

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Сибирский государственный университет
телекоммуникаций и информатики» (СибГУТИ)

09.03.01 "Информатика и вычислительная техника"
профиль "Программное обеспечение средств
вычислительной техники и
автоматизированных систем"

ОТЧЕТ

по дисциплине «Визуальное программирование и
человеко-машинное взаимодействие»

Курсовая работа

Выполнил:
студент группы ИП-811 Разумов Д.Б.

Проверил:
преподаватель Мерзлякова Е.Ю.

Новосибирск 2020

Оглавление

ЧАСТЬ 1.....	3
ЗАДАНИЕ.....	3
ВАРИАНТ ЗАДАНИЯ.....	4
АНАЛИЗ ЗАДАЧ И ПОЛЬЗОВАТЕЛЕЙ.....	5
ВЫБОР РЕПРЕЗЕНТАТИВНЫХ ЗАДАЧ.....	6
ЗАИМСТВОВАНИЕ.....	7
ЧЕРНОВОЕ ОПИСАНИЕ ДИЗАЙНА.....	9
ЧАСТЬ 2.....	12
ЗАДАНИЕ.....	12
ВЫПОЛНЕНИЕ ЗАДАНИЯ.....	13
ЧАСТЬ 3.....	21
ЗАДАНИЕ.....	21
ПЕРВАЯ ЗАДАЧА.....	22
Описание задачи.....	22
SWT-анализ.....	23
GOMS-анализ.....	26
Выявленные проблемы.....	29
ВТОРАЯ ЗАДАЧА.....	30
Описание задачи.....	30
SWT-анализ.....	30
GOMS-анализ.....	32
Выявленные проблемы.....	32
ПРАВИЛА НИЛЬСЕНА-МОЛИХА.....	33
Теория и анализ.....	33
Выявленные проблемы.....	35
ПРИНЦИПЫ ОРГАНИЗАЦИИ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА.....	36
Теория и анализ.....	36
Выявленные проблемы.....	39
РЕШЕНИЕ ПРОБЛЕМ.....	40

ЧАСТЬ 1

ЗАДАНИЕ

Необходимо разделить на подгруппы по 1-2 человека и выбрать вариант задачи таким образом, чтобы они не повторялись в Вашей группе. Варианты приведены ниже. Можно согласовать свой вариант.

Затем, выполнить следующие этапы разработки приложения:

- **анализ задач и пользователей;**

Найти двух человек, которые могут быть заинтересованы в решении предложенной задачи. Дайте их краткое описание (возраст, образование, профессия, навыки в выбранной сфере, навыки владения компьютером).

- **выбор репрезентативных задач;**

Перечислите репрезентативные задачи; затем все задачи, решение которых будет поддерживать разрабатываемая программа.

- **заимствование;**

Найдите приложения или сайты, с которых можно заимствовать какие-либо решения интерфейса, приведите ссылку на источник. Эти приложения не обязательно должны выполнять точно такие же задачи. Заимствовать можно что угодно, даже расположение кнопок. Выберите и напишите, что именно Вы будете заимствовать из данных приложений и зачем.

- **черновое описание дизайна;**

Опишите черновой вариант дизайна словами и графически (иллюстрации с пояснениями). Черновой вариант должен отражать все внешние элементы интерфейса и их назначение.

ВАРИАНТ ЗАДАНИЯ

Я выбрал (и согласовал) свой вариант: личный дневник. Программа предназначена заменить бумажный личный дневник, в котором человек делает записи о событиях своей жизни или о чем-то личном. Обычно в бумажном личном дневнике перед каждой записью ставится дата. Поэтому главный функционал моей программы — это возможность для пользователя создавать текстовые записи, которые привязываются к выбранной дате.

Программа должна использовать базу данных.

АНАЛИЗ ЗАДАЧ И ПОЛЬЗОВАТЕЛЕЙ

Цель: найти двух человек, которые могут быть заинтересованы в решении предложенной задачи. Дайте их краткое описание (возраст, образование, профессия, навыки в выбранной сфере, навыки владения компьютером).

Решение: программу можно использовать по назначению и вести личный дневник. Но с другой стороны можно использовать программу для расписания дел на день, чтобы не держать все это в голове. Например, можно сделать запись на завтрашний день, где пользователь запишет, какие дела завтра у него будут.

Люди:

1. Мой приятель. Возраст — 20 лет, образование — неоконченное высшее, профессия — программист, навыки владения компьютером — высокие. Заинтересован в программе, так как ему нравится вести бумажный личный дневник.
2. Мой знакомый, который много работает. Возраст — 32 года, образование — высшее, профессия — экономист, навыки владения компьютером — средние. Заинтересован в программе, так как ему нужно планировать дела на каждый день.

Примечание: «навыки в выбранной сфере» отсутствуют, так как для ведения электронного дневника нужны только навыки владения компьютером.

ВЫБОР РЕПРЕЗЕНТАТИВНЫХ ЗАДАЧ

Цель: Перечислите репрезентативные задачи; затем все задачи, решение которых будет поддерживать разрабатываемая программа.

Репрезентативные задачи:

1. Создание записей. Можно выбрать дату, для любой даты можно сделать любое количество записей.
2. Выбор даты. Календарь, в котором отображаются все дни выбранного месяца. Можно менять месяц и год. В каждом дне видно, были записи сделаны или нет. Если нажать по дню, то будет выведен список записей, сделанных в этот день.
3. Поиск по записям.

Второстепенные задачи:

1. Добавление необязательных тегов к записям, по которым также можно искать. Например, #работа или #кино.
2. Возможность выгрузки нужных записей в виде .txt.
3. Возможность менять цветовую тему программы.
4. Возможность добавлять фото к записям.
5. Возможность форматирования текста.
6. Выход из приложения.

ЗАИМСТВОВАНИЕ

Задача: найти приложения или сайты, с которых можно заимствовать какие-либо решения интерфейса, приведите ссылку на источник. Эти приложения не обязательно должны выполнять точно такие же задачи. Заимствовать можно что угодно, даже расположение кнопок. Выберите и напишите, что именно Вы будете заимствовать из данных приложений и зачем.

Решение:

1. Программа Rednotebook

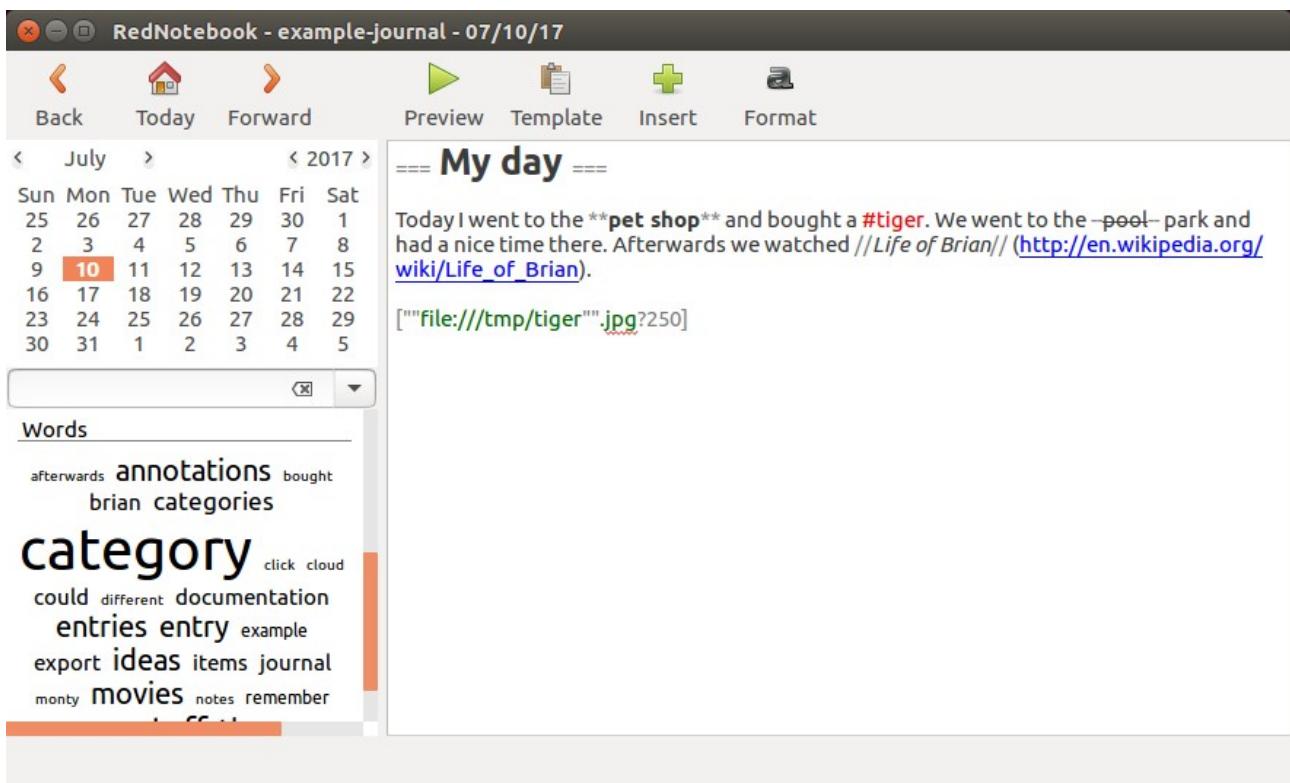


Рисунок 1.1

Отсюда взято расположение главных элементов, возможность одновременно менять месяц и год, функционал.

2. Сайт <https://diaroapp.com>

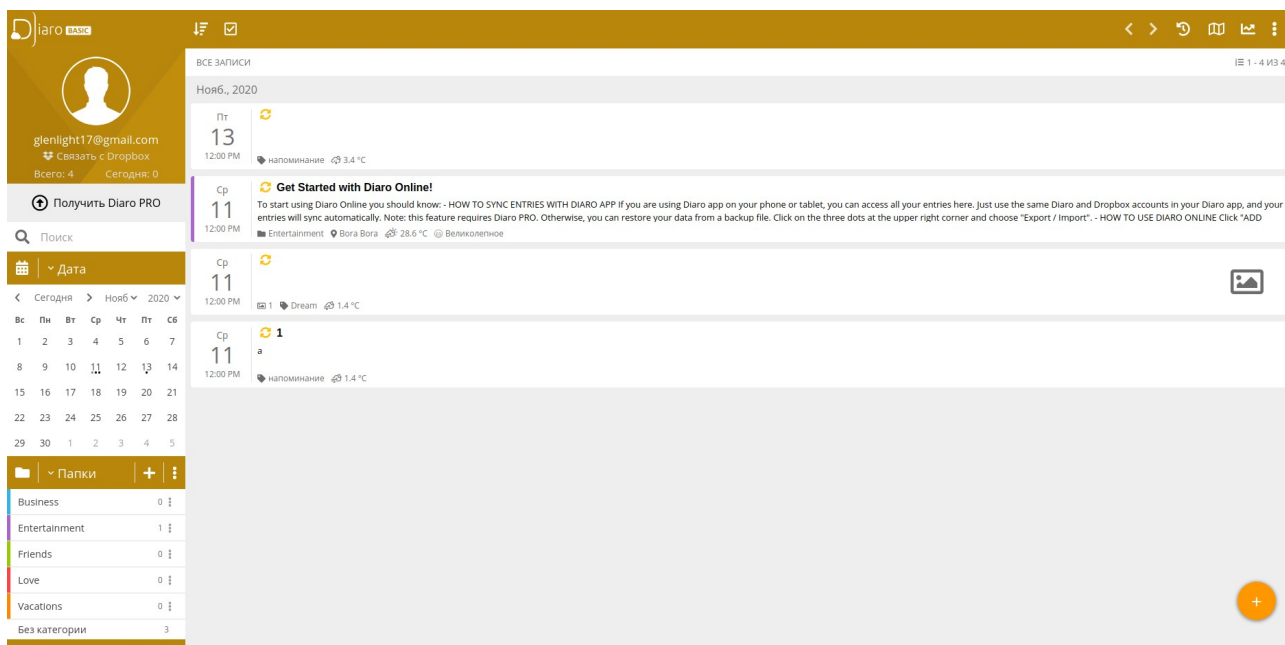


Рисунок 1.2

Отсюда взят вид нескольких записей вместе и возможность менять цветовую тему приложения.

ЧЕРНОВОЕ ОПИСАНИЕ ДИЗАЙНА

Цель: Опишите черновой вариант дизайна словами и графически (иллюстрации с пояснениями). Черновой вариант должен отражать все внешние элементы интерфейса и их назначение.

Выполнение:

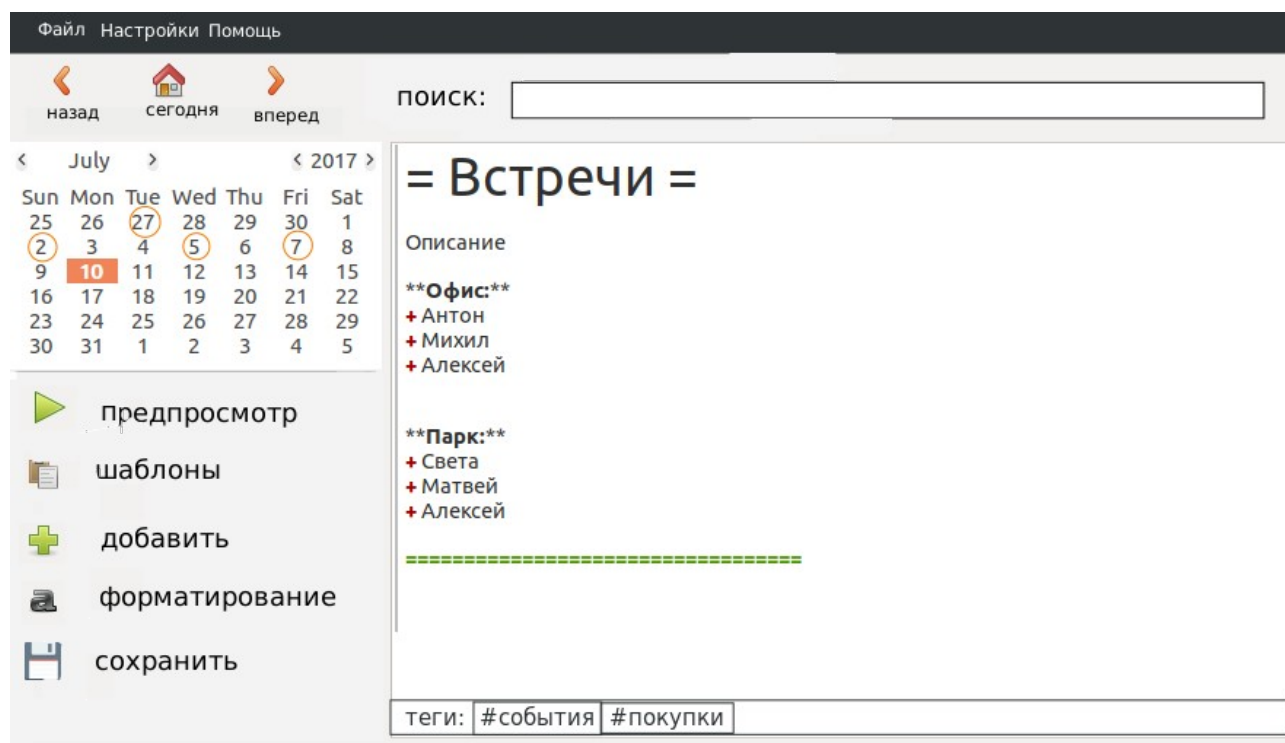


Рисунок 1.3

В верхнем меню подменю «Файл» и раздел «Помощь». В «Настройках» можно настроить внешний вид приложения. «Помощь» отображает окно с руководством по пользованию программой. В «Файле» пользователь увидит следующее:

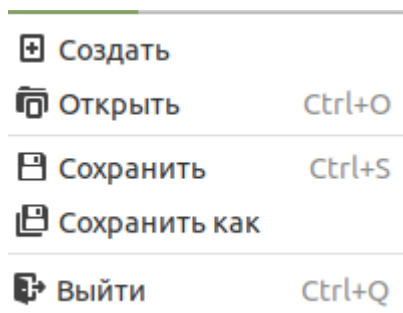


Рисунок 1.4

«Создать» позволяет создать новый дневник, то есть использовать другую базу данных для записей. «Открыть» позволяет переключаться между дневниками. «Сохранить» позволяет сохранить изменения, сделанные в текущем дневнике. «Сохранить как» позволяет сохранить записи в .txt файле.

Кнопки под главным меню позволяют перемещаться по датам, которые видны на календаре. В календаре текущая дата отмечается закрашенным фоном. Даты, в которых были сделаны записи, помечаются кружками.

Самое большое поле справа-внизу — это текстовый редактор, в котором и делаются записи. Здесь также можно применять оформление. Например, заголовки задаются знаками «===» слева и справа. Внизу добавляются теги (через Enter).

Сверху от редактора есть строка для поиска. Через него ищутся записи по совпадениям.

Слева от редактора есть пять кнопок. «Предпросмотр» - редактор меняет вид в соответствии с оформлением, например, так:

Встречи

Описание

Офис:

1. Антон
2. Михил
3. Алексей

Парк:

1. Света
 2. Матвей
 3. Алексей
-

Рисунок 1.5

Кнопка «Шаблоны» - представляет пользователю несколько шаблонов на выбор. Например, можно выбрать шаблон для путешествий, где уже будут написаны заголовки разделов: дата, локация, участники, описание, картинки.

Кнопка «Добавить» позволяет добавить картинку, ссылку, горизонтальный разделитель и т. д.

Кнопка «Форматирование» позволяет форматировать выделенный текст в редакторе. Например, выделенный текст можно сделать заголовком нужного

уровня или сделать текст курсивным.

Кнопка «Сохранить» сохраняет изменения в редакторе.

ЧАСТЬ 2

ЗАДАНИЕ

Запрограммировать приложение в среде Qt Creator. Обязательно использование минимальной базы данных, заставки, навигатора помощи, стилей.

ВЫПОЛНЕНИЕ ЗАДАНИЯ

Я написал программу (в ОС GNU/Linux), которая использует базу данных SQLite. В базе данных (далее - БД) есть таблица с названием «records» - по этому названию программа обращается к БД. В данной таблице есть следующие поля:

- `_id` — уникальный номер для каждой записи, который устанавливается автоматически;
- `year, month, day` — дата, к которой привязана запись;
- `title` — заголовок записи;
- `text` — текст записи;
- `hashtags` — хэштеги.

```
QString str = "CREATE TABLE records ("  
              "_id INTEGER PRIMARY KEY AUTOINCREMENT,"  
              "year INTEGER,"  
              "month INTEGER,"  
              "day INTEGER,"  
              "title TEXT,"  
              "text TEXT,"  
              "hashtags TEXT);";
```

Рисунок 2.1 — QString в программе, с помощью которого создается таблица.

Последние три поля можно оставить пустыми. С помощью SQL-запросов программа создает и отображает нужные пользователю записи.

Loading

Loading modules: 72%

Рисунок 2.2 — заставка загрузки приложения.

Личный дневник

Файл Настройки Помощь

< декабрь > 🏠 < 2020 >

пн	вт	ср	чт	пт	сб	вс
30	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10

Заголовок:

Содержимое записи:

Поиск

Удалить запись

Сохранить запись

Форматирование текста

Шаблоны записей

Добавить

Хэштеги:

Рисунок 2.3 — начальный вид приложения (уже сделано несколько записей).

Слева вверху — календарь, в котором можно выбрать дату. Месяц и год меняются кнопками сверху от календаря (кнопка с изображением дома —

кнопка, которая возвращает календарь на текущий месяц и год). Чтобы выбрать день — надо кликнуть ЛКМ по нужному числу дня.


<	декабрь	>		<	2020	>
пн	вт	ср	чт	пт	сб	вс
30	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10

Рисунок 2.4 - выбранный день отображается синим цветом

<	ноябрь	>		<	2021	>
пн	вт	ср	чт	пт	сб	вс
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5
6	7	8	9	10	11	12

Рисунок 2.5 - измененные месяц и год (дни недели всегда правильные)

После выбора даты пользователь может выбрать уже созданные записи или создать новую запись. Сделать это можно, кликнув по нужной кнопке справа от календаря. Также, можно выбрать нужные записи с помощью одновременного поиска по заголовку, текста и хэштегам записи.

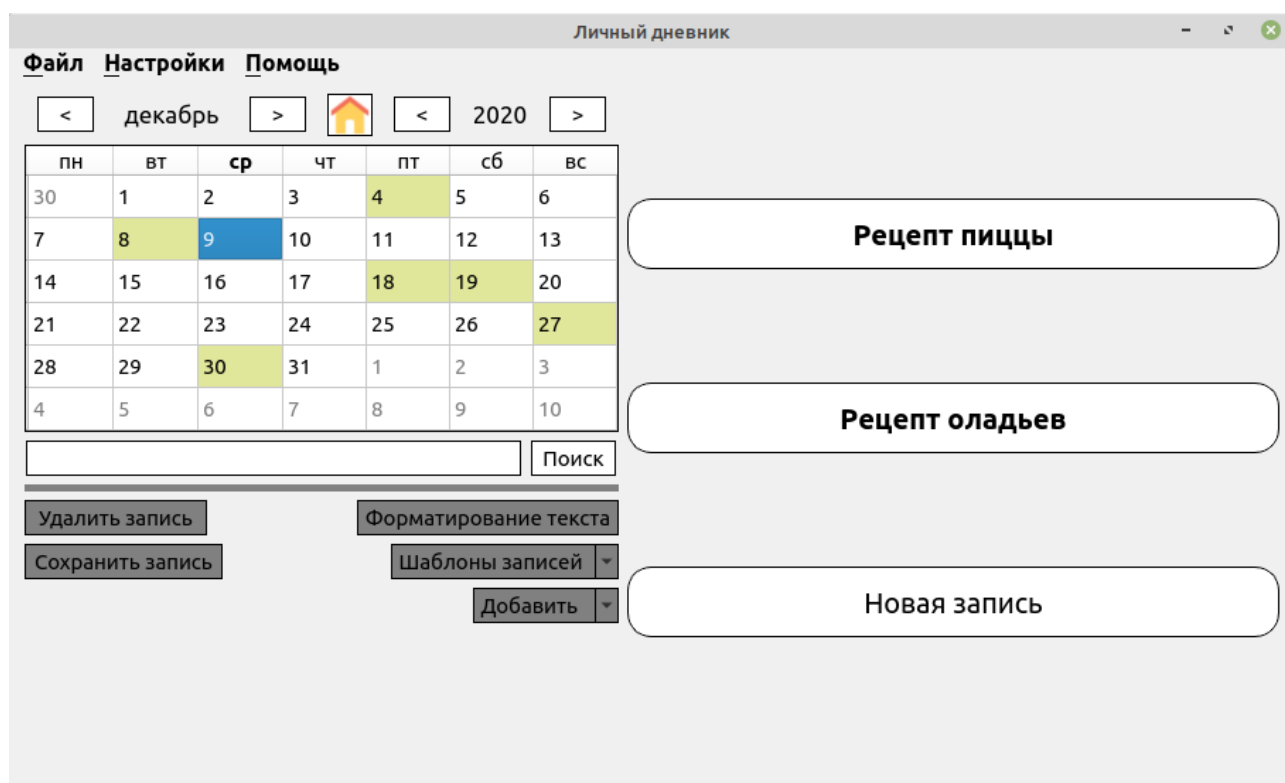


Рисунок 2.6 — отображение записей, у которых дата равна выбранной дате.

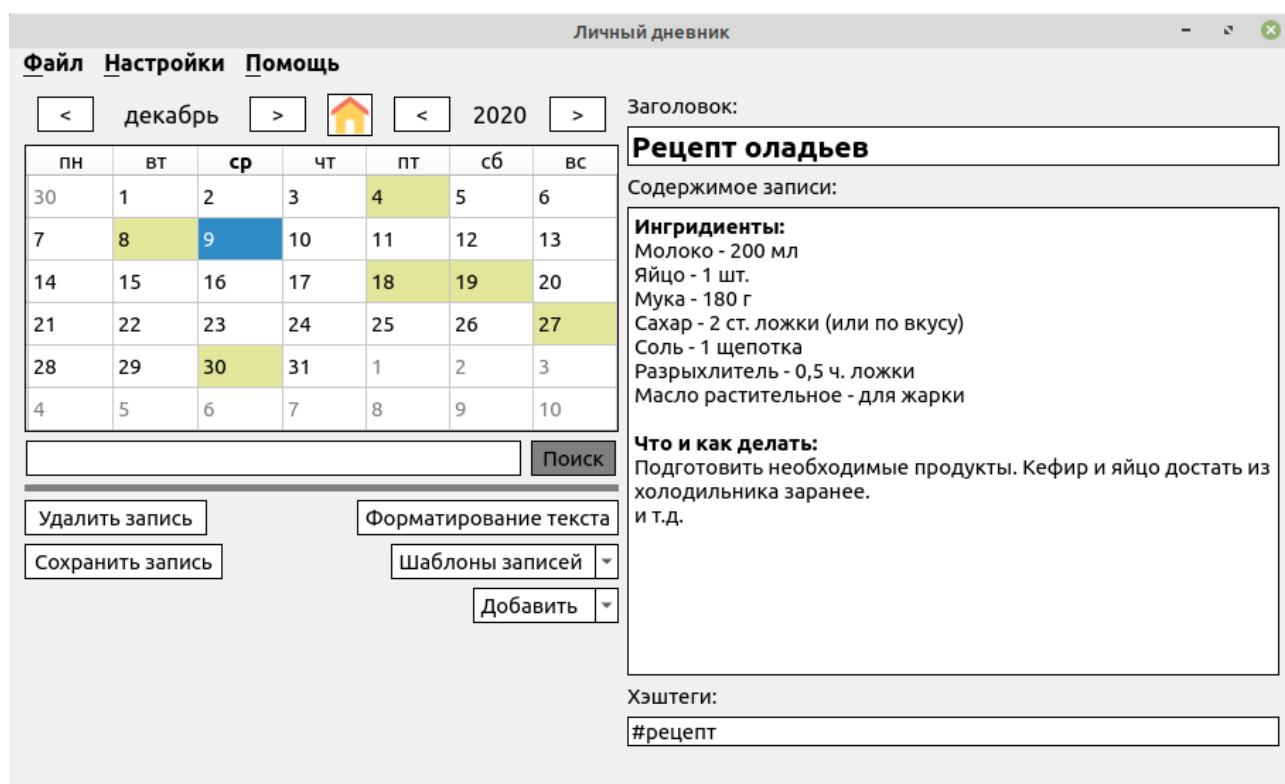


Рисунок 2.7 — редактирование записи (после нажатия кнопки).

Запись редактируется как текст в нужных полях (заголовок, текст, хэштеги). К полю «текст» применяется форматирование HTML. То есть с помощью тэгов

HTML можно делать текст жирным, курсивным, добавлять изображения, списки и т. д. Форматирование включается и отключается кнопкой «Форматирование текста».

```
<b>Ингридиенты: </b>
<br>Молоко - 200 мл
<br>Яйцо - 1 шт.
<br>Мука - 180 г
<br>Сахар - 2 ст. ложки (или по вкусу)
<br>Соль - 1 щепотка
<br>Разрыхлитель - 0,5 ч. ложки
<br>Масло растительное - для жарки

<br><br><b>Что и как делать:</b> <br>
Подготовить необходимые продукты. Кефир и яйцо достать из
холодильника заранее.
<br> и т.д.
```

Рисунок 2.8 - текст записи без форматирования

Запись сохраняется с помощью кнопки «Сохранить запись». Удаляется через «Удалить запись». Также есть кнопки «Шаблоны записей», которые создают «заготовки» для создания записей, и кнопка «Добавить», которая добавляет в текст записи HTML-теги.

Заголовок:

Путешествие в...

Содержимое записи:

Дата:
Страна:
Участники:

Хэштеги:

#путешествие

Рисунок 2.9 - пример шаблона

В вернем меню есть три раздела: «Файл», «Настройки», «Помощь». В разделе файл пользователь может открыть другую базу данных, сохранить изменения в текущей записи, сохранить текущую базу данных в новом файле, выгрузить базу данных в txt-файл и выйти из программы.

Открыть	
Сохранить	Ctrl+S
Сохранить как	
Выгрузить в txt-файл	
Выход	

Рисунок 2.10 - раздел "Файл"

В разделе «Настройки» пользователь может менять цветовую тему программы: стандартную светлую или темную.

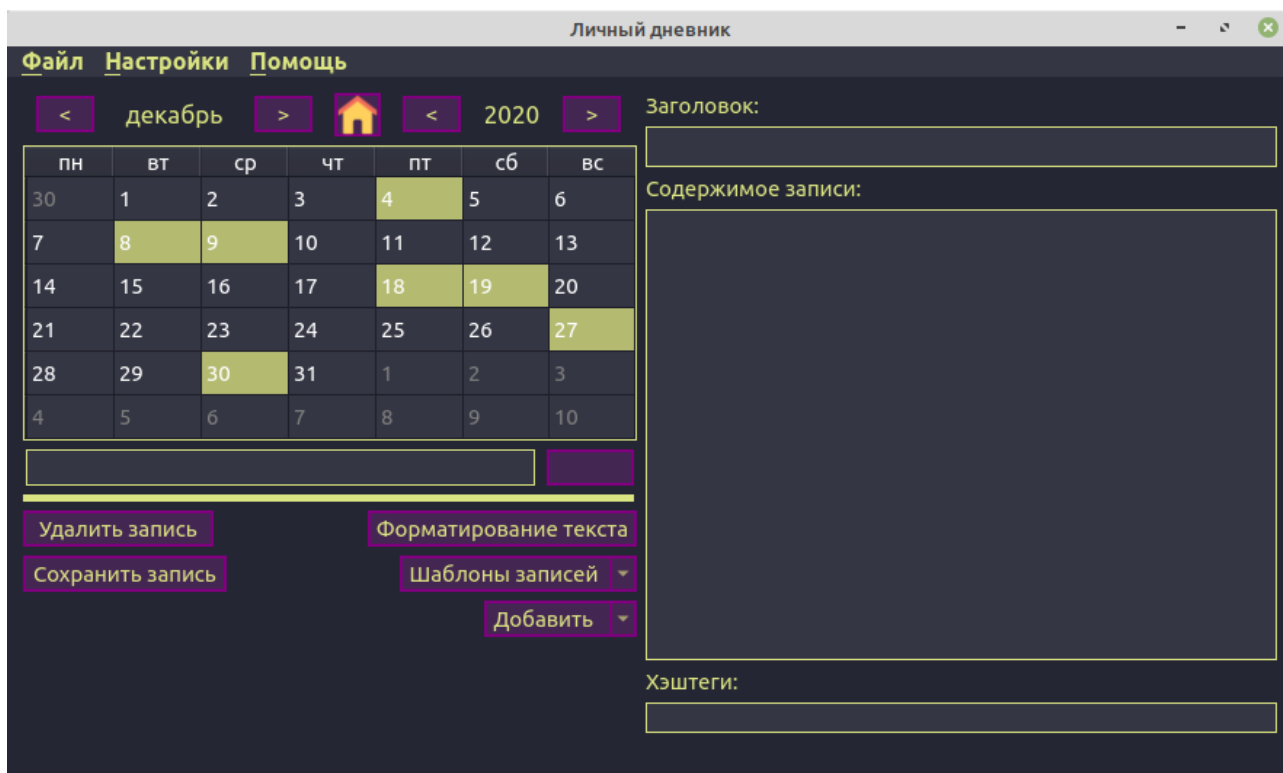


Рисунок 2.11 — темная тема программы

В разделе «Помощь» пользователь может вызвать небольшой справочник, который объясняет работу программы.

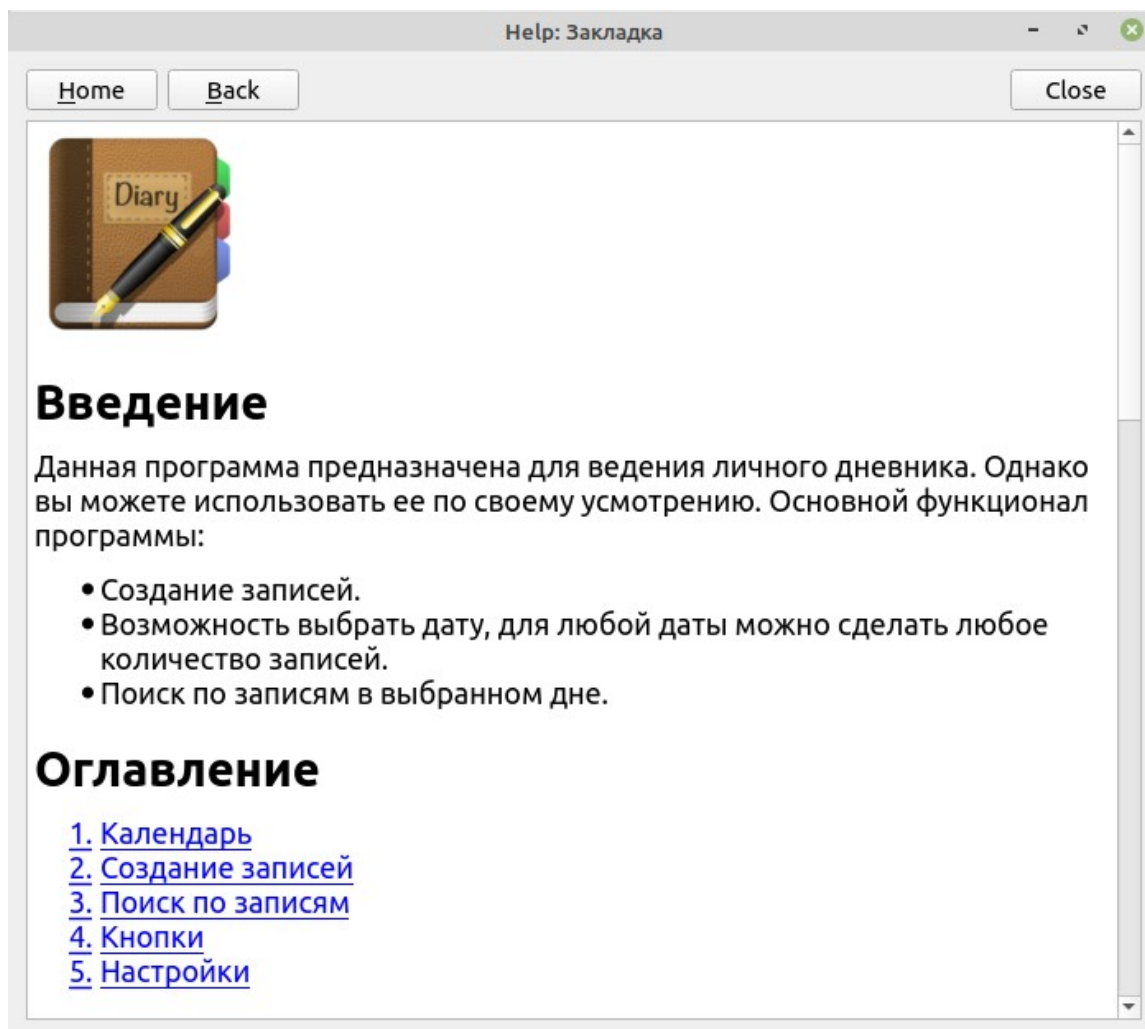


Рисунок 2.12 — справочник программы

ЧАСТЬ 3

ЗАДАНИЕ

Проведите CWT и GOMS анализы интерфейса на примере двух репрезентативных задач, а также подробный анализ интерфейса на соответствие Правилам Нильсена-Молиха и правилам организации графического интерфейса, приводя примеры для обоснования результата анализа. По итогу анализа необходимо составить список проблем и исправить интерфейс, привести конечный результат.

ПЕРВАЯ ЗАДАЧА

Описание задачи

Сделать запись для 11-го января 2021-го года. Запись должна содержать информацию о списке дел. Точная последовательность действий будет такой (для моей программы):

1. Выбрать в программе месяц «Январь» и год «2021». Предположим, что перед выполнением этой задачи в календаре отображается сентябрь 2019-го года.
2. Кликнуть по дню с числом «9».
3. Нажать на кнопку «Новая запись».
4. Нажать на «Шаблоны записей» и выбрать «Дела»
5. В «Содержимом записи» создать следующий текст:
 1. Выгулять собаку.
 2. Позвонить Ване.
 3. Купить помидоры, лук, печенье, сок.
 4. Приготовить ужин на четверых.
6. Выделить последнюю строку курсивом.
7. Посмотреть на запись с форматированием тегов HTML, т. е. нажать на кнопку «Форматирование текста».
8. Сохранить запись.

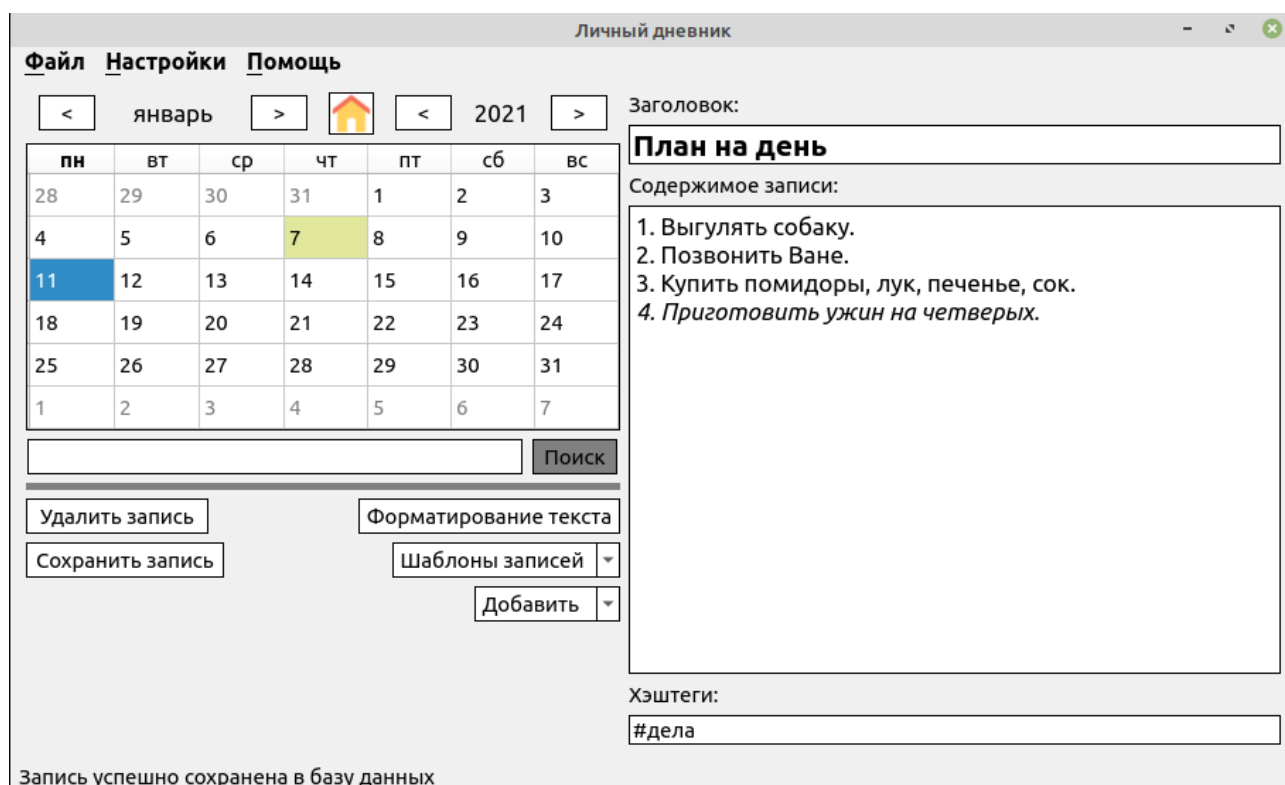


Рисунок 3.1 — как выглядит приложение после выполнения этой задачи

CWT-анализ

Выполнение 1-го пункта не должно вызвать проблем. Календарь имеет стандартный вид, который используется во многих приложениях и операционных системах. Для выполнения этого пункта пользователю надо нажать 4 раза на кнопку «>» у месяца и 2 раза на кнопку «>» у года.

Выполнение 2-го пункта тоже довольно легкое — достаточно кликнуть по любому дню в календаре, и выбранный день подсветится синим цветом. Возможно, у крайне малоопытных пользователей возникнет непонимание, почему надо именно кликнуть по дню в календаре. Если это случится, то пользователь скорее всего догадается кликнуть по «Помощи», где написано «Кликнув по нужной дате, вы выбираете этот день».

Сразу после выполнения 2-го пункта справа от календаря пользователь сразу увидит кнопку для создания новой записи.

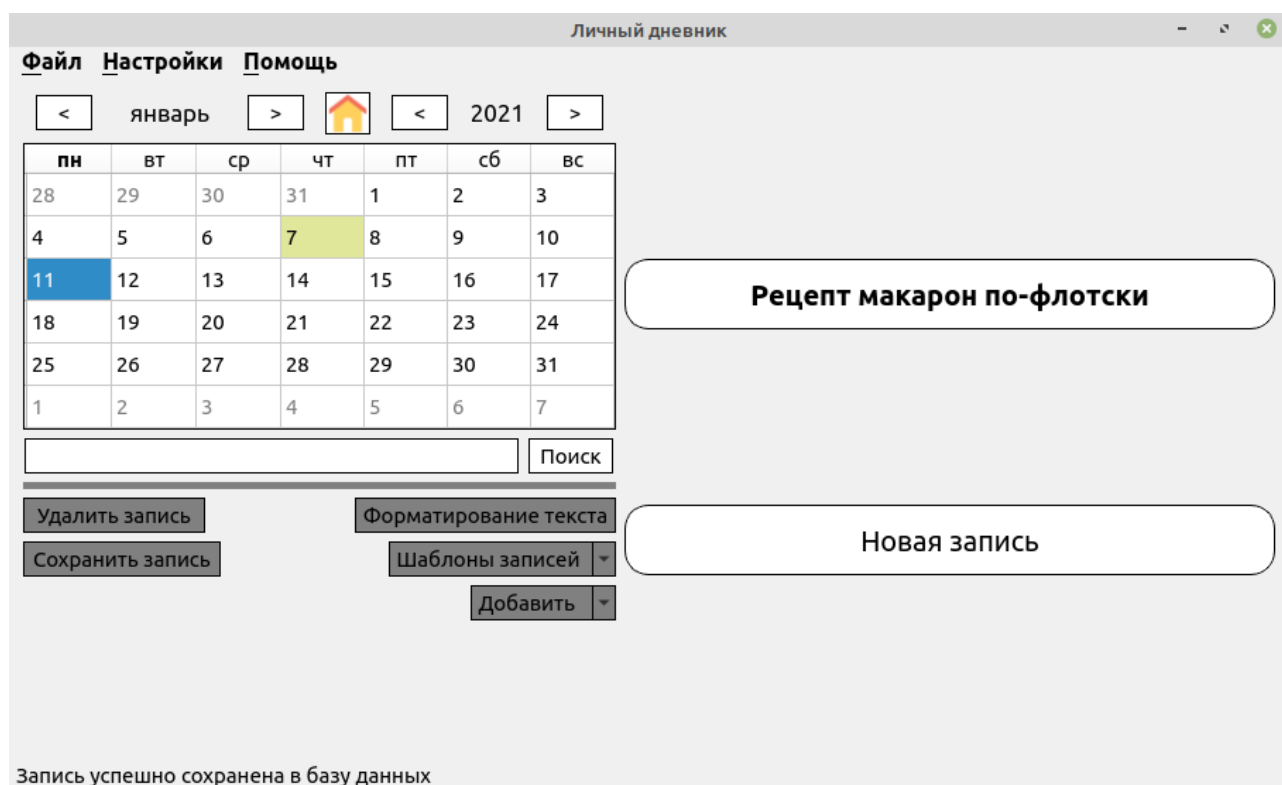


Рисунок 3.2 — после выбора дня программа сразу отображает сделанные записи (если они есть), выделенные жирным, и кнопку «Новая запись»

Выполнение 4-го пункта должно не должно создать проблем, так как легко заметить кнопку «Шаблоны записей». Даже если пользователей не знает о применении шаблонов, то он с легкостью может создать текст из задачи самостоятельно.

Заголовок:

План на день

Содержимое записи:

- 1.
- 2.
- 3.

Хэштеги:

#дела

Рисунок 3.3 - применение шаблона "Дела"

5-ый пункт скорее всего будет проблемным для нового пользователя. В моем приложении указано об HTML-форматировании только в справке. После использования шаблона к нему автоматически применяется форматирование и «Содержимое записи» нельзя изменять. Пользователь может **не понять сразу, что надо нажать на кнопку «Форматирование текста», чтобы показать текст записи без форматирования** (с HTML-тегами). Только после нажатия этой кнопки пользователь сможет изменить текст.

Содержимое записи:

1. Выгулять собаку.

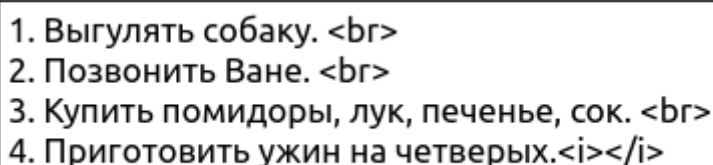
2. Позвонить Ване.

3. Купить помидоры, лук, печенье, сок.

4. Приготовить ужин на четверых.

Рисунок 3.4 - текст записи без форматирования

Для выполнения 6-го пункта пользователь может самостоятельно написать теги курсивного текста. Если он не знает теги HTML, то он может добавить их через кнопку «Добавить». Здесь есть следующая проблема: **теги через эту кнопку всегда добавляются к концу записи**. Было бы лучше, если бы пользователь мог выделить нужный текст и поставить вокруг него нужные теги через «Добавить». Это удобнее. Если пользователь уже напечатал нужный текст, то ему придется вырезать открывающий тег и вставлять его перед началом нужного текста.



```
1. Выгулять собаку. <br>
2. Позвонить Ване. <br>
3. Купить помидоры, лук, печенье, сок. <br>
4. Приготовить ужин на четверых.<i></i>
```

*Рисунок 3.5 - теги курсивного текста
добавлены через кнопку "Добавить"*

Применение форматирования (7-ой пункт) уже не должно вызвать проблем, пользователь познакомился с этим функционалом моей программы.

8-й пункт легкий и понятный — кнопку «Сохранить запись» хорошо видно. Также пользователь получает обратную связь в виде текста слева внизу окна (см. рисунок 3.1).

GOMS-анализ

1. Выбрать в программе месяц «Январь» и год «2021». Предположим, что перед выполнением этой задачи в календаре отображается сентябрь 2019-го года.

Н (переместить руку на мышь, т.к. до этого набирался текст)

М (мыслительная операция)

Р (переместить указатель мыши на кнопку «>» у месяца)

4В (четыре раза кликнуть по кнопке)

М (мыслительная операция)

Р (переместить указатель мыши на кнопку «>» у года)

2В (два раза кликнуть по кнопке)

2. Кликнуть по дню с числом «9».

М (мыслительная операция)

Р (переместить указатель мыши)

В (клик мыши)

3. Нажать на кнопку «Новая запись».

М (мыслительная операция)

Р (переместить указатель мыши)

В (клик мыши)

4. Нажать на «Шаблоны записей» и выбрать «Дела»

М (мыслительная операция)

Р (переместить указатель мыши)

2В (клик мыши на список «Шаблоны», потом на «Дела»)

5.1. В «Содержимом записи» написать «Выгулять собаку.» в 1-й строке.

М (мыслительная операция)

Р (переместить указатель мыши)

Н (переместить руку на клавиатуру)

М (мыслительная операция)

17К (напечатать «Выгулять собаку.»)

5.2 В «Содержимом записи» написать «Позвонить Ване.» во 2-й строке.

М (мыслительная операция)

К (нажать клавишу «стрелка вниз», чтобы перейти на 2-ую строку)

17К (напечатать «Позвонить Ване.»)

5.3 В «Содержимом записи» написать «Купить помидоры, лук, печенье, сок.» во 3-й строке.

М (мыслительная операция)

К (нажать клавишу «стрелка вниз», чтобы перейти на 3-ую строку)

16К (напечатать «Купить помидоры»)

М (мыслительная операция, вспомнить что купить еще)

6К (напечатать «, лук»)

М (мыслительная операция, вспомнить что купить еще)

10К (напечатать «, печенье»)

М (мыслительная операция, вспомнить что купить еще)

7К (напечатать «, сок.»)

5.4 В «Содержимом записи» написать новую строку «4. Приготовить ужин на четверых.»

М (мыслительная операция)

4К (напечатать «
» - тег перевода строки)

М (мыслительная операция)

К (нажать клавишу «стрелка вниз», чтобы перейти на 4-ую строку)

33К (напечатать «4. Приготовить ужин на четверых.»)

6. Выделить последнюю строку курсивом.

М (мыслительная операция)

Н (переместить руку на мышь)

Р (переместить указатель мыши на кнопку «Добавить»)

В (клик мыши)

Р (переместить указатель мыши на кнопку «Наклонный текст»)

В (клик мыши)

М (мыслительная операция)

Р (переместить указатель мыши открывающий тег <i>)

Д (выделить этот тег)

2К (нажать «Ctrl» + «С» одной рукой)

М (мыслительная операция — посмотреть глазами где начало строки)

Р (переместить указатель мыши на начало 4-ой строки)

В (клик мыши)

2К (нажать «Ctrl» + «V» одной рукой)

7. Посмотреть на запись с форматированием тегов HTML, т. е. нажать на кнопку «Форматирование текста».

М (мыслительная операция)

Р (переместить указатель мыши на кнопку «Форматирование текста»)

В (клик мыши)

8. Сохранить запись.

М (мыслительная операция)

Р (переместить указатель мыши на кнопку «Сохранить запись»)

В (клик мыши)

Итого:

$$\begin{aligned} & \text{НМР4ВМР2В} + \text{МРВ} + \text{МРВ} + \text{МР2В} + \text{МРНМ17К} + \text{МК17К} + \\ & \text{МК16КМ6КМ10КМ7К} + \text{М4КМК33К} + \text{МНРВРВМРD2КМРВ2К} + \text{МРВ} + \\ & \text{МРВ} = 6.3 + 2.65 + 2.65 + 2.85 + 7.6 + 4.95 + 13.4 + 10.3 + 12.25 + 2.65 + 2.65 \\ & \text{секунд} = 68.25 \text{ секунд} \end{aligned}$$

Выявленные проблемы

1. Интерфейс программы не может сразу сообщить пользователю о том, что после использования шаблона применяется форматирование текста и как его убрать.
2. Теги через кнопку «Добавить» добавляются только к концу текста — это неудобно для пользователя.

ВТОРАЯ ЗАДАЧА

Описание задачи

Нужно найти все записи в последней неделе декабря 2020-го года, в которых встречается слово «Ваня». В моей программе поиск по записям можно произвести только по одному дню. Поэтому точная последовательность действий будет такой:

1. Кликнуть по дню 28 (предположим, что пользователь уже выбрал на календаре декабрь 2020-го).
2. Ввести в поле поиска «Ваня» и нажать на кнопку «Поиск».
3. Повторить пункты 1-2 для 29, 30, 31 декабря.

CWT-анализ

Выполнение 1-го пункта не должно вызвать проблем.

Выполнение 2-го пункта, возможно, вызовет небольшие проблемы, так как **поле для поиска никак не обозначается** (хоть оно рядом с кнопкой «Поиск»).

Выполнение 3-го пункта вызовет скорее всего недовольство пользователя — ему придется делать одни и те же действия 4 раза. Недостаток моей программы в том, что в ней **нет гибкого поиска**: нельзя сделать поиск по нескольким датам, нельзя искать исключительно по заголовку, тексту записи или хэштегам и т. д.

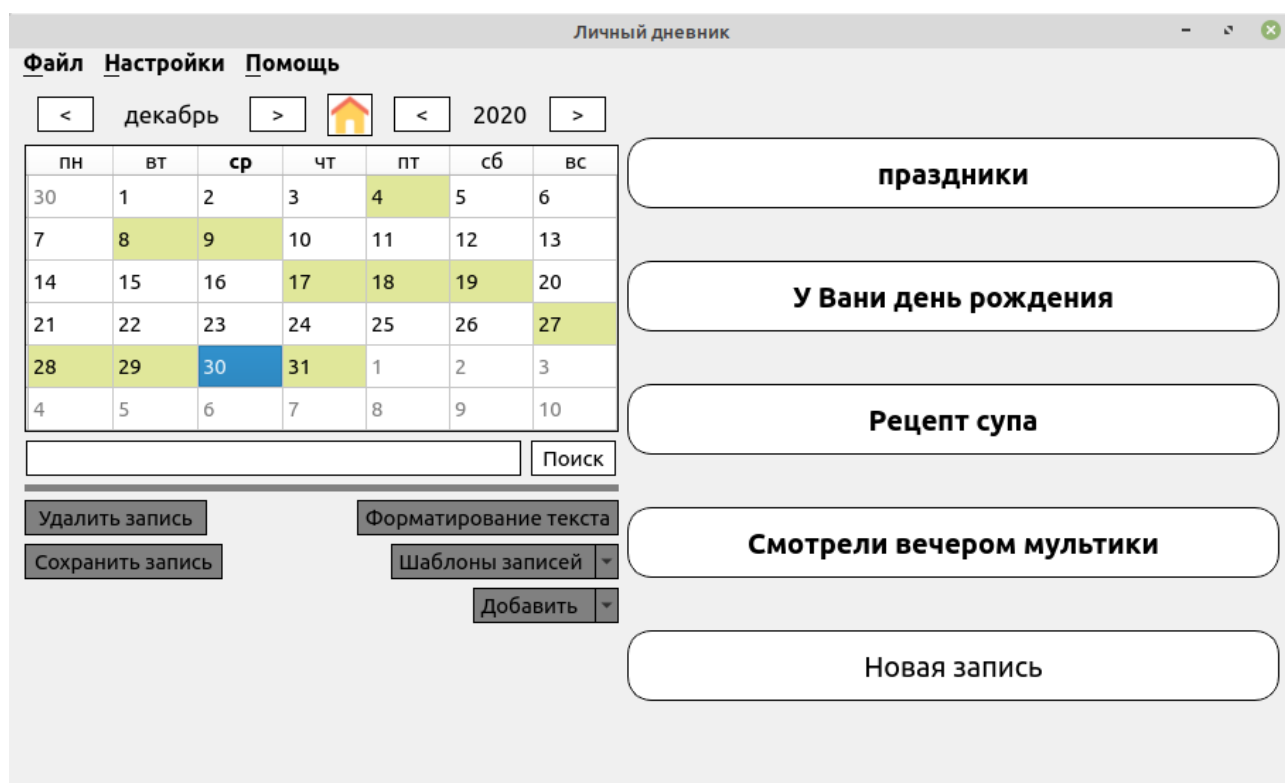


Рисунок 3.6 — вид программы до поиска

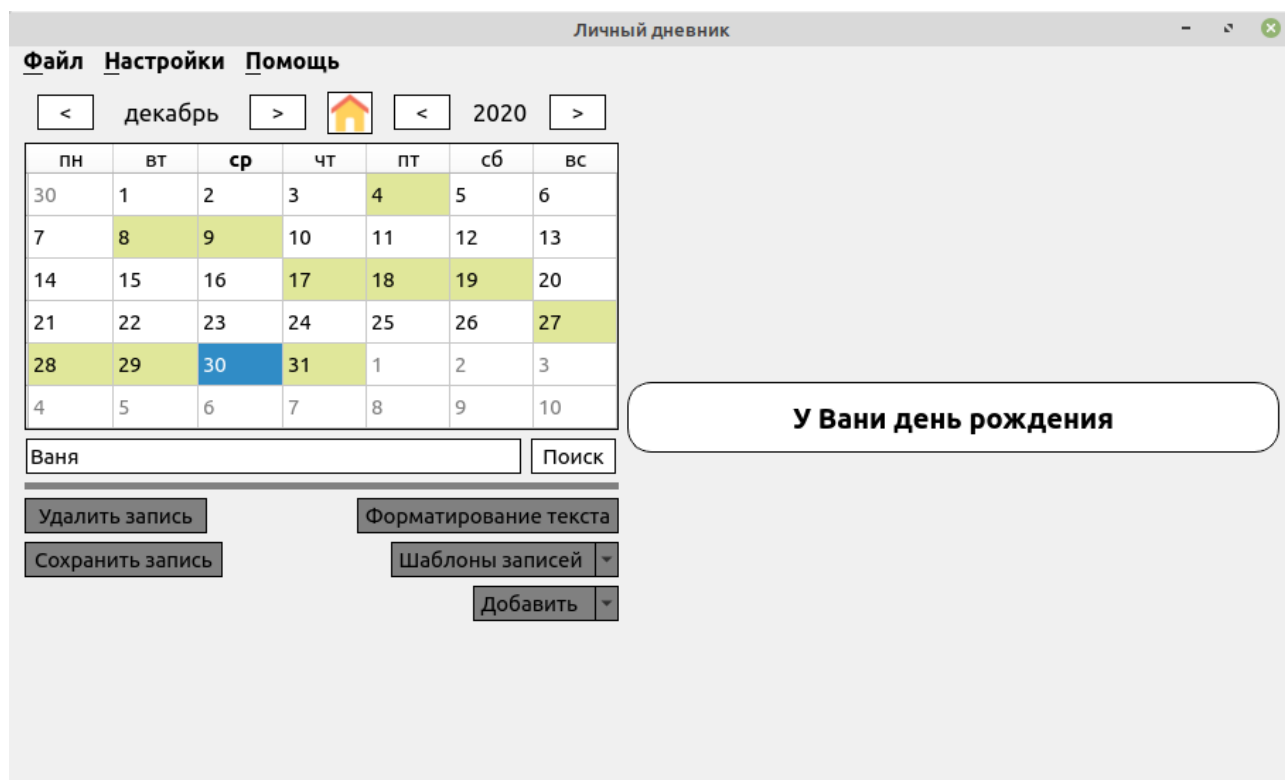


Рисунок 3.7 — вид программы после поиска (слово «Ваня» встречается в тексте записи)

GOMS-анализ

1. Кликнуть по дню 28 (предположим, что пользователь уже выбрал на календаре декабрь 2020-го).

М (мыслительная операция)

Р (переместить указатель мыши)

В (клик мыши по дню)

2. Ввести в поле поиска «Ваня» и нажать на кнопку «Поиск»

М (мыслительная операция)

Р (переместить указатель мыши на поле поиска)

В (клик мыши)

Н (перенести руку на клавиатуру)

5К (напечатать «Ваня»)

М (мыслительная операция)

Н (перенести руку на мышь)

Р (переместить указатель мыши на кнопку «Поиск»)

В (клик мыши)

3. Повторить пункты 1-2 для 29, 30, 31 декабря.

(МРВ+МРВН5КМНРВ) * 3

Итого:

(МРВ+МРВН5КМНРВ) * 4 = (2.65 + 7.1 секунд) * 4 = 39 секунд

Выявленные проблемы

1. Нет обозначения, что поле для поиска — это поле для поиска.
2. Неудобный поиск, можно искать только по всем записям в одном дне.

ПРАВИЛА НИЛЬСЕНА-МОЛИХА

Теория и анализ

1. Простой и естественный диалог. Простота означает, что не должно присутствовать не относящейся к теме или редко используемой информации. Старайтесь добиваться максимального следования задачам пользователя, минимизируя сложность поиска соответствия между семантиками задачи и интерфейсом. Важно представить точно ту информацию, в которой нуждается пользователь, следуя принципу "лучше меньше да лучше". Вся редко используемая информация должна быть спрятана. Естественность означает, что информация, которая выводится на экран, должна появляться в естественном порядке, соответствующем ожиданиям пользователя. Вся взаимосвязанная информация должна группироваться в одном месте.

По-моему, моя программа не содержит ничего лишнего. Я постарался скрывать все неиспользуемые элементы для пользователя. Например, при поиске кнопки для редактирования записи становятся неактивными (см. рис. 3.7).

2. Говорите на языке пользователя. Используйте слова и понятия из мира пользователя. Не используйте специфических инженерных терминов. Например, для большинства людей представление цвета в виде RGB-компонент и их шестнадцатеричная кодировка являются совершенно непонятными вещами. Лучшим и довольно простым решением будет показать вместо кода (или наряду с ним) образец цвета.

В моем интерфейсе используется простой язык.

3. Минимизируйте загрузку памяти пользователя. Не заставляйте пользователя помнить вещи от одного действия к следующему. Оставляйте информацию на экране до тех пор, пока она не перестанет быть нужной. Например, хорошим стилем считается делать только один ряд закладок.

С этим нет проблем. Дни, в которых были сделаны записи, помечаются желтым цветом. Правда, нет гибкого поиска по всем датам, что было описано выше.

4. Будьте последовательны. У пользователей должна быть возможность изучить действия в одной части системы и применить их снова, чтобы получить похожие результаты в других местах.

В моей программе пользователь может использовать любые приобретенные знания к любой записи.

5. Обеспечьте обратную связь. Дайте пользователю возможность видеть,

какой эффект оказывают его действия на систему.

В моей программе есть обратная связь. При любом сохранении записи (успешным или с ошибкой), пользователь получит обратную связь в виде текста внизу программы (см. рис. 3.1). Правда пользователь может и **не заметить небольшой текст**.

6. Обеспечьте хорошо обозначенные выходы. Если пользователь попадает в часть системы, которая его не интересует, у него всегда должна быть возможность быстро выйти оттуда, ничего не повредив.

С этим вроде бы нет проблем. Всегда можно закрыть любое окно на стандартный «крестик». Закрыть запись можно без сохранения — достаточно кликнуть по календарю.

7. Обеспечьте быстрые клавиши и ярлыки. Элементы быстрого доступа могут помочь опытным пользователям избегать длинных диалогов и информационных сообщений, которые им не нужны.

В моей программы вообще нет сообщений, которые надо закрывать.

8. Хорошие сообщения об ошибках. Хорошее сообщение об ошибке помогает пользователю понять, в чём проблема и как это исправить.

С этим у меня явная проблема — если возникает ошибка, то внизу программа выводит **только сообщение об ошибке и в какой функции эта ошибка возникла**. Что никак не помогает пользователю исправить ошибку.

```
///сделать запрос к базе данных:  
createQuery qry;  
if (!qry.exec(str)) {  
    this->statusBar()->showMessage("Ошибка: неправильный запрос в on_pushButton_Search_clicked()");  
}
```

Рисунок 3.8 — код программы, который выводит сообщения об ошибках

9. Предотвращайте ошибки. Всегда, когда вы пишете сообщение об ошибке, вы должны спросить себя, можно ли избежать этой ошибки? Хороший пример построения интерфейса, предотвращающего ошибки – стандартная программа «Калькулятор», в которой при переключении в двоичный режим интерфейс делает недоступными числовые кнопки, кроме 0 и 1, а также делает недоступными некоторые функции, имеющие смысл только для чисел с плавающей точкой. Этот простой приём позволяет минимизировать возможные ошибки пользователя.

В моей программе есть следующая проблема — **если пользователь введет в запись хотя бы один апостроф «'», то возникнет ошибка**. Возникнет ошибка при добавлении записи в базу данных, так как в запросе строчные переменные обрамляются апострофами.

10. Снабдите программу системой помощи. Справка должна быть максимально подробной и рассчитана на абсолютно неопытного пользователя.

В моем приложении есть справка. Но, пожалуй, **она рассчитана только на опытных пользователей.**

Выявленные проблемы

1. Плохая обратная связь. Текст сложно заметить.
2. Неправильные сообщения об ошибках.
3. Нет обработки ошибки, когда пользователь вводит знак апострофа.
4. Справка не рассчитана на неопытных пользователей.

ПРИНЦИПЫ ОРГАНИЗАЦИИ ГРАФИЧЕСКОГО ИНТЕРФЕЙСА

Теория и анализ

1. Принцип кластеризации. Организуйте экран в виде визуально разделённых блоков с похожими элементами управления, предпочтительно с названием для каждого блока. Элементы управления включают меню, диалоговые боксы, экранные кнопки и любые другие графические элементы, позволяющие пользователю взаимодействовать с компьютером. Подобные команды должны быть в одном меню: это позволяет им быть визуально близко и идти под одним заголовком. Команды, относящиеся к некоторой конкретной области функциональности, могут также быть показаны в диалоговых боксах, в визуально определяемых блоках. Тот же самый принцип распространяется на специальные управляющие экраны с многочисленными кнопками и значками, такие как сенсорные панели. Кнопки для конкретной функции должны группироваться вместе, а затем выделяться цветом, обрамлением или окружающим пробелом.

Моя программа более-менее соответствует этому принципу. Блоки выделены красным на следующем рисунке:

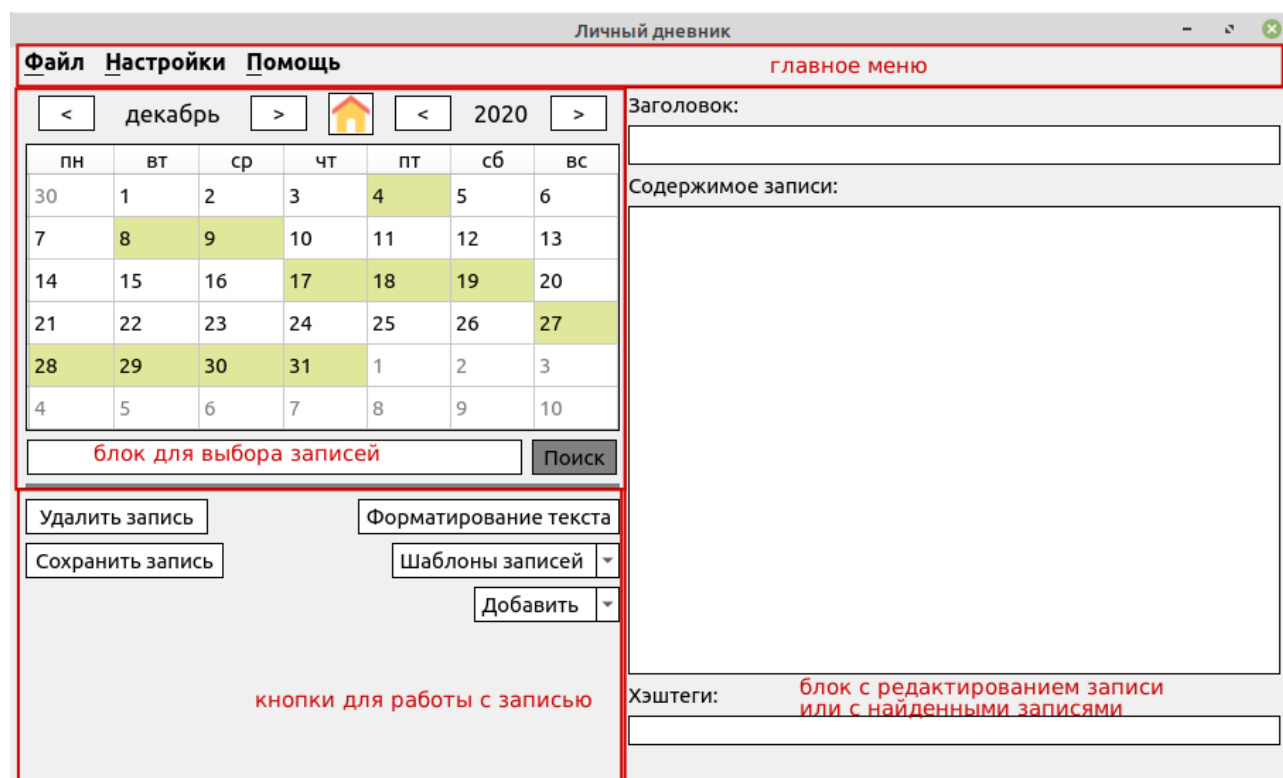


Рисунок 3.9

2. Принцип "видимость отражает полезность". Делайте часто используемые элементы управления заметными, видимыми и легко

доступными; и наоборот, прячьте или сжимайте редко используемые элементы. Пример реализации этого принципа в современных системах – панели инструментов, т.е. наборы иконок для часто используемых функций. Обоснование этого принципа заключается в том, что человек может быстро находить что-то среди небольшого числа объектов. Для редко используемых функций можно позволить поиск в длинных списках.

В моей программе более важные элементы больше по размеру. Например, кнопки, после выбора нужной даты кнопки для записей самые большие, так как, скорее всего, именно эти кнопки будет использовать пользователь.

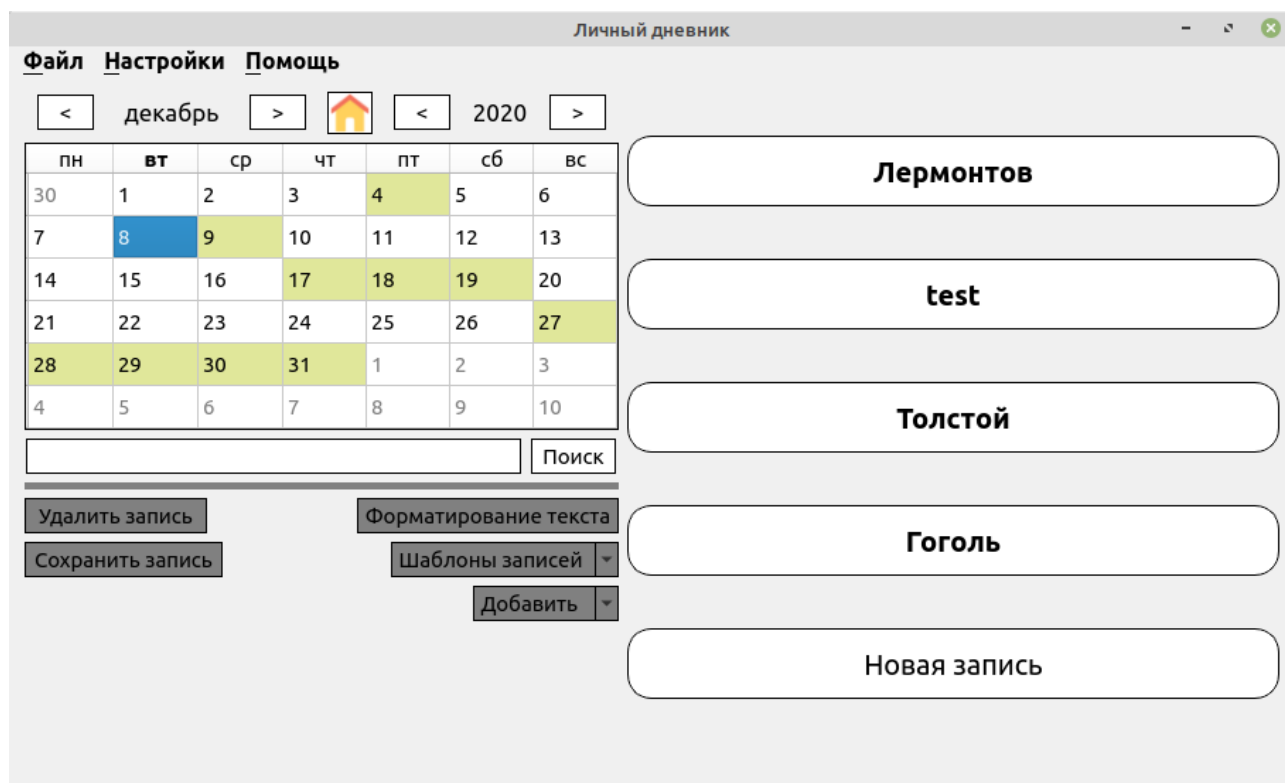


Рисунок 3.10 — отображение кнопок, которые отвечают за переход к записи

3. Принцип интеллектуальной последовательности. Используйте похожие экраны для похожих функций. Это похоже на заимствование, но в этом случае вы заимствуете что-то из одной части дизайна и применяете это к другой части. Обоснование этого принципа очевидно: раз пользователи выучили расположение элементов управления на экране (например, где находится кнопка "Помощь"), они могут легко приложить это знание к другим экранам внутри той же системы. Этот подход позволяет вам сосредоточить усилия на разработке лишь нескольких привлекательных работоспособных экранов, а затем их слегка модифицировать для использования в других частях приложения. Но экраны не должны выглядеть одинаково, если в действительности они должны отражать совершенно разные вещи. Предупреждение о критической ошибке в системе реального времени должно иметь вид, значительно отличающийся от экрана помощи или

информационного сообщения.

В моем приложении только два окна: основное и справка. Они сильно отличаются, так как у них разное предназначение.

4. Принцип "цвет как приложение". Не полагайтесь на цвет как носитель информации; используйте его умеренно, чтобы лишь акцентировать информацию, передаваемую другими средствами. Хорошее правило для большинства интерфейсов – разрабатывать их в чёрно-белом варианте, убедиться, что они работоспособны, а затем добавить минимальный цвет для их окончательного облика. Цвет, безусловно, полезен, когда необходимо выделить предупреждение или информационное сообщение, но обязательно дайте дополнительные ключи для пользователей, не способных воспринимать изменения цвета.

В моем приложении есть две цветовые темы. Первая, светлая, соответствует этому принципу — все в черно-белых тонах, кроме дней с записями и кнопки «Домой». Вторая, темная, не соответствует: **желтый цвет для всего текста, фиолетовый цвет для всех кнопок.**

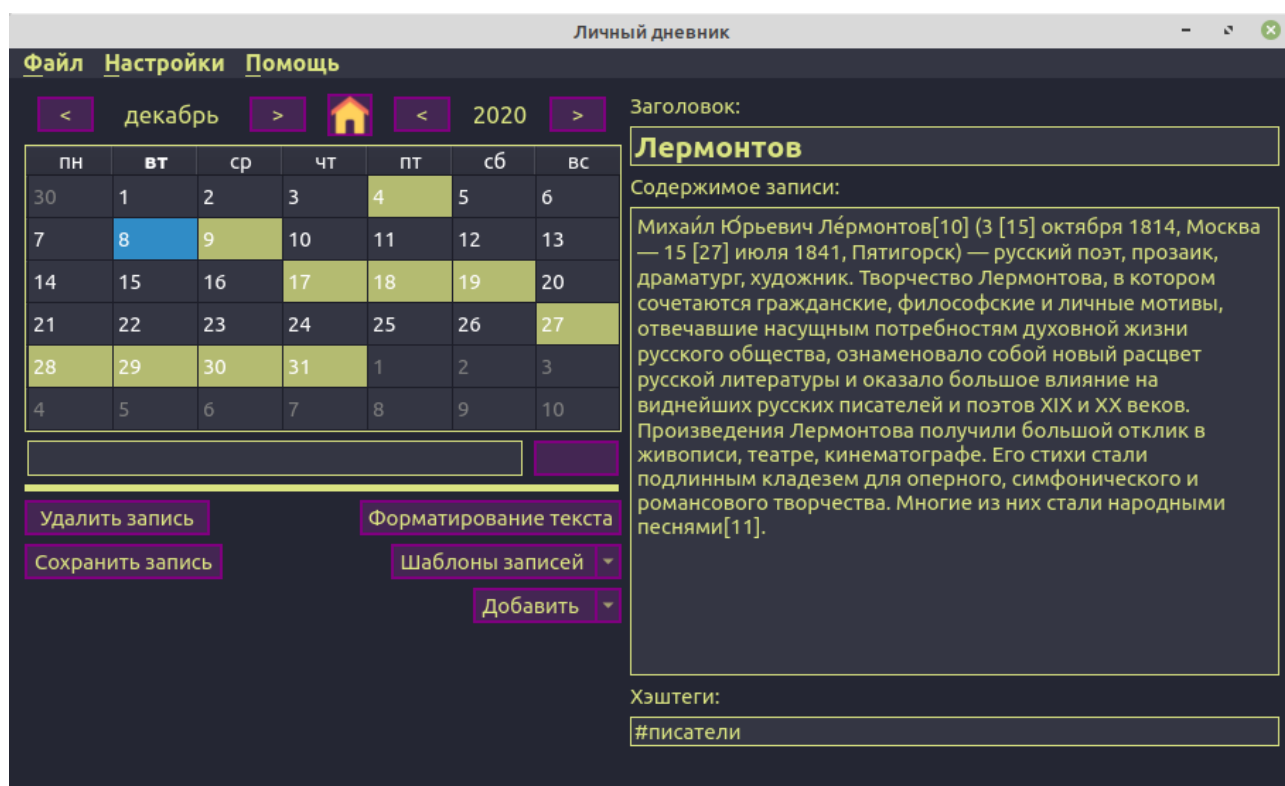


Рисунок 3.11 — темная тема

5. Принцип уменьшения беспорядка. Не помещайте на экран слишком много всего. Этот неформально определяемый принцип – хорошая проверочная точка, чтобы подтвердить, что ваш дизайн отражает перечисленные выше принципы. Если видимы только наиболее часто используемые элементы, все они сгруппированы в небольшое число

визуальных кластеров, использован минимум цвета, то экран должен получиться графически привлекательным. Не пытайтесь наделять каждое меню собственным шрифтом или работать с большим набором размеров. Как правило, пользователи заметят не столько различия, сколько беспорядок.

По-моему, мое приложение соответствует этому принципу.

Выявленные проблемы

1. Слишком много желтого и фиолетового в темной теме.

РЕШЕНИЕ ПРОБЛЕМ

Описание проблемы	Решение проблемы
1. Интерфейс программы не может сразу сообщить пользователю о том, что после использования шаблона применяется форматирование текста и как его убрать.	Можно добавить сообщение об этом внизу программы, как и для других сообщений обратной связи
2. Теги через кнопку «Добавить» добавляются только к концу текста — это неудобно для пользователя.	Это требует доработки программы
3. Нет обозначения, что поле для поиска — это поле для поиска.	Можно добавить текст серого цвета «Поле для поиска», которое автоматически скроется при вводе первой буквы
4. Неудобный поиск, можно искать только по всем записям в одном дне.	Это требует доработки программы
5. Плохая обратная связь. Текст сложно заметить.	Можно сообщения выделять ярким цветом, который постепенно исчезнет через 1-2 секунды
6. Неправильные сообщения об ошибках.	Это требует доработки программы
7. Нет обработки ошибки, когда пользователь вводит знак апострофа.	Можно запретить пользователю вводить этот символ
8. Справка не рассчитана на неопытных пользователей.	Это требует доработки справки
9. Слишком много желтого и фиолетового в темной теме.	Это требует доработки программы — сделать текст и кнопки на тон светлее фонового черного