Министерство образования И НАУКИ Российской Федерации

**отделение среднего профессионального образования**

**«Казанский национальный исследовательский технический университет**

**им. А.Н. Туполева-КАИ»**

**(КНИТУ-КАИ)**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Колледж информационных технологий**

## **КУРСОВАЯ РАБОТА**

**по дисциплине**

**«Прикладное программирование»**

**Тема: “Создание и разработа системы 'Пропускной пункт'”**

**Руководитель**:

преподаватель каф.ПМИ

Мухутдинов Т.А.

**Исполнитель**:

Хасанов А. Р.

студент группы 4333

Оценка \_\_\_\_\_\_\_ Принял \_\_\_\_\_\_\_\_\_\_\_\_

Дата «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_.2018г.

Казань

2018

Содержание

Введение 3

Анализ предметной области 4

Инфологическая модель базы данных 6

Физическая модель базы данных 7

Таблицы БД 7

Разработка экранных форм и алгоритма программы 8

Заключение 15

Используемая литература 16

Приложение 17

**Введение**

Целью данного курсового проекта является создание автоматизиро - ванной системы "Пропускной пункт".

АС "Пропускной пункт" - это система для ведения журнала прохода сотрудников через турникеты.

Введение пропускного режима на предприятии (ПРП) предназначено для:

1. Обеспечения прохода сотрудников, сторонних посетителей, ввоза и вывоза грузов по пропускам установленной формы;

2. Создания условий, препятствующих бесконтрольному проникновению посторонних лиц и транспорта на охраняемую территорию;

3. Оперативного выявления обстоятельств, которые могут нанести ущерб интересам предприятия (например, хищений или пожара), создания системы противодействия этим опасностям.

Цели пропускного пункта:

1. Поддержание внутреннего порядка с учетом специфики предприятия;

2. Рост прибыли за счет борьбы с расхищением имущества и другими уголовными и административными нарушениями;

3. Регистрация и учет посетителей, времени прихода и ухода сотрудников;

4. Охрана интеллектуальной собственности и коммерческих секретов.

## **Анализ предметной области**

Имеется кондитерская фабрика. Данное приложение осуществляет контроль над проходом в предприятие.

Поскольку данная программа является приложением Windows, то для ее реализации будем использовать стандартный подход при написании программ под Windows с использованием графического интерфейса. Так отдельно создадим форму с визуальными компонентами, для различных визуальных компонент сопоставляем процедуры – обработчики. Далее, как только происходит событие на каком-то из визуальных компонент, (щелчок мыши, нажатие на кнопку или пункт меню), операционная система посылает приложению соответствующее сообщение, и запускается соответствующий обработчик именно для этого события. Тогда решение задачи можно разбить на 4 этапов:

1)Разработка Базы данных

2) Разработка визуального интерфейса пользователя, на основе экранных форм среды Windows Forms VisualStudio 2015 и визуальных компонент, предназначенных для ввода исходных данных, запуска действий по обработке исходных данных, вывода результата обработки, вывода результатов на экранную форму. Еще можно предусмотреть вывод информации разработчике и задании курсового проекта.

3) Разработка обработчиков событий от элементов экранных форм, разработка алгоритмов и программных кодов действий обработки текстового файла согласно заданию.

4) Подготовка тестовых исходных данных и отладка программы.

На первом этапе целесообразно выполнить разработку базы данных

На втором этапе целесообразно выполнить разработку интерфейса таким образом, чтобы в начале работы вахтер вводил ФИО сотрудника. Затем вахтер приступает к работе с журналом. По окончании работы выводятся проходы через турникет и количество проходов за определенное время

При выполнении третьего этапа следует использовать модульный принцип построения программы, то есть разбить программу на отдельные части, каждая из которых выполняет какую-то законченную задачу, либо ее малую часть. Роль таких частей будут выполнять процедуры и функции. Программы, состоящие из процедур и функций, называются модульными.

Процедуры и функции в свою очередь могут вызывать процедуры и функции более низкого уровня и так далее. Таким образом, каждая модульная программа имеет иерархическую структуру. Следует заметить, что обработчики событий от объектов экранных форм тоже реализуются как процедуры.

На четвертом этапе при отладке приложения обязательно должна анализироваться правильность работы на исходных тестовых образцах. На этом этапе нужно проверить соответствие разработанной программы исходному заданию, а также проверить ее на отсутствие ошибок.

Подытожив все вышесказанное, можно сформулировать требования к разрабатываемому программному обеспечению и выполнить постановку задания на проектирование.

Программа должна быть реализована с графическим интерфейсом пользователя в среде Visual Studio и должна выполнять следующие функции:

– запрашивать логин и пароль от аккаунта;

– вводить ФИО сотрудников

– выводить журнал посещений;

– добавлять/продлевать/удалять пропуски;

– имитировать посещения;

– вести журнал посещений в базе данных;

– по окончании нужной функции выводить результаты на экран;

## **Инфологическая модель базы данных**

Анализ описанной предметной области и решаемых задач позволяет выделить следующие сущности:

1. Данные о сотрудниках
2. Данные о посещениях

В таблице 1. Данные о сотрудниках: представлена следующая информация:

ФИО, логин (если есть), пароль (если есть), приоритет аккаунта (если есть), имя, должность, срок пропуска.

В таблице 2**.**Данные о посещении: представлена следующая информация: ФИО, время прохода.

## **Физическая модель базы данных**

Авторами была поставлена задача создания СУБД. В качестве сервера базы данных было решено использовать сервер MS SQLServer.

### **Таблицы БД**

База данных, как следует из разработки инфологической модели, содержит 3 таблицы:

DN (Данные)

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип** | **Описание** |
| Id | int | Первичный ключ |
| Full\_name | nvarchar(MAX) | ФИО |
| Login | nvarchar(30) | Логин |
| Password | nvarchar(10) | Пароль |
| Priority | nvarchar(10) | Приоритет |
| Short\_name | nvarchar(50) | Имя |
| Position\_ | nvarchar(MAX) | Должность |
| pass\_from | date | Пропуск с |
| pass\_until | date | Пропуск до |

Attendance(Таблица посещений)

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип** | **Описание** |
| Id | int | Первичный ключ |
| Full\_name | nvarchar(MAX) | ФИО |
| EnterTime | datetime2(0) | Время прохода |

**Разработка экранных форм и алгоритма программы**

По сформулированной в предыдущем задании постановке должна иметься возможность авторизации вахтера или администратора – ввода его логина и пароляы. Для этого предусмотрим отдельную форму авторизации Authorization. Для возможности ввода логина и пароля поместим на форме поля ввода и соответствующие метки описания к ним. Список объектов экранной формы приведен в таблице 2.1

Таблица 2.1 – Объекты экранной формы Authorization

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| Label1 | TLabel | надпись «Введите логин:» |
| Label2 | TLabel | надпись «Введите пароль:» |
| Label3 | TLabel | надпись «ПропускникАвто» |
| Label4 | TLabel | надпись «СладкаяЖизнь©» |
| Label5 | TLabel | надпись «Date» |
| Label6 | TLabel | надпись «Time» |
| textBox1 | TEdit | поле для вода логина |
| textBox1 | TEdit | поле для ввода пароля |
| button1 | TBtn | кнопка для авторизации «Войти» |
| pictureBox1 | Image | рсунок знака выключения |

Для авторизации предусмотрена кнопка типа TBitBtn. В обработчике нажатия на эту кнопку должна выводиться форма с журналом и другими кнопками для работы с журналом и не только. Внешний вид спроектированной формы приведен на рисунке 2.1

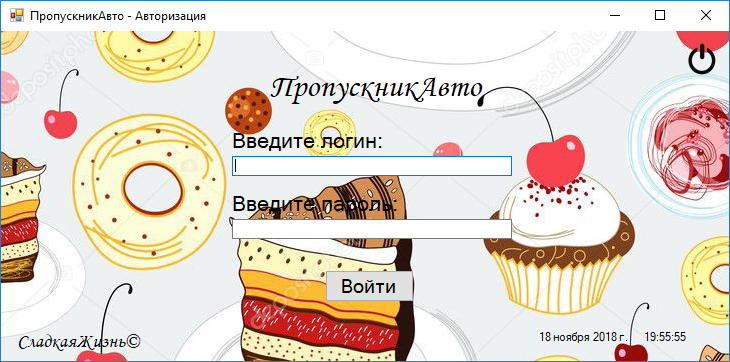


Рисунок 2.1 – Внешний вид формы Authorization

Для вывода журнала посещений предусмотрим отдельную форму UserForm. Список объектов, используемых в форме приведен в таблице 2.2, а внешний вид приведен на рисунке 2.2.

Таблица 2.2 – Объекты экранной формы UserForm

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| Label1 | TLabel | надпись «ФИО:» |
| Label2 | TLabel | надпись «Здравствуйте, Name» |
| Label3 | TLabel | надпись «Месяц:» |
| Label4 | TLabel | надпись «Time» |
| Label5 | TLabel | надпись «Date» |
| button1 | TButton | кнопка для просмотра посещений за все время |
| button2 | TButton | кнопка для просмотра списка сотрудников |
| button3 | TButton | кнопка для просмотра посещений за месяц |
| button4 | TButton | кнопка для перехода на форму имитации прохода |
| button5 | TButton | кнопка для перехода на форму просмотра кол-ва проходов за день и год |
| listBox1 | TListBox | поле для вывода журнала посещений на экран |
| textBox1 | TEdit | поле для ввода ФИО |
| textBox2 | TEdit | поле для ввода месяца |
| pictureBox1 | Image | рисунок для выхода из аккаунта |
| pictureBox2 | Image | рисунок для перехода на форму администратора, если имеются необходимые права администратора |

При построении формы предполагалось, что ФИО сотрудника будет вводиться с клавиатуры. ФИО должно быть полным. Месяц так же вводитя с клавиатуры. При вводе ФИО показываются посещения только этого человека, иначе весь журнал посещений. Спиок сотрудников всегда выводится полностью. После нажатия на кнопки просмотра посещений так же появляется окно с количеством посещений.

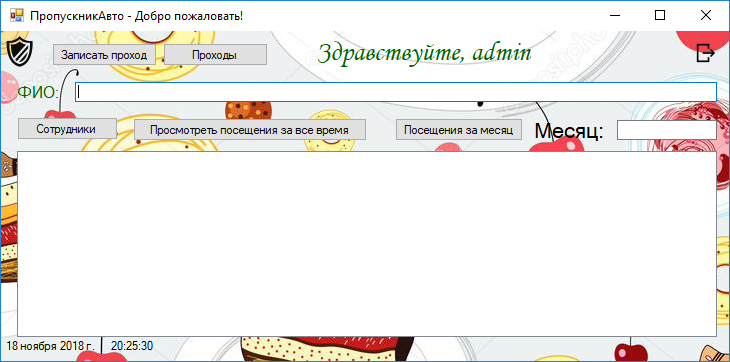


Рисунок 2.2 – Внешний вид формы UserForm

Для работы с пропусками предусмотрим отдельную форму AdminForm, который доступен только ползоателям с правами администратора. Список объектов, используемых в форме приведен в таблице 2.3, а внешний вид приведен на рисунке 2.3.

Таблица 2.3 – Объекты экранной формы AdminForm

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| Label1 | TLabel | надпись «ФИО:» |
| Label2 | TLabel | надпись «\*Приоритет:» |
| Label3 | TLabel | надпись «\*Пароль:» |
| Label4 | TLabel | надпись «\*Логин:» |
| Label5 | TLabel | надпись «Доступен с» |
| Label6 | TLabel | надпись «Должность:» |
| Label7 | TLabel | надпись «Доступен по» |
| Label8 | TLabel | надпись «\* - поля только для вахтеров или администраторов» |
| Label9 | TLabel | надпись «Date» |
| Label10 | TLabel | надпись «Time» |
| Label11 | TLabel | надпись «Имя:» |
| textBox1 | TEdit | поле для ввода ФИО |
| textBox2 | TEdit | поле для ввода приоритета |
| textBox3 | TEdit | поле для ввода пароля |
| textBox4 | TEdit | поле для ввода логина |
| textBox5 | TEdit | поле для ввода срока начала работы пропуска |
| textBox6 | TEdit | поле для ввода должности |
| textBox7 | TEdit | поле для ввода срока окончания работы пропуска |
| textBox9 | TEdit | поле для ввода имени |
| button1 | TButton | кнопка для добавления пропуска |
| button2 | TButton | кнопка для удаления пропуска |
| button3 | TButton | кнопка для продления пропуска |
| pictureBox1 | Image | картинка для выхода из аккаунта |
| pictureBox2 | Image | картинка для возвращения на UserForm |
| pictureBox3 | Image | картинка для вызова окна с сообщением "Формат даты MM-DD-YYYY! (при продлении DD-MM-YYYY)" |
| pictureBox4 | Image | картинка для вызова окна с сообщением "Формат даты MM-DD-YYYY! (при продлении DD-MM-YYYY)" |

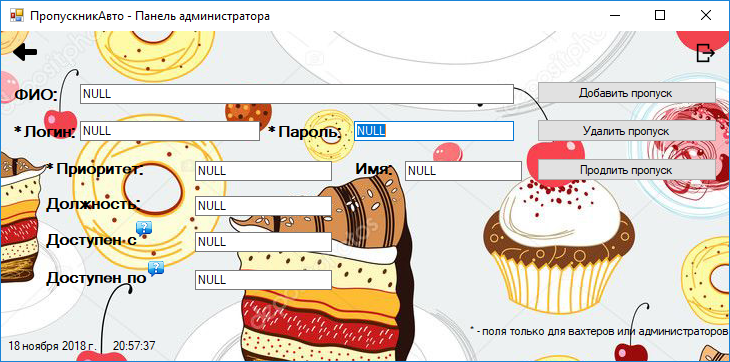
При построении формы предполагалось, что поля без знака “\*” обязательно должны быть заполнены. Формат вводимой даты при добавлении/удалении пропуска MM-DD-YYYY, а при продлении DD-MM-YYYY.

Рисунок 2.3 – Внешний вид формы AdminForm

Для имитации проходов предусмотрим отдельную форму EnterAdd. Список объектов, используемых в форме приведен в таблице 2.4, а внешний вид приведен на рисунке 2.4.

Таблица 2.4 – Объекты экранной формы EnterAdd

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| Label1 | TLabel | надпись «ФИО:» |
| Label5 | TLabel | надпись «Date» |
| Label6 | TLabel | надпись «Time» |
| button1 | TButton | кнопка для пропуска сотрудника и записи посещения в журнал |
| button2 | TButton | кнопка для не пропуска сотрудника |
| pictureBox1 | Image | картинка для возвращения на UserForm |
| pictureBox2 | Image | картинка для выхода из аккаунта |
| listBox1 | TListBox | поле для вывода журнала посещений на экран |

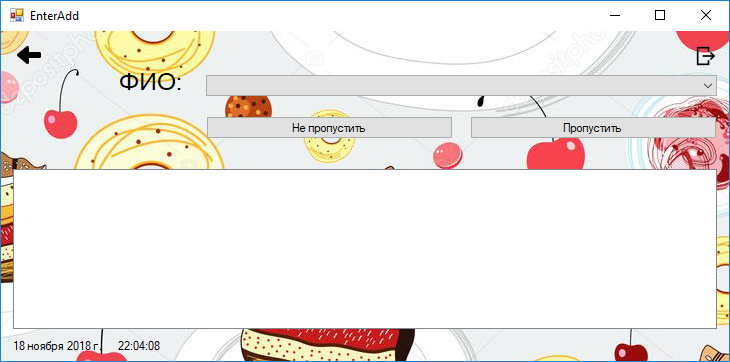
При построении формы предполагалось, что пропуски будут выбираться из выпадающего списка. Если пропуск просрочен, то сотрудник не пропускается.

Рисунок 2.4 – Внешний вид формы EnterAdd

Для узнавания количества проходов в определенные дни и года предусмотрим отдельную форму Days. Список объектов, используемых в форме приведен в таблице 2.5, а внешний вид приведен на рисунке 2.5.

Таблица 2.5 – Объекты экранной формы Days

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Назначение |
| Label1 | TLabel | надпись «День» |
| Label2 | TLabel | надпись «Год» |
| Label3 | TLabel | надпись «Сотрудник» |
| Label5 | TLabel | надпись «Date» |
| Label4 | TLabel | надпись «Time» |
| textBox1 | TEdit | поле для ввода дня |
| textBox2 | TEdit | поле для ввода года |
| textBox3 | TEdit | поле для ввода ФИО сотрудника |
| button1 | TButton | кнопка для показа проходов за день |
| button2 | TButton | кнопка для показа проходов за год |
| pictureBox1 | Image | картинка для возвращения на UserForm |
| pictureBox2 | Image | картинка для выхода из аккаунта |
| listBox1 | TListBox | поле для вывода журнала посещений на экран |

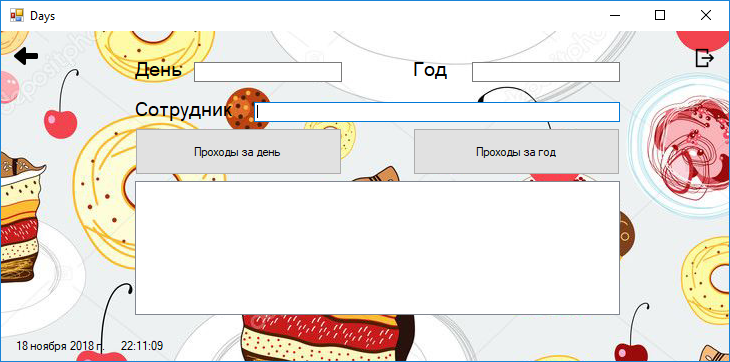
При построении формы предполагалось, ФИО сотрудника будет вводиться полностью, если не введен вовсе, то показывается общее количесто проходов.

Рисунок 2.5 – Внешний вид формы Days

Анализируя все вышеизложенные предположения и разработанные формы, можем составить словесные алгоритмы приложения.

1. Запустить экранную форму авторизации Authorization

2. При нажатии на кнопку «Войти» следует выполнить действия:

2.1 Проверить наличие аккаунта в базе;

2.2 При удачном нахождении войти в учетную запись, иначе показать окошко с ошибкой;

2.3 Скрыть форму Authorizationи показать форму с тестами AdminForm;

3. При нажатии на кнопку «Сотрудники» вывести в listBox1 список сотрудников из базы;

4. При нажатии на кнопку «Просмотреть посещения за все время» происходит следующее:

4.1 Если поле ФИО пустое, то выводится список посещений всех сотрудников за все время;

4.2 Если поле ФИО заполнено и если ФИО сотрудника есть в базе, то выводится список посещений данного сотрудника за все время;

5. При нажатии на кнопку «Просмотреть посещения за месяц» происходит следующее:

5.1 Если поле ФИО пустое, то выводится список посещений всех сотрудников за введенный месяц;

5.2 Если поле ФИО заполнено и если ФИО сотрудника есть в базе, то выводится список посещений данного сотрудника за введенный месяц;

6. При нажатии на картинку со щитом, если у вас есть права администратора, то открывается форма AdminForm, а UserForm закроется, иначе сообщение о том, что этих прав у вас нет.

6.1 Если поля, не помеченные звездочкой \* заполнены, то при нажатии на кнопки «Продлить пропуск», «Удалить пропуск» и «Добавить пропуск» происходит определенное действие, а именно продление / удаление / добавление пропуска соответственно, а иначе выходит сообщение о том, что не все обязательные поля заполнены.

7. При нажатии на кнопку «Записать проход» на форме UserForm, то откроется форма EnterAdd, а UserForm закроется.

7.1 При нажатии на кнопку «Не пропустить» выходит ошибка о том, что сотрудник не пропущен.

7.2 При нажатии на кнопку пропустить, если сотрудник не выбран, то выводится сообщение с ошибкой. Если пропуск выбран и он действителен (не просрочен), то ыводится сообщение о проходе и проход добавляется в базу и список ниже (listBox1). Если пропуск просрочен, то выходит соответствующая ошибка.

8. При нажатии на кнопку проходы на UserForm открывается форма Days, а форма UserForm скрывается.

8.1 Если поле день пуст, то выходит соответствующая ошибка при нажатии на кнопку «Проходы за день», аналогичная ситуация с полем год и кнопкой «Проходы за год». Иначе, если поле ФИО пусто, то выводится спиок посещений за день/год и сообщение с количеством проходов. Если ФИО написано и сотрудник есть в базе, то выводится его посещаемость и количество посещений. Если ФИО нет, то выходит ошибка.

Анализируя требования к функциям разрабатываемого приложения под разработанные экранные формы и словесный алгоритм, можно выделить следующие программные модули:

1) Модуль проверки логина и пароля;

2) Модуль вывода журнала посещений и списка сотрудников;

3) Модуль добавления пропусков;

4) Модуль удаления пропусков;

5) Модуль продления пропусков;

5) Модули перехода между экранными формами.

Каждый из вышеприведенных модулей реализуем при помощи процедуры. Модуль 1 целесообразно запускать каждый раз при нажатии на кнопку “Войти”. Модуль 2 следует запускать каждый раз при нажатии на кнопку вывода сотрудников. Модуль 3 следует запускать каждый раз при нажатии на кнопку добавления пропусков. Модуль 4 следует запускать каждый раз при нажатии на кнопку удаления пропусков. Модуль 5 следует запускать каждый раз при нажатии на кнопку продления пропусков.

**Заключение**

В ходе выполнения курсовой работы была разработана программа для ведения журнала в проходном пункте. Программа может бать использована в целях упрощения работы вахтеров.

В качестве дальнейшего развития возможна доработка программы с целью добавления новых возможностей для упращения работы вахтерам и сотрудникам. Например, более удобный выбор сотрудников, дня / месяца или года для просмотра посещения и многое другое.

В результате выполнения данной курсовой работы, я убедился в широких возможностях языка программирования С#.

**Используемая литература**

1) Биллинг В.А. Основы программирования на С#.-М.: Изд-во " Интернет-университет информационных технологий-ИНТУИТ.ру", 2006.-488с.

2) Ватсон К. С#.- М.: Лори, 2004.-880с.

3) Микелсен К. Язык программирования С#. Лекции и упражнения: Учебник. -Киев: ДиаСофт, 2002.-656 с.

**Приложение**

Приложение 1. Код формы авторизации

using System;

using System.Windows.Forms;

using PropusknoiPunkt.Models;

namespace PropusknoiPunkt

{

public partial class Authorization : Form

{

//////////////////////////////////

/\*\*\*\*\* инициализация и т.д. \*\*\*\*\*/

//////////////////////////////////

public Authorization()

{

InitializeComponent();// инициализация компонента

textBox2.PasswordChar = '\*';// для поля пароль визуально устанавливает \* вместо вводимых значений

textBox2.MaxLength = 10; // для поля пароль устанавливает максимальную длину 10 символов

}

//////////////////////////////////

/\*\*\*\*\* клик на кнопку войти \*\*\*\*\*/

//////////////////////////////////

private void button1\_Click(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

try //попытаться

{

bool lgn\_true = false, pswrd\_true = false, lgn=false, pswrd=false;

for (int i = 0; i < Connection.getAccountsList().Count; i++) //прохождение по списку аккаунтов

{

if (Connection.getAccountsList()[i].Login == textBox1.Text)// если элемент списка совпадает с полем логин

{

if(textBox1.Text != "") //если поле логин не пуста

lgn\_true = true;

}

if (Connection.getAccountsList()[i].Password == textBox2.Text)// если элемент списка совпадает с полем пароль

{

if(textBox2.Text != "") //если поле пароль не пуста

pswrd\_true = true;

}

if (lgn\_true == true && pswrd\_true == true)//если логин и пароль были найдены

{

UserForm form3 = new UserForm(this);//объявление другой формы(UserForm)

form3.Priority = Connection.getAccountsList()[i].Priority;//передача приоритета из списка в переменную другой формы

form3.Hello\_name = Connection.getAccountsList()[i].Short\_name;//передача имени из списка в переменную другой формы

form3.Show();//показать UserForm

textBox1.Text = "";//очистить поле логин

textBox2.Text = "";//очистить поле пароль

Hide();//спрятать данную форму

lgn\_true = false;

pswrd\_true = false;

lgn = true;

pswrd = true;

}

}

if (lgn == false && pswrd == false)//если логин и пароль были не найдены

MessageBox.Show("Логин или пароль неверны! Проверьте данные.");//показать сообщение

}

catch //иначе

{

MessageBox.Show("Не возможно подключиться к базе данных! Попробуйте повторить вход позже.");//показать сообщение

}

Cursor = Cursors.Arrow;//замена курсора

}

/////////////////////////////////////////////////////

/\*\*\*\*\* нажатие кнопки вход при нажатии энетера \*\*\*\*\*/

/////////////////////////////////////////////////////

private void enter(object sender, KeyEventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

if (e.KeyCode == Keys.Enter) button1.PerformClick();//еслии нажатая кнопка - энтер, то нажимается кнопка вход

Cursor = Cursors.Arrow;//замена курсора

}

///////////////////////////////////////////////////////////////

/\*\*\*\*\* Выход из приложения при нажатии на картинку power \*\*\*\*\*/

///////////////////////////////////////////////////////////////

private void Quit(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

Application.Exit();//выйти из проекта (закрыть)

Cursor = Cursors.Arrow;//замена курсора

}

//////////////////////////////////////////

/\*\*\*\*\* При загрузке формы(открытии) \*\*\*\*\*/

//////////////////////////////////////////

private void UserForm\_Load(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

timer1.Start();//запустить таймер

label5.Text = DateTime.Now.ToLongDateString();//поставить сегодняшнюю дату в label даты

label6.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

Cursor = Cursors.Arrow;//замена курсора

}

/////////////////////////

/\*\*\*\*\* Тик таймера \*\*\*\*\*/

/////////////////////////

private void timer1\_Tick(object sender, EventArgs e)

{

label6.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

timer1.Start();//запустить таймер(продолжить его работу)

}

}

}

Приложение 2. Код формы UserForm

using System;

using System.Windows.Forms;

using PropusknoiPunkt.Models;

namespace PropusknoiPunkt

{

public partial class UserForm : Form

{

////////////////////////////////////////////////////////

private Form \_form1;//объявление формы /////////////////

string priority, name;//объявление переменных

////////////////////////////////////////////////////////

////////////////////////////////////////////

/\*\*\*\*\* публичная строковая переменная \*\*\*\*\*/

////////////////////////////////////////////

public string Priority//для передачи приоритета из формы авторизации

{

get

{

return priority;

}

set

{

priority = value;

}

}

///////////////////////////////////////////////

public string Hello\_name//для передачи имени из формы авторизации

{

get

{

return name;

}

set

{

name = value;

}

}

//////////////////////////////////

/\*\*\*\*\* Инициализация и т.д. \*\*\*\*\*/

//////////////////////////////////

public UserForm(Form form1)

{

InitializeComponent();//инициализация компонента

\_form1 = form1;

}

///////////////////////////////

/\*\*\*\*\* Выход из аккаунта \*\*\*\*\*/

///////////////////////////////

private void LogOut(object sender, EventArgs e) //выход из аккаунта

{

Cursor = Cursors.WaitCursor;//замена курсора

this.Close();//закрытие этой формы

\_form1.Show();//показать окно авторизации

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////////////

/\*\*\*\*\* Вывод посещаемости \*\*\*\*\*/

////////////////////////////////

private void Attendance(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

try//попытаться

{

if (textBox1.Text == "")//если поле ввода пустое

{

listBox1.Items.Clear();//очистка поля вывода (в этом случае листбокс)

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробегаемся по списку посещений

{

if (Connection.getAttendanceList()[j].Full\_name != "")

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавляем нужные значения в листбокс

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

else

{

listBox1.Items.Clear();//очистка поля вывода (в этом случае листбокс)

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробегаемся по списку посещений

{

if (Connection.getAttendanceList()[j].Full\_name == textBox1.Text)//если фио в списке совпадает с фио в поле ввода

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавляем нужные значения в листбокс

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

}

catch//иначе

{

MessageBox.Show("Что-то пошло не так! Проверьте введенные данные или попробуйте повторить попытку позже.");//выводим сообщение

}

Cursor = Cursors.Arrow;//замена курсора

}

//////////////////////////////////////////

/\*\*\*\*\* При загрузке формы(открытии) \*\*\*\*\*/

//////////////////////////////////////////

private void UserForm\_Load(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

timer1.Start();//запустить таймер

label5.Text = DateTime.Now.ToLongDateString();//поставить сегодняшнюю дату в label даты

label4.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

label2.Text = "Здравствуйте, " + Hello\_name;//текст здравствуйте, [имя владельца аккаунта]

Cursor = Cursors.Arrow;//замена курсора

}

/////////////////////////

/\*\*\*\*\* Тик таймера \*\*\*\*\*/

/////////////////////////

private void timer1\_Tick(object sender, EventArgs e)

{

label4.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

timer1.Start();//запустить таймер(продолжить его работу)

}

///////////////////////////////////////////////////

/\*\*\*\*\* Клик по щиту для перехода в AdminForm \*\*\*\*\*/

///////////////////////////////////////////////////

private void pictureBox2\_Click(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

if (priority == "admin")//если приоритет аккаунта админ

{

AdminForm form2 = new AdminForm(this, \_form1);//объявление формы администратора

form2.Show();//показать форму администратора

Hide();//спрятать

}

else

MessageBox.Show("Вы не являетесь администратором!");//вывод сообщения

Cursor = Cursors.Arrow;//замена курсора

}

///////////////////////////////////////////////

/\*\*\*\*\* Клик по кнопке показа сотрудников \*\*\*\*\*/

///////////////////////////////////////////////

private void WorkersBttn(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

try//попытаться

{

listBox1.Items.Clear();//очистить листбокс

int j=0;

foreach (var i in Connection.getAccountsList())//пробежаться по списку аккаунтов

{

listBox1.Items.Add(Connection.getAccountsList()[j].Id + " " + Connection.getAccountsList()[j].Full\_name + " " + Connection.getAccountsList()[j].Position\_ + " Пропуск с " + Connection.getAccountsList()[j].pass\_from + " Пропуск по " + Connection.getAccountsList()[j].pass\_untill);//добавить в листбокс нужные значения из connection.cs

j++;

}

}

catch

{

MessageBox.Show("Что-то пошло не так! Попробуйте повторить попытку позже.");//вывод сообщения

}

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////////////////////////

/\*\*\*\*\* Клик по кнопке записать проход \*\*\*\*\*/

////////////////////////////////////////////

private void AddEnter(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

EnterAdd formadd = new EnterAdd(this, \_form1);//объявление формы администратора

formadd.Show();//показать форму администратора

Hide();//спрятать

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////////////////

/\*\*\*\*\* Клик по кнопке проходы \*\*\*\*\*/

////////////////////////////////////

private void Enters(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

Days formdays = new Days(this, \_form1);//объявление формы администратора

formdays.Show();//показать форму администратора

Hide();//спрятать

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////////////////////////////////////

/\*\*\*\*\* Клик по щиту для показа посещения за месяц \*\*\*\*\*/

////////////////////////////////////////////////////////

private void Attendance\_month(object sender, EventArgs e)//посещение за месяц

{

Cursor = Cursors.WaitCursor;//замена курсора

listBox1.Items.Clear();//очистить листбокс

try

{

if (textBox1.Text == "")//если поле фио пусто

{

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробегаемся по списку посещений

{

if ((Connection.getAttendanceList()[j].EnterTime.Month == byte.Parse(textBox2.Text)) && Connection.getAttendanceList()[j].Full\_name != "")//если месяц в списке совпал с месяцем в текстовом поле месяц

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавление в листбокс нужных значений списка посещений

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

else

{

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробежаться по списку посещений

{

if ((Connection.getAttendanceList()[j].EnterTime.Month == byte.Parse(textBox2.Text)) && (Connection.getAttendanceList()[j].Full\_name == textBox1.Text))//если месяц в списке совпал с месяцем в текстовом поле месяц и фио в списке совпала с фио в поле фио

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавление в листбокс нужных значений списка посещений

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

}

catch

{

MessageBox.Show("Что-то пошло не так! Проверьте введенные данные или попробуйте повторить попытку позже.");//вывод сообщения

}

Cursor = Cursors.Arrow;//замена курсора

}

}

}

Приложение 3. Код формы AdminForm

using System;

using System.Windows.Forms;

using PropusknoiPunkt.Models;

namespace PropusknoiPunkt

{

public partial class AdminForm : Form

{

//////////////////////////////////////

private Form \_form2;//объявление формы

private Form \_form1;//объявление формы

//////////////////////////////////////

//////////////////////////////////

/\*\*\*\*\* инициализация и т.д. \*\*\*\*\*/

//////////////////////////////////

public AdminForm(Form form2, Form form1)

{

InitializeComponent();//инициализация компонента

\_form2 = form2;

\_form1 = form1;

textBox3.MaxLength = 10;//максимальная длина поля пароль = 10

foreach (Control textbox in this.Controls)//пробегаемся по объектам на форме

if (textbox is TextBox)//если это текстовое поле

{

if ((textbox as TextBox).Text == "")//если поле пусто, то ставим туда NULL

(textbox as TextBox).Text = "NULL";

}

}

///////////////////////////////

/\*\*\*\*\* Выход из аккаунта \*\*\*\*\*/

///////////////////////////////

private void LogOut(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

this.Hide();//спрятать эту форму

\_form1.Show();//показать форму авторизации

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////////////////

/\*\*\*\*\* Клик на текстовое поле \*\*\*\*\*/

////////////////////////////////////

private void txtbx\_click(object sender, EventArgs e)//поле фио

{

foreach (Control textbox in this.Controls)//пробегаемся по объектам на форме

if (textbox is TextBox)//если это текстовое поле

{

if ((textbox as TextBox).Text == "")//если поле пусто, то ставим туда NULL

(textbox as TextBox).Text = "NULL";

}

if (textBox1.Text == "NULL")//если в поле написано NULL, то очищаем поле

textBox1.Text = "";

}

///////////////////////////////////////////////

private void txtbx2\_click(object sender, EventArgs e)//поле приоритет

{

foreach (Control textbox in this.Controls)

if (textbox is TextBox)

{

if ((textbox as TextBox).Text == "")

(textbox as TextBox).Text = "NULL";

}

if (textBox2.Text == "NULL")

textBox2.Text = "";

}

///////////////////////////////////////////////

private void txtbx3\_click(object sender, EventArgs e)//поле пароль

{

foreach (Control textbox in this.Controls)

if (textbox is TextBox)

{

if ((textbox as TextBox).Text == "")

(textbox as TextBox).Text = "NULL";

}

if (textBox3.Text == "NULL")

textBox3.Text = "";

}

///////////////////////////////////////////////

private void txtbx4\_click(object sender, EventArgs e)//поле логин

{

foreach (Control textbox in this.Controls)

if (textbox is TextBox)

{

if ((textbox as TextBox).Text == "")

(textbox as TextBox).Text = "NULL";

}

if (textBox4.Text == "NULL")

textBox4.Text = "";

}

///////////////////////////////////////////////

private void txtbx5\_click(object sender, EventArgs e)//поле пропуск с

{

foreach (Control textbox in this.Controls)

if (textbox is TextBox)

{

if ((textbox as TextBox).Text == "")

(textbox as TextBox).Text = "NULL";

}

if (textBox5.Text == "NULL")

textBox5.Text = "";

}

///////////////////////////////////////////////

private void txtbx6\_click(object sender, EventArgs e)//поле должность

{

foreach (Control textbox in this.Controls)

if (textbox is TextBox)

{

if ((textbox as TextBox).Text == "")

(textbox as TextBox).Text = "NULL";

}

if (textBox6.Text == "NULL")

textBox6.Text = "";

}

///////////////////////////////////////////////

private void txtbx7\_click(object sender, EventArgs e)//поле пропуск по

{

foreach (Control textbox in this.Controls)

if (textbox is TextBox)

{

if ((textbox as TextBox).Text == "")

(textbox as TextBox).Text = "NULL";

}

if (textBox7.Text == "NULL")

textBox7.Text = "";

}

///////////////////////////////////////////////

private void txtbx9\_click(object sender, EventArgs e)//поле имя

{

foreach (Control textbox in this.Controls)

if (textbox is TextBox)

{

if ((textbox as TextBox).Text == "")

(textbox as TextBox).Text = "NULL";

}

if (textBox9.Text == "NULL")

textBox9.Text = "";

}

//////////////////////////////////////////////

/\*\*\*\*\* Клик на форму или текст на форме \*\*\*\*\*/

//////////////////////////////////////////////

private void form\_click(object sender, EventArgs e)

{

foreach (Control textbox in this.Controls)//пробегаемся по объектам на форме

if (textbox is TextBox)//если это текстовое поле

{

if ((textbox as TextBox).Text == "")//если поле пусто, то ставим туда NULL

(textbox as TextBox).Text = "NULL";

}

}

//////////////////////////////////

/\*\*\*\*\* Клик на кнопку назад \*\*\*\*\*/

//////////////////////////////////

private void Back(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

this.Hide();//спрятать эту форму

\_form2.Show();//показать UserForm

Cursor = Cursors.Arrow;//замена курсора

}

//////////////////////////////////////////

/\*\*\*\*\* При загрузке формы(открытии) \*\*\*\*\*/

//////////////////////////////////////////

private void UserForm\_Load(object sender, EventArgs e)

{

timer1.Start();//запустить таймер

label9.Text = DateTime.Now.ToLongDateString();//поставить сегодняшнюю дату в label даты

label10.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

}

/////////////////////////

/\*\*\*\*\* Тик таймера \*\*\*\*\*/

/////////////////////////

private void timer1\_Tick(object sender, EventArgs e)

{

label10.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

timer1.Start();//запустить таймер(продолжить его работу)

}

//////////////////////////////////////

/\*\*\*\*\* Клик добавить сотрудника \*\*\*\*\*/

//////////////////////////////////////

public void insertCard(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

Connection.insert(textBox1.Text, textBox4.Text, textBox3.Text, textBox2.Text, textBox9.Text, textBox6.Text, textBox5.Text, textBox7.Text);//выполнить insert в connection, передав значения текстбоксов

Cursor = Cursors.Arrow;//замена курсора

}

//////////////////////////////////////

/\*\*\*\*\* Клик удалить сотрудника \*\*\*\*\*/

//////////////////////////////////////

private void deleteCard(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

Connection.delete(textBox1.Text, textBox4.Text, textBox3.Text, textBox2.Text, textBox9.Text, textBox6.Text, textBox5.Text, textBox7.Text);//выполнить delete в connection, передав значения текстбоксов

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////////////////////////

/\*\*\*\*\* Клик по картинкам с вопросиком \*\*\*\*\*/

////////////////////////////////////////////

private void question\_Click(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

foreach (Control textbox in this.Controls)//пробегаемся по объектам на форме

if (textbox is TextBox)//если это текстовое поле

{

if ((textbox as TextBox).Text == "")//если поле пусто, то ставим туда NULL

(textbox as TextBox).Text = "NULL";

}

MessageBox.Show("Формат даты MM-DD-YYYY! (при продлении DD-MM-YYYY)");//показать сообщение

Cursor = Cursors.Arrow;//замена курсора

}

///////////////////////////////////////////////

/\*\*\*\*\* Нажатие кнопки продления пропуска \*\*\*\*\*/

///////////////////////////////////////////////

private void cardRenewal(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

Connection.update(textBox1.Text, textBox5.Text, textBox7.Text);

Cursor = Cursors.Arrow;//замена курсора

}

}

}

Приложение 4. Код формы Days

using System;

using System.Windows.Forms;

using PropusknoiPunkt.Models;

namespace PropusknoiPunkt

{

public partial class Days : Form

{

//////////////////////////////////////

private Form \_form2;//объявление формы

private Form \_form1;//объявление формы

//////////////////////////////////////

//////////////////

/\*\*\*\*\* Назад\*\*\*\*\*/

//////////////////

private void Back(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

\_form2.Show();

Close();

Cursor = Cursors.Arrow;//замена курсора

}

///////////////////////////////

/\*\*\*\*\* Выход из аккаунта \*\*\*\*\*/

///////////////////////////////

private void LogOut(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

\_form1.Show();

Close();

Cursor = Cursors.Arrow;//замена курсора

}

//////////////////////////////////

/\*\*\*\*\* Инициализация и т.д. \*\*\*\*\*/

//////////////////////////////////

public Days(Form form2, Form form1)

{

InitializeComponent();

\_form2 = form2;

\_form1 = form1;

}

/////////////////////////////////////////

/\*\*\*\*\* Клик по кнопке проходы день \*\*\*\*\*/

/////////////////////////////////////////

private void enter\_kol(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

listBox1.Items.Clear();//очистить листбокс

try

{

if (textBox3.Text == "")//если поле фио пусто

{

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробегаемся по списку посещений

{

if ((Connection.getAttendanceList()[j].EnterTime.Day == byte.Parse(textBox1.Text)) && Connection.getAttendanceList()[j].Full\_name != "")//если day в списке совпал с day в текстовом поле day

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавление в листбокс нужных значений списка посещений

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

else

{

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробежаться по списку посещений

{

if ((Connection.getAttendanceList()[j].EnterTime.Day == byte.Parse(textBox1.Text)) && (Connection.getAttendanceList()[j].Full\_name == textBox3.Text))//если day в списке совпал с day в текстовом поле day и фио в списке совпала с фио в поле фио

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавление в листбокс нужных значений списка посещений

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

}

catch

{

MessageBox.Show("Что-то пошло не так! Проверьте введенные данные или попробуйте повторить попытку позже.");//вывод сообщения

}

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////////////////////

/\*\*\*\*\* Клик по кнопке проходы год \*\*\*\*\*/

////////////////////////////////////////

private void enter\_kol\_year(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

listBox1.Items.Clear();//очистить листбокс

try

{

if (textBox3.Text == "")//если поле фио пусто

{

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробегаемся по списку посещений

{

if ((Connection.getAttendanceList()[j].EnterTime.Year == int.Parse(textBox2.Text)) && Connection.getAttendanceList()[j].Full\_name != "")//если year в списке совпал с year в текстовом поле year

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавление в листбокс нужных значений списка посещений

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

else

{

int j = 0, k = 0;

foreach (var i in Connection.getAttendanceList())//пробежаться по списку посещений

{

if ((Connection.getAttendanceList()[j].EnterTime.Year == int.Parse(textBox2.Text)) && (Connection.getAttendanceList()[j].Full\_name == textBox3.Text))//если year в списке совпал с year в текстовом поле year и фио в списке совпала с фио в поле фио

{

k++;//порядковый номер

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавление в листбокс нужных значений списка посещений

}

j++;

}

MessageBox.Show("Кол-во проходов = " + k);

}

}

catch

{

MessageBox.Show("Что-то пошло не так! Проверьте введенные данные или попробуйте повторить попытку позже.");//вывод сообщения

}

Cursor = Cursors.Arrow;//замена курсора

}

/////////////////////////

/\*\*\*\*\* Тик таймера \*\*\*\*\*/

/////////////////////////

private void timer1\_Tick(object sender, EventArgs e)

{

label4.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

timer1.Start();//запустить таймер(продолжить его работу)

}

//////////////////////////////////////////

/\*\*\*\*\* При загрузке формы(открытии) \*\*\*\*\*/

//////////////////////////////////////////

private void Days\_Load(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

timer1.Start();//запустить таймер

label5.Text = DateTime.Now.ToLongDateString();//поставить сегодняшнюю дату в label даты

label4.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

Cursor = Cursors.Arrow;//замена курсора

}

}

}

Приложение 5. Код формы EnterAdd

using System;

using System.Windows.Forms;

using PropusknoiPunkt.Models;

namespace PropusknoiPunkt

{

public partial class EnterAdd : Form

{

//////////////////////////////////////

private Form \_form2;//объявление формы

private Form \_form1;//объявление формы

//////////////////////////////////////

//////////////////////////////////

/\*\*\*\*\* Инициализация и т.д. \*\*\*\*\*/

//////////////////////////////////

public EnterAdd(Form form2, Form form1)

{

InitializeComponent();

\_form2 = form2;

\_form1 = form1;

}

///////////////////////////

/\*\*\*\*\* Загрузка формы\*\*\*\*\*/

///////////////////////////

private void EnterAdd\_Load(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

Connection.addCombo(comboBox1);//заполнение комбобокса

timer1.Start();//запустить таймер

label5.Text = DateTime.Now.ToLongDateString();//поставить сегодняшнюю дату в label даты

label4.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

Cursor = Cursors.Arrow;//замена курсора

}

/////////////////////////

/\*\*\*\*\* Тик таймера \*\*\*\*\*/

/////////////////////////

private void timer1\_Tick(object sender, EventArgs e)

{

label4.Text = DateTime.Now.ToLongTimeString();//поставить нынешнее время в label времени

timer1.Start();//запустить таймер(продолжить его работу)

}

//////////////////

/\*\*\*\*\* Назад\*\*\*\*\*/

//////////////////

private void Back(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

\_form2.Show();

Close();

Cursor = Cursors.Arrow;//замена курсора

}

///////////////////////////////

/\*\*\*\*\* Выход из аккаунта \*\*\*\*\*/

///////////////////////////////

private void LogOut(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

\_form1.Show();

Close();

Cursor = Cursors.Arrow;//замена курсора

}

///////////////////////

/\*\*\*\*\* Не пропуск\*\*\*\*\*/

///////////////////////

private void noEnter(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

MessageBox.Show("Сотрудник не пропущен!");//вывести сообщение

Cursor = Cursors.Arrow;//замена курсора

}

////////////////////////

/\*\*\*\*\* Пропустить \*\*\*\*\*/

////////////////////////

private void enter(object sender, EventArgs e)

{

Cursor = Cursors.WaitCursor;//замена курсора

if (comboBox1.SelectedItem != null)//если комбобокс не выбран

{

try

{

Connection.entr(comboBox1);//выполнить entr из connection.cs, передав значение комбобокса

}

catch

{

MessageBox.Show("Произошла ошибка при проверке пропуска!");//вывести сообщение

}

listBox1.Items.Clear();//очистить листбокс

int j = 0, k = 1;

foreach (var i in Connection.getAttendanceList())//пробегаемся по списку посещений

{

listBox1.Items.Add(k + " " + Connection.getAttendanceList()[j].Full\_name + " " + Connection.getAttendanceList()[j].EnterTime);//добавление в листбокс нужных значений списка посещений

k++;//порядковый номер

j++;

}

}

else

MessageBox.Show("Вы не выбрали пропуск!");//вывести сообщение

Cursor = Cursors.Arrow;//замена курсора

}

}

}

Приложение 6. Код файла Connection.cs

using System;

using System.Collections.Generic;

using System.Data.SqlClient;

using System.Windows.Forms;

namespace PropusknoiPunkt.Models

{

public static class Connection

{

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

static string connectString = @"Data Source=(localdb)\MSSQLLocalDB;AttachDbFilename=|DataDirectory|\APP\_DATA\PPDB.MDF;Integrated Security=True;";//строка подключения для БД PPDB.MDF ////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

//////////////////////////////

/\*\*\*\*\* Список посещений \*\*\*\*\*/

//////////////////////////////

public static List<Attendance> getAttendanceList()

{

List<Attendance> atttendanceList = new List<Attendance>();//объявление списка

SqlConnection sqlConnection = new SqlConnection(connectString);//объявление соединения

atttendanceList.Clear();//очистка списка

sqlConnection.Open();//открытие соединения

SqlCommand command = new SqlCommand("SELECT \* FROM Attendance", sqlConnection);//команда на SQL

SqlDataReader reader = command.ExecuteReader();//объявление ридера (для считывания нескольких данных)

while (reader.Read())//пока ридеру есть чего ситывать

{

atttendanceList.Add(new Attendance() { Id = uint.Parse(reader[0].ToString()), Full\_name = reader[1].ToString(), EnterTime = DateTime.Parse(reader[2].ToString())});//добавление в список нужных значений

}

sqlConnection.Close();//закрытие соединения

return atttendanceList;//возврат списка

}

//////////////////////////////

/\*\*\*\*\* Список аккаунтов \*\*\*\*\*/

//////////////////////////////

public static List<Account> getAccountsList()//объявление списка

{

List<Account> accountsList = new List<Account>();//объявление списка

SqlConnection sqlConnection = new SqlConnection(connectString);//объявление соединения

accountsList.Clear();//очистка списка

sqlConnection.Open();//открытие соединения

SqlCommand command = new SqlCommand("SELECT \* FROM Accounts", sqlConnection);//команда на SQL

SqlDataReader reader = command.ExecuteReader();//объявление ридера (для считывания нескольких данных)

while (reader.Read())//пока ридеру есть чего ситывать

{

accountsList.Add(new Account() { Id = uint.Parse(reader[0].ToString()), Full\_name = reader[1].ToString(), Login = reader[2].ToString(), Password = reader[3].ToString(), Priority = reader[4].ToString(), Short\_name = reader[5].ToString(), Position\_ = reader[6].ToString(), pass\_from = Convert.ToDateTime(reader[7].ToString()), pass\_untill = Convert.ToDateTime(reader[8].ToString()) });//добавление в список нужных значений

}

sqlConnection.Close();//закрытие соединения

return accountsList;//возврат списка

}

///////////////////////////////

/\*\*\*\*\* INSERT INTO (SQL) \*\*\*\*\*/

///////////////////////////////

public static bool insert(string full\_name, string login, string password, string priority, string name, string position, string pass\_from, string pass\_untill)

{

try//попытаться

{

if (full\_name == "" || name == "" || position == "" || pass\_from == "" || pass\_untill == "" || full\_name == "NULL" || name == "NULL" || position == "NULL" || pass\_from == "NULL" || pass\_untill == "NULL")//если обязательные поля пучты или NULL

{

MessageBox.Show("Не все обязательные поля были заполнены! Пожалуйста, заполните их и повторите попытку.");//вывод сообщения

}

else

{

if ((login != "" && password != "" && priority != "") || (login != "NULL" && password != "NULL" && priority != "NULL"))//если необязательные поля не заполнены

{

SqlConnection sqlConnection = new SqlConnection(connectString);//объявление соединения

sqlConnection.Open();//открытие соединения

SqlCommand command = new SqlCommand("INSERT INTO [dbo].[Accounts] ([Full\_name], [Login], [Password], [Priority], [Short\_name], [Position\_], [pass\_from], [pass\_until]) VALUES (N'" + full\_name + "', N'" + login + "', N'" + password + "', N'" + priority + "', N'" + name + "', N'" + position + "', N'" + DateTime.Parse(pass\_from).Date + "', N'" + DateTime.Parse(pass\_untill).Date + "');", sqlConnection);//команда на SQL

command.ExecuteNonQuery();//выполнение команды без возврата значений

sqlConnection.Close();//закрыть соединение

MessageBox.Show("Пропуск добален!");//показать сообщение

}

else

{

SqlConnection sqlConnection = new SqlConnection(connectString);

sqlConnection.Open();

SqlCommand command = new SqlCommand("INSERT INTO [dbo].[Accounts] ([Full\_name], [Short\_name], [Position\_], [pass\_from], [pass\_until]) VALUES (N'" + full\_name + "', N'" + name + "', N'" + position + "', N'" + DateTime.Parse(pass\_from).Date + "', N'" + DateTime.Parse(pass\_untill).Date + "');", sqlConnection);

command.ExecuteNonQuery();

sqlConnection.Close();

MessageBox.Show("Пропуск проден!");//показать сообщение

}

}

}

catch//иначе

{

MessageBox.Show("Не удалось продлить пропуск! Попробуйте позже.");//показать сообщение

}

return false;//возврат false

}

///////////////////////////////

/\*\*\*\*\* DELETE FROM (SQL) \*\*\*\*\*/

///////////////////////////////

public static bool delete(string full\_name, string login, string password, string priority, string name, string position, string pass\_from, string pass\_untill)

{

try

{

if (full\_name == "" || name == "" || position == "" || pass\_from == "" || pass\_untill == "" || full\_name == "NULL" || name == "NULL" || position == "NULL" || pass\_from == "NULL" || pass\_untill == "NULL")

{

MessageBox.Show("Не все обязательные поля были заполнены! Пожалуйста, заполните их и повторите попытку.");

}

else

{

if ((login != "" && password != "" && priority != "")||(login != "NULL" && password != "NULL" && priority != "NULL"))

{

SqlConnection sqlConnection = new SqlConnection(connectString);

sqlConnection.Open();

SqlCommand command = new SqlCommand("DELETE FROM [dbo].[Accounts] WHERE [Full\_name] = N'" + full\_name + "' AND [Short\_name] = N'" + name + "' AND [Position\_] = N'" + position + "' AND [pass\_from] = N'" + DateTime.Parse(pass\_from).Date + "' AND [pass\_until] = N'" + DateTime.Parse(pass\_untill).Date + "';", sqlConnection);

command.ExecuteNonQuery();

sqlConnection.Close();

MessageBox.Show("Сотрудник удален!");

}

else

{

SqlConnection sqlConnection = new SqlConnection(connectString);

sqlConnection.Open();

SqlCommand command = new SqlCommand("DELETE FROM [dbo].[Accounts] WHERE [Full\_name] = N'" + full\_name + "' AND [Login] = N'" + login + "' AND [Password] = N'" + password + "' AND [Priority] = N'" + priority + "' AND [Short\_name] = N'" + name + "' AND [Position\_] = N'" + position + "' AND [pass\_from] = N'" + DateTime.Parse(pass\_from).Date + "' AND [pass\_until] = N'" + DateTime.Parse(pass\_untill).Date + "';", sqlConnection);

command.ExecuteNonQuery();

sqlConnection.Close();

MessageBox.Show("Сотрудник удален!");

}

}

}

catch

{

MessageBox.Show("Не удалось удалить сотрудника! Попробуйте позже.");

}

return false;

}

////////////////////////////////////////////////////////////////

/\*\*\*\*\* Запись нужных значений в combobox (форма enteradd) \*\*\*\*\*/

////////////////////////////////////////////////////////////////

public static bool addCombo(ComboBox comboBox1)

{

try

{

SqlConnection sqlConnection = new SqlConnection(connectString);//объявление соединения

sqlConnection.Open();//открытие соединения

SqlCommand command = new SqlCommand("SELECT Full\_name, Position\_, pass\_from, pass\_until FROM Accounts", sqlConnection);//команда на SQL

SqlDataReader reader = command.ExecuteReader();//объявление ридера (для считывания нескольких данных)

while (reader.Read())//пока ридеру есть чего ситывать

{

comboBox1.Items.Add(reader[0] + " " + reader[1] + " " + reader[2] + " " + reader[3]);//запись нужных значений из БД в combobox (фио, должность, дата срока пропуска)

}

sqlConnection.Close();//закрытие соединения

}

catch

{

MessageBox.Show("Не удалось связаться с БД! Попробуйте позже!");//вывести сообщение

}

return false;//возврат false

}

//////////////////////////////////////////////////////////////

/\*\*\*\*\* Запись прохода по нажатии на кнопку "пропустить" \*\*\*\*\*/

//////////////////////////////////////////////////////////////

public static bool entr(ComboBox comboBox1)

{

try

{

SqlConnection sqlConnection = new SqlConnection(connectString);//объявление соединения

string[] itms = comboBox1.SelectedItem.ToString().Split(new char[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);//разделение строки из combobox по пробелам

string today = DateTime.Now.Day + "." + DateTime.Now.Month + "." + DateTime.Now.Year;//сегодняшняя дата

if (DateTime.Parse(itms[4]) > DateTime.Parse(today) || DateTime.Parse(itms[6]) < DateTime.Parse(today))//сравнение нынешней даты со сроком пропуска

{

MessageBox.Show("Пропуск недействителен!");//вывести сообщение

}

else

{

sqlConnection.Open();//открытие соединения

SqlCommand command = new SqlCommand("INSERT INTO [dbo].[Attendance] ([Full\_name], [EnterTime]) VALUES (N'" + itms[0] + " " + itms[1] + " " + itms[2] + "', N'" + DateTime.Now.Month + "." + DateTime.Now.Day + "." + DateTime.Now.Year + " " + DateTime.Now.Hour + ":" + DateTime.Now.Minute + ":" + DateTime.Now.Second + "');", sqlConnection);//команда на SQL

command.ExecuteNonQuery();//выполнить команду без ожидания возврата

sqlConnection.Close();//закрытие соединения

MessageBox.Show("Сотрудник пропущен!");//вывести сообщение

}

}

catch

{

MessageBox.Show("Не удалось связаться с БД! Попробуйте позже!");//вывести сообщение

}

return false;//возврат false

}

//////////////////////////

/\*\*\*\*\* UPDATE (SQL) \*\*\*\*\*/

//////////////////////////

public static bool update(string full\_name, string pass\_from, string pass\_untill)

{

try//попытаться

{

string[] a = new string[1] { "" };

if ((full\_name != "" && pass\_from != "" && pass\_untill != "") && (full\_name != "NULL" && pass\_from != "NULL" && pass\_untill != "NULL"))

{

DateTime from\_ = new DateTime();//объявление даты

DateTime untill\_ = new DateTime();//объявление даты

from\_ = DateTime.Parse(pass\_from).Date;//присвоение строки с переводом в дату время

untill\_ = DateTime.Parse(pass\_untill).Date;//присвоение строки с переводом в дату время

SqlConnection sqlConnection = new SqlConnection(connectString);//объявление соединения

sqlConnection.Open();//открытие соединения

try

{

SqlCommand select = new SqlCommand("SELECT \* FROM [Accounts] WHERE Full\_name = @full\_name;", sqlConnection);

select.Parameters.Add("@full\_name", full\_name);

SqlDataReader reader = select.ExecuteReader();//объявление ридера (для считывания нескольких данных)

while (reader.Read())//пока ридеру есть чего ситывать

{

a[0] = reader[1].ToString();

}

reader.Close();

}

catch

{

MessageBox.Show("Пропуск не найден!");

}

if (a[0] != "")

{

SqlCommand command = new SqlCommand("UPDATE [Accounts] SET pass\_from = @from, pass\_until= @untill WHERE Full\_name = @full\_name", sqlConnection);//команда на SQL

command.Parameters.Add("@from", from\_);//добавить парамет со значением выше в @ из строки команды

command.Parameters.Add("@untill", untill\_);//добавить парамет со значением выше в @ из строки команды

command.Parameters.Add("@full\_name", full\_name);//добавить парамет со значением выше в @ из строки команды

command.ExecuteNonQuery();//выполнение команды без возврата значений

sqlConnection.Close();//закрыть соединение

MessageBox.Show("Пропуск продлен!");//показать сообщение

}

else

{

MessageBox.Show("Пропуск неверен!");

}

}

else

{

MessageBox.Show("Пропуск неверен!");

}

}

catch//иначе

{

MessageBox.Show("Не удалось продлить пропуск! Попробуйте позже.");//показать сообщение

}

return false;//возврат false

}

}

}

Приложение 7. Код файла Account.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PropusknoiPunkt.Models

{

public class Account

{

public uint Id { get; set; }

public string Full\_name { get; set; }

public string Login { get; set; }

public string Password { get; set; }

public string Priority { get; set; }

public string Short\_name { get; set; }

public string Position\_ { get; set; }

public DateTime pass\_from { get; set; }

public DateTime pass\_untill { get; set; }

}

}

Приложение 8. Код файла Attendance.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace PropusknoiPunkt.Models

{

public class Attendance

{

public uint Id { get; set; }

public string Full\_name { get; set; }

public DateTime EnterTime { get; set; }

public DateTime ExitTime { get; set; }

}

}