**Министерство образования Иркутской области**

Государственное бюджетное профессиональное образовательное учреждение Иркутской области

«Иркутский авиационный техникум»

(ГБПОУИО «ИАТ»)

|  |  |
| --- | --- |
| **ДП.09.02.03.22.191.10.ПЗ** | УТВЕРЖДАЮ  Зам. директора по УР, к.т.н.  \_\_\_\_\_\_\_\_\_\_\_Е.А. Коробкова |

**ПРИЛОЖЕНИЕ**

**«Умная камера»**

|  |  |  |
| --- | --- | --- |
| Нормконтролер: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (В.А. Пролыгина) |
| Консультант по экономической части: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Рачкова) |
| Руководитель: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (М.А. Кудрявцева) |
| Студент: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) | (C.А. Евдокимов) |

Иркутск 2023

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc135782436)

[1 Предпроектное исследование 5](#_Toc135782437)

[1.1 Описание предметной области 5](#_Toc135782438)

[1.2 Анализ инструментальных средств реализации 7](#_Toc135782439)

[2 Техническое задание на разработку программного продукта 9](#_Toc135782440)

[3 Проектирование 10](#_Toc135782441)

[3.1 Архитектура программного обеспечения 10](#_Toc135782442)

[3.2 Функциональное проектирование 10](#_Toc135782443)

[3.3 Проектирование базы данных 17](#_Toc135782444)

[3.4 Проектирование пользовательского интерфейса 20](#_Toc135782445)

[4. Реализация программного обеспечения 22](#_Toc135782446)

[4.1 Разработка базы данных 22](#_Toc135782447)

[4.2 Разработка программного продукта 23](#_Toc135782448)

[5 Документирование программного обеспечения 31](#_Toc135782449)

[5.1 Руководство пользователя 31](#_Toc135782450)

[6. Стоимость разработки и внедрения программного продукта 34](#_Toc135782451)

[6.1 Организационно-экономическое обоснование проекта 34](#_Toc135782452)

[6.2 Расчет затрат на разработку программного продукта 34](#_Toc135782455)

[6.3 Расчет затрат на внедрение программного продукта 39](#_Toc135782456)

[6.4 Основные выводы 39](#_Toc135782457)

[ЗАКЛЮЧЕНИЕ 40](#_Toc135782458)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 41](#_Toc135782459)

[Приложение А – Обнаружения лиц с камеры 43](#_Toc135782460)

[Приложение Б – Сохранение списка прошедших на предприятие 46](#_Toc135782461)

ВВЕДЕНИЕ

Приложение – это программа, функции которой состоят в выполнении специфических преобразований информации и вычислений, предоставления для пользователя удобного и легко осваиваемого интерфейса.

В настоящее время автоматизация процессов в различных сферах деятельности становится все более актуальной. Одним из таких процессов является контроль за рабочим временем сотрудников. Традиционные методы контроля, такие как журналы учета рабочего времени или механические устройства для регистрации входа и выхода, становятся все менее эффективными и удобными для использования.

В этой связи, все большую популярность получают умные камеры, которые способны автоматически определять время прихода и ухода сотрудников на рабочее место. Такие камеры обладают широким функционалом и могут использоваться в различных организациях, начиная от малых предприятий и заканчивая крупными корпорациями.

В данной работе будет рассмотрена технология умных камер и ее применение для контроля за рабочим временем сотрудников. В частности, будут рассмотрены принципы работы умной камеры, а также ее основные функции и возможности. Будет проанализировано применение умных камер в различных организациях и оценены их преимущества и недостатки по сравнению с традиционными методами контроля за рабочим временем.

Кроме того, в работе будут рассмотрены вопросы безопасности и конфиденциальности при использовании умных камер, а также будут предложены рекомендации по оптимальному использованию этой технологии в организациях.

Необходимо выполнить следующие задачи:

1. исследование предметной области;
2. выборка среды разработки приложения;
3. создание приложения, используя различные средства разработки;
4. описание руководство пользователя;
5. подготовка технической документации;
6. проектирование программного обеспечения;
7. разработка и отладка программного продукта;
   1. Предпроектное исследование
   2. Описание предметной области

Предметная область для диплома по теме "Умная камера" для определения времени прихода и ухода сотрудников на рабочее место включает в себя различные аспекты, связанные с техническими и организационными аспектами создания и эксплуатации системы контроля доступа и учета рабочего времени.

Технические аспекты охватывают разработку и установку умных камер, оснащенных компьютерным зрением и искусственным интеллектом для автоматического распознавания лиц, а также необходимых систем хранения и обработки данных.

Организационные аспекты включают в себя разработку политики безопасности, установление процедур учета рабочего времени и контроля доступа, обеспечение совместимости системы с законодательными требованиями и обязательствами по защите персональных данных.

Для решения задачи определения времени прихода и ухода сотрудников на рабочее место, система умной камеры может включать в себя следующие функции:

1. Распознавание лиц: камера автоматически распознает лицо сотрудника при прибытии и уходе с рабочего места.
2. Регистрация времени: система регистрирует время прихода и ухода каждого сотрудника на рабочее место, используя информацию, полученную от умной камеры.
3. Хранение данных: система хранит данные о времени прихода и ухода каждого сотрудника на сервере, который может быть доступен только уполномоченным лицам.
4. Обработка данных: система может автоматически обрабатывать данные о рабочем времени, вычислять количество отработанных часов, делать расчеты заработной платы, и предоставлять отчеты.
5. Уведомления: система может отправлять уведомления сотрудникам о просроченном прибытии или раннем уходе, а также уведомления руководству о невыполнении графика работы.

Одной из основных проблем, связанных с использованием системы умной камеры для учета рабочего времени, является защита персональных данных сотрудников. Поэтому необходимо учитывать законодательные требования и разработать политику безопасности, которая обеспечивает конфиденциальность персональных данных, таких как изображения лиц, время прихода и ухода, а также гарантирует их защиту от несанкционированного доступа.

Также возможны некоторые трудности при распознавании лиц сотрудников при использовании умной камеры. Например, различные условия освещения или недостаточное качество изображения могут повлиять на точность распознавания. Для улучшения точности распознавания можно использовать дополнительные технические решения, такие как дополнительные источники освещения или более высококачественные камеры.

Также важно обеспечить корректное функционирование системы умной камеры, ее регулярное обслуживание и техническую поддержку. В случае сбоев в работе системы, необходимо обеспечить быстрое устранение проблем и восстановление работоспособности.

Итак, для реализации проекта "Умная камера для определения времени прихода и ухода сотрудников на рабочее место" необходимо учитывать все вышеперечисленные технические и организационные аспекты, провести соответствующую подготовку и разработку необходимой инфраструктуры, а также обеспечить правильную эксплуатацию и поддержку системы.

* 1. Анализ инструментальных средств реализации

Для разработки приложения можно использовать следующие инструментальные средства разработки и языки программирования:

Visual Studio – это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Помимо стандартного редактора и отладчика, которые есть в большинстве сред IDE, Visual Studio включает в себя компиляторы, средства автозавершения кода, графические конструкторы и многие другие функции для улучшения процесса разработки.

C# – современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет разработчикам создавать разные типы безопасных и надежных приложений, выполняющихся в .NET.

SQL («язык структурированных запросов») – декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Draw.io — инструмент для создания диаграмм, блок-схем, интеллект-карт, бизнес-макетов, отношений сущностей, программных блоков и другого. Сервис распространяется на бесплатной основе с открытым исходным кодом.

Для выбора инструментов разработки были произведены сравнения, которые показаны в таблице 1, таблице 2 и таблице 3.

Таблица 1 – Сравнение языков программирования для разработки программного продукта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Критерии | C# | Java | Python | C++ |
| Наличие библиотек | + | + | + | + |
| Инструменты для работы с БД | + | + | + | + |
| Объектно-ориентированные возможности | + | + | + | + |

Таблица 2 – Сравнение сред разработки

|  |  |  |
| --- | --- | --- |
| Название IDE | Visual Studio | Intellije IDEA |
| Автосохранение | + | + |
| Автодополнение | + | + |
| Поиск по коду | + | + |

Таблица 3 – Сравнение средств реализации базы данных

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

Вывод, из представленных языков программирования был выбран C#, так как именно этот язык более перспективный в разработке desktop-приложений.

Для разработки данного приложения лучше всего подойдет Visual Studio потому, что данная программа имеет очень хороший функционал.

Для реализации приложения подходит MySQL, поскольку предлагает высокую производительность и масштабируемость, что делает его идеальным для крупных проектов и приложений с большим объемом данных, так же она имеет большое и активное сообщество разработчиков, а также широкий выбор ресурсов для поддержки и обучения.

1. Техническое задание на разработку программного продукта

В начале разработки создавалось техническое задание, в котором указывались основные требования.

Для создания технического задания использовался стандарт ГОСТ 34.602-89.

Согласно ГОСТ 34.602-89 техническое задание должно включать следующие разделы:

1. общие сведения;
2. назначение и цели создания системы;
3. требования к системе в целом;
   1. требования к структуре и функционированию системы;
   2. требования к надежности;
   3. требования к безопасности;
   4. требования к эксплуатации, техническому обслуживанию, ремонту и хранению компонентов системы;
4. требования к документированию;
5. состав и содержание работ по созданию системы.

Техническое задание на разработку приложения представлено отдельным документом.

1. Проектирование
   1. Архитектура программного обеспечения

Архитектура программного обеспечения для системы умной камеры для определения времени прихода и ухода сотрудников на рабочее место может состоять из нескольких компонентов, которые будут взаимодействовать друг с другом:

Компонент умной камеры - представляет собой аппаратное обеспечение, которое осуществляет захват изображений сотрудников при проходе через зону контроля. Для распознавания лиц используется как классический алгоритм распознавания, так и нейросетевые алгоритмы.

Клиент распознавания лиц - представляет собой программное обеспечение, которое получает изображение с камеры, производит распознавание лица сотрудника и определяет время его прихода и ухода. Этот компонент использует библиотеку распознавания лиц – OpenCV.

База данных - хранит информацию о сотрудниках, их фотографии и информацию о времени их прихода и ухода на рабочее место.

Интерфейс пользователя - предоставляет интерфейс для пользователей для просмотра данных о приходе и уходе сотрудников и взаимодействие с базой данных для получения и отображения информации о сотрудниках.

* 1. Функциональное проектирование

Проектирование приложения происходит при помощи CASE средств, которые помогают обеспечить высокое качество программ, отсутствие ошибок и простоту в обслуживании программных продуктов.

На рисунке 2 изображена диаграмма прецедентов, которая показывает структурную схему приложения «Умная камера» для пользователя «Охранник», «Администратор».

Данная схема отображает действия, выполняемые оператором и администратором. «Охранник» и «Администратор» являются – актерами. «Просмотр заявок», «Работа с заявками» и «Работа с товарами» – общие действия. «Работа с сотрудниками» и «Вывод на печать» являются действиями Администратора.

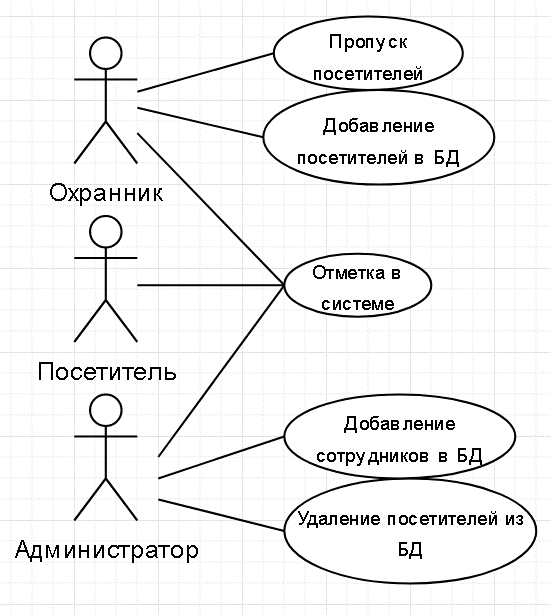


Рисунок 2 – Диаграмма прецедентов

Таким образом, представленные диаграммы демонстрируют взаимодействие основных объектов приложения и их действия.

Диаграмма последовательности является видом диаграмм взаимодействия языка UML, которые описывают отношения объектов в различных условиях. Условия взаимодействия задаются сценарием, полученным на этапе разработки [диаграмм вариантов использования](https://pro-prof.com/archives/2594).

Сценарий заполнения расписания показан на рисунке 3.

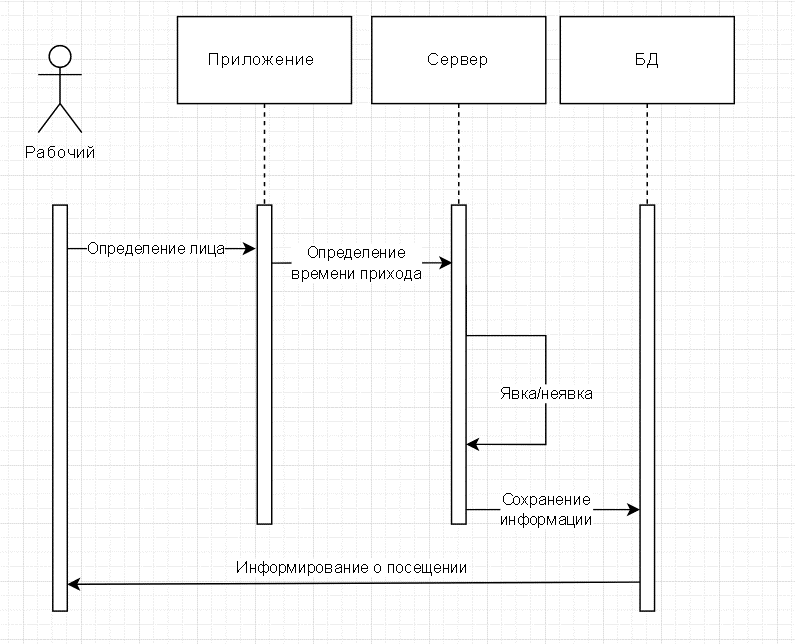


Рисунок 3 –Диаграмма последовательности заполнения заявок

Диаграмма деятельности – это UML**-**диаграмма, на которой показаны действия, состояния которых описано на диаграмме состояний. Под деятельностью понимается спецификация исполняемого поведения в виде координированного последовательного и параллельного выполнения подчинённых элементов **-** вложенных видов деятельности и отдельных действий, соединённых между собой потоками. Диаграмма моделирует динамические аспекты поведения системы или, другими словами, последовательные (а иногда и параллельные) шаги процесса.

Рисунок 4 представляет собой диаграмму деятельности, которая поэтапно описывает все процессы. На данной диаграмме будут описаны все действия, которые совершат актеры. В нашем случае, актерами являются «Охранник», «Посетитель» и «Администратор».

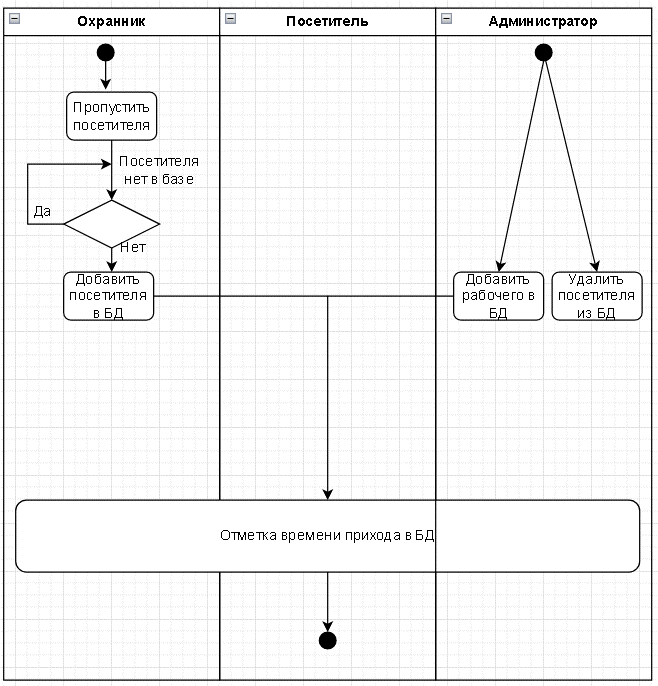


Рисунок 4 – Диаграмма деятельности информационной системы «Расписание занятий в техникуме»

Диаграмма компонентов (рисунок 5) показывает разбиение программной системы на структурные компоненты и связи между компонентами.

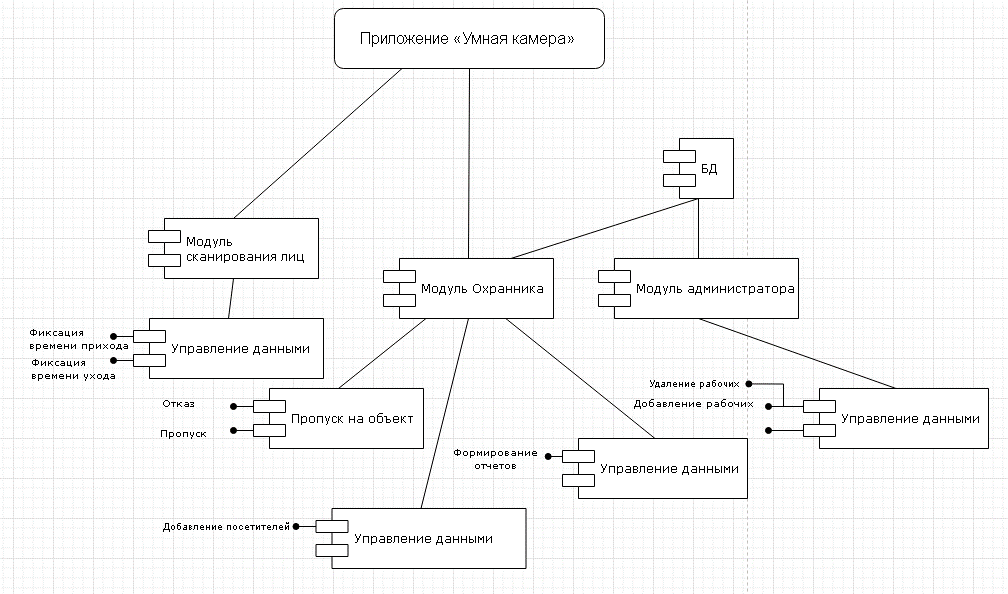


Рисунок 5 – Диаграмма компонентов

На рисунке 6 представлена диаграмма развертывания – это тип диаграммы, которая определяет физическое оборудование, на котором будет работать программная система. Он также определяет способ развертывания программного обеспечения на базовом оборудовании. Он отображает программные части системы на устройство, которое будет выполнять его.

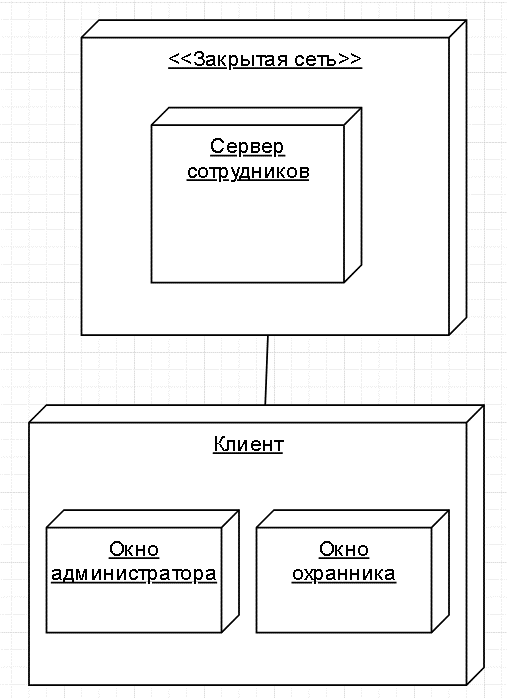


Рисунок 6 – Диаграмма развертывания

Контекстная диаграмма – это модель, представляющая систему как набор иерархических действий, в которой каждое действие преобразует некоторый объект или набор объектов.

На рисунке 7 изображена контекстная диаграмма, на которой показаны входные данные, управление, механизм, выходные данные и функция.

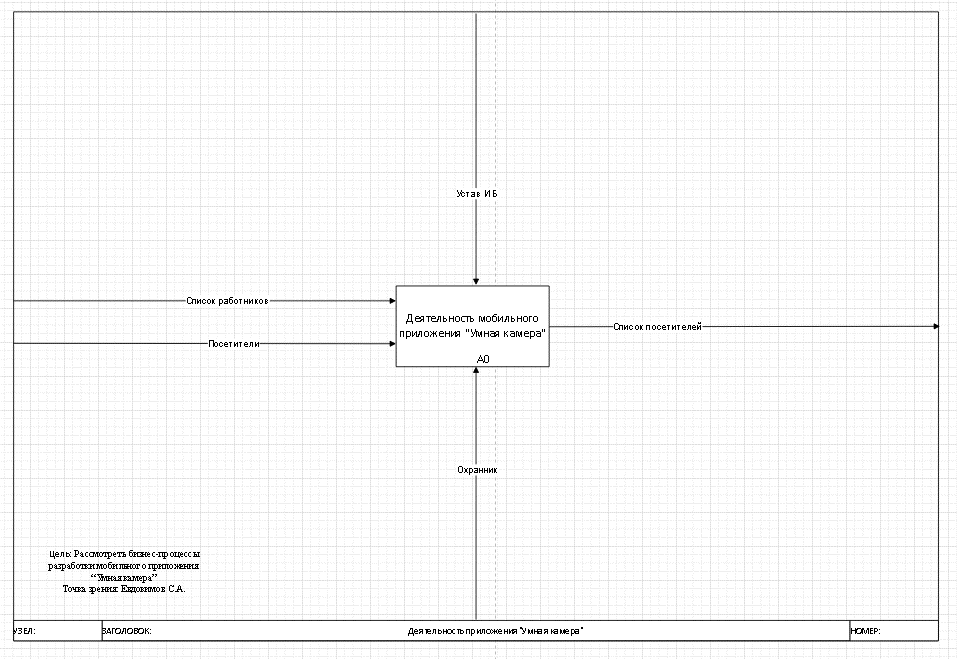


Рисунок 7 – Контекстная диаграмма

На рисунке 8 показана диаграмма декомпозиций, которая расписывает функции программы. На этой диаграмме показаны такие функции, как: «Учет статуса преподавателей», «Учет статуса кабинетов», «Внесение изменений в расписание занятий»

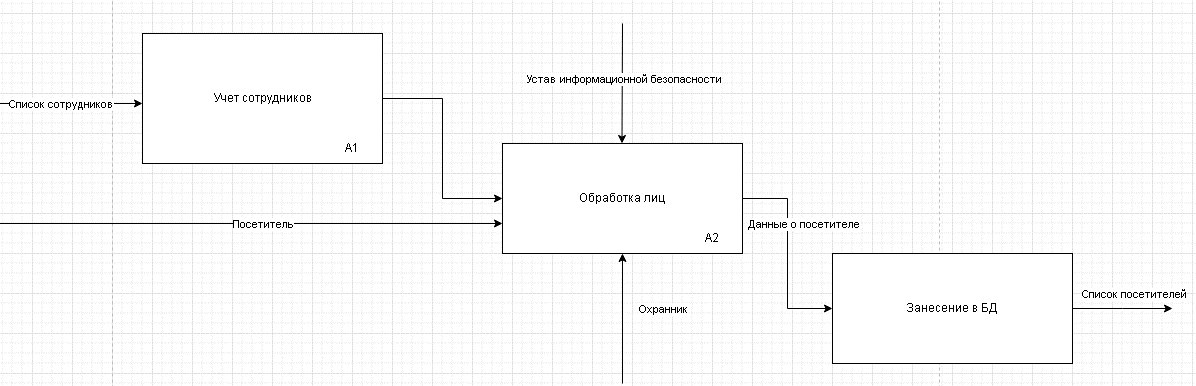


Рисунок 8 – Диаграмма декомпозиции времени прихода

Также в ходе работы была разработана диаграмма классов, приведенная на рисунке 9, которая отображает внутреннюю работу ИС.

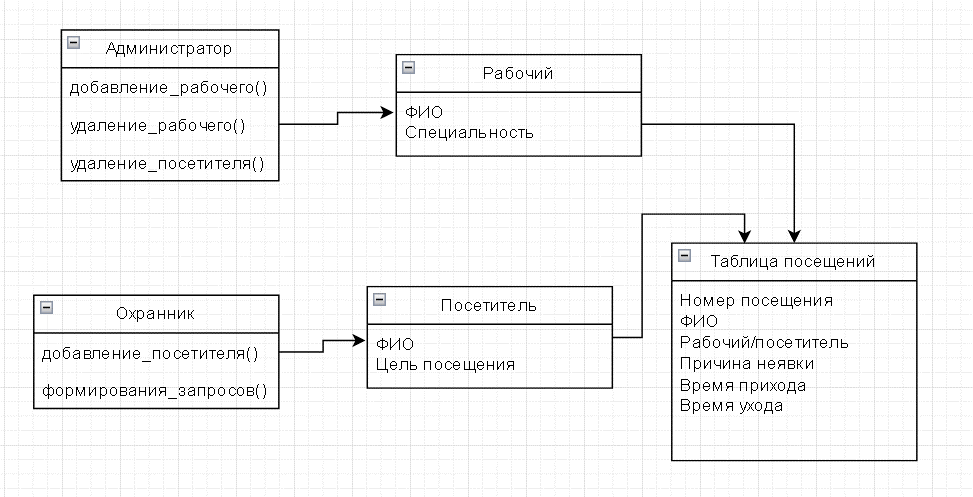


Рисунок 9 – Диаграмма классов

Диаграмма потоков данных, представленная на рисунке 10. Так называется методология графического структурного анализа, описывающая внешние по отношению к системе источники, и адресаты данных, логические функции, потоки данных и хранилища данных, к которым осуществляется доступ.

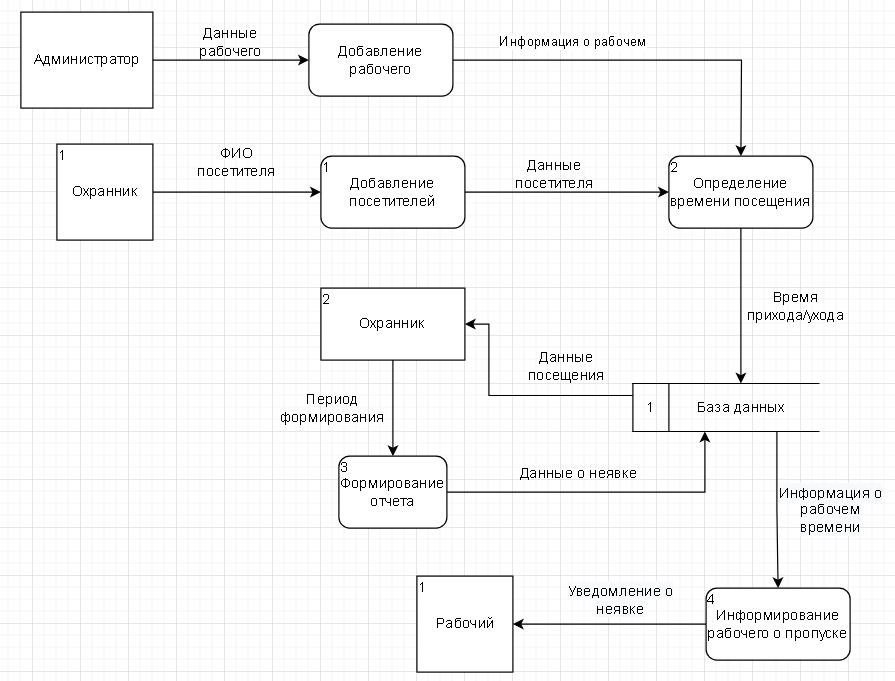


Рисунок 10 – Диаграмма потоков данных

Функциональное проектирование представляет наиболее общий подход к описанию систем. Определяются граничные условия и желательные входы, и выходы, составляется подробный перечень функций или операций, которые должны выполняться. При функциональном проектировании осуществляется синтез структуры и определяются основные параметры объекта и его составных частей (элементов), оцениваются показатели эффективности и качества процессов функционирования.

* 1. Проектирование базы данных

Для проектирования будущей базы данных использовался MySQL Workbench.

На схеме базы данных представлены 4 таблицы: «person», «visiter», «recognition» и «time\_to\_work». Поля id являются первичными ключами.

На рисунке 11 изображена ER-модель базы данных.

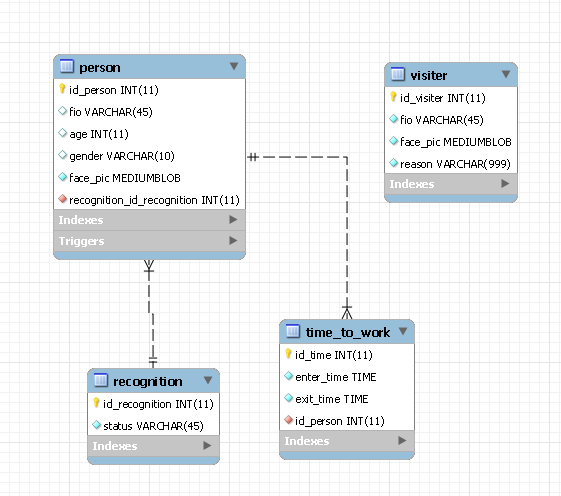


Рисунок 11 – ER-модель базы данных

В таблицах 4, 5, 6, 7 представлены описания полей с типами данных для каждой таблицы данной ER-модели.

Таблица 4 – «person»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_person | INT | Уникальный идентификатор человека |
| fio | VARCHAR(45) | ФИО человека |
| age | INT | Возраст человека |
| gender | VARCHAR(10) | Пол человека |
| face\_pic | VARBINARY(999999) | Фотография лица человека в байтах |
| recognition\_id\_recognition | INT | Идентификатор группы доступа к которой относится человек |

Таблица 5 – «recognition»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_recognition | INT | Уникальный идентификатор опознавания |
| status | VARCHAR(45) | Статус опознанного человека |

Таблица 6 – «visiter»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_visiter | INT | Уникальный идентификатор посетителя |
| fio | VARCHAR(45) | ФИО посетителя |
| face\_pic | VARBINARY(999999) | Фотография лица посетителя в байтах |

Таблица 7 – «time\_to\_work»

|  |  |  |
| --- | --- | --- |
| Поле | Тип данных | Описание |
| id\_time | INT | Уникальный идентификатор времени |
| enter\_time | TIME | Время начала рабочей смены |
| exit\_time | TIME | Время окончания рабочей смены |
| id\_person | INT | Идентификатор рабочего которому присвоено время работы |

База данных приведена к 3-ой нормальной форме, так как все поля, принимаемые больше одного значения, декомпозированы, и каждый не ключевой атрибут не транзитивно зависит от первичного ключа.

* 1. Проектирование пользовательского интерфейса

Пользовательский интерфейс — средства, позволяющие пользователю эффективно взаимодействовать с устройствами компьютера достаточно удобным для себя образом.

На рисунке 12 изображён прототип главной страницы приложения. На данном прототипе можно увидеть приблизительный дизайн страницы. В левой части находятся изображение с камеры. В средней части находятся быстрое добавление гостей в базу, чтобы зафиксировать информацию о посетителе и переход в «Админ-панель». В правой части находится таблица с временем прихода и информацией о сотруднике.

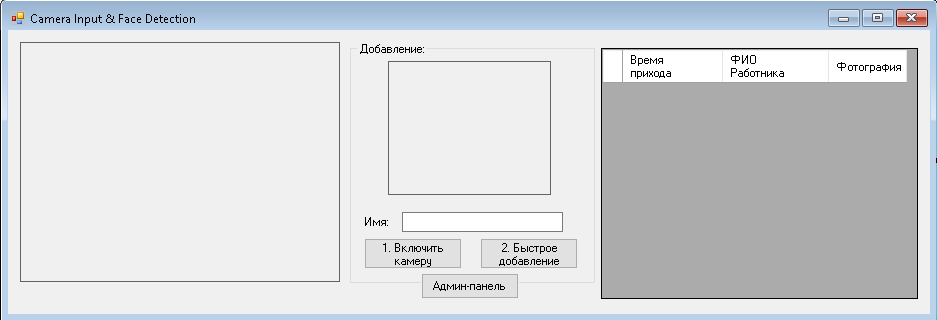


Рисунок 12 – Прототип главного окна.

На рисунке 13 изображен прототип страницы «Админ-панель». На данном прототипе можно будет добавлять сотрудников в базу данных для дальнейшего распознавания.

В левой части находится поле изображение с камерой для добавления фотографии сотрудника, чтобы программа определяла лицо при входе. В правой части находятся поля для заполнения данными о сотруднике.

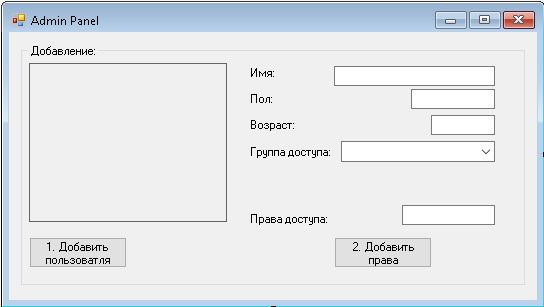


Рисунок 13 – Прототип окна администратора.

1. Реализация программного обеспечения

4.1 Разработка базы данных

Разработка базы данных приложения «Умная камера» реализовывалась в СУБД PhpMyAdmin. База данных состоит из 4 таблиц (рисунок 14).

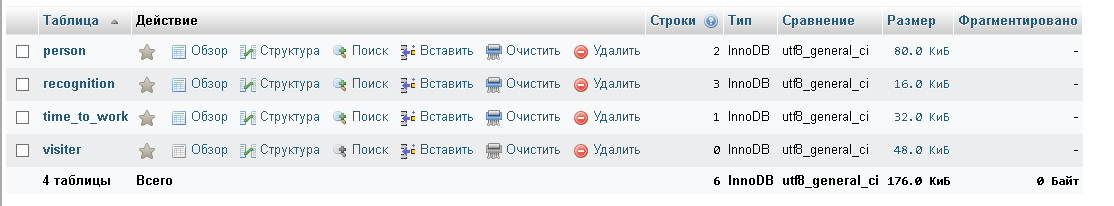


Рисунок 14 – СУБД phpMyAdmin и БД

Структуры таблиц соответствуют схеме базы данных из пункта 4.3.

Таблица «person» на рисунке 15.

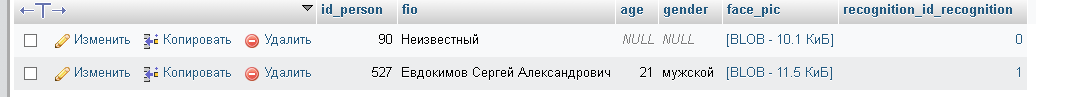


Рисунок 15 – Структура таблицы «person»

Таблица «visiter» на рисунке 16.

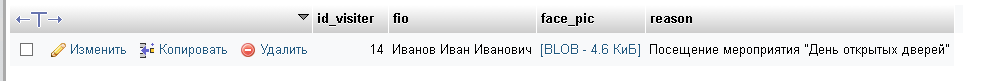


Рисунок 16 – Структура таблицы «visiter»

Таблица «recognition» на рисунке 17.

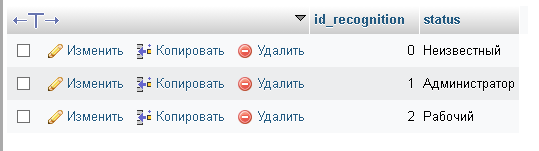


Рисунок 17 – Структура таблицы «recognition»

Таблица «time\_to\_work» на рисунке 18.

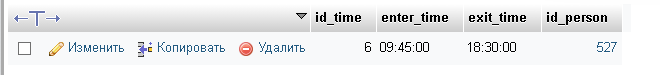


Рисунок 18 – Структура таблицы «time\_to\_work»

После создания всех таблиц и связей в базе данных, она готова к работе.

4.2 Разработка программного продукта

Для разработки приложения был использован UI-фреймворк Windows Forms. Данный фреймворк позволил сделать простой интерфейс со всем необходимым функционалом.

Основной задачей разработки программного продукта было обеспечение точного определения характеристик лиц сотрудников. Для этого были реализованы следующие функциональные модули:

* + захват видео-потока: был реализован модуль, который осуществляет захват видео-потока с камеры. Для этого были использованы соответствующие API и библиотеки (рисунок 19);
  + обработка видео-потока: полученный видео-поток был подвергнут обработке с использованием алгоритмов компьютерного зрения (рисунок 20);
  + идентификация сотрудников: на основе извлеченных характеристик лиц сотрудников, был реализован модуль идентификации, который сопоставлял полученные данные с заранее сохраненными в базе данных. (рисунок 21) ;
  + определение времени прихода и ухода: после успешной идентификации сотрудника, было осуществлено определение времени прихода и ухода на рабочее место (рисунок 22);
  + визуализация и управление: был реализован модуль, который обеспечивал визуализацию результатов работы приложения, например, путем отображения времени прихода и ухода сотрудников в приложении, и возможность сохранения в виде .XLSX файла (рисунок 23);

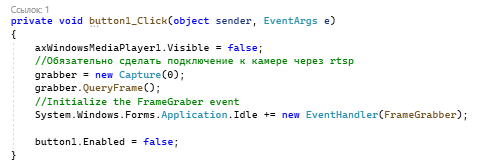


Рисунок 19 – Фрагмент кода захват видео-потока

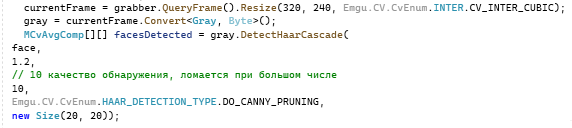


Рисунок 20 – Фрагмент кода обработки видео-потока

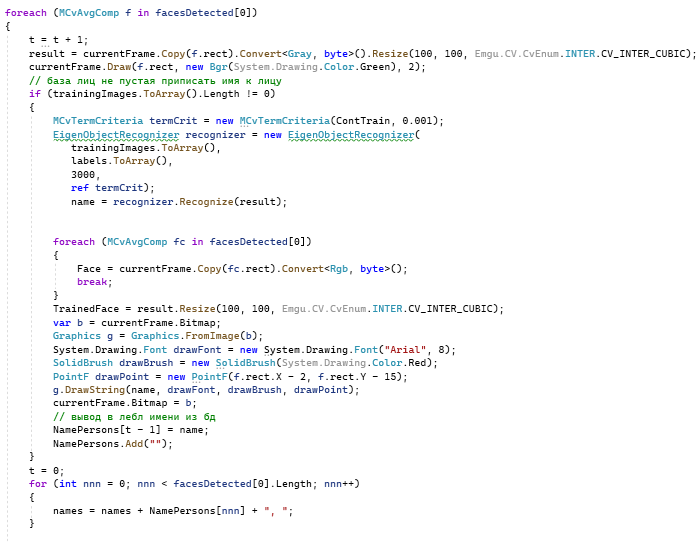


Рисунок 21 – Фрагмент кода идентификации сотрудников

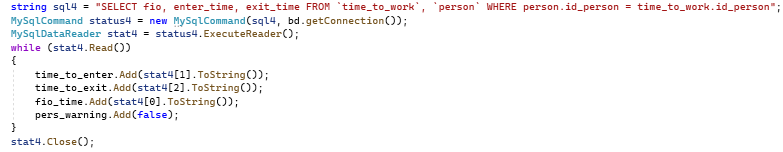


Рисунок 22 – Фрагмент кода определения времени прихода и ухода

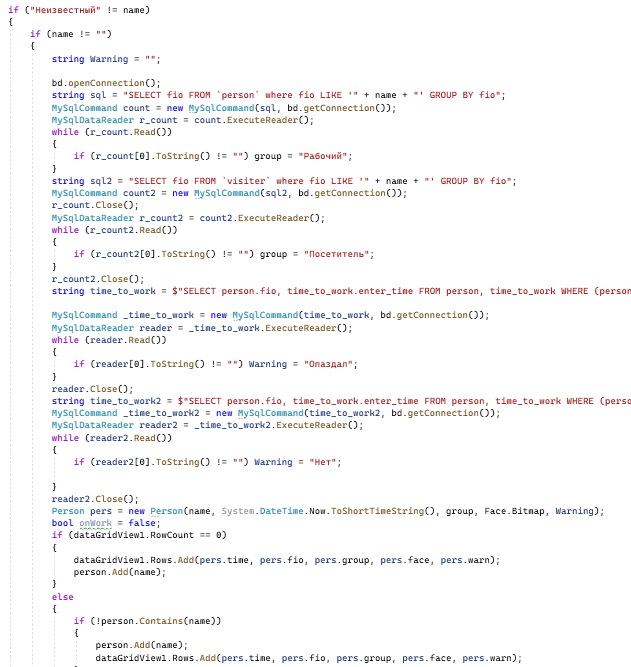


Рисунок 23 – Фрагмент кода визуализации и управления

Вся основная работа с информацией построена на sql запросах в \*.cs:

1. Получение лиц сотрудников/посетителей (рисунок 24).
2. Получение рабочего времени сотрудников (рисунок 25).
3. Получение прав доступа сотрудников (рисунок 26).
4. Добавление новых сотрудников/посетителей в БД (рисунок 27).



Рисунок 24 – Фрагмент кода получения лиц

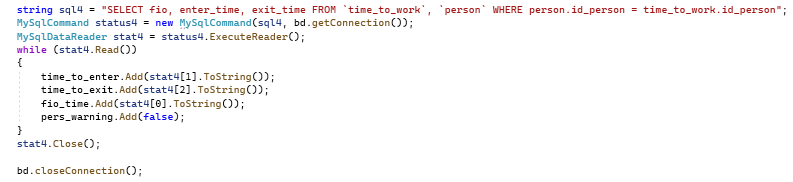


Рисунок 25 – Фрагмент кода получения рабочего времени сотрудников

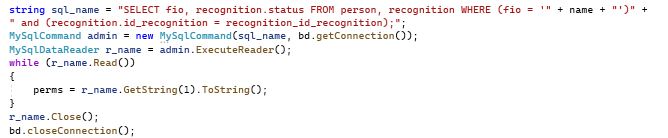


Рисунок 26 – Фрагмент кода получения прав доступа сотрудников



Рисунок 27 – Фрагмент кода добавления новых сотрудников в БД

Реализация программного кода обнаружения лиц представлена в Приложение А. Вывод прошедших на предприятие реализован с помощью языка C#. Данные выводятся в таблицу dataGridView1 и сохраняются в Excel-таблицу (рисунок 28). Программный код реализации сохранения списка прошедших на предприятие представлен в приложении Б

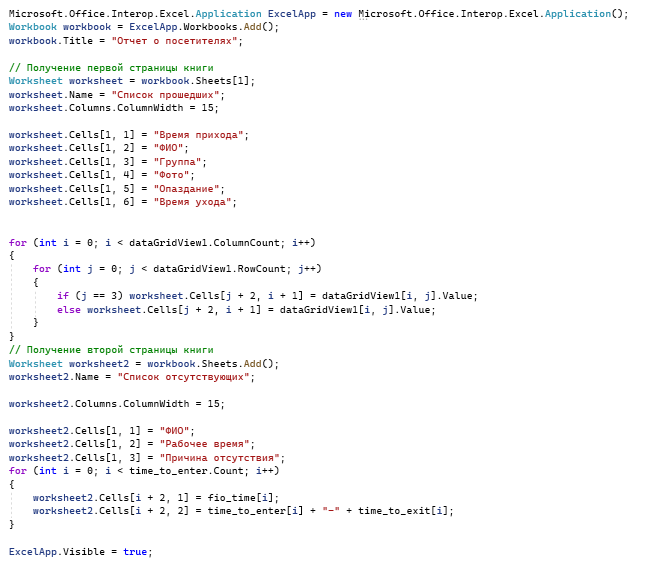


Рисунок 28 – Фрагмент кода вывода и сохранения данных о прошедших

Так же были реализованы следующие функции на главной форме (рисунок 29):

1. кнопка – «Включить камеру», подключается к камере, через удаленное соединение и транслирует изображение в специальном окне;
2. кнопка – «Добавить посетителя», отправляет запрос на добавление информации о посетителе в БД;
3. кнопка – «Админ-панель», открывает новую форму с панелью администратора;
4. кнопка – «Сформировать отчет», создает .xlsx документ с информацией о посетителях, времени прихода рабочих, и не пришедших на работу;
5. кнопка – «Сохранить видео», сохраняет транслируемое видео на сервер;
6. кнопка – «Открыть видео», воспроизводит сохраненное видео с сервера;

Функции реализованные на форме панели администратора (рисунок 30):

1. Кнопка «Добавить пользователя», отправляет запрос на добавление информации о новом сотруднике в БД;
2. Кнопка «Добавить права», отправляет запрос на добавление новых прав доступа.

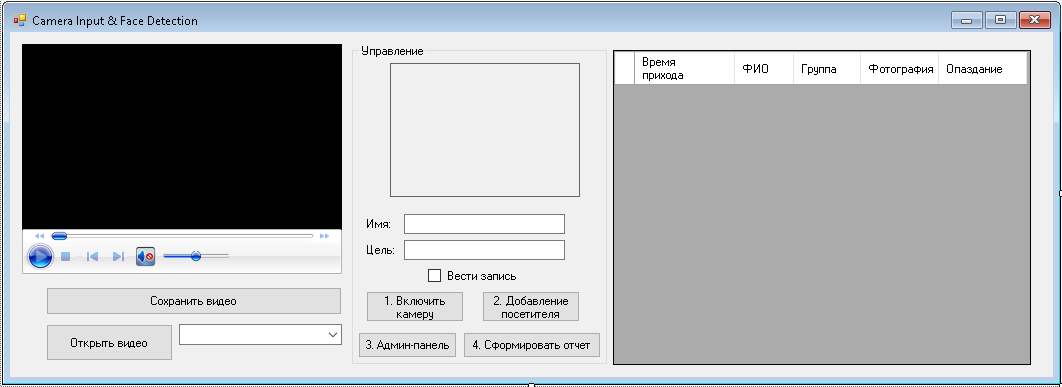


Рисунок 29 – Главная страница

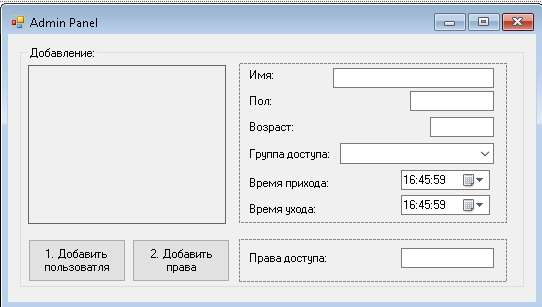


Рисунок 30 – Страница панели администратора

Была реализована валидация необходимых полей. За валидацию полей отвечает условия в \*.cs, которые отвечают за минимальное число символов в поле для ввода. К примеру, валидация данных при добавлении новых сотрудников (рисунок 31).

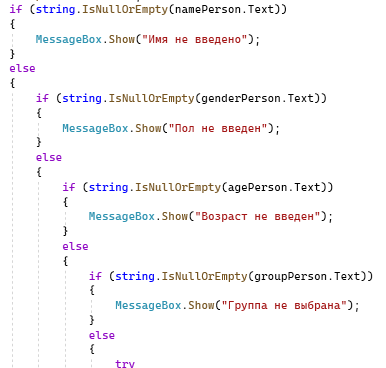


Рисунок 31 – Фрагмент кода валидации полей

Приложение будет использоваться для контроля посещений охраняемых объектов и контроль посещений сотрудников. Для работы приложения используется терминал на системе Windows и камера со встроенным сервером на системе Linux. Приложение разработано для использования на охраняемых объектах, либо для облегчения контроля явки сотрудников.

1. Документирование программного обеспечения

Процесс написания документов программного обеспечения – важный этап в процессе создания и эксплуатации программного обеспечения. Именно с руководства пользователя начинает свое знакомство с программным продуктом.

* 1. Руководство пользователя

Приложение имеет простой и понятный интерфейс, что позволяет легко использовать его.

При открытии приложения открывается страница со всем основным функционалом (рисунок 32). После запуска программы необходимо включить камеру и активировать запись, для начала отслеживания и сохранения видео на сервере. Если нужно пропустить постороннего, необходимо заполнить информацию в определенных полях, зафиксировать лицо посетителя на камере и добавить ее в базу (рисунок 33). При необходимости, можно сформировать отчет об прошедших посетителях, опозданиях и неявкам (рисунок 34).

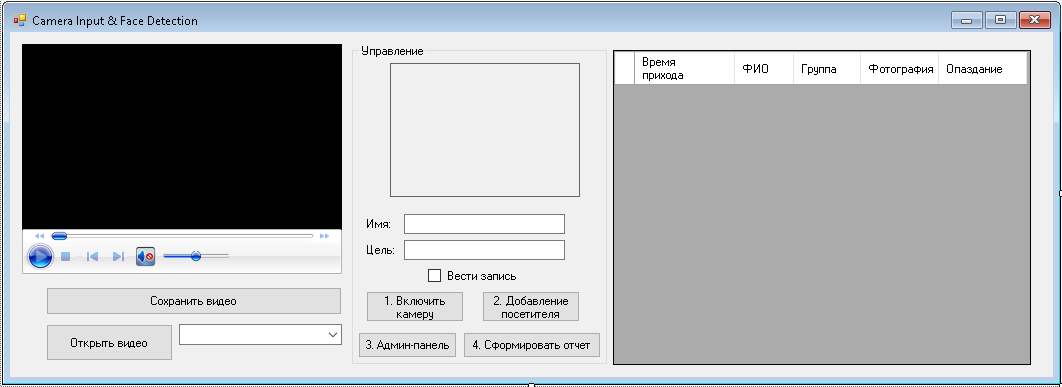


Рисунок 32 – Главная страница

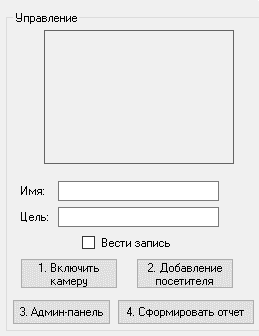


Рисунок 33 – Панель добавления посетителей

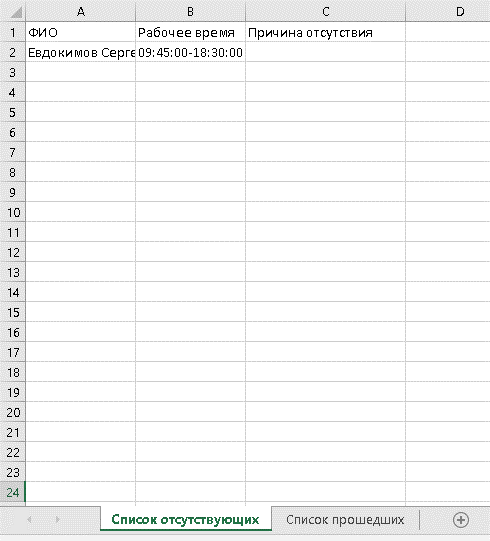


Рисунок 34 – Сформированный отчет

Чтобы получить доступ к функциям администратора необходимо пройти авторизацию по лицу. Если пользователь обладает необходимым уровнем допуска, то откроется окно панели администратора (рисунок 35). Для добавления новых сотрудников или прав доступа, в админ-панели необходимо заполнить соответствующие поля, все данные должны быть валидны, иначе приложение выдаст предупреждение и попросит заново заполнить все поля (рисунок 36).

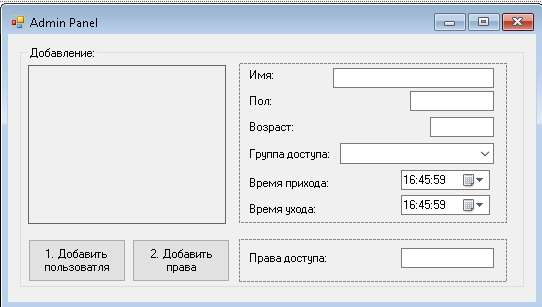


Рисунок 35 – Страница администратора

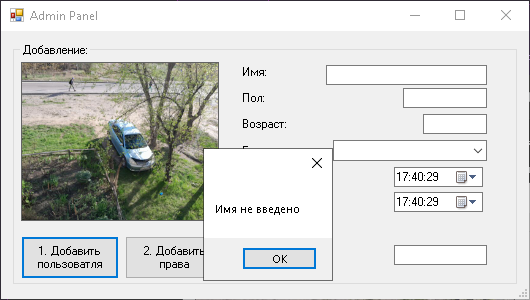


Рисунок 36 – Проверка валидности данных

6. Стоимость разработки и внедрения программного продукта

* 1. Организационно-экономическое обоснование проекта

Разрабатываемое приложение «Умная камера» будет использоваться на частных предприятий, с ограниченным допуском и компаниях, нуждающихся в отслеживании времени посещений сотрудников.

Средняя стоимость разработки приложения составляет (таблица 8).

Таблица 8 – стоимость аналогов приложения «Умная камера»

|  |  |
| --- | --- |
| Название сайта | Стоимость разработки аналога, руб. |
| Tanda | от $50/мес. |
| TSheets | от $20/мес. |
| Deputy | от $3.50/мес. за человека |

1. 2. Расчет затрат на разработку программного продукта

Расчет полных затрат на разработку проектного решения (КРПР) осуществляется по формуле:

(6.1)

КРПР=ЗОТР+ЗЭВМ+ЗСПП+ЗХОН+Е+А,

гдеЗОТР – сумма оплаты труда разработчика/разработчиков ПП;

ЗЭВМ – затраты, связанные с эксплуатацией техники;

ЗСПП – затраты на специальные программные продукты, необходимые для разработки ПП;

ЗХОН – затраты на хозяйственно-операционные нужды (бумага, литература, носители информации и т.п.);

Е – затраты на электроэнергию, руб.;

А – амортизация ПК, руб.

Для подсчета фонда оплаты труда разработчика необходимо определить общее время разработки (таблица 9). Время, затрачиваемое на разработку проектного решения j-м разработчиком, определяется методом экспертных оценок или хронометража. Итоговое значение рассчитывается на основании приведенных исходных данных по формуле:

(6.2)

где tβ– время β-го этапа разработки проектного решения, дн.

Таблица 9 - Затраты времени на создание программного продукта

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Этап создания | | Затраты времени  (в днях) | Затраты времени (в часах) | Машинное время работы над ПП  (в часах) |
| Разработка | Обследование объекта  автоматизации | 2 | 16 | 16 |
| Анализ и уточнение требований | 2 | 16 | 16 |
| Разработка технического задания | 2 | 16 | 16 |
| Проектирование структуры | 1 | 8 | 8 |
|  | Программная реализация | 22 | 176 | 176 |
| Тестирование программного продукта | 3 | 24 | 24 |
| Отладка программного продукта | 7 | 56 | 56 |
| Разработка описания | 2 | 16 | 16 |
| ИТОГО | 41 | 384 | 384 |

Рабочий день принимается равным 8 часам.

Сумму оплаты труда разработчика за время работы над программным продуктом рассчитаем исходя из часовой тарифной ставки и фонда фактического времени, затраченного на разработку программного продукта (по формуле 6.3):

Зотр=Ст1\*Фвр,

(6.3)

где Ст1 - часовой тарифной ставки (принимаем равной 120 руб./час);

Фвр – фонд фактического времени, затраченного на разработку программного продукта, час.

Начисления на заработную плату рассчитываются в таблице 10.

Таблица 10 – Начисление на заработную плату

|  |  |  |
| --- | --- | --- |
| Начисление на заработную плату | Процент,% | Сумма,руб. |
| Пенсионный фонд(ПФ):  -страховая часть  -накопительная часть | 16  6 | 5529,6  2073,6 |
| Фонд социального страхования(ФСС) | 2,9 | 1002,2 |
| Федеральный фонд обязательного медицинского страхования(ФФОМС) | 5,1 | 1762,6 |
| Итого | 30 | 10368 |

Затраты, связанные с использованием вычислительной и оргтехники (формула 6.4):

,

(6.4)

где ТМРПР – машинное время работы над программным продуктом, час;

kГ– коэффициент готовности ЭВМ, ;

n – количество единиц техники, равно 1;

СМ-Ч – себестоимость машино-часа, СМ-Ч=60руб.

6912

Затраты на электроэнергию рассчитываются по следующей формуле 6.5:

(6.5)

Е=W×t×T,

где W – мощность, потребляемая ПК, кВт/час;

t – время работы ПК, час;

T – тариф электроэнергии, руб.

Тариф 1,42 рубля за киловатт.

Амортизация ПК рассчитывается по следующей формуле 6.6:

(6.6)

где – первоначальная стоимость ПК, руб;

– процент амортизации в год.

Для разработки программного продукта не произведено затрат на специальные программные продукты. Отсюда следует, что ЗСПП = 0.

Затраты на хозяйственно-организационные нужды приводятся в таблице 11 и вычисляются по формуле 6.7:

,

(6.7)

где Цτ – цена τ-го товара, руб.;

Кτ – количество τ-го товара.

Таблица 11 - Затраты на хозяйственно-организационные нужды

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Цена за единицу (руб.) | Кол-во (шт.) | Всего (руб.) |
| Флеш-карты | 600 | 1 | 600 |
| Внешний жесткий диск | 3000 | 1 | 3000 |
| Итого | | | 3600 |

Затраты на специальные программные продукты, необходимые для разработки ПП рассчитываются по формуле:

,

(6.8)

где Цρ – цена ρ-го специального программного продукта .

Перечень программных продуктов специального назначения приведен в таблице 12.

Таблица 12 - Программные продукты специального назначения

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование | Цена за единицу (руб.) | Кол-во (шт.) | Всего (руб.) |
| Одноплатный компьютер Raspberry Pi 3B+ | 9350 | 1 | 9350 |
| Модуль камеры для Raspberry Pi | 1635 | 1 | 1635 |
| Инжектор+сплиттер PoE DEXP PZ1S | 250 | 1 | 350 |
| Сетевой адаптер5V-1A | 250 | 1 | 250 |
| Итого | | | 3600 |

Результаты выполненных расчетов сводятся в общей таблице (таблица 13).

Таблица 13 – Затраты на разработку

|  |  |  |
| --- | --- | --- |
| Наименование затрат | Условное обозначение | Значение |
| Оплата труда разработчика программного продукта |  | 46080 |
| Затраты, связанные с эксплуатацией техники |  | 6912 |
| Затраты на электроэнергию | Е | 328,68 |
| Затраты на амортизацию ПК | А | 2425 |
| Затраты на хозяйственно-операционные нужды |  | 3600 |
| Затраты на специальные программные продукты | *З*СПП | 11585 |
| Итого затрат на разработку |  | 70930,68 |

* 1. Расчет затрат на внедрение программного продукта

Внедрение модуля в работу предприятия не требует дополнительную оплату.

6.4 Основные выводы

Итоговая стоимость затрат на разработку программного продукта, в которую включена оплата труда разработчика, затраты на эксплуатацию техники, в том числе и амортизации ПК, затраты на электроэнергию, затраты на специальные программные продукты и затраты на хозяйственно-операционные нужды, составила 70930,68 рублей.

Из проведенных расчетов можно сделать вывод, что разработка приложения для предприятия является дорогостоящим процессом.

# ЗАКЛЮЧЕНИЕ

В рамках данной дипломной работы было разработано и реализовано приложение, основанное на использовании умной камеры, которое предназначено для учета прибытия сотрудников на рабочее место.

Основными задачами приложения было автоматизировать процесс учета рабочего времени сотрудников, обеспечить точность и надежность получаемых данных, а также улучшить общую эффективность управления рабочим персоналом.

В результате разработки и тестирования приложения было достигнуто несколько важных целей. Во-первых, умная камера, интегрированная в приложение, обеспечивает высокую точность и скорость распознавания лиц, что позволяет автоматически идентифицировать сотрудников при их прибытии на рабочее место. Это существенно сокращает время и ресурсы, затрачиваемые на ручной учет трудозатрат и приводит к повышению производительности организации. Во-вторых, приложение обладает удобным и интуитивно понятным пользовательским интерфейсом, что делает его простым в использовании для каких-либо пользователей. Сотрудники могут легко отсканировать свое лицо перед входом на рабочее место, и данные об их прибытии моментально регистрируются и сохраняются в центральной системе. Администраторы имеют доступ к информации о рабочем времени каждого сотрудника в режиме реального времени, что помогает им принимать обоснованные решения в планировании и управлении трудовыми ресурсами.

Приложение обладает потенциалом для дальнейшего развития и расширения функциональности. Например, можно добавить модуль аналитики, который позволит проводить анализ эффективности работы сотрудников, оптимизировать рабочие процессы и выявлять потенциальные узкие места в организации. Также можно рассмотреть возможность интеграции приложения с другими системами учета и управления персоналом, чтобы создать единую централизованную платформу для управления рабочим времен.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Accord.NET [Электронный ресурс] - Режим доступа: http://accord-framework.net/. - Библиотека машинного обучения и компьютерного зрения на C# (Дата обращения: 15.05.2023).
2. ACM Digital Library [Электронный ресурс] - Режим доступа: https://dl.acm.org/. - Цифровая библиотека ACM (Дата обращения: 01.05.2023).
3. AForge.NET [Электронный ресурс] - Режим доступа: http://www.aforgenet.com/framework/. - Библиотека компьютерного зрения и обработки изображений на C# (Дата обращения: 18.05.2023).
4. DlibDotNet [Электронный ресурс] - Режим доступа: https://github.com/takuya-takeuchi/DlibDotNet. - Репозиторий библиотеки DlibDotNet для обработки изображений и распознавания лиц (Дата обращения: 27.04.2023).
5. Emgu CV [Электронный ресурс] - Режим доступа: https://www.emgu.com/. - Официальный сайт библиотеки компьютерного зрения на C# (Дата обращения: 13.04.2023).
6. Face API [Электронный ресурс] - Режим доступа: https://docs.microsoft.com/en-us/azure/cognitive-services/Face/. - Документация по Face API от Microsoft (Дата обращения: 02.04.2023).
7. Face Detection and Recognition in C# [Электронный ресурс] - Режим доступа: https://www.codeproject.com/Articles/239849/Face-Detection-and-Recognition-in-Csharp. - Статья о распознавании и обнаружении лиц на C# (Дата обращения: 24.04.2023).
8. Google Scholar [Электронный ресурс] - Режим доступа: https://scholar.google.com/. - Поисковик научных статей (Дата обращения: 16.05.2023).
9. IEEE Xplore [Электронный ресурс] - Режим доступа: https://ieeexplore.ieee.org/. - Ресурс для научных статей и публикаций (Дата обращения: 10.04.2023).
10. Microsoft Documentation [Электронный ресурс] - Режим доступа: https://docs.microsoft.com/en-us/dotnet/csharp/. - Руководство по разработке на C# (Дата обращения: 30.04.2023).
11. OpenCV [Электронный ресурс] - Режим доступа: https://opencv.org/. - Официальный сайт (Дата обращения: 23.04.2023).
12. OpenCVSharp [Электронный ресурс] - Режим доступа: https://github.com/shimat/opencvsharp. - GitHub репозиторий библиотеки OpenCVSharp для C# (Дата обращения: 01.04.2023).
13. OpenFace.NET [Электронный ресурс] - Режим доступа: https://github.com/theshadowx/OpenFace.NET. - Репозиторий библиотеки OpenFace.NET для распознавания лиц (Дата обращения: 20.04.2023).
14. Python.org [Электронный ресурс] - Режим доступа: https://www.python.org/. - Официальный сайт языка программирования Python (Дата обращения: 19.05.2023).
15. PyTorch [Электронный ресурс] - Режим доступа: https://pytorch.org/. - Официальный сайт (Дата обращения: 19.05.2023).
16. ResearchGate [Электронный ресурс] - Режим доступа: https://www.researchgate.net/. - Онлайн-платформа для научных исследований (Дата обращения: 16.05.2023).
17. TensorFlow [Электронный ресурс] - Режим доступа: https://www.tensorflow.org/. - Официальная документация (Дата обращения: 03.04.2023).
18. Прохоров Д.В. Распознавание лиц с использованием глубокого обучения. М.: НИЦ ИНФОРМАТИКА, 2018. ISBN 978-5-9765-3554-3 (Дата обращения: 08.04.2023).
19. Смирнов Д.А., Камалов Р.Н. Обработка изображений и видео в OpenCV на языке Python. М.: ДМК Пресс, 2019. ISBN 978-5-97060-723-1 (Дата обращения: 05.04.2023).

# Приложение А – Обнаружения лиц с камеры

NamePersons.Add("");

currentFrame = grabber.QueryFrame().Resize(320, 240, Emgu.CV.CvEnum.INTER.CV\_INTER\_CUBIC);

gray = currentFrame.Convert<Gray, Byte>();

MCvAvgComp[][] facesDetected = gray.DetectHaarCascade(

face,

1.2,

// 10 качество обнаружения, ломается при большом числе

10,

Emgu.CV.CvEnum.HAAR\_DETECTION\_TYPE.DO\_CANNY\_PRUNING,

new Size(20, 20));

foreach (MCvAvgComp f in facesDetected[0])

{

t = t + 1;

result = currentFrame.Copy(f.rect).Convert<Gray, byte>().Resize(100, 100, Emgu.CV.CvEnum.INTER.CV\_INTER\_CUBIC);

currentFrame.Draw(f.rect, new Bgr(System.Drawing.Color.Green), 2);

// база лиц не пустая приписать имя к лицу

if (trainingImages.ToArray().Length != 0)

{

MCvTermCriteria termCrit = new MCvTermCriteria(ContTrain, 0.001);

EigenObjectRecognizer recognizer = new EigenObjectRecognizer(

trainingImages.ToArray(),

labels.ToArray(),

3000,

ref termCrit);

name = recognizer.Recognize(result);

foreach (MCvAvgComp fc in facesDetected[0])

{

Face = currentFrame.Copy(fc.rect).Convert<Rgb, byte>();

break;

}

TrainedFace = result.Resize(100, 100, Emgu.CV.CvEnum.INTER.CV\_INTER\_CUBIC);

var b = currentFrame.Bitmap;

Graphics g = Graphics.FromImage(b);

System.Drawing.Font drawFont = new System.Drawing.Font("Arial", 8);

SolidBrush drawBrush = new SolidBrush(System.Drawing.Color.Red);

PointF drawPoint = new PointF(f.rect.X - 2, f.rect.Y - 15);

g.DrawString(name, drawFont, drawBrush, drawPoint);

currentFrame.Bitmap = b;

// вывод в лебл имени из бд

NamePersons[t - 1] = name;

NamePersons.Add("");

}

t = 0;

for (int nnn = 0; nnn < facesDetected[0].Length; nnn++)

{

names = names + NamePersons[nnn] + ", ";

}

if ("Неизвестный" != name)

{

if (name != "")

{

string Warning = "";

bd.openConnection();

string sql = "SELECT fio FROM `person` where fio LIKE '" + name + "' GROUP BY fio";

MySqlCommand count = new MySqlCommand(sql, bd.getConnection());

MySqlDataReader r\_count = count.ExecuteReader();

while (r\_count.Read())

{

if (r\_count[0].ToString() != "") group = "Рабочий";

}

string sql2 = "SELECT fio FROM `visiter` where fio LIKE '" + name + "' GROUP BY fio";

MySqlCommand count2 = new MySqlCommand(sql2, bd.getConnection());

r\_count.Close();

MySqlDataReader r\_count2 = count2.ExecuteReader();

while (r\_count2.Read())

{

if (r\_count2[0].ToString() != "") group = "Посетитель";

}

r\_count2.Close();

string time\_to\_work = $"SELECT person.fio, time\_to\_work.enter\_time FROM person, time\_to\_work WHERE (person.id\_person = time\_to\_work.id\_person && person.fio = '{name}') && (enter\_time < '{DateTime.Now.ToShortTimeString()}:00')";

MySqlCommand \_time\_to\_work = new MySqlCommand(time\_to\_work, bd.getConnection());

MySqlDataReader reader = \_time\_to\_work.ExecuteReader();

while (reader.Read())

{

if (reader[0].ToString() != "") Warning = "Опаздал";

}

reader.Close();

string time\_to\_work2 = $"SELECT person.fio, time\_to\_work.enter\_time FROM person, time\_to\_work WHERE (person.id\_person = time\_to\_work.id\_person && person.fio = '{name}') && (enter\_time > '{DateTime.Now.ToShortTimeString()}:00')";

MySqlCommand \_time\_to\_work2 = new MySqlCommand(time\_to\_work2, bd.getConnection());

MySqlDataReader reader2 = \_time\_to\_work2.ExecuteReader();

while (reader2.Read())

{

if (reader2[0].ToString() != "") Warning = "Нет";

}

reader2.Close();

Person pers = new Person(name, System.DateTime.Now.ToShortTimeString(), group, Face.Bitmap, Warning);

bool onWork = false;

if (dataGridView1.RowCount == 0)

{

dataGridView1.Rows.Add(pers.time, pers.fio, pers.group, pers.face, pers.warn);

person.Add(name);

}

else

{

if (!person.Contains(name))

{

person.Add(name);

dataGridView1.Rows.Add(pers.time, pers.fio, pers.group, pers.face, pers.warn);

}

}

string sql\_name = "SELECT fio, recognition.status FROM person, recognition WHERE (fio = '" + name + "')" +

" and (recognition.id\_recognition = recognition\_id\_recognition);";

MySqlCommand admin = new MySqlCommand(sql\_name, bd.getConnection());

MySqlDataReader r\_name = admin.ExecuteReader();

while (r\_name.Read())

{

perms = r\_name.GetString(1).ToString();

}

r\_name.Close();

bd.closeConnection();

}

}

}

imageBox1.Image = TrainedFace;

TrainedFace = null;

NamePersons.Clear();

imageBoxFrameGrabber.Image = currentFrame;

# Приложение Б – Сохранение списка прошедших на предприятие

Microsoft.Office.Interop.Excel.Application ExcelApp = new Microsoft.Office.Interop.Excel.Application();

Workbook workbook = ExcelApp.Workbooks.Add();

workbook.Title = "Отчет о посетителях";

// Получение первой страницы книги

Worksheet worksheet = workbook.Sheets[1];

worksheet.Name = "Список прошедших";

worksheet.Columns.ColumnWidth = 15;

worksheet.Cells[1, 1] = "Время прихода";

worksheet.Cells[1, 2] = "ФИО";

worksheet.Cells[1, 3] = "Группа";

worksheet.Cells[1, 4] = "Фото";

worksheet.Cells[1, 5] = "Опаздание";

worksheet.Cells[1, 6] = "Время ухода";

for (int i = 0; i < dataGridView1.ColumnCount; i++)

{

for (int j = 0; j < dataGridView1.RowCount; j++)

{

if (j == 3) worksheet.Cells[j + 2, i + 1] = dataGridView1[i, j].Value;

else worksheet.Cells[j + 2, i + 1] = dataGridView1[i, j].Value;

}

}

// Получение второй страницы книги

Worksheet worksheet2 = workbook.Sheets.Add();

worksheet2.Name = "Список отсутствующих";

worksheet2.Columns.ColumnWidth = 15;

worksheet2.Cells[1, 1] = "ФИО";

worksheet2.Cells[1, 2] = "Рабочее время";

worksheet2.Cells[1, 3] = "Причина отсутствия";

for (int i = 0; i < time\_to\_enter.Count; i++)

{

worksheet2.Cells[i + 2, 1] = fio\_time[i];

worksheet2.Cells[i + 2, 2] = time\_to\_enter[i] + "-" + time\_to\_exit[i];

}

ExcelApp.Visible = true;