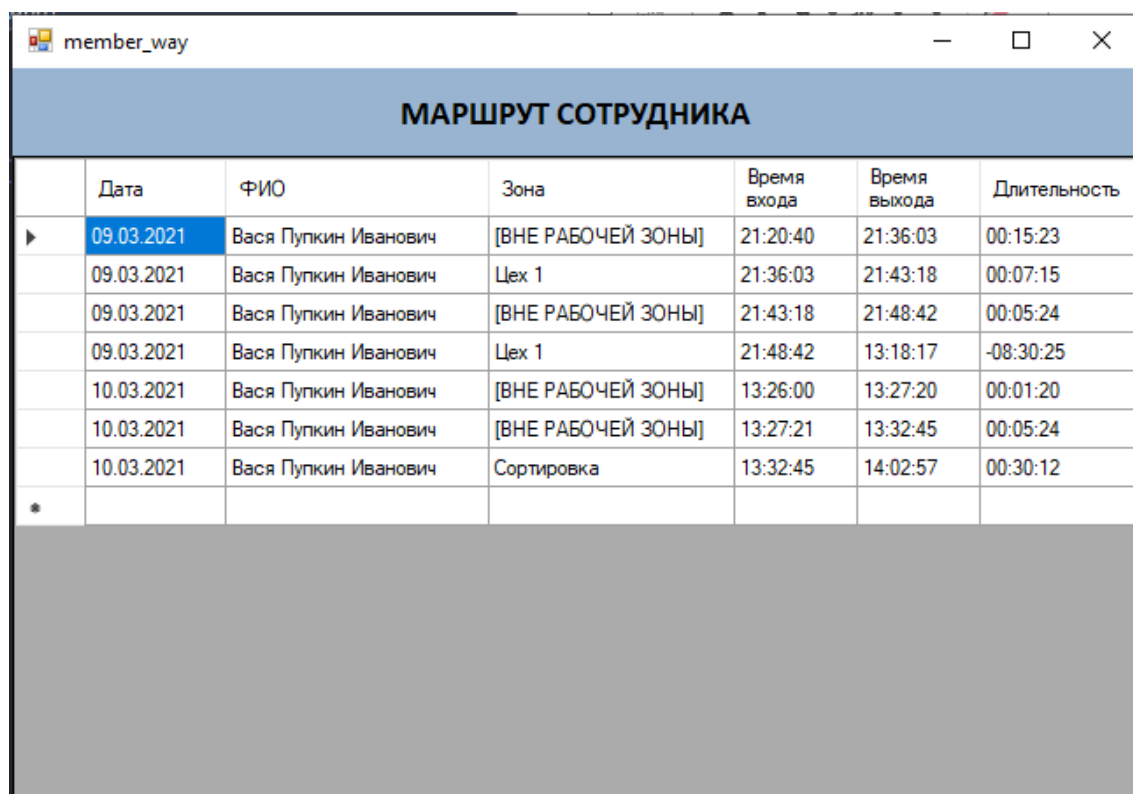


Рисунок 7 - Главное меню

Для добавления перемещения работника необходимо в левой области «ДОБАВЛЕНИЕ ПЕРЕМЕЩЕНИЯ» выбрать полное имя нужного сотрудника в поле «ФИО» и зону в поле «Помещение», в которую он должен переместиться, нажать кнопку «Добавить».

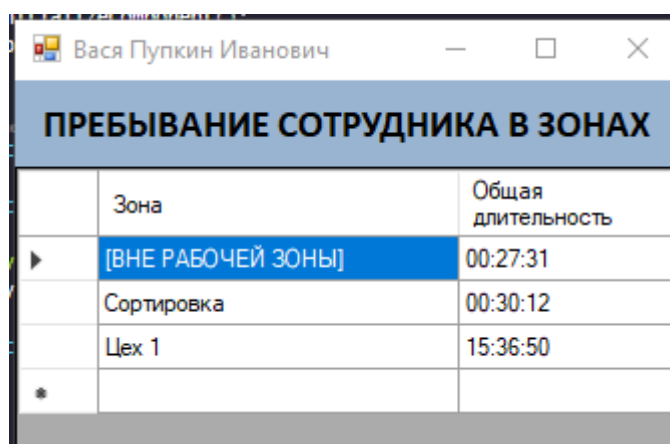
Для просмотра сводок необходимо в правой области «СВОДКА О ПЕРЕМЕЩЕНИЯХ» выбрать вид желаемого отчёта в поле «Вид отчёта» и полное имя сотрудника в поле «ФИО» и нажать кнопку «Получить». После нажатия кнопки может быть открыто одно из трёх окон: «Маршрут сотрудника» (Рис. 8), «Пребывание сотрудника в зонах» (Рис. 9), «Переработка сотрудника» (Рис. 10).



The screenshot shows a window titled 'member_way' with a blue header 'МАРШРУТ СОТРУДНИКА'. Below the header is a table with 7 columns: an empty column with a right-pointing triangle, 'Дата', 'ФИО', 'Зона', 'Время входа', 'Время выхода', and 'Длительность'. The table contains 8 rows of data for the employee 'Вася Пупкин Иванович' on various dates in March 2021, showing his movement between different zones and the duration of each stay.

	Дата	ФИО	Зона	Время входа	Время выхода	Длительность
▶	09.03.2021	Вася Пупкин Иванович	[ВНЕ РАБОЧЕЙ ЗОНЫ]	21:20:40	21:36:03	00:15:23
	09.03.2021	Вася Пупкин Иванович	Цех 1	21:36:03	21:43:18	00:07:15
	09.03.2021	Вася Пупкин Иванович	[ВНЕ РАБОЧЕЙ ЗОНЫ]	21:43:18	21:48:42	00:05:24
	09.03.2021	Вася Пупкин Иванович	Цех 1	21:48:42	13:18:17	-08:30:25
	10.03.2021	Вася Пупкин Иванович	[ВНЕ РАБОЧЕЙ ЗОНЫ]	13:26:00	13:27:20	00:01:20
	10.03.2021	Вася Пупкин Иванович	[ВНЕ РАБОЧЕЙ ЗОНЫ]	13:27:21	13:32:45	00:05:24
	10.03.2021	Вася Пупкин Иванович	Сортировка	13:32:45	14:02:57	00:30:12
*						

Рисунок 8 – Окно «Маршрут сотрудника»



The screenshot shows a window titled 'Вася Пупкин Иванович' with a blue header 'ПРЕБЫВАНИЕ СОТРУДНИКА В ЗОНАХ'. Below the header is a table with 3 columns: an empty column with a right-pointing triangle, 'Зона', and 'Общая длительность'. The table contains 4 rows of data, summarizing the total duration of the employee's stays in different zones.

	Зона	Общая длительность
▶	[ВНЕ РАБОЧЕЙ ЗОНЫ]	00:27:31
	Сортировка	00:30:12
	Цех 1	15:36:50
*		

Рисунок 9 – Окно «Пребывание сотрудника в зонах»



The screenshot shows a window titled 'overtime' with a blue header bar containing the text 'ПЕРЕРАБОТКА СОТРУДНИКА'. Below the header is a table with five columns: 'ФИО', 'План, чч', 'Фактически, чч', and 'Переработка, чч'. The first row of data contains 'Вася Пупкин Иванович', '8,0000', '16,5758', and '8,5758'. The second row is empty. The table has a grey background and a blue selection bar on the left side of the first row.

	ФИО	План, чч	Фактически, чч	Переработка, чч
▶	Вася Пупкин Иванович	8,0000	16,5758	8,5758
*				

Рисунок 10 – Форма «Переработка сотрудника»

Для того, чтобы покинуть чат, необходимо нажать на крестик, в правом верхнем углу.

4.4 Системные требования

Рекомендуемые операционные системы:

Windows 7

Windows 8

Windows 8.1

Windows 10

Рекомендуемая аппаратная конфигурация для сервера:

- Процессор Pentium 4 или более новый
- 512 МБ RAM для 64-битной сборки
- 100 МБ свободного дискового пространства

Рекомендуемая аппаратная конфигурация для клиента:

- Процессор Pentium 4 или более новый
- 200 МБ RAM
- 200 МБ RAM для 64-битной сборки
- 10 МБ свободного дискового пространства

Требования подключения к сети: Соединение клиента с сервером через витую пару третьей категории, подключение к сети интернет не обязательно. Минимальная скорость соединения: 50 кбит/с.

Заключение

При выполнении курсовой работы на тему «Приложение для определения местоположения сотрудников» была исследована и описана предметная область, проведен анализ аналогов данной системы, а также выбор инструментов и платформы для разработки.

Проведен анализ объектов автоматизации и разработаны методы решения технических задач. Также была разработана и реализована структура и архитектура системы, структура системы передачи информации и интерфейса взаимодействия пользователя с системой. Также проведено тестирование и отладка приложения.

Результатом данной курсовой работы является рабочее приложение для определения местоположения сотрудников.

Список использованных источников

- 1) Ачилов Р.Н. Построение защищенных корпоративных сетей. М: ДМКпресс, 2013. https://e.lanbook.com/book/66472?category_pk=1547#book_name
- 2) Васильков А.В., Васильков А.А., Васильков И.А. Информационные системы и их безопасность. М.: Форум, 2013.
- 3) Е.В. Смирнова, А.В. Пролетарский, Е.А. Ромашкина и др. Технологии коммутации и маршрутизации в локальных компьютерных сетях. Учебное пособие. М.: МГТУ им. Н.Э. Баумана, 2013.
- 4) Сидоров В.Н., Сломинская Е.Н., Полникова Т.В., Макарова О.Ю. Оформление графической части выпускной квалификационной работы. Учебное пособие. М.: МГТУ им. Н.Э. Баумана, 2016.
- 5) Э.Таненбаум, Д.Уэзеролл. Компьютерные сети. 5-е изд. СПб.: Питер, 2012.
- 6) MahApps.Metro. Вопросы и ответы. [Электронный ресурс] – URL:// <https://progi.pro/mahappsmetro-t8102>
- 7) Асинхронные методы, `async` и `await`. [Электронный ресурс] – URL:// <https://metanit.com/sharp/tutorial/13.3.php>.
- 8) Маршрутизация событий. [Электронный ресурс] – URL:// <https://metanit.com/sharp/wpf/6.php>
- 9) Полное руководство по языку программирования C# 9.0 и платформе .NET 5. [Электронный ресурс] – URL:// <https://metanit.com/sharp/tutorial/>
- 10) Реализация уведомления об изменении свойства. [Электронный ресурс] – URL:// <https://docs.microsoft.com/ru-ru/dotnet/desktop/wpf/data/how-toimplement-property-change-notification?view=netframeworkdesktop-4.8>
- 11) Сокеты в C# и .Net. [Электронный ресурс] – URL:// <https://metanit.com/sharp/net/3.1.php>