Министерство науки и высшего образования Российской Федерации

**Муромский институт (филиал)**

федерального государственного бюджетного образовательного учреждения высшего образования

**«Владимирский государственный университет   
имени Александра Григорьевича и Николая Григорьевича Столетовых»**

(МИВлГУ)

Факультет НАЗВАНИЕ ФАКУЛЬТЕТА



Кафедра НАЗВАНИЕ КАФЕДРЫ

КУРСОВАЯ

РАБОТА

по НАИМЕНОВАНИЕ ДИСЦИПЛИНЫ

(наименование дисциплины)

Тема ТЕМА РАБОТЫ

Руководитель

ФИО РУКОВОДИТЕЛЯ

(оценка) (фамилия, инициалы)

Члены комиссии

(подпись) (дата)

Студент НАЗВАНИЕ ГРУППЫ

(подпись) (Ф.И.О.) (группа)

ТВОЕ ФИО

(подпись) (Ф.И.О.) (фамилия, инициалы)

(подпись) (дата)

Муром 2024

В данной курсовой работе в соответствии с заданием разработана информационная система для организации планирования рабочего времени пользователя. Проанализировано техническое задание, сформирован список требований к системе и произведено обоснование выбора средств реализации информационной системы. Выполнено проектирование структуры базы данных, реализована база данных с использованием СУБД Firebird. Разработано пользовательское приложение с использованием среды разработки Microsoft Visual Studio 2022.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc155006154)

[1. Анализ технического задания 5](#_Toc155006155)

[1.1 Анализ предметной области 5](#_Toc155006156)

[1.2 Формирование требований к системе 6](#_Toc155006157)

[2. Проектирование базы данных 8](#_Toc155006158)

[3. Разработка базы данных 12](#_Toc155006159)

[4. Разработка клиентского приложения 14](#_Toc155006160)

[4.1 Реализация работы с данными 14](#_Toc155006161)

[4.2 Реализация логики приложения и пользовательского интерфейса 14](#_Toc155006162)

[5. Тестирование программного продукта 18](#_Toc155006163)

[6. Руководство по программному продукту 20](#_Toc155006164)

[6.1 Руководство пользователя 20](#_Toc155006165)

[6.2 Руководство администратора 26](#_Toc155006166)

[ЗАКЛЮЧЕНИЕ 27](#_Toc155006167)

ВВЕДЕНИЕ

В настоящее время, в связи с возросшими потоками информации, которую необходимо обработать для принятия верного решения, разрабатываются специальные программные продукты, позволяющие представить информацию в более наглядной для пользователя форме. К подобным программным продуктам относятся базы данных.

База данных – это совокупность записей различного типа, содержащая перекрестные ссылки. Базы данных не помогают пользователю в принятии решения, они только предоставляют всю полноту информации необходимую для принятия этого решения, а вся ответственность за результат применения этого решения возлагается на пользователя.

Широкое применение базы данных нашли в областях формирования различного рода документации, где необходимо не только создать бумажную копию некоторого документа, но и сохранить ее для возможного обращения в будущем. Типовая информационная система ориентирована на сбор и обработку информации и включает в себя базу данных и средства работы с ней.

Внедрение информационных технологий на любой фирме обуславливает не только эффективность процессов, но также облегчает всю работу большим количеством данных.

В данном курсовом проекте необходимо разработать информационную систему планирования рабочего времени пользователя. Разработанный программный продукт может быть полезен для компаний или отдельных пользователей с большим количеством ежедневных задач, выполнение которых нужно отслеживать, а также получать различные отчеты по разным категориям задач.

1. Анализ технического задания

1.1 Анализ предметной области

В данном курсовом проекте необходимо разработать информационную систему для планирования рабочего времени пользователя.

Ниже описан процесс работы сотрудника условной компании при работе с рабочими задачами.

При поступлении рабочего задания сотрудник фиксирует его. При этом нужно указать:

- название;

- дату и время принятия задачи в работу;

- указывается метка важности (срочная задача или нет);

- указывается категория задачи.

При выполнении рабочего задания сотруднику нужно указать дату и время выполнения рабочего задания

При необходимости пользователю нужно вносить изменения рабочего задания:

- переносить дату выполнения;

- изменять важность задания;

- изменять категорию задания;

При ведении планирования без использования информационной системы пользователю придется работать с большим количеством не структурированной информации. Как правило учет рабочих заданий ведется в бумажной форме, при этом поиск и отслеживание становится сложной задачей. Если нужно контролировать работу нескольких сотрудников задача в разы усложняется.

1.2 Формирование требований к системе

Исходя из приведенного выше анализа предметной области, сформулированы следующие основные требования к разрабатываемой информационной системе:

- информационная система должна быть многопользовательской, в базе данных необходимо независимо хранить задачи разных пользователей;

- в базе данных нужно хранить список задач (журнал задач);

- в журнале задач нужно реализовать добавление задачи, удаление задачи, редактирование задачи;

- в базе данных задачи разбиваются на категории (справочник категории);

- в базе данных у задач есть метка важности (справочник важность задачи);

- для отслеживания статуса выполнения задачи (справочник статус задачи);

- для организации многопользовательской работы с базой данных (справочник пользователи);

- информационная система должна иметь клиент – серверную архитектуру, для организации одновременной работы нескольких пользователей.

- по запросу пользователя информационная система должна предоставлять отчеты:

- по невыполненным задачам (важная задача);

- по отложенным задачам (обычная задача).

Система должна удовлетворять следующим требованиям надежности работы:

- система должна корректно обрабатывать ошибки ввода данных, допущенные пользователем;

- при возникновении ошибок, система должна сообщать пользователю описание ошибки и рекомендации по ее устранению (если возможно).

2. Проектирование базы данных

Для проектирования и реализации информационной системы в курсовом проекте использовалось программное обеспечение IBExpert. Данное программное обеспечение является бесплатным и имеет графический интерфейс, что для проектирования базы данных очень важно. В IBExpert можно сгенерировать ER-диаграмму, которая будет графически отражать базу данных.

Процесс построения информационной модели состоит из следующих шагов:

- определение сущностей;

- определение зависимостей между сущностями;

- задание первичных и внешних ключей;

- определение атрибутов сущностей;

- приведение модели к требуемому уровню нормальной формы.

На основании анализа предметной области, в модели базы данных выделены следующие основные сущности:

- категория задачи;

- важность задачи;

- статус выполнения задачи;

- пользователь системы;

- список задач.

- задача;

При проектировании базы данных сущности связываются один ко многим.

Следующим этапом является проектирование физической модели базы данных.

Физические модели баз данных определяют способы размещения данных в среде хранения и способы доступа к этим данным, которые поддерживаются на физическом уровне.

Для реализации поставленной задачи, используется СУБД Firebird. Данная СУБД является реляционной, следовательно, физическая модель данных будет построена по следующим правилам:

- каждой сущности соответствует таблица физической модели;

- каждому атрибуту сущностей соответствует атрибут таблицы физической модели;

- реляционные отношения преобразуются в ограничения целостности данных.

Таблица 1 Структура таблицы CATEGORY (Категория)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Первичный ключ | Внешний ключ |
| ID | INTEGER | + |  |
| NAME | VARCHAR |  |  |

Таблица 2 Структура таблицы IMPORTANCE (Важность)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Первичный ключ | Внешний ключ |
| ID | INTEGER | + |  |
| NAME\_IMPORTANCE | VARCHAR |  |  |

Таблица 3 Структура таблицы STATUS (Статус выполнения)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Первичный ключ | Внешний ключ |
| ID | INTEGER | + |  |
| NAME\_STATUS | VARCHAR |  |  |

Таблица 4 Структура таблицы USER\_TODO (Пользователь системы)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Первичный ключ | Внешний ключ |
| ID | INTEGER | + |  |
| LOGIN | VARCHAR |  |  |
| LOGO | VARCHAR |  |  |
| PHRASE | VARCHAR |  |  |

Таблица 5 Структура таблицы TODO (Журнал задач)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя поля | Тип данных | Первичный ключ | Внешний ключ |
| ID | INTEGER | + |  |
| NAME\_TASK | VARCHAR |  |  |
| DATE\_TASK\_START | TIMESTAMP |  |  |
| DATE\_TASK\_END | TIMESTAMP |  |  |
| ID\_STATUS | INTEGER |  | + |
| ID\_IMPORTANCE | INTEGER |  | + |
| ID\_CATEGORY | INTEGER |  | + |
| ID\_USER | INTEGER |  | + |

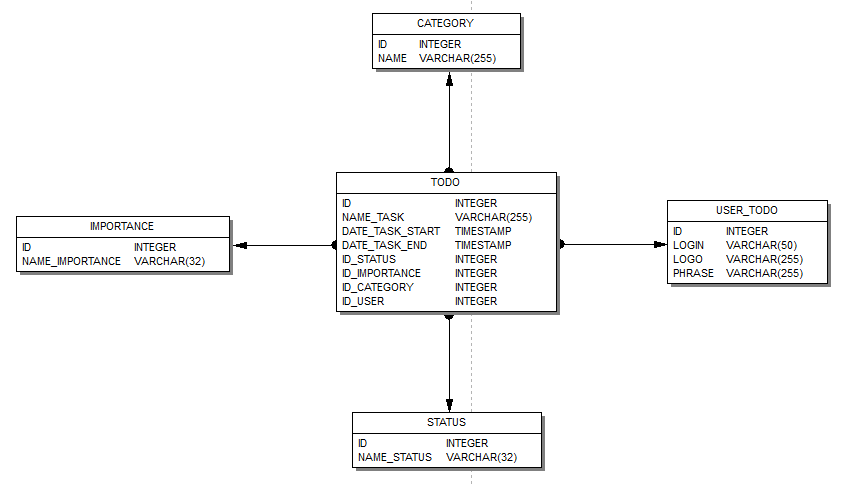


Рисунок 1 - ER-диаграмма

Первичные ключи в базе данных генерируются внутренними средствами (автоинкрементном).

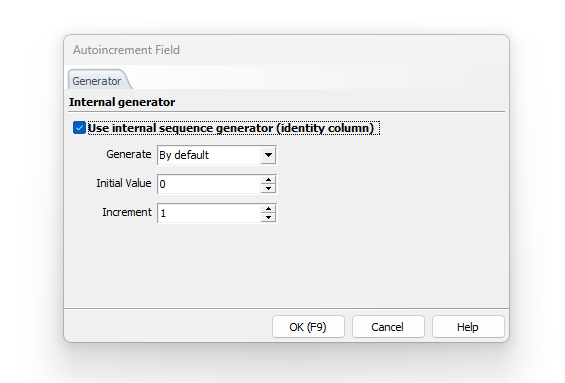


Рисунок 2 – генерация первичного ключа

3. Разработка базы данных

База данных реализована на СУБД Firebird версии 4.0. Для управления СУБД и реализации объектов базы данных использовано средство администрирования IBExpert версии 2022.

Таблицы базы данных созданы в соответствии с разработанной ранее физической моделью данных. Ниже приведены SQL скрипты для создания таблиц и связей между ними.

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\* Generated by IBExpert 2022.3.4.1 29.12.2023 10:38:37 \*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

SET SQL DIALECT 3;

SET NAMES UTF8;

CREATE DATABASE 'C:\Users\admin\source\repos\C-Sharp-ToDo\_DataManagement\bin\Debug\db\TODO.FDB'

USER 'SYSDBA' PASSWORD 'masterkey'

PAGE\_SIZE 16384

DEFAULT CHARACTER SET UTF8 COLLATION UTF8;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\* Tables \*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

CREATE TABLE CATEGORY (

ID INTEGER GENERATED BY DEFAULT AS IDENTITY,

NAME VARCHAR(255)

);

CREATE TABLE IMPORTANCE (

ID INTEGER GENERATED BY DEFAULT AS IDENTITY,

NAME\_IMPORTANCE VARCHAR(32)

);

CREATE TABLE STATUS (

ID INTEGER GENERATED BY DEFAULT AS IDENTITY,

NAME\_STATUS VARCHAR(32)

);

CREATE TABLE TODO (

ID INTEGER GENERATED BY DEFAULT AS IDENTITY,

NAME\_TASK VARCHAR(255),

DATE\_TASK\_START TIMESTAMP,

DATE\_TASK\_END TIMESTAMP,

ID\_STATUS INTEGER,

ID\_IMPORTANCE INTEGER,

ID\_CATEGORY INTEGER,

ID\_USER INTEGER

);

CREATE TABLE USER\_TODO (

ID INTEGER GENERATED BY DEFAULT AS IDENTITY,

LOGIN VARCHAR(50),

LOGO VARCHAR(255),

PHRASE VARCHAR(255)

);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\* Autoincrement generators \*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ALTER TABLE CATEGORY ALTER ID RESTART WITH 9;

ALTER TABLE IMPORTANCE ALTER ID RESTART WITH 3;

ALTER TABLE STATUS ALTER ID RESTART WITH 4;

ALTER TABLE TODO ALTER ID RESTART WITH 21;

ALTER TABLE USER\_TODO ALTER ID RESTART WITH 3;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\* Primary keys \*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ALTER TABLE CATEGORY ADD CONSTRAINT PK\_CATEGORY PRIMARY KEY (ID);

ALTER TABLE IMPORTANCE ADD CONSTRAINT PK\_IMPORTANCE PRIMARY KEY (ID);

ALTER TABLE STATUS ADD CONSTRAINT PK\_STATUS PRIMARY KEY (ID);

ALTER TABLE TODO ADD CONSTRAINT PK\_TODO PRIMARY KEY (ID);

ALTER TABLE USER\_TODO ADD CONSTRAINT PK\_USER\_TODO PRIMARY KEY (ID);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\* Foreign keys \*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

ALTER TABLE TODO ADD CONSTRAINT FK\_TODO\_1 FOREIGN KEY (ID\_STATUS) REFERENCES STATUS (ID);

ALTER TABLE TODO ADD CONSTRAINT FK\_TODO\_2 FOREIGN KEY (ID\_IMPORTANCE) REFERENCES IMPORTANCE (ID);

ALTER TABLE TODO ADD CONSTRAINT FK\_TODO\_3 FOREIGN KEY (ID\_CATEGORY) REFERENCES CATEGORY (ID);

ALTER TABLE TODO ADD CONSTRAINT FK\_TODO\_4 FOREIGN KEY (ID\_USER) REFERENCES USER\_TODO (ID);

4. Разработка клиентского приложения

4.1 Реализация работы с данными

Программа для работы с базой данных реализована на платформе .NET Framework версии 4.8. Среда разработки – Microsoft Visual Studio 2022, язык программирования – C#. Для разработки проекта использована полноценная версия Firebird 4.0. При разработке используем локальный сервер Firebird 4.0.

Работа с базой данных реализована с помощью компонента для Microsoft Visual Studio - FirebirdSql.Data.FirebirdClient, с помощью которого можно делать запросы непосредственно из проекта.

Для генерации отчетов из информационной системы используем компонент Microsoft.Office.Interop.Excel, который позволяет легко выгрузить отчет в формат \*.xlsx.

4.2 Реализация логики приложения и пользовательского интерфейса

Клиентское программное обеспечение построено на платформе .NET Framework 4.8. Проект реализован как приложение WindowsForms. Ниже приведен перечень программных модулей проекта.

Windows Forms — интерфейс программирования приложений (API), отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework. Данный интерфейс упрощает доступ к элементам интерфейса Microsoft Windows за счет создания обёртки для существующего Win32 API в управляемом коде. Причём управляемый код — классы, реализующие API для Windows Forms, не зависят от языка разработки. То есть программист одинаково может использовать Windows Forms как при написании ПО на C#, C++, так и на VB.Net, J# и др.

Таблица 6 Структура проекта пользовательского приложения

|  |  |
| --- | --- |
| Имя файла исходного кода | Описание |
| Program.cs | Точка входа в приложение |
| Login.cs | Форма авторизации пользователя |
| Main.cs | Форма журнала задач |
| AddEditTask.cs | Форма добавления и редактирования задачи |
| CategoryList.cs | Форма выбора категории задачи |

Подключение к базе данных происходит после выбора базы данных и ввода логина и пароля в форме авторизации пользователя.

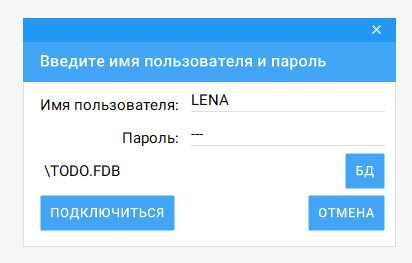


Рисунок 3 – Форма авторизации пользователя

private void materialRaisedButton1\_Click(object sender, EventArgs e) {

if (this.materialSingleLineTextField1.Text.Length != 0 && this.materialSingleLineTextField2.Text.Length != 0){

fb\_cons = new FbConnectionStringBuilder();

fb\_cons.Charset = "UTF8";

fb\_cons.UserID = this.materialSingleLineTextField1.Text;

fb\_cons.Password = this.materialSingleLineTextField2.Text;

fb\_cons.Dialect = 3;

fb\_cons.Database = Path.Combine(Directory.GetCurrentDirectory(), "db") + materialLabel3.Text;

fb\_cons.ServerType = 0;

fb\_cons.DataSource = "localhost";

fb\_cons.Port = 3050;

try {

fbCon = new FbConnection(fb\_cons.ToString());

fbCon.Open();

fbCon.Close();

Program.AddUpdateAppSettings("ConnectionString", fb\_cons.ToString());

Program.AddUpdateAppSettings("Login", fb\_cons.UserID.ToString());

Program.AddUpdateAppSettings("Database", materialLabel3.Text.Replace("\\", ""));

Main toDoList = new Main();

toDoList.Show();

this.Hide();

}

catch (Exception){

MessageBox.Show("Введен неверный логин или пароль или указан неверный путь к БД!");

}

finally {

fbCon.Close();

}

}

else

{

materialSingleLineTextField1.BackColor = System.Drawing.Color.Tomato;

materialSingleLineTextField2.BackColor = System.Drawing.Color.Tomato;

}

}

private void materialRaisedButton3\_Click(object sender, EventArgs e) {

openFileDialog1.InitialDirectory = Path.Combine(Directory.GetCurrentDirectory(), "db");

if (openFileDialog1.ShowDialog() == DialogResult.OK) {

if (Path.GetDirectoryName(openFileDialog1.FileName) !=

Path.Combine(Directory.GetCurrentDirectory(), "db")) {

MessageBox.Show("Пожалуйста выберите БД в директории db в корне программы", "Неправильная директория с БД", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else {

materialLabel3.Text

= openFileDialog1.FileName.Replace(Path.Combine(Directory.GetCurrentDirectory(), "db"), "");

}

}

}

Работа пользователя с приложением построена в диалоговом режиме. Для выполнения команд пользователя, отображается соответствующее модальное диалоговое окно, в котором реализован определенный функционал. Логика работы программы после закрытия модального окна может быть различной, в зависимости от результата выполнения операции в модальном окне. Для этого производится анализ возвращаемого результата диалогового окна.

5. Тестирование программного продукта

Тестирование разработанного программного продукта выполняется с целью проверки соответствия реализованного функционала требованиям технического задания, а также проверки надежности работы и отсутствия ошибок в реализации функционала. Ниже приведена методика тестирования программного продукта.

Таблица 7 Методика тестирования программного продукта

|  |  |
| --- | --- |
| Выполняемая функция | Требуемый результат |
| 1. Запустить файл C-Sharp-ToDo\_DataManagement.exe | Отобразится окно авторизации |
| 2. Указать пусть к базе данных, а также логин и пароль и нажать кнопку подключиться | Отобразится главное окно клиентского приложения с задачами на текущий день |
| 3. В главном окне нажать кнопку ДОБАВИТЬ | Отобразится модальное окно формы добавления задачи. |
| 4. Заполнить данные о задаче и нажать кнопку ПРИМЕНИТЬ | Запись добавиться в базу данных |
| 5. В главном окне нажать кнопку ИЗМЕНИТЬ | Появится модальное окно формы редактирования задачи |
| 6. Изменить данные о задаче и нажать кнопку ПРИМЕНИТЬ | Измененная запись добавится в базу данных |
| 7. В главном окне нажать на кнопку УДАЛИТЬ | Выделенная запись удалится из базы данных |
| 8. В главном окне выбрать нужную категорию | Отобразятся задачи выбранной категории |
| 9. В главном окне ввести поисковую фразу | Отобразятся задачи, соответствующие поисковому запросу |
| 10. В главном окне нажать кнопку Обновить | В главном окне отобразятся задачи на текущий день, всех категорий |
| 11. Выбрать на календаре нужную дату в главном окне | В главном окне отобразятся задачи на выбранный день |
| 12. Нажать на кнопку ЗАВЕРШИТЬ ЗАДАЧУ | В базе данных изменится задача и ей присвоится статус Выполнено |
| 13. Нажать на кнопку ОТЧЕТ ПО НЕВЫПОЛНЕННЫМ ЗАДАЧАМ | Создастся файл \*xlsx с просроченными задачами (статус обязательно, но просроченными) |
| 14. Нажать на кнопку ОТЧЕТ ПО ОТЛОЖЕННЫМ ДЕЛАМ | Создастся файл \*.xlsx с не обязательными задачами (статус - Можно отложить) |
| 15. Нажимаем на кнопку ЗАКРЫТЬ | Выход из клиентского приложения |

Результаты, полученные в ходе тестирования, позволяют сделать заключение в том, что реализация программного продукта соответствует требованиям технического задания.

6. Руководство по программному продукту

6.1 Руководство пользователя

Для начала работы с программой, необходимо запустить исполняемый файл приложения C-Sharp-ToDo\_DataManagement.exe. После запуска приложения, будет отображено окно входа в систему.

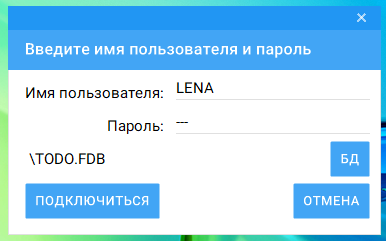


Рисунок 3 – Форма авторизации пользователя

После успешной авторизации отобразится главное окно приложения.

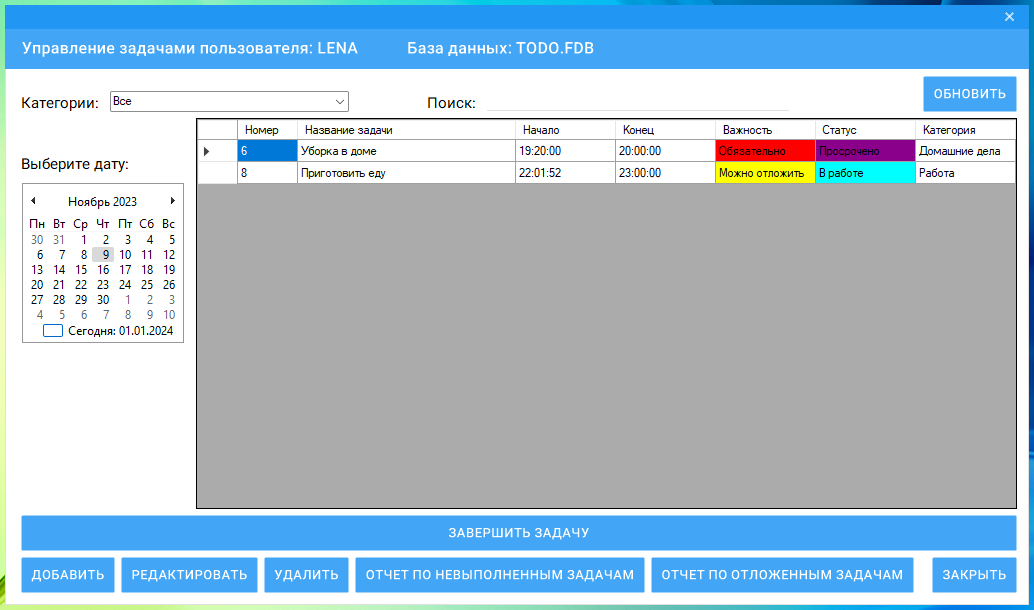


Рисунок 4 – Главное окно приложения

Пользователь может управлять задачами (добавление, редактирование, удаление), присваивать статус завершения задачи, производить поиск по задачам, а также генерировать отчеты:

- по невыполненным задачам;

- по отложенным задачам.

Для добавления задачи нужно нажать кнопку Добавить.

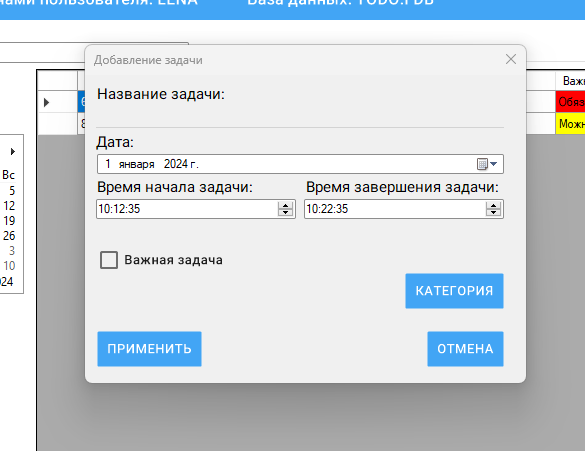


Рисунок 5 – Окно добавления задачи

Для задачи нужно установить нужные параметры:

- название;

- дату;

- время начала;

- время завершения;

- выбрать категорию;

- установить при необходимости статус (Важная задача)

Нажать кнопку ПРИМЕНИТЬ, для добавления задачи или ОТМЕНА, если пользователь передумал создавать задачу.

Для выбора категории в базе данных есть справочник КАТЕГОРИИ:

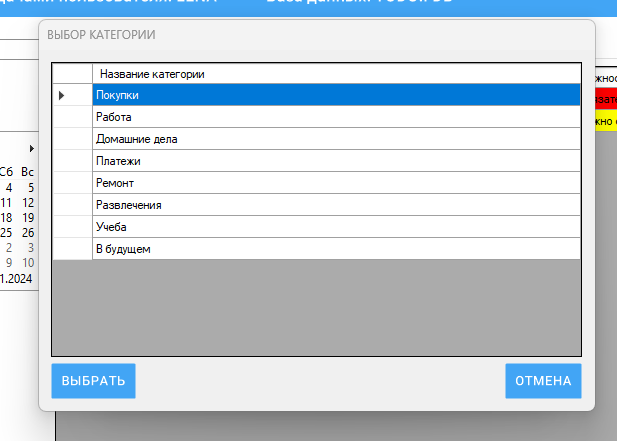


Рисунок 6 – Справочник категории

Если необходимо, пользователь может изменить задачу, для этого нужно выбрать ее и нажать на кнопку РЕДАКТИРОВАТЬ.

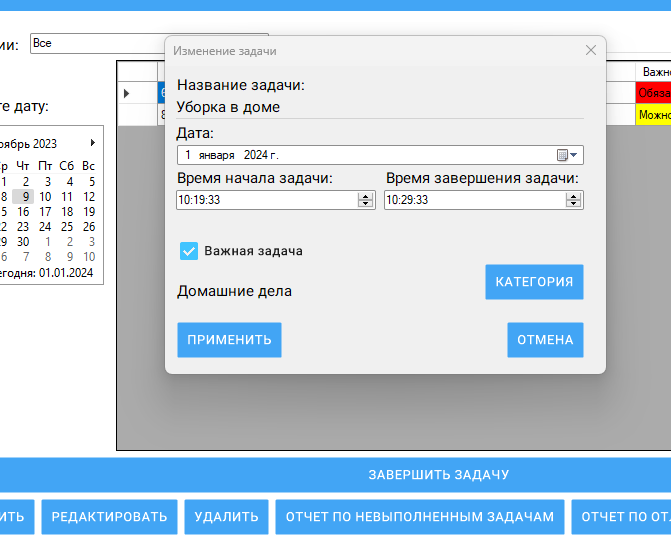


Рисунок 7 – Окно редактирования задачи

После внесения необходимых изменений нужно нажать кнопку ПРИМЕНИТЬ, или кнопку ОТМЕНА, если пользователь передумал редактировать задачу.

Для удаления задачи нужно выделить ее и нажать на кнопку УДАЛИТЬ, после этого задача будет удалена из базы данных.



Рисунок 8 – Кнопка удаления задачи

Для применения фильтра по категориям в главном окне можно выбрать нужную категорию в выпадающем списке, после этого в списке задач будут только задачи нужной категории.

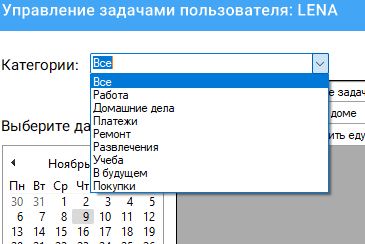


Рисунок 9 – Список выбора категории

Для поиска задач, в системе предусмотрен данный функционал. Если нужно найти задачу в длинном списке, пользователь может ввести частичное название в поле и система произведет поиск.



Рисунок 10 – Поиск по списку задач

Для сортировки списка задач, можно воспользоваться заголовками у таблицы.



Рисунок 11 – Сортировка в таблице задач

Для выбора задач на определенную дату можно воспользоваться календарем, после выбора даты, в списке задач отобразятся задачи на этот день.

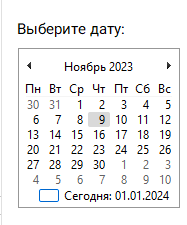


Рисунок 12 – Выбор задач на определенную дату

Для завершения задачи (когда она выполнена) пользователь выбирает ее и нажимает на кнопку ЗАВЕРШИТЬ ЗАДАЧУ, после этого задаче будет присвоен статус ЗАВЕРШЕНО.



Рисунок 13 – Кнопка завершения задачи

В системе предусмотрено 2 вида отчетов:

- по невыполненным задачам;

- по отложенным задачам.

Для генерации нужного отчета, пользователь нажимает на нужную кнопку, после этого создается файл \*.xlsx в директории reports, с префиксом нужного отчета.

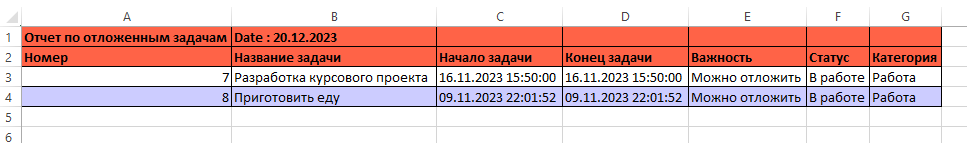


Рисунок 14 – Отчет по отложенным задачам

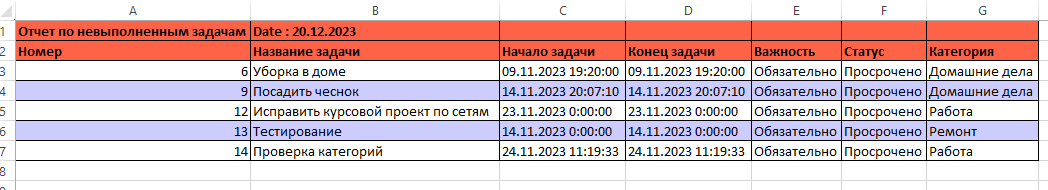


Рисунок 15 – Отчет по просроченным задачам

Для выхода из системы пользователь должен нажать на кнопку ЗАКРЫТЬ.



Рисунок 16 – Кнопка выхода из системы

6.2 Руководство администратора

Информационная система состоит из клиентского приложения и базы данных. База данных может быть размещена как на ПК, где непосредственно используется программа, так и на отдельном сервере в сети. Для функционирования клиентских приложений, необходимо наличие следующих продуктов, установленных на ПК пользователей:

- операционная система Microsoft Windows 7/8/10;

- .NET Framework 4.8

Для обеспечения связи клиентских рабочих мест с сервером БД, необходима локальная сеть с поддержкой протокола TCP/IP.

Для развертывания серверной стороны ИС, необходимо:

- операционная система Microsoft Windows 7/8/10;

- Firebird SQL Server версии 4.0.

При первоначальном развертывании системы, на серверной стороне необходимо подключить и настроить базу данных со следующими параметрами:

- Имя базы данных: TODO.FDB;

- Пользователь: SYSDBA;

- Пароль: masterkey.

- Кодовая страница базы данных: UTF-8.

ЗАКЛЮЧЕНИЕ

В данном курсовом проекте в соответствии с заданием разработана информационная система для планирования рабочего времени.

В ходе работы над курсовым проектом, были выполнены следующие задачи:

- анализ технического задания;

- анализ предметной области;

- проектирование базы данных;

- реализация базы данных;

- реализация клиентского программного обеспечения.

Для проектирования базы данных было использовано ПО IBExpert. Реализация базы данных осуществлена с помощью языке SQL-запросов. Разработанная программа позволяет выполнять следующие задачи:

- вести учет задач пользователей;

- формировать отчет по невыполненным задачам;

- формировать отчет по отложенным задачам;

- в информационной системе реализован удобный поиск по базе данных;

- в информационной системе реализован фильтр по категориям задач;

- в информационной системе реализован просмотр задач на определенный день.

Разработанный программный продукт может быть полезен для применения в различных фирмах, а также отдельных пользователей, для организации планирования рабочего времени.