Министерство образования и науки Российской Федерации

федеральное государственное бюджетное образовательное

учреждение высшего образования

«Алтайский государственный технический университет

им. И.И. Ползунова»

Факультет \_\_информационных технологий\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кафедра \_\_\_прикладной математики\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Специальность (направление, профиль) \_\_ПИ\_\_\_\_\_\_\_\_\_\_\_\_\_

Курсовой проект

защищен с оценкой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.С. Троицкий

(подпись руководителя проекта) (инициалы, фамилия)

“\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Курсовой проект

Разработка программного обеспечения для создания и хранения заметок \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(тема курсового проекта)

Пояснительная записка

по дисциплине \_\_\_\_\_\_Программирование\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_КП 09.03.04.19.000 ПЗ\_\_\_\_\_\_\_

(обозначение документа)



Студент группы ПИ91 Карнаухова В.О.\_ \_\_\_\_\_\_06.01.2021

(фамилия, имя, отчество) (подпись) (дата)



Студент группы ПИ91 Арестова В.И.\_\_ \_\_\_\_\_\_06.01.2021

(фамилия, имя, отчество) (подпись) (дата)

Руководитель

проекта доцент, к.т.н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.С. Троицкий

(должность, ученое звание) (подпись) (инициалы, фамилия)

БАРНАУЛ 2020

Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО «Алтайский государственный технический университет

имени И.И. Ползунова»

Факультет информационных технологий

Кафедра «Прикладная математика»

З А Д А Н И Е

на курсовой проект по дисциплине «Программирование»

студенту группы ПИ-91 Карнауховой Виктории Олеговне

Тема курсового проекта: «Разработка программного обеспечения для создания и хранения заметок».

Календарный план работы:

|  |  |  |
| --- | --- | --- |
| № этапа | Содержание этапа | Недели семестра |
| 1 | Получение задания | 1 |
| 2 | Постановка задачи. Работа с документацией | 2 |
| 3 | Проектирование программы | 3-4 |
| 4 | Реализация программы | 5-13 |
| 5 | Оформление пояснительной записки | 14 |
| 6 | Защита курсового проекта | 15-16 |

Руководитель проекта\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Троицкий В.С., кандидат технических наук, доцент подпись

Дата выдачи задания «05» февраля 2020 г.

число месяц год



Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Карнаухова В.О.

подпись



Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Арестова В.И.

подпись

Содержание

[Введение 4](#_Toc60785500)

[1 Обзор предметной области и постановка задачи 5](#_Toc60785501)

[1.1 Обзор предметной области 5](#_Toc60785502)

[1.2 Постановка задачи 6](#_Toc60785503)

[2 Проектирование 7](#_Toc60785504)

[3 Реализация 9](#_Toc60785505)

[3.1 Выбор средств реализации Desktop-приложения 9](#_Toc60785506)

[3.2 Структура программы 9](#_Toc60785507)

[3.3 Описание реализации ПО 9](#_Toc60785508)

[3.4 Выбор средств реализации Android-приложения 10](#_Toc60785509)

[3.5 Структура программы 11](#_Toc60785510)

[3.6 Описание реализации ПО 11](#_Toc60785511)

[Заключение 13](#_Toc60785512)

[Список использованной литературы 14](#_Toc60785513)

[Приложение А 15](#_Toc60785514)

[Приложение Б 43](#_Toc60785515)

# ****Введение****

В современном мире человеку приходится постоянно сталкиваться с огромным количеством разнообразной информации. Если постоянно держать в голове эту информацию, можно попросту забыть о ней. Чтобы такого не происходило, нужно как-то систематизировать свои мысли и планы в виде тезисов-заметок. Так как чаще всего у нас под рукой оказывается телефон, чем блокнот и ручка, мы решили создать приложение для создания и хранения заметок для одной из самых популярных мобильных операционных систем-Android. Однако в рабочее время большую часть времени человек проводит за компьютером, поэтому также мы решили создать аналогичное десктопное приложение с графическим интерфейсом под распространенную настольную операционную систему Windows.

Мы выделили несколько причин для использования наших приложений для создания и хранения заметок:

1)Удобство в использовании. Благодаря двум версиям ежедневника для разных устройств, он будет доступен практически всегда. Вы всегда сможете записать какую-то идею, нужные контакты людей.

2)Планирование своего времени. В заметках можно распланировать не только день, но и неделю, месяц, а также поставить какие-то долгосрочные цели.

3)Изучение иностранных языков. Заметками можно пользоваться как Вашим личным словарем и записывать в него иностранные слова и словосочетания.

4) Учет доходов и расходов. Если Вам стало сложно отслеживать свои расходы, записывайте их в заметки. Тем самым Вы сможете увидеть, сколько потратили денег и на что.

Существует множество приложений ,которые помогают создавать заметки, как для телефона, так и для ПК. Цель данной курсовой работы-создать такой программный продукт, который бы сочетал в себе простоту использования и необходимый функционал.

Главная задача: создать два приложения (Desktop/Android) с удобным интерфейсом для работы с заметками.

# 1 Обзор предметной области и постановка задачи

* 1. Обзор предметной области

В современном мире человеку приходится постоянно сталкиваться с огромным количеством разнообразной информации. Если постоянно держать в голове эту информацию, можно попросту забыть о ней. Чтобы такого не происходило, нужно как-то систематизировать свои мысли и планы в виде тезисов-заметок.

*Заметка* - это краткая запись о чем-либо.

Различные приложения для цифровых устройств решают проблему поиска блокнота и ручки, а также нужной заметки. Кроме того, во многих приложениях существует функция напоминания, что существенно облегчает жизнь человеку.

Заметки можно создавать по разным причинам. Например, можно пользоваться ими как личным словарем и записывать в него иностранные слова и словосочетания. Также в приложении можно вести учет доходов и расходов, тем самым контролируя свой бюджет. Записывать бизнес идеи для реализации в будущем или составлять план на день, чтобы он всегда был под рукой.

*Создать заметку* – создать файл с произвольным текстом с помощью приложения «Заметки», который может быть открыт в этом же приложении для дальнейших действий: редактирования, удаления. Этот файл будет отображаться в приложении как строка с заголовком, при нажатии на которую открывается полный текст заметки и отображается дата создания заметки.

*Удалить заметку* – удалить файл с произвольным текстом с помощью приложения «Заметки». При этом из перечня заметок исчезает строка с заголовком и текст полностью удаляется из памяти устройства.

Заметки также можно *редактировать* – вносить изменения, исправления, добавления, изменения структуры текста. Ненужные заметки легко удалить, чтобы они не мешались впоследствии.

*Работа с персональными данными* - любыми сведениями, относящимися к прямо или косвенно определённому физическому лицу - в приложении осуществляться не должна. Также не должно быть авторизации в приложении, чтобы пользователю не приходилось каждый раз вводить свои логин и пароль от учетной записи, в любой момент можно было быстро открыть приложение и создать новую заметку. Ведь наша главная цель - создать такой программный продукт, который бы сочетал в себе простоту использования и необходимый функционал.

1.2 Постановка задачи

Создать приложение с дружественным интерфейсом, организующее работу с заметками. Приложение должно работать оффлайн (не требовать подключения к интернет-соединению) и быть доступным на двух устройствах в качестве обособленных приложений с одинаковыми функциями.

Реализация на разных устройствах:

Desktop-приложение – Карнаухова В.О.

Android-приложение – Арестова.В.И.

**Основные функции приложения:**

-создание заметки

-редактирование заметки

-удаление заметки

-возможность работать как на компьютере, так и в мобильной версии

# 2 Проектирование

**Проект программного продукта для Desktop-приложения**

**2.1.Структура данных**

В данном приложении реализованы:

Формы:

* Form\_main – главная форма проекта. Содержит меню, в котором есть пункты «Заметки», «Важное» и «Ежедневник» (объект menuStrip) и текст приветствия (объект label). При нажатии на какой-либо пункт меню текст приветствия скрывается (свойство Hide()) и появляются (свойство Show()) заметки/важные заметки/заметки из ежедневника (из файла note.txt/important\_note.txt/planner.txt с помощью TextBox), содержит кнопки «Создать заметку», «Создать важную заметку», «Создать заметку для ежедневника» (объект Button).
* New\_Note - форма для создания новой заметки. Содержит поля для ввода заголовка, текста заметки и поле, в котором отображается текущая дата создания заметки(объекты TextBox). Также содержит кнопку «Сохранить заметку» (объект Button).
* Note – форма для работы с конкретной заметкой. Отображает заголовок заметки, дату создания и текст заметки (из файла note.txt с помощью TextBox). Содержит кнопки для редактирования и удаления заметки ( объекты Button).
* Edit\_note-форма для редактирования заметки. Отображает заголовок заметки, текст заметки (из файла note.txt с помощью TextBox) и дату создания (с помощью компонента dateTimePicker). Содержит кнопку для сохранения заметки, при нажатии на которую происходит изменение текста заметки в файле note.txt/important\_note.txt (заметки/важные заметки).
* Planner\_note- форма для работы с конкретной заметкой ежедневника. Отображает заголовок заметки, дату создания, срок выполнения и текст заметки (из файла note.txt с помощью TextBox). Содержит кнопки для редактирования и удаления заметки ( объекты Button).
* New\_planner\_note - форма для создания новой заметки в ежедневнике. Содержит поля для ввода заголовка, текста заметки,поле, в котором отображается текущая дата создания заметки(объекты TextBox) и поле для выбора срока выполнения (компонент dateTimePicker). Также содержит кнопку «Сохранить заметку в ежедневник» (объект Button).
* Edit\_planner\_note-форма для редактирования заметки в ежедневнике. Отображает заголовок заметки, текст заметки (из файла note.txt с помощью TextBox), дату создания и срок выполнения (с помощью компонента dateTimePicker). Содержит кнопку для сохранения заметки, при нажатии на которую происходит изменение текста заметки в файле planner.txt.

**2.2. Структура файлов**

Для работы программы необходим исполняемый файл программы.

Заметки хранятся в файле note.txt. Важные заметки хранятся в файле important\_note.txt . Заметки ежедневника хранятся в файле planner.txt. При создании заметки вся информация записывается в файл, при демонстрации заметки нужные данные считываются из файла, соответственно, при удалении файла, заметки, сделанные пользователем, недоступны.

**Проект программного продукта для Android-приложения**

**2.3.Структура данных**

В данном приложении реализованы:

Активности:

* mainActivity – главная активность (форма) проекта. Отображает заметки пользователя (из базы данных SQLite с помощью TextView) в виде прокручивающегося горизонтального списка (в основе объект recycler для прокручивания списка, список LinearLayout и объекты TextView внутри), содержит кнопку «+» (объект Button) для создания новой заметки.
* CreateActivity - активность для создания новой заметки. Содержит поля для ввода заголовка заметки (объект EditText), дату создания заметки (объект TextView, с информацией о дате на настоящий момент) и текст заметки (объект EditText). Содержит кнопку «сохранить» (объект Button) для сохранения данных в БД.
* OutputNoteActivity – активность для работы с конкретной заметкой. Отображает заголовок заметки, дату создания и текст заметки (из базы данных SQLite с помощью TextView). Предоставляет доступ к редактированию заметки без дополнительных действий. Содержит кнопки сохранения изменений и удаления заметки (объекты Button).

**2.4. Структура файлов**

Для работы программы необходима база данных.

Заметки хранятся в базе данных. При создании заметки вся информация записывается в строку базы данных, при демонстрации заметки нужные данные считываются из базы, при редактировании новая информация записывается в базе данных, соответственно, при удалении файла, заметки, сделанные пользователем, удаляются из базы.

# 3 Реализация

3.1 Выбор средств реализации Desktop-приложения

Данное ПО реализовано на языке С# с использованием среды разработки программного обеспечения Microsoft Visual Studio 2017. Язык С# - статически типизированный объектно-ориентированный язык программирования.

Программа соответствует всем парадигмам объектно-ориентированного программирования.

Для полноценного функционирования программы требуется:

* ОС Windows XP или выше;
* Оперативная память 5МБ или выше;
* Жесткий диск: 1МБ для хранения программы; дополнительное место для хранения файлов, используемых при работе с базой данных;
* Наличие мыши (или её аналоги), клавиатуры, монитора.

3.2 Структура программы

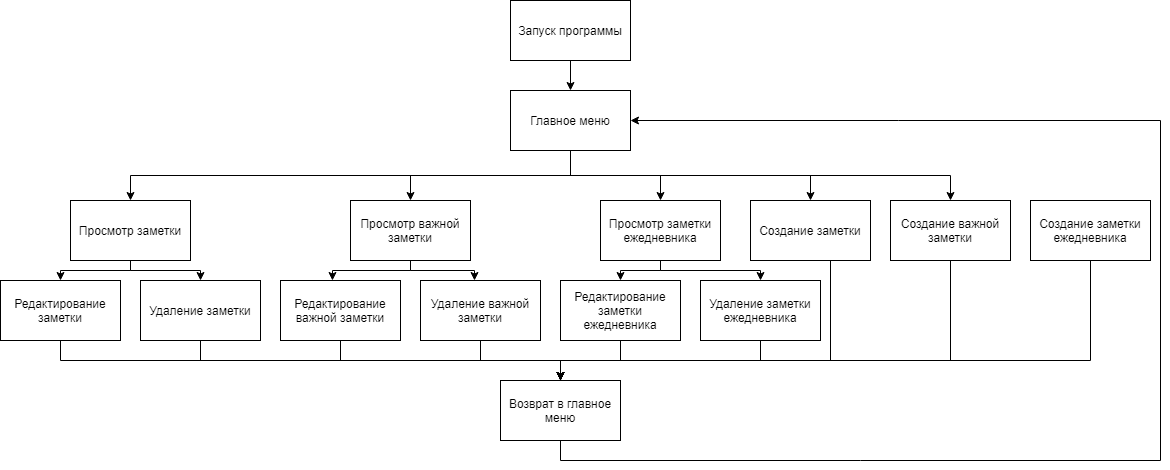


Рисунок 1-Структура программы

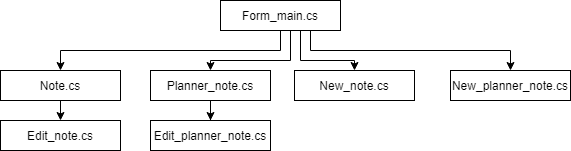


Рисунок 2-Диаграмма UML

3.3 Описание реализации ПО

Главная форма (Form\_main.cs) содержит меню с такими пунктами, как «Заметки», «Важное» и «Ежедневник» и текст приветствия. При нажатии на какой-либо пункт меню текст приветствия скрывается (здесь и далее с помощью свойства Hide()).

При нажатии на пункты меню «Заметки» и «Важное» появляются (здесь и далее свойство Show()) заголовки и даты создания заметок/важных заметок (из файла note.txt/important\_note.txt). Кроме этого, появляются кнопки «Создать заметку»/«Создать важную заметку». Для фиксирования нажатия кнопок используется обработчик событий Click.

При нажатии на пункт меню «Ежедневник» появляются заголовки, даты создания и сроки выполнения заметок ежедневника (из файла planner.txt). Также появляется кнопка «Создать заметку для ежедневника».

При нажатии на заметку откроется форма для просмотра текста заметки. Также эта форма содержит кнопки «Редактировать заметку» и «Удалить заметку»

При нажатии на кнопку «Редактировать заметку» откроется форма для редактирования заметки. Также эта форма содержит кнопку «Сохранить заметку», при нажатии на которую произойдет изменение заметки в файле note.txt/important\_note.txt/planner.txt с помощью вспомогательного файла new\_note.txt. После появится сообщение(здесь и далее MessageBox::Show()) с информацией об успешном изменении заметки. При закрытии данной формы произойдет возврат на главное форму программы.

При нажатии на кнопку «Удалить заметку» появится сообщение с просьбой к пользователю подтвердить свой выбор. Если пользователь нажмет «Да», то произойдет удаление заметки из файла note.txt/important\_note.txt/planner.txt с помощью вспомогательного файла new\_note.txt. После появится сообщение(здесь и далее MessageBox::Show()) с информацией об успешном удалении заметки. Затем произойдет возврат на главное форму программы.

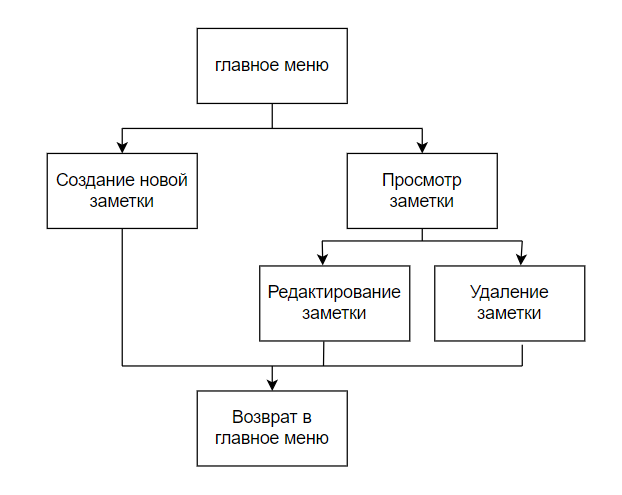
3.4 Выбор средств реализации Android-приложения

Данное ПО реализовано на языке Java с использованием среды разработки программного обеспечения Android Studio 4.0.1. Язык Java -  строго типизированный объектно-ориентированный язык программирования общего назначения.

Программа соответствует всем парадигмам объектно-ориентированного программирования.

Для полноценного функционирования программы требуется:

* ОС Android 7.0 или выше;
* Оперативная память 5 МБ или выше;
* Встроенная память 6 МБ для хранения программы; дополнительное место для хранения файлов, используемых при работе с базой данных;

3.5 Структура программы

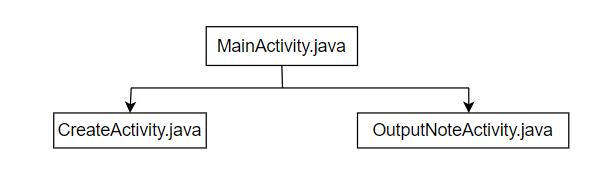
Рисунок 1-Структура программы

Рисунок 2-Диаграмма UML

3.6 Описание реализации ПО

Главная активность (MainActivity.java) содержит такие поля как: список имеющихся заметок, с отображением названия и даты создания заметки, и кнопка добавления новой заметки.

При нажатии на какую-либо заметку из списка открывается активность просмотра заметки (OutputNoteActivity.java) c помощью намерения Intent и свойства StartActivity() здесь и дальше. Заголовок, дата и текст заметки считываются из базы данных и отображаются на экране. Здесь же есть возможность редактирования заметки. При этом редактирование названия недоступно, дата меняется на системную дату устройства в текущее время, а текст заметки редактируется при нажатии на него.

При нажатии кнопки «сохранить» изменения вносятся в базу и активность закрывается (здесь и дальше finich()). Для фиксирования нажатия кнопок используется обработчик событий onClick. Для фиксирования нажатия конкретной заметки используется обработчик событий onItemClick.

При нажатии кнопки «удалить» заметка удаляется из базы и активность закрывается.

При нажатии кнопки добавления новой заметки «+» на главной активности открывается активность добавления новой заметки (CreateActivity.java). Пользователю предлагается ввести название и текст заметки. Дата устанавливается системно и выводится на экран.

При нажатии кнопки «сохранить» данные вносятся в базу данных и активность закрывается.

Для того, чтобы новая заметка отобразилась в главной активности, необходимо вновь зайти в приложение. Для того, чтобы удаленная заметка исчезла из списка, необходимо поступить также.

# Заключение

Был создан программный продукт, позволяющий эффективно работать с заметками. Реализованы создание заметок, их хранение, а также редактирование и удаление в различных режимах : обычные заметки, важные заметки и ежедневник.

В процессе выполнения задания были подробно изучены методы работы с файлами на языке С#, визуальными компонентами CLI, документацией MSDN.

Преимущества приложения:

* Интуитивно понятный интерфейс;
* Возможность оперативного редактирования/ добавления/ удаления записей;

Недочёты приложения:

* Для работы необходимы специально отформатированные файлы;

Возможны дальнейшие усовершенствования программного продукта с течением времени:

* Добавление push-уведомлений с напоминаниями о сроке выполнения (для заметок ежедневника);

# Список использованной литературы

1. Сайт Программирование [Электронный ресурс]. Режим доступа: http://www.prog-cpp.ru/, свободный.
2. Сайт CyberForum [Электронный ресурс]. Режим доступа: http://www.cyberforum.ru/, свободный.
3. “MSDN” – информационный сервис для разработчиков [Электронный ресурс] – Режим доступа: http://msdn.microsoft.com/ru-ru/ms348103/library, свободный.

# Приложение А

**Исходный код программы Desktop**

Form\_main.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Note

{

public partial class Form\_main : Form

{

public static int number = 0;

public static int count\_notes = 0;

public static int count\_important\_notes = 0;

public static int count\_planner = 0;

public static string file\_name;

public static string help\_file\_name;

public Form\_main()

{

InitializeComponent();

label\_notes.Hide();

button\_create\_note.Hide();

button\_create\_important\_note.Hide();

button\_create\_planner\_note.Hide();

}

private void Form\_main\_Load(object sender, EventArgs e)

{

label\_about.Text += Environment.NewLine;

label\_about.Text += Environment.NewLine;

label\_about.Text += "В разделе \"Заметки\" Вы можете создавать обычные заметки,а также редактировать и удалять их" + Environment.NewLine;

label\_about.Text += Environment.NewLine;

label\_about.Text += "Важные заметки Вы можете создавать в отдельном разделе \"Важное\" " + Environment.NewLine;

label\_about.Text += Environment.NewLine;

label\_about.Text += "Если Вам нужно создать заметку,для которой предусмотрен срок выполнения,воспользуйтесь разделом \"Ежедневник\" " + Environment.NewLine;

}

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

{

this.Size= new Size(816, 489);

file\_name = "note.txt";

foreach (Control c in this.Controls)

{

if (c is RichTextBox)

{

c.Hide();

}

}

file\_name = "note.txt";

count\_notes = 0;

StreamReader f = new StreamReader(file\_name, Encoding.UTF8);

while (!f.EndOfStream)

{

string line = f.ReadLine();

if (line == "---###---")

++count\_notes;

}

f.Close();

label\_hello.Hide();

label\_about.Hide();

label\_notes.Text = "Ваши заметки:";

label\_notes.Show();

button\_create\_note.Show();

button\_create\_important\_note.Hide();

button\_create\_planner\_note.Hide();

if (count\_notes == 0)

{

MessageBox.Show("У Вас нет созданных заметок");

}

else

{

StreamReader file = new StreamReader(file\_name, Encoding.UTF8);

string line;

RichTextBox[] tb = new RichTextBox[count\_notes];

for (int i = 0; i < tb.Length; ++i)

{

tb[i] = new RichTextBox();

tb[i].Location = new Point(190, i \* 50 + 70);

tb[i].Width = 400;

tb[i].Height = 50;

tb[i].Font = new Font("Georgia", this.Height / 40);

tb[i].Tag = i;

Controls.Add(tb[i]);

tb[i].Click += new System.EventHandler(this.textBox\_Click);

line = file.ReadLine();

tb[i].Text = line.ToString() + Environment.NewLine;

line = file.ReadLine();

tb[i].Text += "Дата создания заметки: "+line.ToString() + Environment.NewLine;

while ((line = file.ReadLine()) != "---###---")

{

}

}

file.Close();

}

}

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

{

this.Hide();

number = Convert.ToInt32((sender as RichTextBox).Tag);

Note note = new Note();

note.Show();

}

private void textBox\_planner\_note\_Click(object sender, EventArgs e)

{

this.Hide();

number = Convert.ToInt32((sender as RichTextBox).Tag);

Planner\_note planner\_note = new Planner\_note();

planner\_note.Show();

}

private void button\_create\_note\_Click(object sender, EventArgs e)

{

file\_name = "note.txt";

this.Hide();

New\_Note new\_note = new New\_Note();

new\_note.Show();

}

private void button\_create\_important\_note\_Click(object sender, EventArgs e)

{

file\_name = "important\_note.txt";

this.Hide();

New\_Note new\_note = new New\_Note();

new\_note.Show();

}

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

{

this.Size = new Size(816, 489);

file\_name = "important\_note.txt";

foreach (Control c in this.Controls)

{

if(c is RichTextBox)

{

c.Hide();

}

}

label\_hello.Hide();

label\_about.Hide();

label\_notes.Text = "Ваши важные заметки:";

label\_notes.Show();

button\_create\_note.Hide();

button\_create\_important\_note.Show();

button\_create\_planner\_note.Hide();

count\_important\_notes = 0;

StreamReader f = new StreamReader("important\_note.txt", Encoding.UTF8);

while (!f.EndOfStream)

{

string line = f.ReadLine();

if (line == "---###---")

++count\_important\_notes;

}

f.Close();

if (count\_important\_notes == 0)

{

MessageBox.Show("У Вас нет созданных важных заметок");

}

else

{

StreamReader file = new StreamReader("important\_note.txt", Encoding.UTF8);

string line;

RichTextBox[] tb = new RichTextBox[count\_important\_notes];

for (int i = 0; i < tb.Length; ++i)

{

tb[i] = new RichTextBox();

tb[i].Location = new Point(190, i \* 50 + 70);

tb[i].Width = 400;

tb[i].Height = 50;

tb[i].Font = new Font("Georgia", this.Height / 40);

tb[i].Tag = i;

Controls.Add(tb[i]);

tb[i].Click += new System.EventHandler(this.textBox\_Click);

line = file.ReadLine();

tb[i].Text = line.ToString() + Environment.NewLine;

line = file.ReadLine();

tb[i].Text += "Дата создания заметки: "+line.ToString() + Environment.NewLine;

while ((line = file.ReadLine()) != "---###---")

{

}

}

file.Close();

}

}

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

{

this.Size = new Size(816, 489);

file\_name = "planner.txt";

foreach (Control c in this.Controls)

{

if (c is RichTextBox)

{

c.Hide();

}

}

label\_hello.Hide();

label\_about.Hide();

label\_notes.Text = "Ежедневник:";

label\_notes.Show();

button\_create\_note.Hide();

button\_create\_important\_note.Hide();

button\_create\_planner\_note.Show();

count\_planner = 0;

StreamReader f = new StreamReader("planner.txt", Encoding.UTF8);

while (!f.EndOfStream)

{

string line = f.ReadLine();

if (line == "---###---")

++count\_planner;

}

f.Close();

if (count\_planner == 0)

{

MessageBox.Show("У Вас нет заметок в ежедневнике");

}

else

{

StreamReader file = new StreamReader("planner.txt", Encoding.UTF8);

string line;

RichTextBox[] tb = new RichTextBox[count\_planner];

for (int i = 0; i < tb.Length; ++i)

{

tb[i] = new RichTextBox();

tb[i].Location = new Point(190, i \* 50 + 70);

tb[i].Width = 400;

tb[i].Height = 60;

tb[i].Font = new Font("Georgia", this.Height / 40);

tb[i].Tag = i;

Controls.Add(tb[i]);

tb[i].Click += new EventHandler(this.textBox\_planner\_note\_Click);

line = file.ReadLine();

tb[i].Text = line.ToString() + Environment.NewLine;

line = file.ReadLine();

tb[i].Text += "Дата создания заметки: "+line.ToString() + Environment.NewLine;

line = file.ReadLine();

tb[i].Text += "Срок выполнения: "+line.ToString() + Environment.NewLine;

while ((line = file.ReadLine()) != "---###---")

{

}

}

file.Close();

}

}

private void button\_create\_planner\_note\_Click(object sender, EventArgs e)

{

this.Hide();

New\_planner\_note new\_planner\_note = new New\_planner\_note();

new\_planner\_note.Show();

}

}

}

Note.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Note

{

public partial class Note : Form

{

public Note()

{

InitializeComponent();

}

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

{

textBox\_title.Font = new Font("Georgia", this.Height / 40);

textBox\_date.Font = new Font("Georgia", this.Height / 40);

richTextBox\_text.Font = new Font("Georgia", this.Height / 40);

StreamReader fi = new StreamReader(Form\_main.file\_name, true);

string line;

int in\_number = 0;

for (in\_number = 0; in\_number <= Form\_main.number; ++in\_number)

{

line = fi.ReadLine();

textBox\_title.Text = line.ToString() + System.Environment.NewLine;

line = fi.ReadLine();

textBox\_date.Text = line.ToString() + System.Environment.NewLine;

while ((line = fi.ReadLine()) != "---###---")

{

if (in\_number == Form\_main.number)

richTextBox\_text.Text += line.ToString() + System.Environment.NewLine;

}

}

fi.Close();

}

private void button\_edit\_Click(object sender, EventArgs e)

{

Edit\_note edit\_note = new Edit\_note();

edit\_note.ShowDialog();

this.Hide();

}

private void button\_delete\_Click(object sender, EventArgs e)

{

DialogResult dialog = MessageBox.Show(this, "Вы действительно хотите удалить заметку?", "Предупреждение", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (dialog == DialogResult.Yes)

{

int count = Form\_main.number;

StreamReader file = new StreamReader(Form\_main.file\_name);

StreamWriter new\_file = new StreamWriter("new\_note.txt", false);

string line;

for (int i = 0; i < count; ++i)

{

while ((line = file.ReadLine()) != "---###---")

{

line = file.ReadLine();

new\_file.WriteLine(line);

}

}

while ((line = file.ReadLine()) != "---###---")

{

}

while (!file.EndOfStream)

{

line = file.ReadLine();

new\_file.WriteLine(line);

}

new\_file.Close();

file.Close();

StreamWriter file\_edit = new StreamWriter(Form\_main.file\_name, false);

StreamReader new\_file\_edit = new StreamReader("new\_note.txt");

while (!new\_file\_edit.EndOfStream)

{

line = new\_file\_edit.ReadLine();

file\_edit.WriteLine(line);

}

file\_edit.Close();

new\_file\_edit.Close();

--Form\_main.count\_notes;

}

MessageBox.Show(this, "Заметка успешно удалена", "Удаление заметки", MessageBoxButtons.OK, MessageBoxIcon.None);

this.Hide();

Form\_main form = new Form\_main();

form.Show();

}

private void Form\_main\_Closed(object sender, FormClosedEventArgs e)

{

Form\_main form = new Form\_main();

form.Show();

}

}

}

New\_Note.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Note

{

public partial class Note : Form

{

public Note()

{

InitializeComponent();

}

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

{

textBox\_title.Font = new Font("Georgia", this.Height / 40);

textBox\_date.Font = new Font("Georgia", this.Height / 40);

richTextBox\_text.Font = new Font("Georgia", this.Height / 40);

StreamReader fi = new StreamReader(Form\_main.file\_name, true);

string line;

int in\_number = 0;

for (in\_number = 0; in\_number <= Form\_main.number; ++in\_number)

{

line = fi.ReadLine();

textBox\_title.Text = line.ToString() + System.Environment.NewLine;

line = fi.ReadLine();

textBox\_date.Text = line.ToString() + System.Environment.NewLine;

while ((line = fi.ReadLine()) != "---###---")

{

if (in\_number == Form\_main.number)

richTextBox\_text.Text += line.ToString() + System.Environment.NewLine;

}

}

fi.Close();

}

private void button\_edit\_Click(object sender, EventArgs e)

{

Edit\_note edit\_note = new Edit\_note();

edit\_note.ShowDialog();

this.Hide();

}

private void button\_delete\_Click(object sender, EventArgs e)

{

DialogResult dialog = MessageBox.Show(this, "Вы действительно хотите удалить заметку?", "Предупреждение", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (dialog == DialogResult.Yes)

{

int count = Form\_main.number;

StreamReader file = new StreamReader(Form\_main.file\_name);

StreamWriter new\_file = new StreamWriter("new\_note.txt", false);

string line;

for (int i = 0; i < count; ++i)

{

while ((line = file.ReadLine()) != "---###---")

{

line = file.ReadLine();

new\_file.WriteLine(line);

}

}

while ((line = file.ReadLine()) != "---###---")

{

}

while (!file.EndOfStream)

{

line = file.ReadLine();

new\_file.WriteLine(line);

}

new\_file.Close();

file.Close();

StreamWriter file\_edit = new StreamWriter(Form\_main.file\_name, false);

StreamReader new\_file\_edit = new StreamReader("new\_note.txt");

while (!new\_file\_edit.EndOfStream)

{

line = new\_file\_edit.ReadLine();

file\_edit.WriteLine(line);

}

file\_edit.Close();

new\_file\_edit.Close();

--Form\_main.count\_notes;

}

MessageBox.Show(this, "Заметка успешно удалена", "Удаление заметки", MessageBoxButtons.OK, MessageBoxIcon.None);

this.Hide();

Form\_main form = new Form\_main();

form.Show();

}

private void Form\_main\_Closed(object sender, FormClosedEventArgs e)

{

Form\_main form = new Form\_main();

form.Show();

}

}

}

Edit\_note.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Note

{

public partial class Edit\_note : Form

{

public Edit\_note()

{

InitializeComponent();

}

private void Edit\_note\_Load(object sender, EventArgs e)

{

textBox\_title.Font = new Font("Georgia", this.Height / 40);

dateTimePicker\_date.Font = new Font("Georgia", this.Height / 40);

richTextBox\_text.Font = new Font("Georgia", this.Height / 40);

StreamReader fi = new StreamReader(Form\_main.file\_name, true);

string line;

int in\_number = 0;

for (in\_number = 0; in\_number <= Form\_main.number; ++in\_number)

{

line = fi.ReadLine();

textBox\_title.Text = line.ToString() + System.Environment.NewLine;

line = fi.ReadLine();

dateTimePicker\_date.Text = line.ToString() + System.Environment.NewLine;

while ((line = fi.ReadLine()) != "---###---")

{

if (in\_number == Form\_main.number)

richTextBox\_text.Text += line.ToString() + System.Environment.NewLine;

}

}

fi.Close();

}

private void button\_save\_edit\_note\_Click(object sender, EventArgs e)

{

int count = Form\_main.number;

StreamReader file = new StreamReader(Form\_main.file\_name);

StreamWriter new\_file = new StreamWriter("new\_note.txt", false);

string line;

for (int i = 0; i < count; ++i)

{

while ((line = file.ReadLine()) != "---###---")

{

new\_file.WriteLine(line);

}

new\_file.WriteLine(line);

}

new\_file.WriteLine((textBox\_title.Text).Replace(Environment.NewLine, " "));

new\_file.WriteLine((dateTimePicker\_date.Text).Replace(Environment.NewLine, " "));

new\_file.WriteLine((richTextBox\_text.Text).Replace(Environment.NewLine, " "));

new\_file.WriteLine("---###---");

while ((line = file.ReadLine()) != "---###---")

{

}

while (!file.EndOfStream)

{

line = file.ReadLine();

new\_file.WriteLine(line);

}

new\_file.Close();

file.Close();

StreamWriter file\_edit = new StreamWriter(Form\_main.file\_name, false);

StreamReader new\_file\_edit = new StreamReader("new\_note.txt");

while (!new\_file\_edit.EndOfStream)

{

line = new\_file\_edit.ReadLine();

file\_edit.WriteLine(line);

}

file\_edit.Close();

new\_file\_edit.Close();

File.Delete("new\_note.txt");

MessageBox.Show(this, "Заметка успешно отредактирована", "Редактирование заметки", MessageBoxButtons.OK, MessageBoxIcon.None);

this.Hide();

Form\_main form = new Form\_main();

form.Show();

}

private void Edit\_note\_Closed(object sender, FormClosedEventArgs e)

{

Form\_main form = new Form\_main();

form.Show();

}

}

}

Planner\_note.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Note

{

public partial class Planner\_note : Form

{

public Planner\_note()

{

InitializeComponent();

}

private void Planner\_note\_Load(object sender, EventArgs e)

{

textBox\_title.Font = new Font("Georgia", this.Height / 40);

textBox\_date.Font = new Font("Georgia", this.Height / 40);

textBox\_deadline.Font = new Font("Georgia", this.Height / 40);

richTextBox\_text.Font = new Font("Georgia", this.Height / 40);

StreamReader fi = new StreamReader(Form\_main.file\_name, true);

string line;

int in\_number = 0;

for (in\_number = 0; in\_number <= Form\_main.number; ++in\_number)

{

line = fi.ReadLine();

textBox\_title.Text = line.ToString() + System.Environment.NewLine;

line = fi.ReadLine();

textBox\_date.Text = line.ToString() + System.Environment.NewLine;

line = fi.ReadLine();

textBox\_deadline.Text = line.ToString() + System.Environment.NewLine;

while ((line = fi.ReadLine()) != "---###---")

{

if (in\_number == Form\_main.number)

richTextBox\_text.Text += line.ToString() + System.Environment.NewLine;

}

}

fi.Close();

}

private void Planner\_note\_closed(object sender, FormClosedEventArgs e)

{

Form\_main form = new Form\_main();

form.Show();

}

private void button\_edit\_Click(object sender, EventArgs e)

{

this.Hide();

Edit\_planner\_note edit\_planner\_note = new Edit\_planner\_note();

edit\_planner\_note.Show();

}

private void button\_delete\_Click(object sender, EventArgs e)

{

DialogResult dialog = MessageBox.Show(this, "Вы действительно хотите удалить заметку из ежедневника?", "Предупреждение", MessageBoxButtons.YesNo, MessageBoxIcon.Warning);

if (dialog == DialogResult.Yes)

{

int count = Form\_main.number;

StreamReader file = new StreamReader(Form\_main.file\_name);

StreamWriter new\_file = new StreamWriter("new\_note.txt", false);

string line;

for (int i = 0; i < count; ++i)

{

while ((line = file.ReadLine()) != "---###---")

{

new\_file.WriteLine(line);

}

}

while ((line = file.ReadLine()) != "---###---")

{

}

while (!file.EndOfStream)

{

line = file.ReadLine();

new\_file.WriteLine(line);

}

new\_file.Close();

file.Close();

StreamWriter file\_edit = new StreamWriter(Form\_main.file\_name, false);

StreamReader new\_file\_edit = new StreamReader("new\_note.txt");

while (!new\_file\_edit.EndOfStream)

{

line = new\_file\_edit.ReadLine();

file\_edit.WriteLine(line);

}

file\_edit.Close();

new\_file\_edit.Close();

--Form\_main.count\_notes;

}

MessageBox.Show(this, "Заметка успешно удалена", "Удаление заметки", MessageBoxButtons.OK, MessageBoxIcon.None);

this.Hide();

Form\_main form = new Form\_main();

form.Show();

}

}

}

New\_planner\_note.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Note

{

public partial class New\_planner\_note : Form

{

public New\_planner\_note()

{

InitializeComponent();

}

private void button\_save\_planner\_Click(object sender, EventArgs e)

{

++Form\_main.count\_planner;

StreamWriter sw = new StreamWriter("new\_note.txt", false, Encoding.UTF8);

sw.WriteLine(textBox\_title.Text);

sw.WriteLine(textBox\_date.Text);

sw.WriteLine(dateTimePicker\_deadline.Text);

sw.WriteLine(richTextBox\_text.Text);

sw.WriteLine("---###---");

textBox\_title.Text = "";

richTextBox\_text.Text = "";

StreamReader file = new StreamReader(Form\_main.file\_name, Encoding.UTF8);

string line;

while (!file.EndOfStream)

{

line = file.ReadLine();

sw.WriteLine(line);

}

sw.Close();

file.Close();

StreamWriter new\_file = new StreamWriter(Form\_main.file\_name, false, Encoding.UTF8);

StreamReader new\_edit\_file = new StreamReader("new\_note.txt", Encoding.UTF8);

while (!new\_edit\_file.EndOfStream)

{

line = new\_edit\_file.ReadLine();

new\_file.WriteLine(line);

}

new\_file.Close();

new\_edit\_file.Close();

File.Delete("new\_note.txt");

this.Hide();

Form\_main form = new Form\_main();

form.Show();

}

private void New\_planner\_note\_Load(object sender, EventArgs e)

{

textBox\_date.Text = DateTime.Now.ToLongDateString();

textBox\_title.Font = new Font("Georgia", this.Height / 40);

textBox\_date.Font = new Font("Georgia", this.Height / 40);

dateTimePicker\_deadline.Font = new Font("Georgia", this.Height / 40);

richTextBox\_text.Font = new Font("Georgia", this.Height / 40);

}

private void New\_planner\_note\_close(object sender, FormClosedEventArgs e)

{

Form\_main form = new Form\_main();

form.Show();

}

}

}

Edit\_planner\_note.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

namespace Note

{

public partial class Edit\_planner\_note : Form

{

public Edit\_planner\_note()

{

InitializeComponent();

}

private void Edit\_planner\_note\_Load(object sender, EventArgs e)

{

textBox\_title.Font = new Font("Georgia", this.Height / 40);

dateTimePicker\_date.Font = new Font("Georgia", this.Height / 40);

dateTimePicker\_deadline.Font = new Font("Georgia", this.Height / 40);

richTextBox\_text.Font = new Font("Georgia", this.Height / 40);

StreamReader fi = new StreamReader(Form\_main.file\_name, true);

string line;

int in\_number = 0;

for (in\_number = 0; in\_number <= Form\_main.number; ++in\_number)

{

line = fi.ReadLine();

textBox\_title.Text = line.ToString() + Environment.NewLine;

line = fi.ReadLine();

dateTimePicker\_date.Text = line.ToString() + Environment.NewLine;

line = fi.ReadLine();

dateTimePicker\_deadline.Text = line.ToString() + Environment.NewLine;

while ((line = fi.ReadLine()) != "---###---")

{

if (in\_number == Form\_main.number)

richTextBox\_text.Text += line.ToString() + Environment.NewLine;

}

}

fi.Close();

}

private void button\_save\_edit\_planner\_note\_Click(object sender, EventArgs e)

{

int count = Form\_main.number;

StreamReader file = new StreamReader(Form\_main.file\_name);

StreamWriter new\_file = new StreamWriter("new\_note.txt", false);

string line;

for (int i = 0; i < count; ++i)

{

while ((line = file.ReadLine()) != "---###---")

{

new\_file.WriteLine(line);

}

new\_file.WriteLine(line);

}

new\_file.WriteLine((textBox\_title.Text).Replace(Environment.NewLine, " "));

new\_file.WriteLine((dateTimePicker\_date.Text).Replace(Environment.NewLine, " "));

new\_file.WriteLine((dateTimePicker\_deadline.Text).Replace(Environment.NewLine, " "));

new\_file.WriteLine((richTextBox\_text.Text).Replace(Environment.NewLine, " "));

new\_file.WriteLine("---###---");

while ((line = file.ReadLine()) != "---###---")

{

}

while (!file.EndOfStream)

{

line = file.ReadLine();

new\_file.WriteLine(line);

}

new\_file.Close();

file.Close();

StreamWriter file\_edit = new StreamWriter(Form\_main.file\_name, false);

StreamReader new\_file\_edit = new StreamReader("new\_note.txt");

while (!new\_file\_edit.EndOfStream)

{

line = new\_file\_edit.ReadLine();

file\_edit.WriteLine(line);

}

file\_edit.Close();

new\_file\_edit.Close();

File.Delete("new\_note.txt");

MessageBox.Show(this, "Заметка успешно отредактирована", "Редактирование заметки", MessageBoxButtons.OK, MessageBoxIcon.None);

this.Hide();

Form\_main form = new Form\_main();

form.Show();

}

private void Edit\_planner\_note\_close(object sender, FormClosedEventArgs e)

{

Form\_main form = new Form\_main();

form.Show();

}

}

}

**Исходный код программы Android**

MainActivity.java

package com.example.zametki.Activities;  
  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import androidx.recyclerview.widget.RecyclerView;  
  
import android.content.Intent;  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
  
  
import com.example.zametki.Adapters.RecyclerViewAdapter;  
import com.example.zametki.Models.Note;  
import com.example.zametki.R;  
import com.example.zametki.Generic.DB;  
import com.example.zametki.Generic.Registry;  
  
import java.util.ArrayList;  
import java.util.List;  
  
public class MainActivity extends AppCompatActivity implements RecyclerViewAdapter.OnItemListener{  
  
 Button create;  
 RecyclerView rv;  
  
 List<Note> notes = new ArrayList<>();  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 Registry.*baseContext* = getBaseContext();  
 try {  
 DB.*init*();  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 try {  
 notes = DB.*getAllNotes*();  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 rv = findViewById(R.id.*recycler*);  
 RecyclerViewAdapter rva = new RecyclerViewAdapter(this, getLayoutInflater(), notes,  
 this);  
 rv.setAdapter(rva);  
 rv.setLayoutManager(new LinearLayoutManager(this));  
  
 create = findViewById(R.id.*button\_create*);  
 create.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Intent cr;  
 cr = new Intent(MainActivity.this, CreateActivity.class);  
 startActivity(cr);  
 }  
  
 });  
 }  
  
 @Override  
 public void onItemClick(int position) throws IllegalAccessException, NoSuchFieldException, InterruptedException {  
 Intent i = new Intent(this, OutputNoteActivity.class);  
 i.putExtra("id", notes.get(position).id);  
 startActivity(i);  
 }  
  
}

CreateActivity.java

package com.example.zametki.Activities;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.example.zametki.Models.Note;

import com.example.zametki.R;

import com.example.zametki.Generic.DB;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.util.Date;

import java.util.Locale;

public class CreateActivity extends AppCompatActivity {

TextView data;

EditText name, descr;

Button bSave;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_create);

//Сделать это при нажатии кнопки сохранить

data = findViewById(R.id.textView\_data);

name = findViewById(R.id.editText\_name);

descr = findViewById(R.id.editText\_note);

bSave = findViewById(R.id.button\_save);

Date currentDate = new Date();

DateFormat dateFormat = new SimpleDateFormat("dd.MM.yyyy", Locale.getDefault());

final String dateText = dateFormat.format(currentDate);

data.setText(dateText);

bSave.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

Note newNote = new Note();

newNote.date = dateText;

EditText txtDescription = (EditText) descr;

newNote.description = txtDescription.getText().toString();

EditText txtName = (EditText) name;

newNote.title = txtName.getText().toString();

try {

DB.addNote(newNote);

} catch (InterruptedException e) {

e.printStackTrace();

}

finish();

}

});

}

}

OutputNoteActivity.java

package com.example.zametki.Activities;  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.Button;  
import android.widget.EditText;  
import android.widget.TextView;  
import android.widget.Toast;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
import com.example.zametki.Models.Note;  
import com.example.zametki.R;  
import com.example.zametki.Generic.DB;  
  
import java.text.DateFormat;  
import java.text.SimpleDateFormat;  
import java.util.ArrayList;  
import java.util.Date;  
import java.util.List;  
import java.util.Locale;  
  
public class OutputNoteActivity extends AppCompatActivity {  
 TextView name, data, descr;  
 Button saveThis, deleteThis;  
 Note note;  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_output\_note*);  
 name = findViewById(R.id.*textView\_name*);  
 data = findViewById(R.id.*textView\_data*);  
 descr = findViewById(R.id.*textView\_note*);  
 saveThis = findViewById(R.id.*button\_save*);  
 deleteThis = findViewById(R.id.*button\_delete*);  
  
 Bundle arguments = getIntent().getExtras();  
 if (null != arguments) {  
 int id = arguments.getInt("id");  
 try {  
 note = DB.*getNote*(id);  
 name.setText(note.title);  
 data.setText(note.date);  
 descr.setText(note.description);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 deleteThis.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
  
 try {  
 DB.*deleteNote*(note.id);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 finish();  
 }  
  
 });  
 saveThis.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View v) {  
 Date currentDate = new Date();  
 DateFormat dateFormat = new SimpleDateFormat("dd.MM.yyyy", Locale.*getDefault*());  
 final String dateText = dateFormat.format(currentDate);  
 data.setText(dateText);  
  
 Note editNote = new Note();  
 editNote.date = dateText;  
 editNote.description = descr.getText().toString();  
 editNote.title = name.getText().toString();  
  
 try {  
 if (null != note) {  
 editNote.id = note.id;  
 DB.*updateNote*(editNote);  
 } else {  
 DB.*addNote*(editNote);  
 }  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 finish();  
 }  
 });  
 }  
}

RecyclerViewAdapter.java

package com.example.zametki.Adapters;  
  
import android.content.Context;  
import android.view.LayoutInflater;  
import android.view.View;  
import android.view.ViewGroup;  
import android.widget.TextView;  
  
import androidx.annotation.NonNull;  
import androidx.recyclerview.widget.RecyclerView;  
  
import com.example.zametki.Models.Note;  
import com.example.zametki.R;  
  
import java.util.List;  
  
public class RecyclerViewAdapter extends RecyclerView.Adapter<RecyclerViewAdapter.ViewHolder> {  
  
 private LayoutInflater inflater;  
 private List<Note> notes;  
 private Context context;  
 private RecyclerViewAdapter.OnItemListener mOnItemListener;  
  
 public RecyclerViewAdapter(Context context, LayoutInflater inflater, List<Note> accounts,  
 RecyclerViewAdapter.OnItemListener mOnItemListener) {  
 this.context = context;  
 this.inflater = inflater;  
 this.notes = accounts;  
 this.mOnItemListener = mOnItemListener;  
 }  
  
 @NonNull  
 @Override  
 public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {  
 View view = inflater.inflate(R.layout.*recycler\_view*, parent, false);  
 return new RecyclerViewAdapter.ViewHolder(view, mOnItemListener);  
 }  
  
 @Override  
 public void onBindViewHolder(@NonNull final RecyclerViewAdapter.ViewHolder holder, final int position) {  
 holder.date.setText(notes.get(position).date);  
 holder.title.setText(notes.get(position).title);  
 }  
  
 @Override  
 public int getItemCount() {  
 return notes.size();  
 }  
  
 class ViewHolder extends RecyclerView.ViewHolder implements View.OnClickListener{  
 final TextView title;  
 final TextView date;  
 RecyclerViewAdapter.OnItemListener onItemListener;  
  
 ViewHolder(View view, RecyclerViewAdapter.OnItemListener onItemListener) {  
 super(view);  
 title = view.findViewById(R.id.*textView\_n*);  
 date = view.findViewById(R.id.*textView\_d*);  
 this.onItemListener = onItemListener;  
 view.setOnClickListener(this);  
 }  
  
 @Override  
 public void onClick(View view) {  
 try {  
 mOnItemListener.onItemClick(getAdapterPosition());  
 } catch (IllegalAccessException | NoSuchFieldException | InterruptedException ignored) {  
  
 }  
 }  
 }  
  
 public interface OnItemListener {  
 void onItemClick(int position) throws IllegalAccessException, NoSuchFieldException, InterruptedException;  
 }  
}

Registry.java

package com.example.zametki.Generic;  
  
import android.content.Context;  
import android.database.sqlite.SQLiteDatabase;  
  
public class Registry {  
  
 public static Context *baseContext*;  
  
 public static SQLiteDatabase *DB*;  
}

DB.java

package com.example.zametki.Generic;  
  
import android.database.Cursor;  
  
import com.example.zametki.Models.Note;  
  
import java.util.ArrayList;  
import java.util.List;  
import java.util.concurrent.Semaphore;  
  
import static android.content.Context.*MODE\_PRIVATE*;  
  
public class DB {  
  
 private static final Semaphore *Connections* = new Semaphore(1, true);  
  
 public static void init() throws InterruptedException {  
 *Connections*.acquire();  
  
 Registry.*DB* = Registry.*baseContext*.openOrCreateDatabase(  
 "app.db", *MODE\_PRIVATE*, null);  
  
 Registry.*DB*.execSQL("CREATE TABLE IF NOT EXISTS notes (" +  
 "id INTEGER PRIMARY KEY AUTOINCREMENT, " +  
 "title TEXT, date TEXT, description TEXT)");  
  
 *Connections*.release();  
 }  
  
 public static void addNote(Note note) throws InterruptedException {  
  
 *Connections*.acquire();  
  
 String query = String.*format*(  
 "INSERT INTO notes (title, date, description) VALUES ('%s', '%s', '%s')",  
 note.title,  
 note.date,  
 note.description  
 );  
  
 Registry.*DB*.execSQL(query);  
  
 *Connections*.release();  
 }  
  
 public static List<Note> getAllNotes() throws InterruptedException {  
 *Connections*.acquire();  
 List<Note> notes = new ArrayList<>();  
  
 Cursor result = Registry.*DB*.rawQuery("SELECT \* FROM notes", null);  
 if (result.moveToFirst()) {  
 do {  
 Note note = new Note();  
 note.id = result.getInt(result.getColumnIndex("id"));  
 note.title = result.getString(result.getColumnIndex("title"));  
 note.date = result.getString(result.getColumnIndex("date"));  
 note.description = result.getString(result.getColumnIndex("description"));  
 notes.add(note);  
 } while (result.moveToNext());  
 }  
  
 *Connections*.release();  
 return notes;  
 }  
  
 public static Note getNote(int id) throws InterruptedException {  
 *Connections*.acquire();  
  
 String query = String.*format*("SELECT \* FROM notes WHERE id = '%d'", id);  
 Cursor result = Registry.*DB*.rawQuery(query, null);  
  
 result.moveToFirst();  
 Note note = new Note();  
 note.description = result.getString(result.getColumnIndex("description"));  
 note.title = result.getString(result.getColumnIndex("title"));  
 note.date = result.getString(result.getColumnIndex("date"));  
 note.id = result.getInt(result.getColumnIndex("id"));  
  
 *Connections*.release();  
 return note;  
 }  
  
 public static void updateNote(Note note) throws InterruptedException {  
 *Connections*.acquire();  
  
 String query = String.*format*(  
 "UPDATE notes SET title='%s', date='%s', description='%s' WHERE id='%d'",  
 note.title,  
 note.date,  
 note.description,  
 note.id  
 );  
  
 Registry.*DB*.execSQL(query);  
  
 *Connections*.release();  
 }  
  
 public static void deleteNote(int id) throws InterruptedException {  
 DB.*Connections*.acquire();  
 Registry.*DB*.execSQL(String.*format*("DELETE FROM notes WHERE id='%d'", id));  
 DB.*Connections*.release();  
 }  
}

Note.java

package com.example.zametki.Models;  
  
public class Note {  
  
 public int id;  
 public String title;  
 public String date;  
 public String description;  
  
}

activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".Activities.MainActivity">  
  
 <Button  
 android:id="@+id/button\_create"  
 style="@style/Widget.AppCompat.Button.Borderless.Colored"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:backgroundTint="#75EBFA"  
 android:text="+"  
 android:textAppearance="@style/TextAppearance.AppCompat.Display1"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/recycler"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 app:layout\_constraintTop\_toBottomOf="@+id/button\_create"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 />  
</androidx.constraintlayout.widget.ConstraintLayout>

activity\_create.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".Activities.CreateActivity">  
  
 <EditText  
 android:id="@+id/editText\_name"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="40dp"  
 android:hint="Введите название"  
 android:textAppearance="@style/TextAppearance.AppCompat.Large"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.0"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <TextView  
 android:id="@+id/textView\_data"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="дата"  
 android:textAppearance="@style/TextAppearance.AppCompat.Body1"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/editText\_name" />  
  
 <EditText  
 android:id="@+id/editText\_note"  
 android:layout\_width="match\_parent"  
 android:layout\_height="0dp"  
 android:hint="Введите заметку"  
 android:inputType="textMultiLine"  
 android:gravity="top"  
 android:textAppearance="@style/TextAppearance.AppCompat.Medium"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView\_data" />  
  
 <Button  
 android:id="@+id/button\_save"  
 android:layout\_width="120dp"  
 android:layout\_height="40dp"  
 android:text="Сохранить"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
</androidx.constraintlayout.widget.ConstraintLayout>

activity\_output\_note.xml

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 xmlns:tools="http://schemas.android.com/tools"  
 tools:context=".Activities.OutputNoteActivity">  
  
 <Button  
 android:id="@+id/button\_delete"  
 android:layout\_width="120dp"  
 android:layout\_height="40dp"  
 android:text="Удалить"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <EditText  
 android:id="@+id/textView\_note"  
 android:layout\_width="match\_parent"  
 android:layout\_height="0dp"  
 android:gravity="top"  
 android:textAppearance="@style/TextAppearance.AppCompat.Medium"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView\_data" />  
  
 <TextView  
 android:id="@+id/textView\_data"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Дата"  
 android:textAppearance="@style/TextAppearance.AppCompat.Body1"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textView\_name" />  
  
 <TextView  
 android:id="@+id/textView\_name"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="40dp"  
 android:text="Название"  
 android:textAppearance="@style/TextAppearance.AppCompat.Large"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="1.0"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <Button  
 android:id="@+id/button\_save"  
 android:layout\_width="120dp"  
 android:layout\_height="40dp"  
 android:text="Сохранить"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="1.0"  
 app:layout\_constraintTop\_toTopOf="parent" />  
</androidx.constraintlayout.widget.ConstraintLayout>

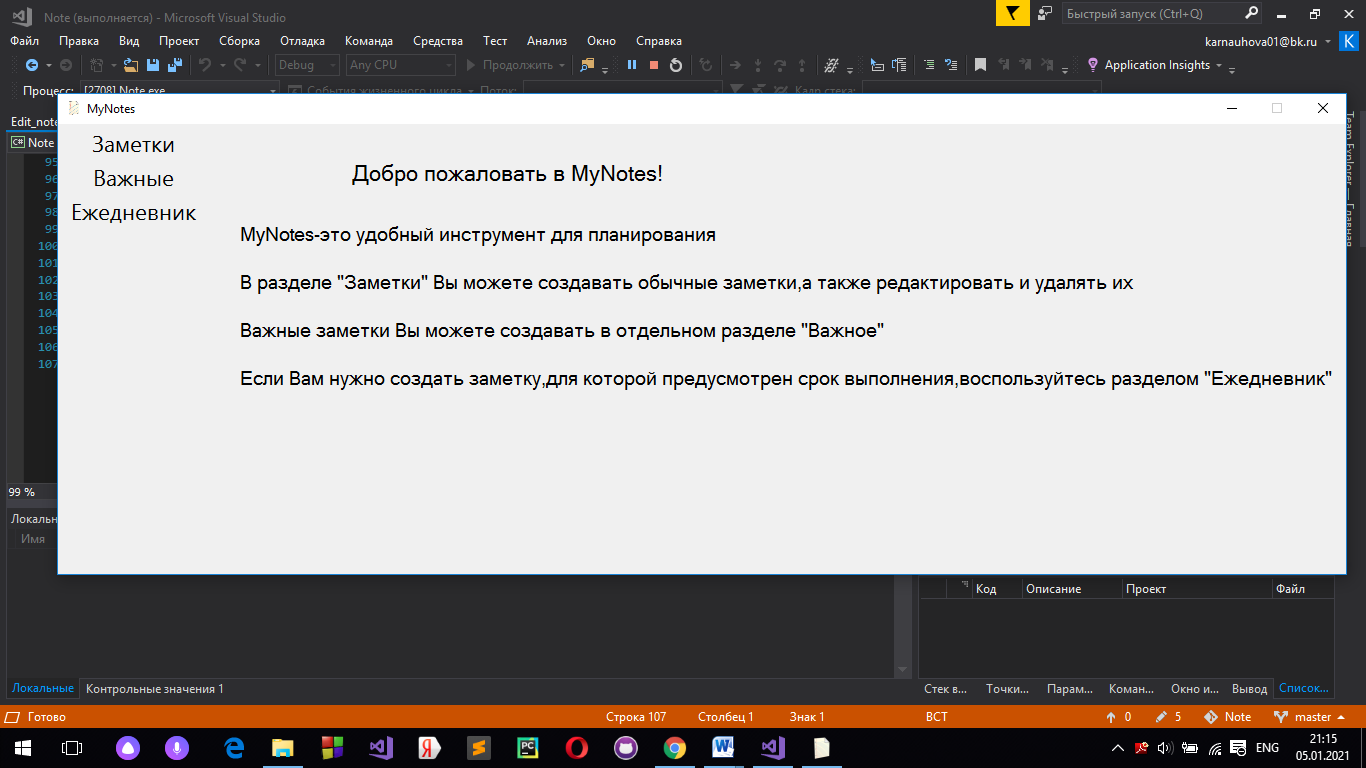
Recycler\_view.xml

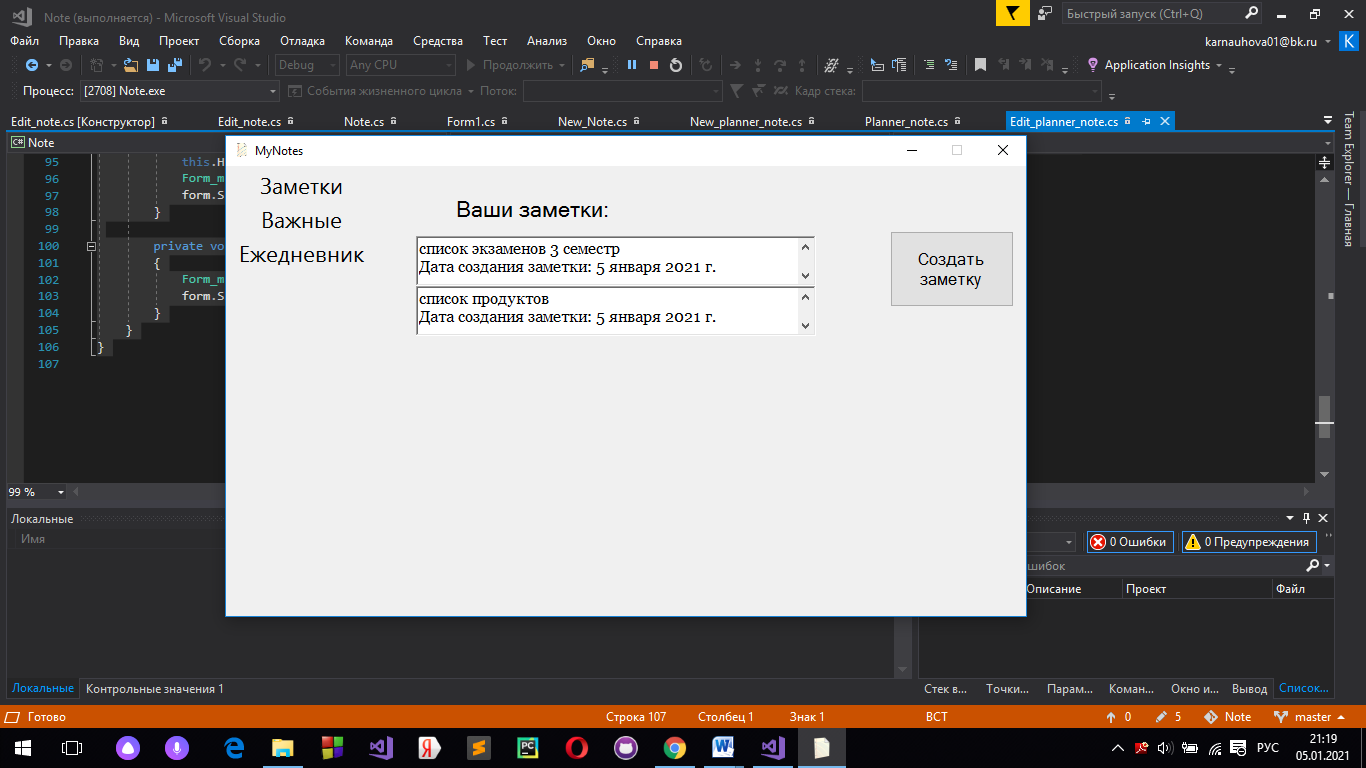
<?xml version="1.0" encoding="utf-8"?>  
<androidx.cardview.widget.CardView  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="10dp"  
 android:layout\_marginEnd="10dp"  
 android:layout\_marginTop="4dp"  
 xmlns:app="http://schemas.android.com/apk/res-auto">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="horizontal"  
 android:weightSum="1">  
  
 <TextView  
 android:id="@+id/textView\_n"  
 android:layout\_width="0px"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="0.75"  
 android:textAppearance="@style/TextAppearance.AppCompat.Large" />  
  
 <TextView  
 android:id="@+id/textView\_d"  
 android:layout\_width="0px"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="0.25" />  
 </LinearLayout>  
</androidx.cardview.widget.CardView>

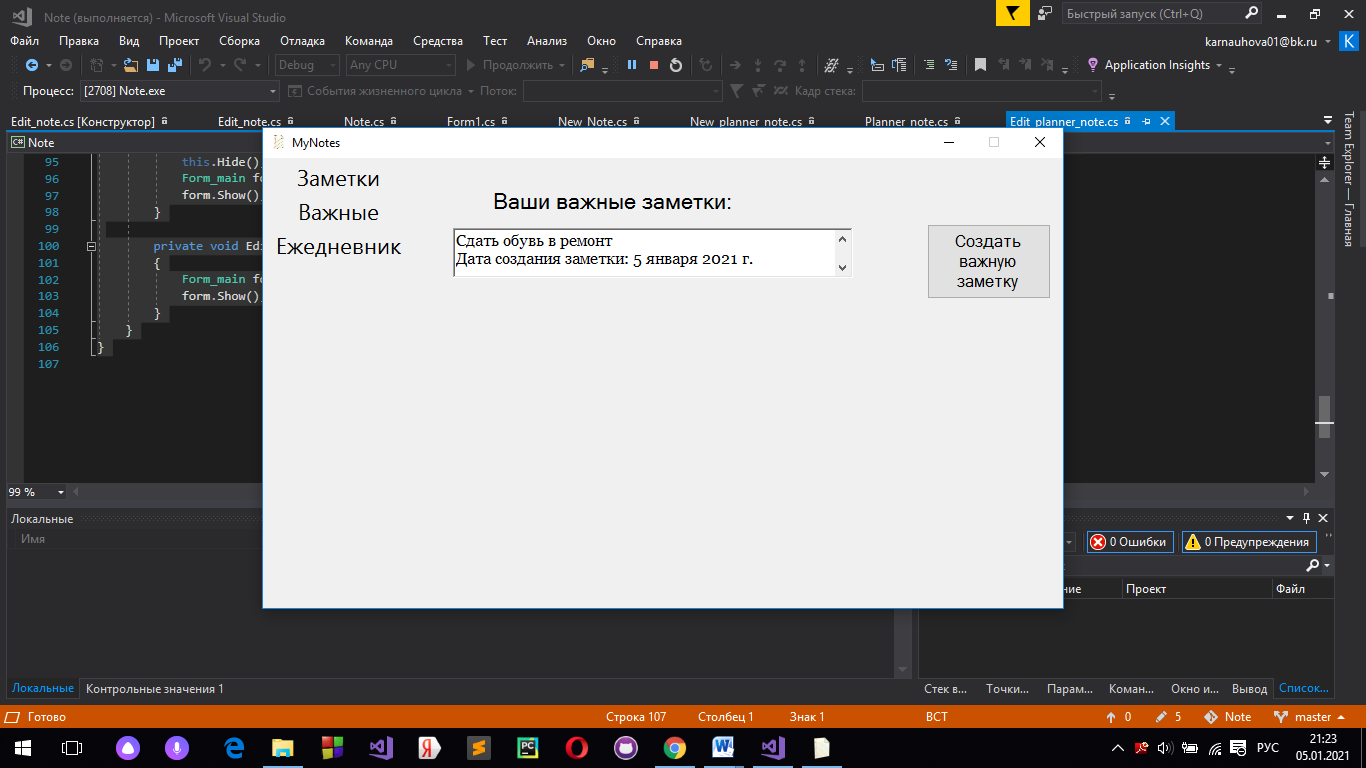
# Приложение Б

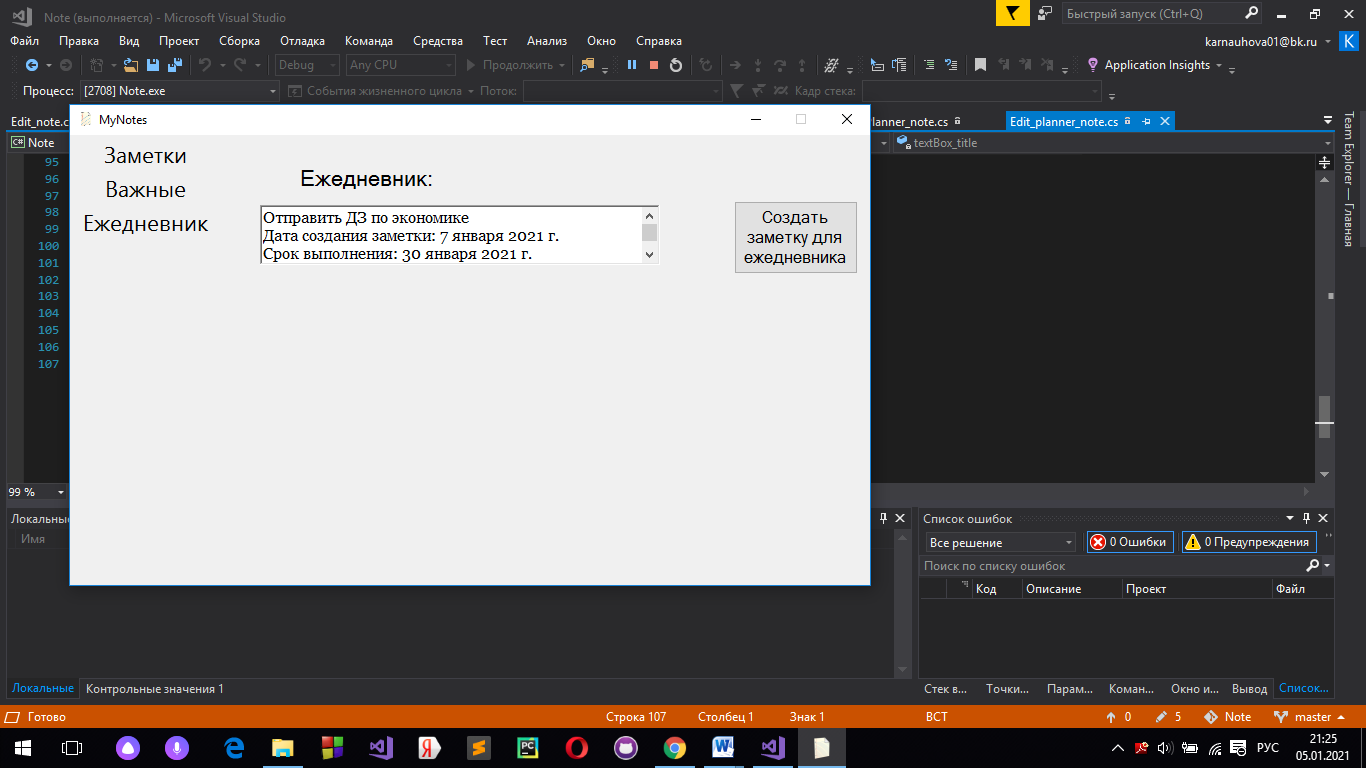
**Снимки экранных форм пользовательского интерфейса Desktop**

Главная форма

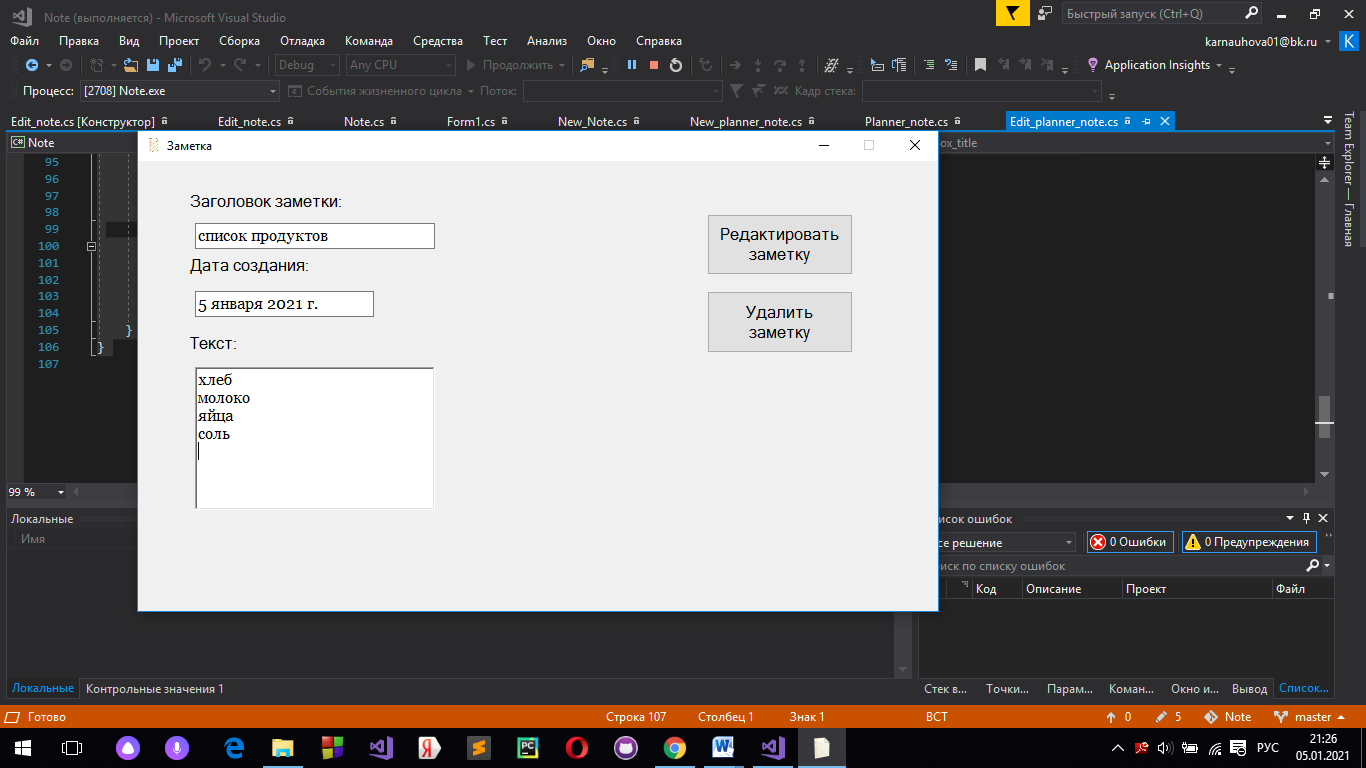




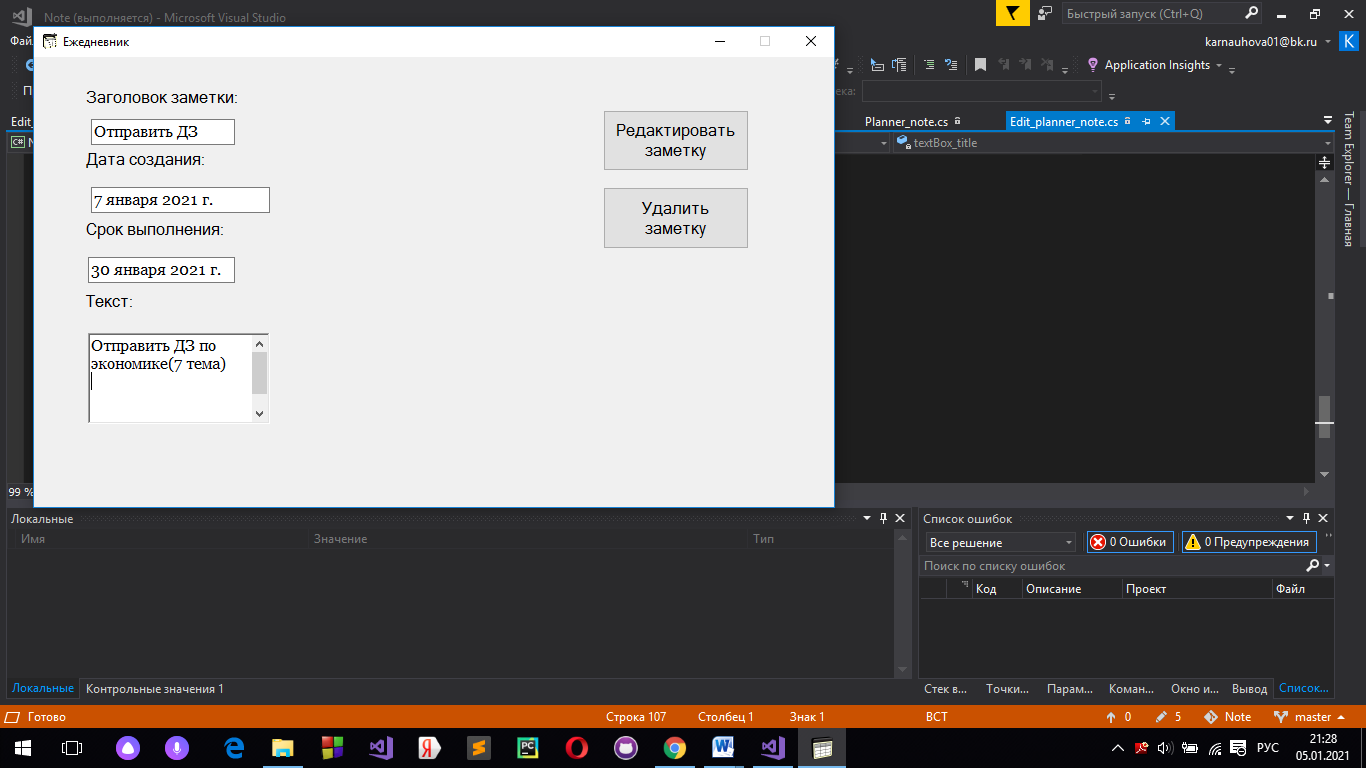




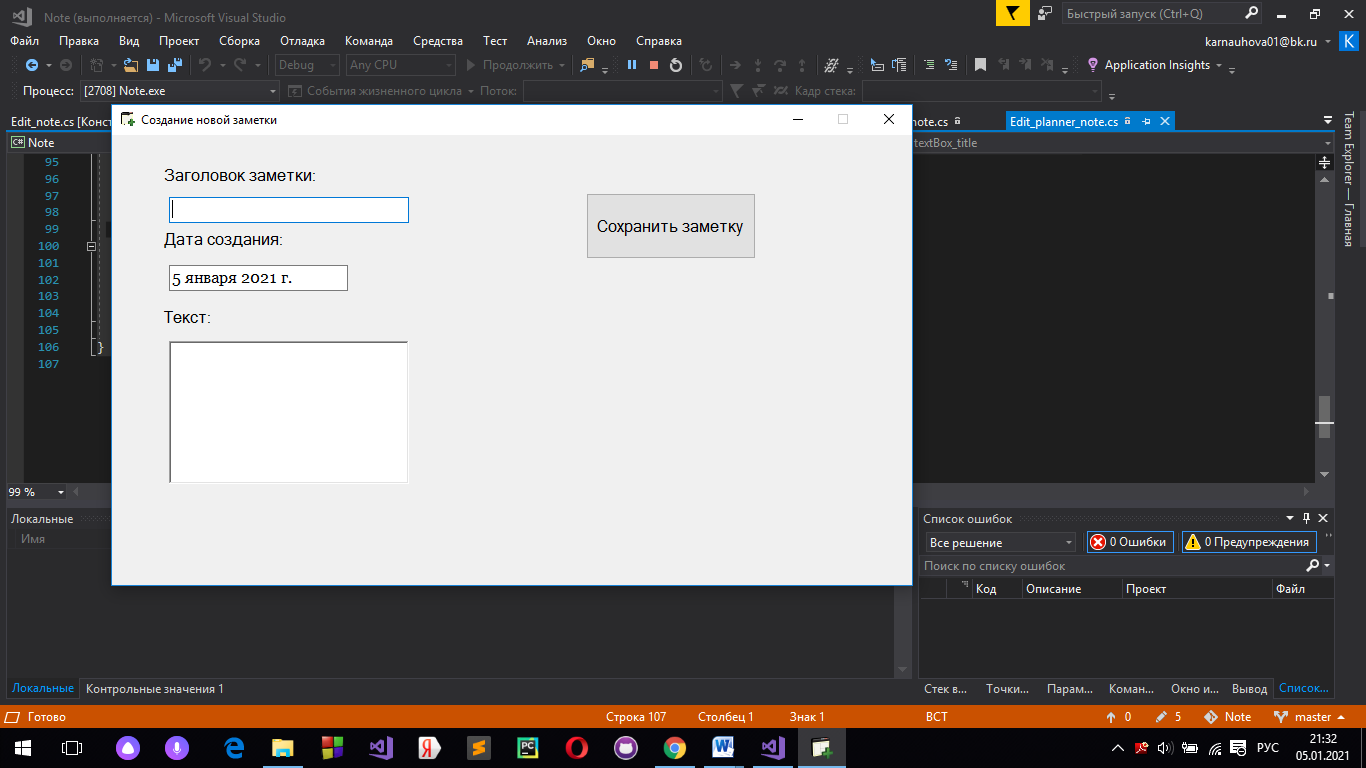
Форма для работы с отдельной заметкой



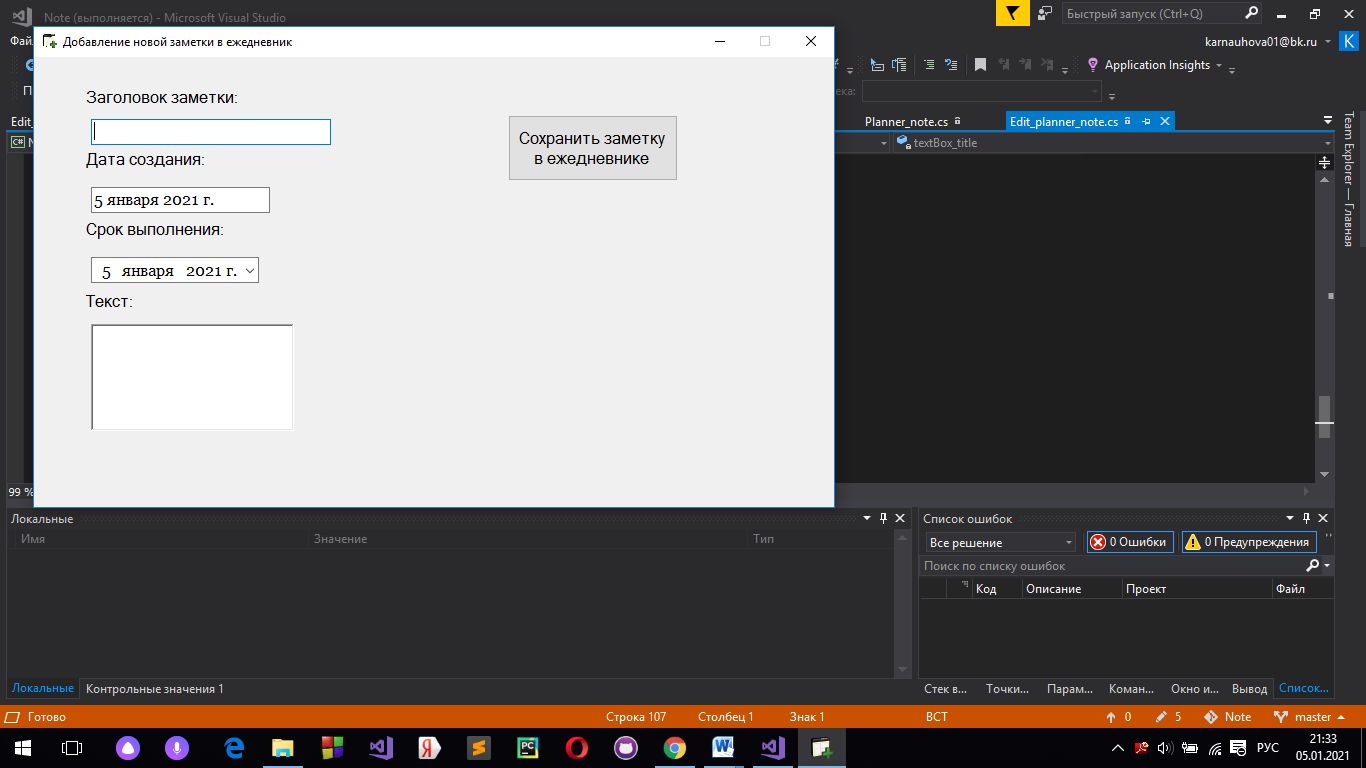
Форма для работы с отдельной заметкой ежедневника



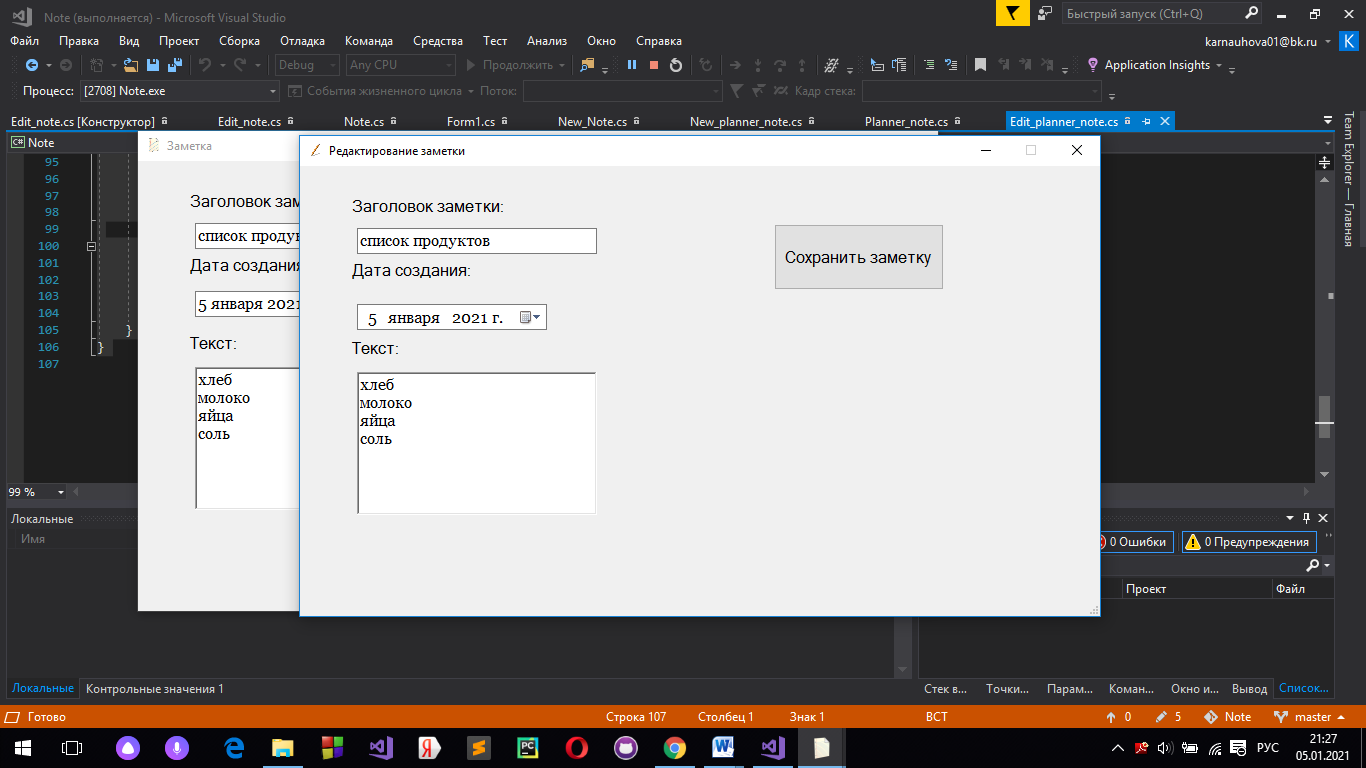
Форма для создания заметки



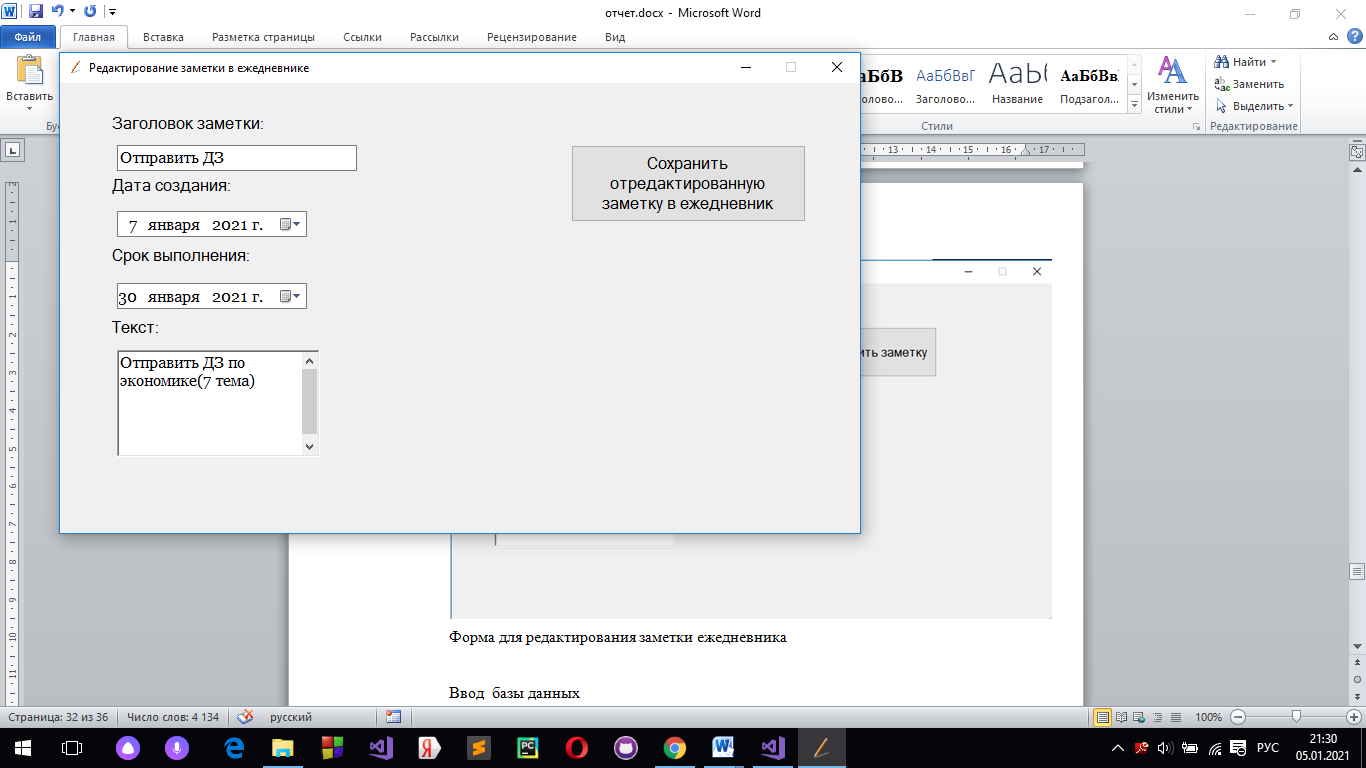
Форма для создания заметки ежедневника



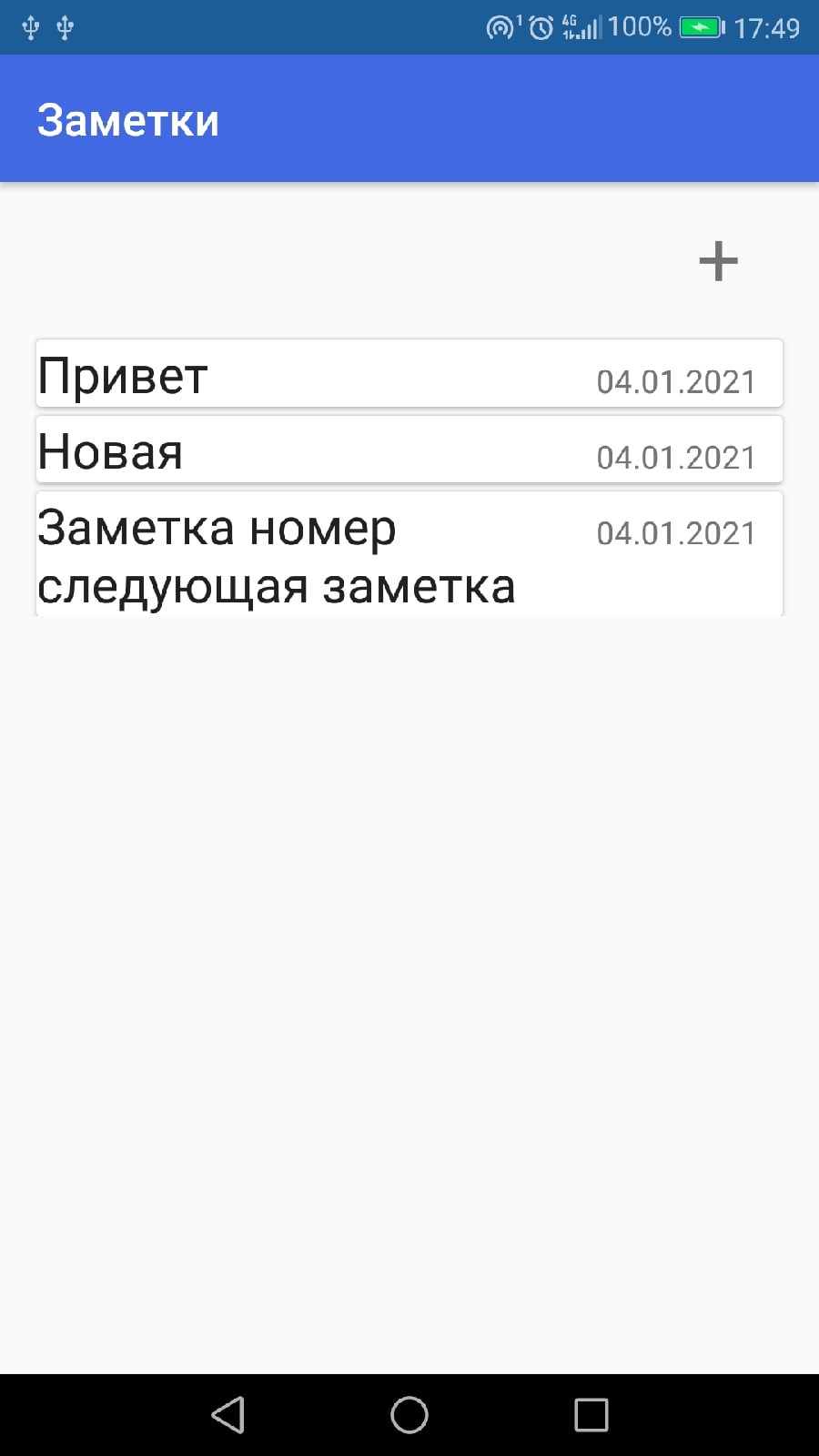
Форма для редактирования заметки



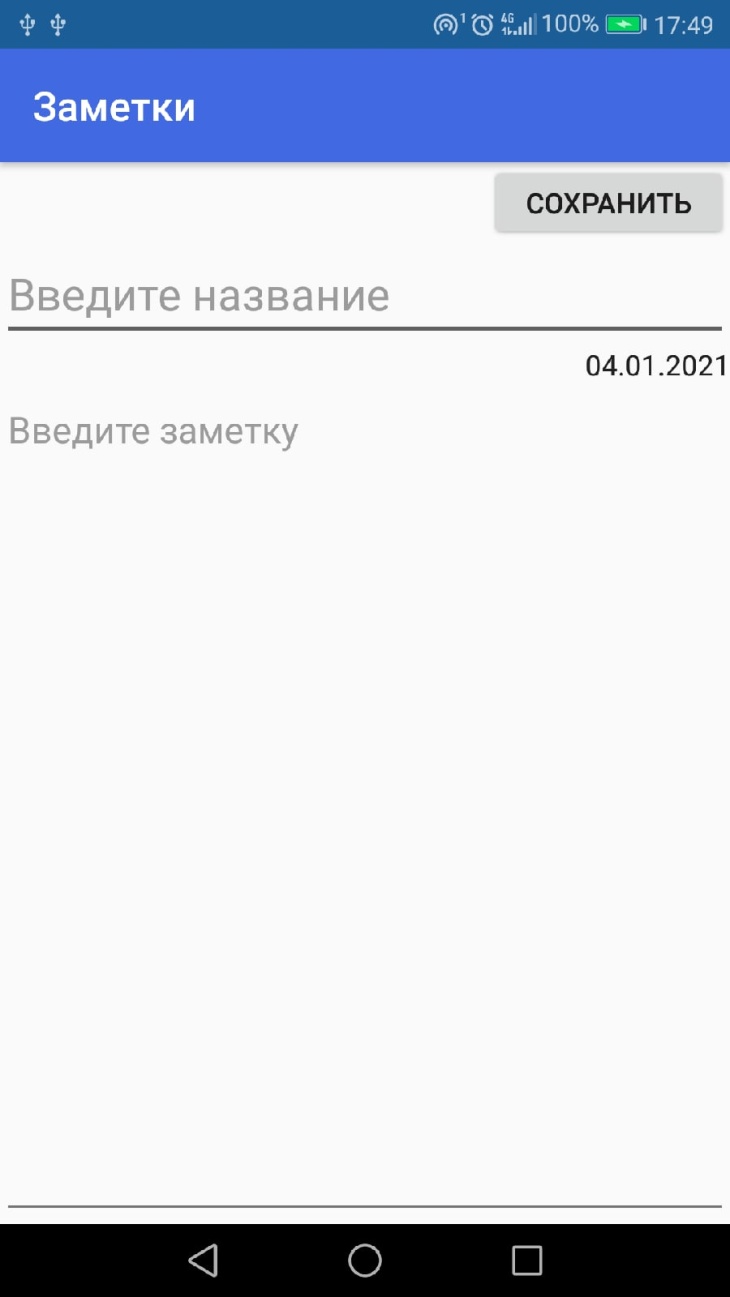
Форма для редактирования заметки ежедневника



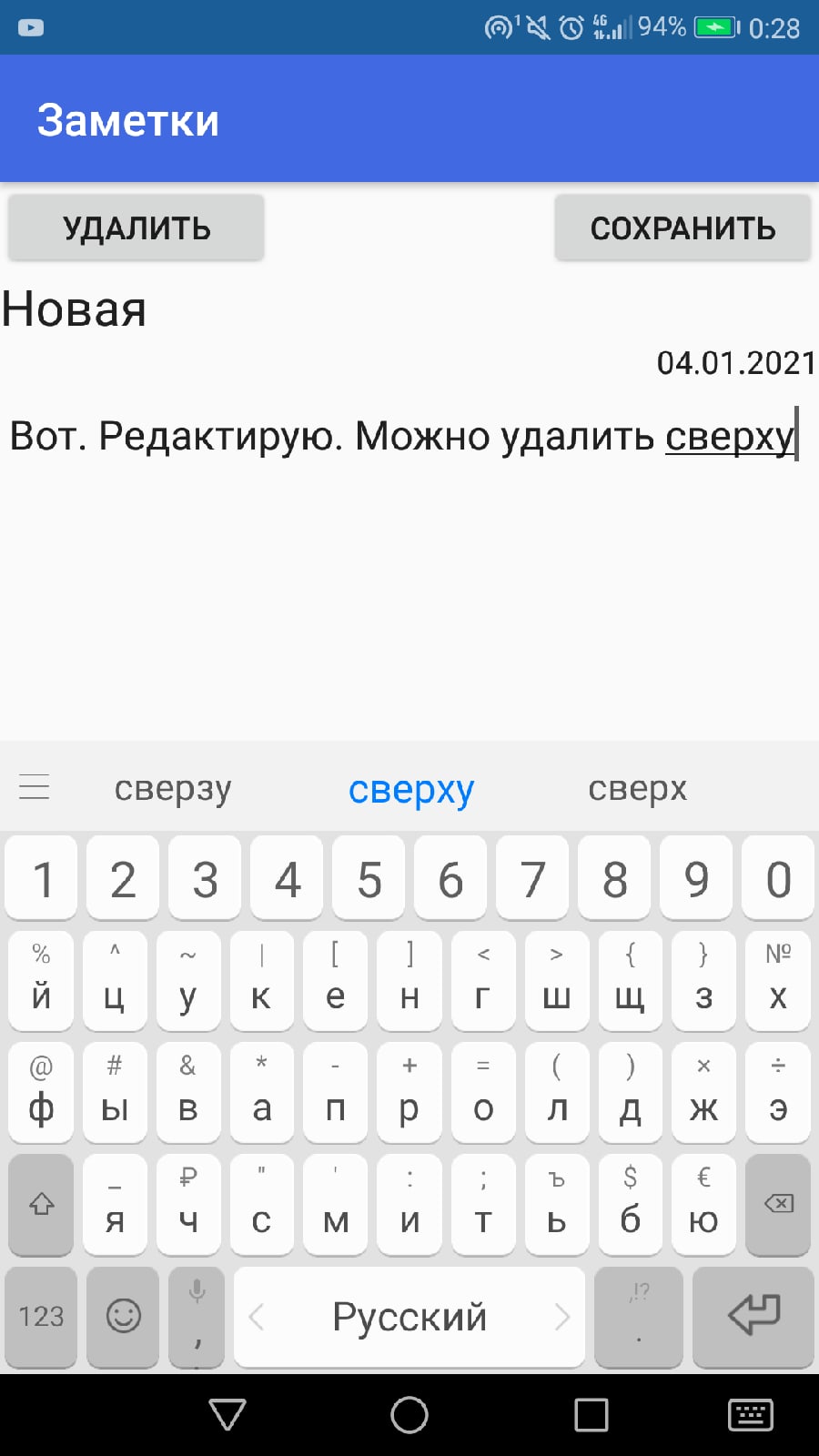
**Снимки экранных форм пользовательского интерфейса Android**



Главная активность заметок



Активность создания новой заметки



Активность просмотра, редактирования и удаления заметки