|  |
| --- |
| 09.02.07 ПР-419  **ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**  **ПМ.11 Разработка, администрирование и защита баз данных** |
| **УП.11.02 Разработка приложения** |

|  |  |  |  |
| --- | --- | --- | --- |
| Практикант |  |  | Мавренков В.Д. |
| Руководитель практики |  |  | **Пьяненкова А.В.** |
|  |  |  |  |

Оглавление

[УП.11.02 Разработка приложения 3](#_Toc118538247)

[**База практики** 3](#_Toc118538248)

[**Период прохождения практики** 3](#_Toc118538249)

[**Основная задача практики** 3](#_Toc118538250)

[**Виды работ** 3](#_Toc118538251)

[**Задание на разработку** 4](#_Toc118538252)

[Описание организации данных 6](#_Toc118538253)

[**Логическая модель данных** 6](#_Toc118538254)

[**Схема данных** 6](#_Toc118538255)

[**Описание таблиц** 7](#_Toc118538256)

[**Описание интерфейса** 9](#_Toc118538257)

[**Описание отчетов** 14](#_Toc118538258)

[**Заключение** 18](#_Toc118538259)

[Используемые источники информации 19](#_Toc118538260)

**УП.11.02 Разработка приложения**

**База практики**

Практика проводилась на базе полигона вычислительной техники БПОУ ОО «Омский авиационный колледж имени Н.Е. Жуковского»

**Период прохождения практики**

Практика Разработка приложения по разработке программных продуктов проходила с **31.10.2022 г.** по **05.11.2022 г.** на полигоне вычислительной техники Омавиат.

**Основная задача практики**

Основная задача практики - формирование практических навыков по разработке кода программного продукта на основе готовой спецификации на уровне модуля, проведения тестирования программного модуля по определенному сценарию, использования основных методологий процессов разработки программного обеспечения, работы с объектами базы данных в конкретной СУБД.

**Виды работ**

* создание базы данных;
* разграничение данных пользователей;
* разработка приложения;
* документирование и презентация решений/

**Задание на разработку**

**Описание**

Требуется разработать программное решение для учета платежей физических лиц. Используя полученный программный продукт, физические лица могут вести учет своих платежей. Кроме того, появится возможность производить несложный анализ затрат в разрезе периодов или категорий. Также можно получать результаты анализа в печатной форме.

**Требования**

**Разграничение данных пользователей**

1. Программное решение должно быть многопользовательским;
2. Пароли должны обязательно храниться в зашифрованном виде;
3. Логин можно выбирать из выпадающего списка;

**Окно с основными данными**

1. В окне основных данных выводятся только данные текущего пользователя;
2. Данные должны помещаться на один экран по ширине;
3. Данные можно отображать за выбранный период (от даты до даты);
4. Данные можно отображать по одной из категорий;
5. Данные можно добавлять;
6. Данные можно удалять;
7. По отображаемым данным можно получить отчет;

**Добавление платежей**

1. При добавлении данных можно указывать категорию платежа;
2. При добавлении данных обязательно указывается назначение платежа;
3. Назначение платежа указывается на русском языке;
4. Назначение платежа состоит как минимум из 3 букв;
5. Едино разово можно совершить несколько одинаковых платежей;
6. Количество обязательно должно быть целым положительным числом;
7. Для платежа обязательно указывается цена в рублях;
8. Цена не может быть отрицательной;
9. Стоимость платежа рассчитывается как количество \* цена;
10. Дата платежа указывается как текущее время на компьютере;

**Удаление платежей**

1. При удалении данных система должна запрашивать подтверждение;
2. В подтверждении нужно выводить назначение платежа и другую значимую информацию;
3. Удаление должно сопровождаться звуковым или визуальным оформлением;

**Отчет**

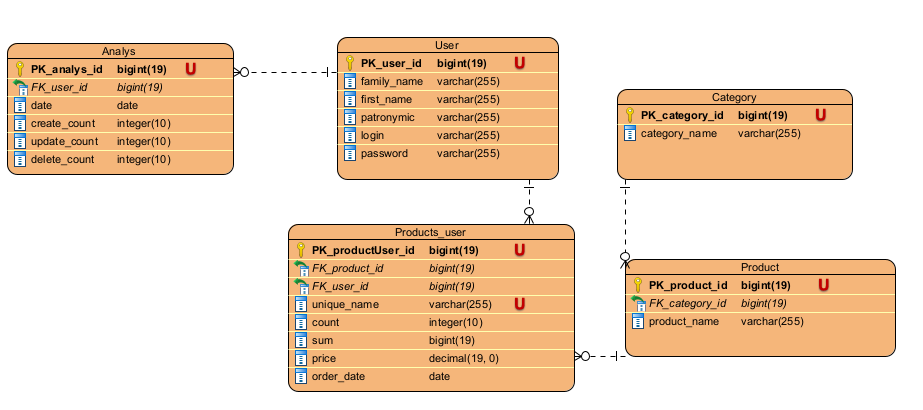
1. Отчет может выводиться встроенными средствами, либо в распространенных программных продуктах (Microsoft Office Word, Microsoft Office Excel, Internet-браузер);
2. У отчета обязательно должен быть заголовок на первой странице;
3. В заголовке может быть указан период и категория платежей;
4. На каждой странице указывается ФИО пользователя и номер страницы;
5. Платежи группируются по категориям;
6. Внутри каждой категории платежи сортируются по дате;
7. В конце отчета выводится суммарная стоимость всех показанных платежей.

Описание организации данных

**Логическая модель данных**

На основе предоставленных для работы данных, были выделены следующие объекты (таблица 1) и разработана ERD-диаграмма в среде MS Visual Paradigm   
(рисунок 1).

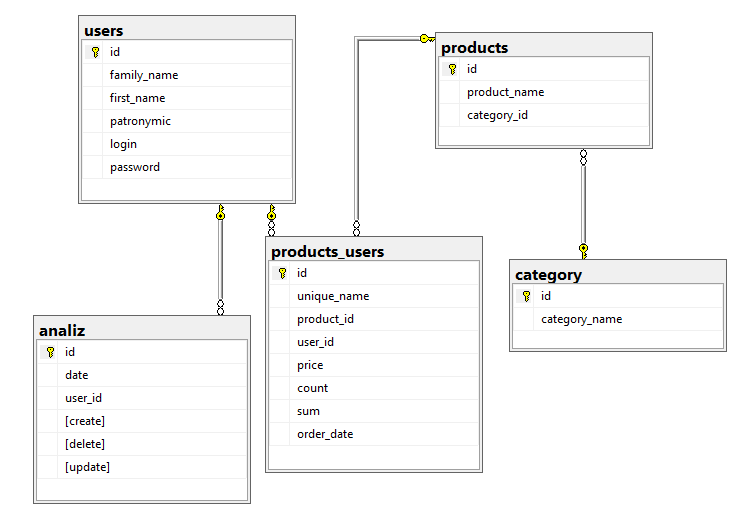
|  |  |
| --- | --- |
| Объект | Описание |
| Пользователь | Информация о пользователях системы. |
| Продукт пользователя | Информация о продуктах определенного пользователя системы |
| Продукт | Информация о продуктах, зарегистрированных в системе |
| Категория | Информация о категориях продуктов, зарегистрированных в системе |
| Анализ | Информация о сессиях пользователя за день |



**Рисунок 1. ERD-диаграмма**

**Схема данных**

На основе разработанной ERD-диаграммы, была создана база данных в среде MS SQL Server Management Studio и схема данных (рисунок 2).



**Рисунок 2 - Схема данных**

**Описание таблиц**

**Таблица 1-Users**

|  |  |  |  |
| --- | --- | --- | --- |
| **Название поля** | **Описание** | **Тип значения/поля** | **Кол-во знаков** |
| user\_id | Уникальный идентификатор пользователя | int | 19 |
| family\_name | Фамилия | nvarchar | 255 |
| first\_name | Имя | nvarchar | 255 |
| patronymic | Отчество | nvarchar | 255 |
| login | Логин | nvarchar | 255 |
| password | Пароль | nvarchar | 255 |

**Таблица 2-Products\_users**

|  |  |  |  |
| --- | --- | --- | --- |
| **Название поля** | **Описание** | **Тип значения/поля** | **Кол-во знаков** |
| product\_id | Уникальный идентификатор продукта пользователя | Int | 19 |
| unique\_name | Уникальное имя продукта пользователя | nvarchar | 255 |
| product\_id | Уникальный идентификатор продукта | int | 19 |
| user\_id | Уникальный идентификатор пользователя | int | 19 |
| price | Цена | money | 20 |
| count | Кол-во продуктов | int | 19 |
| sum | Стоимость | money | 20 |
| order\_date | Дата оформления | date | 10 |

**Таблица 3-Products**

|  |  |  |  |
| --- | --- | --- | --- |
| **Название поля** | **Описание** | **Тип значения/поля** | **Кол-во знаков** |
| product\_id | Уникальный идентификатор продукта | int | 19 |
| product\_name | Название продукта | nvarchar | 255 |
| category\_id | Уникальный идентификатор категории | int | 19 |

**Таблица 4-Category**

|  |  |  |  |
| --- | --- | --- | --- |
| **Название поля** | **Описание** | **Тип значения/поля** | **Кол-во знаков** |
| category\_id | Уникальный идентификатор категории | int | 19 |
| category\_name | Название категории | nvarchar | 255 |

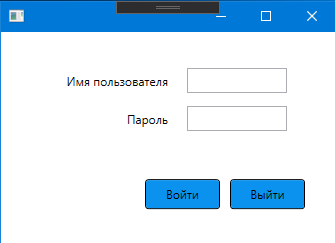
**Таблица 5-Analys**

|  |  |  |  |
| --- | --- | --- | --- |
| **Название поля** | **Описание** | **Тип значения/поля** | **Кол-во знаков** |
| analyz\_id | Уникальный идентификатор анализа активности | int | 19 |
| user\_id | Уникальный идентификатор пользователя | int | 19 |
| date | Дата | date | 10 |
| create\_count | Кол-во добавленных записей | int | 19 |
| update\_count | Кол-во измененных записей | int | 19 |
| delete\_count | Кол-во удаленных записей | int | 19 |

**Описание интерфейса**

На основании предоставленного макета экранных форм был разработан интерфейс системы.

На рисунке 3 представлен экран авторизации, предназначенный для осуществления входа в учётную запись пользователя. В случае некорректного ввода пароля, генерируется CAPTCHA.



**Рисунок 3 - Экран авторизации**

В листинге 1 представлен метод авторизации по нажатию кнопки «Войти». Осуществляется хэширование введенного пароля и проверка на его корректность в таблице Users.

Листинг 1. Авторизация

private void enter\_btn\_Click(object sender, RoutedEventArgs e)

{

password = tbPass.Text.ToString();

login = tbName.Text.ToString();

if (!String.IsNullOrEmpty(login) && !String.IsNullOrEmpty(password))

{

password = hash(password);

//Получение данных о пользователе из таблицы users

var users = Instances.db.users.Where(u => u.login == login && u.password == password).ToList();

if (capcha.Visibility != Visibility.Visible)

{

if (users.Count() > 0)

{

Main main = new Main(users[0]);

main.Show();

Session.dateAuth = DateTime.Now;

Close();

}

else

error\_login();

}

else if (capcha.Visibility == Visibility.Visible)

{

if (!String.IsNullOrEmpty(capcha\_textbox.Text))

{

if (users.Count() > 0 && capcha\_textbox.Text == capcha\_code)

{

Main main = new Main(users[0]);

main.Show();

Session.dateAuth = DateTime.Now;

Close();

}

else

error\_login();

}

else

{

CreateCapcha();

MessageBox.Show("НЕВЕРНАЯ капча");

}

}

}

}

На рисунке 4 представлен экран для управления продуктами пользователя, вошедшего в систему. На ней он может добавлять/изменять/удалять записи, фильтровать по дате/категориям продуктов, формировать отчёт.



**Рисунок 4-Главный экран**

В листинге 2 с помощью библиотеки Syncfusion было реализовано создание отчета в форматах PDF и Excel файлов.

Листинг 2. Создание Excel и PDF файлов с отчётом

private void otchet\_btn\_Click(object sender, RoutedEventArgs e)

{

//Создание настроек для экспорта файла формата Excel

var options = new ExcelExportingOptions();

options.ExportMode = ExportMode.Text;

var excelEngine = ordersDataGrid.ExportToExcel(ordersDataGrid.View, options);

var workBook = excelEngine.Excel.Workbooks[0];

//Сохранение файла в формате .xlsx

workBook.SaveAs("Sample.xlsx");

//Создание настроек для экспорта файла формата PDF

var document = ordersDataGrid.ExportToPdf();

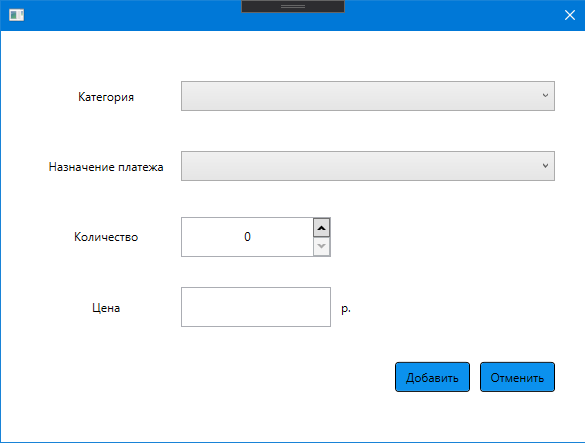
//Сохранение данных в формате .pdf

document.Save("Sample1.pdf");

MessageBox.Show("Отчёт сохранён");

}

На рисунке 5 реализован экран предназначенный для добавления новых записей платежей пользователя в таблицу products\_user. Она имеет поля: Выбор категории товара /Назначение платежа/Количество/Цена.



**Рисунок 5-Экран добавления записи**

В листинге 3 с помощью средств разработки EntityFramework было реализовано добавление выбранных пользователем данных в таблицу products\_users .

Листинг 3. Добавление записи в таблицу на основе введённых данных

private void btnAdd\_Click(object sender, RoutedEventArgs e)

{

string choiceCategory = categ.SelectedValue.ToString();

string choiceProduct = textbox\_name.SelectedValue.ToString();

int count = text\_kol.Value.Value;

string price = text\_cost.Text.ToString();

if ( !choiceCategory.Equals("") && !choiceProduct.Equals("") && !count.Equals(0) && !price.Equals(""))

{

int max = Instances.db.products\_users.Max(u => u.id);

max += 1;

if (create)

{

//Создание объекта таблицы products\_users

products\_users item = new products\_users();

//Получение id элементов

item.user\_id = user.id;

item.product\_id = products.Where(y => y.product\_name == choiceProduct).Select(u => u.id).Max();

item.price = decimal.Parse(price);

item.count = count;

item.sum = count \* decimal.Parse(price);

item.order\_date = DateTime.Now;

item.unique\_name = $"{(choiceCategory.ToCharArray())[0].ToString()}-{max}-{DateTime.Now.ToString("dd.MM.yyyy")}";

//Добавление данных в таблицу products\_users

Instances.db.products\_users.Add(item);

Session.countAdd++;

Session.count++;

}

else

{ //Создание запроса в БД

var obj = Instances.db.products\_users.Where(u => u.id == prod.id).FirstOrDefault();

obj.price = decimal.Parse(price);

obj.sum = count \* decimal.Parse(price);

obj.product\_id = products.Where(y => y.product\_name == choiceProduct).Select(u => u.id).Max();

Session.countUpdate++;

Session.count++;

}

try

{ //Сохранение данных в БД

Instances.db.SaveChanges();

MessageBox.Show("Информация сохранена");

this.DialogResult = true;

} catch(Exception ex)

{

MessageBox.Show(ex.Message.ToString());

}

} else

{

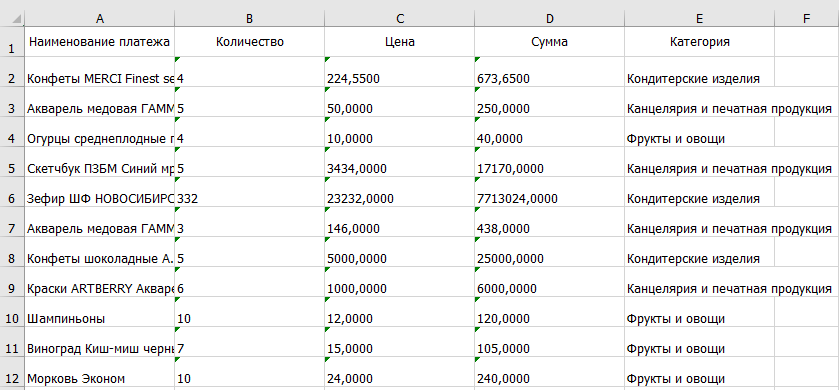
MessageBox.Show("Заполнены не все поля!");

}

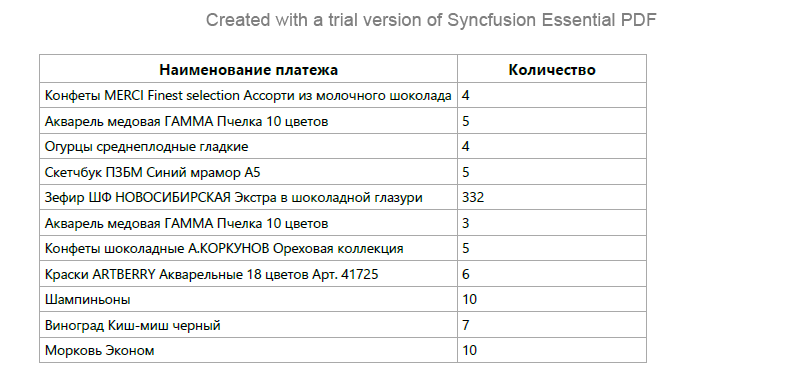
}

**Описание отчетов**

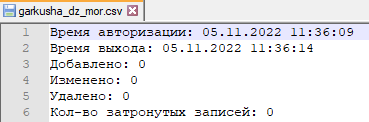
На основе данных о записях пользователя реализовано формирование отчетов в форматах Excel (рисунок 6) и PDF (рисунок 7) ,а также сохранения данных о сессии пользователя в .csv файл (рисунок 8) и её отображение в виде таблицы и   
графиков (рисунок 9).



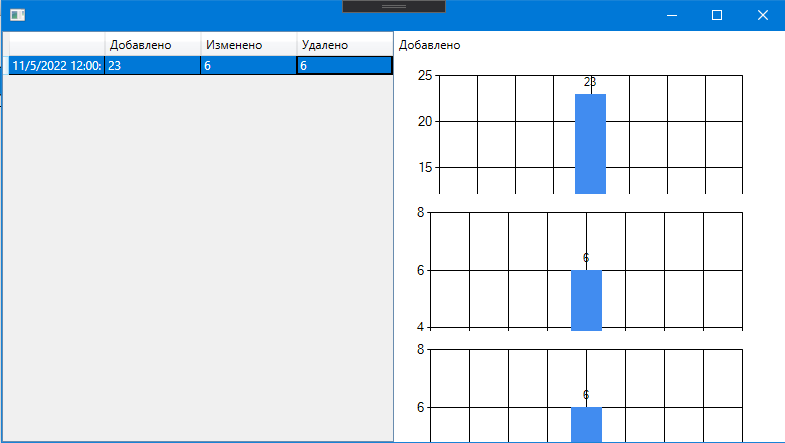
**Рисунок 6 - Отчёт в Excel**



**Рисунок 7 - Отчёт в PDF**



**Рисунок 8 - Отчёт о сессии**



**Рисунок 9 - Отчёт в виде графиков**

В листинге 4 представлен метод генерации Excel и PDF файлов отчётов по нажатию на кнопку «Отчет» с помощью Nuget-пакета Syncfusion.

Листинг 4. Генерация отчётов в Excel и PDF

private void otchet\_btn\_Click(object sender, RoutedEventArgs e)

{

//Добавление настроек для экспорта

var options = new ExcelExportingOptions();

options.ExportMode = ExportMode.Text;

// Генерация Excel файла

var excelEngine = ordersDataGrid.ExportToExcel(ordersDataGrid.View, options);

var workBook = excelEngine.Excel.Workbooks[0];

workBook.SaveAs("Sample.xlsx");

// Генерация PDF файла

var document = ordersDataGrid.ExportToPdf();

document.Save("Sample1.pdf");

MessageBox.Show("Отчёт сохранён");

}

В листинге 5 представлен метод сохранения данных о текущей сессии пользователя в .csv файл.

Листинг 5. Создание .csv файл с данными о сессии

private void Window\_Closed(object sender, EventArgs e)

{

// Указание пути сохранения

string mydocu = Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments);

string pathToSave = mydocu + [$@"\{user.login}.csv](mailto:$@%22\%7buser.login%7d.csv)";

// Сбор всех данных в одну переменную

var csv = new StringBuilder();

csv.AppendLine($"Время авторизации: {Session.dateAuth}");

csv.AppendLine($"Время выхода: {Session.dateExit=DateTime.Now}");

csv.AppendLine($"Добавлено: {Session.countAdd}");

csv.AppendLine($"Изменено: {Session.countUpdate}");

csv.AppendLine($"Удалено: {Session.countDelete}");

csv.AppendLine($"Кол-во затронутых записей: {Session.count}");

// Запись данных по пути сохранения

File.WriteAllText(pathToSave, csv.ToString());

MessageBox.Show("Логи сохранены");

}

В листинге 6 представлены создание графиков и вывод данных в DataGrid при открытии окна анализа активности на основе данных из таблицы «Анализ».

Листинг 7. Создание графиков и вывод в DataGrid данных из таблицы «Анализ»

public OtchetWindow(user user)

{

InitializeComponent();

this.user = user;

// Вывод данных из таблицы "Анализ" в DataGrid

dataGrid.ItemsSource = Instances.db.analizs.Where(u => u.user\_id == user.id).ToList();

// Настройка элементов WindowsFormsHost

chartAnalysAdded.ChartAreas.Add(new ChartArea("Main"));

chartAnalysDeleted.ChartAreas.Add(new ChartArea("Main"));

chartAnalysUpdate.ChartAreas.Add(new ChartArea("Main"));

var currentSeriesAdded = new Series("analys")

{

IsValueShownAsLabel = true

};

var currentSeriesDeleted = new Series("analys")

{

IsValueShownAsLabel = true

};

var currentSeriesUpdate = new Series("analys")

{

IsValueShownAsLabel = true

};

chartAnalysAdded.Series.Add(currentSeriesAdded);

chartAnalysDeleted.Series.Add(currentSeriesDeleted);

chartAnalysUpdate.Series.Add(currentSeriesUpdate);

// Получение данных из таблицы "Анализ" и создание на их основе графиков

var res = Instances.db.analizs.ToList().Where(p => p.user\_id == user.id).ToList();

for (int i = 0; i < res.Count; i++)

{

currentSeriesAdded.Points.AddXY(res[i].date, res[i].create);

currentSeriesUpdate.Points.AddXY(res[i].date, res[i].update);

currentSeriesDeleted.Points.AddXY(res[i].date, res[i].delete);

}

}

**Заключение**

В результате практики УП.11.02 Разработка приложения было разработано программное решение для учета платежей физических лиц. Были выполнены работы, согласно заданию, а именно: разграничение данных пользователей, окно с основным данными и их фильтрация, добавление/удаление/редактирование платежей, формирование отчета по основным данным.

# Используемые источники информации

1. Генерация PDF из WPF-приложения: https://habr.com/ru/post/201836/
2. Создание CAPCHA для Windows Form приложений: Справочник по C# : https://csharpcoderr.com/1346/
3. Построение графиков на WPF форме: https://habr.com/ru/post/145343/
4. SHA256 Класс: https://learn.microsoft.com/ru-ru/dotnet/api/system.security.cryptography.sha256
5. Сохранить данные из WPF DataGrid в CSV: <https://ru.stackoverflow.com/questions/1181568/wpf-datagrid>
6. Экспорт данных из DataGrid в Excel:

https://help.syncfusion.com/wpf/datagrid/export-to-excel