**Министерство цифрового развития, связи и массовых коммуникаций РФ**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Сибирский государственный университет телекоммуникаций и информатики»**

**(СибГУТИ)**

КУРСОВАЯ РАБОТА

Вариант «1»

«Электронный фотоальбом»

Выполнил студент группы ИП-114 Штапов И.Н

Проверил доцент кафедры ПМиК к.т.н Мерзлякова Е.Ю.

Новосибирск \_\_\_\_\_\_г.

Новосибирск 2023

СОДЕРЖАНИЕ

[Введение 3](#_Toc153114939)

[1.Анализ задач и пользователей 4](#_Toc153114940)

[2. Выбор репрезентативных задач 5](#_Toc153114941)

[3. Заимствование 6](#_Toc153114942)

[4. Протопит интерфейса 9](#_Toc153114943)

[5. Реализация 12](#_Toc153114944)

[**CWT-Анализ** 15](#_Toc153114945)

[**GOMS-анализ** 16](#_Toc153114946)

[Список используемых источников 20](#_Toc153114947)

### Введение

В современном мире цифровых технологий, электронные фотоальбомы становятся все более популярным средством сохранения и представления важных моментов в жизни. Отражая переход от традиционных бумажных альбомов к современным цифровым форматам, электронные фотоальбомы предоставляют пользователю удобные и гибкие инструменты для хранения, организации и обмена воспоминаниями.

Данная курсовая работа нацелена на рассмотрение процесса создания электронного фотоальбома с использованием языка программирования C++ и среды разработки Qt Creator. Мы будем исследовать основные шаги проектирования и разработки, начиная с концептуального проектирования пользовательского интерфейса и заканчивая реализацией ключевых функций, таких как добавление фотографий и организация альбомов. Этот процесс позволит нам погрузиться в мир современных технологий программирования и раскрыть потенциал Qt Creator в создании инновационных приложений для обработки и представления фотографий.

### 1.Анализ задач и пользователей

Для варианта задания «Электронный фотоальбом» при рассмотрении заинтересованных лиц были выбраны люди, которые имеют опыт в работе с компьютером.

Из данной категории были выбраны сестра и брат.

1. Сестра: возраст - 30 лет, образование – высшее техническое, навыки работы с компьютером – средние.
2. Брат: возраст – 22 года, образование – среднее, навыки работы с компьютером – средние.

Пользователи программного продукта являются лицами разного пола, имеют незаконченное или полное высшее образование. Навык владения компьютером обычно на уровне базового.

Основные термины пользователя:

* Добавление фотографий – процесс переноса фотографий в определенный фотоальбом
* Удаление фотографий – процесс удаления выбранной фотографии в альбоме
* Создание нового альбома – процесс создания нового альбом и присваивание ему названия

Возможности существующих программ по схожей тематике:

* Добавление фотографий
* Удаление фотографий
* Редактирование альбомов

### Выбор репрезентативных задач

1. Создание и удаление фотоальбома c указанием его имени
2. Добавление и удаление фотографий из фотоальбома
3. Скачивание фотографий из фотоальбома на рабочий стол
4. Сортировка фотографий по дате добавления
5. Предоставление помощи пользователю по интерфейсу реализованное с помощью отдельного окна вывода в виде инструкции

### Заимствование

Приложение XnViewer:

При запуске программы в левой части интерфейса (рис. 1) отображается дерево папок. Для открытия доступны каталоги на жёстких дисках и внешних носителях, в том числе накопители специализированной техники, вроде современных фотоаппаратов.

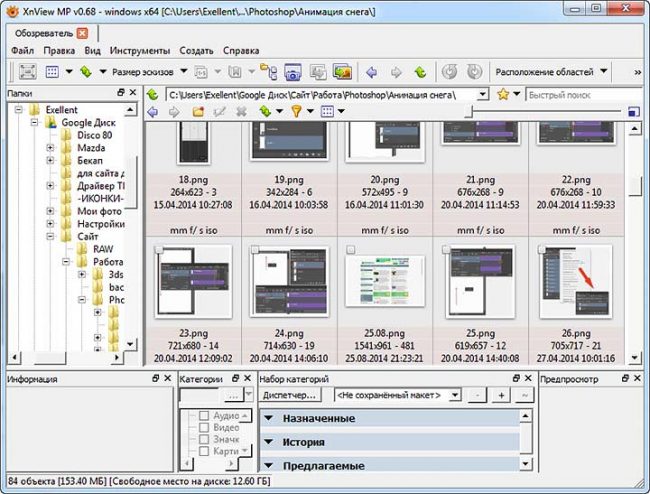


Рисунок 1. Внешний интерфейс приложения XnViewer

По клику на одной из папок в рабочей зоне появляются миниатюры фотографий с кратким описанием: имя файла, формат, развёрстка и дата. В нижней части интерфейса располагается предварительный просмотр (рис. 2).

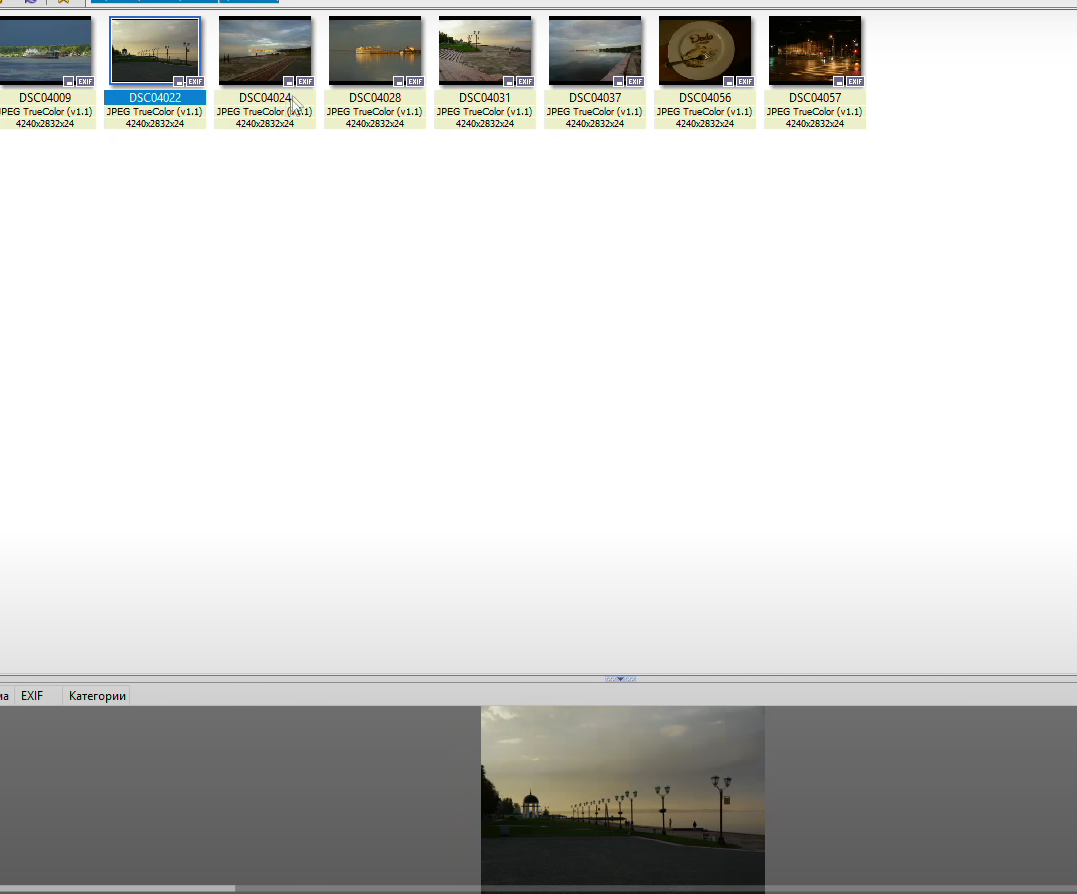


Рисунок 2. Предварительный просмотр фотографии XnViewer

Фотоальбомы Вконтакте:

Внешний вид фотоальбома во ВКонтакте представляет собой набор фотоальбомов с названием и количеством фотографий в нём(рис.3).

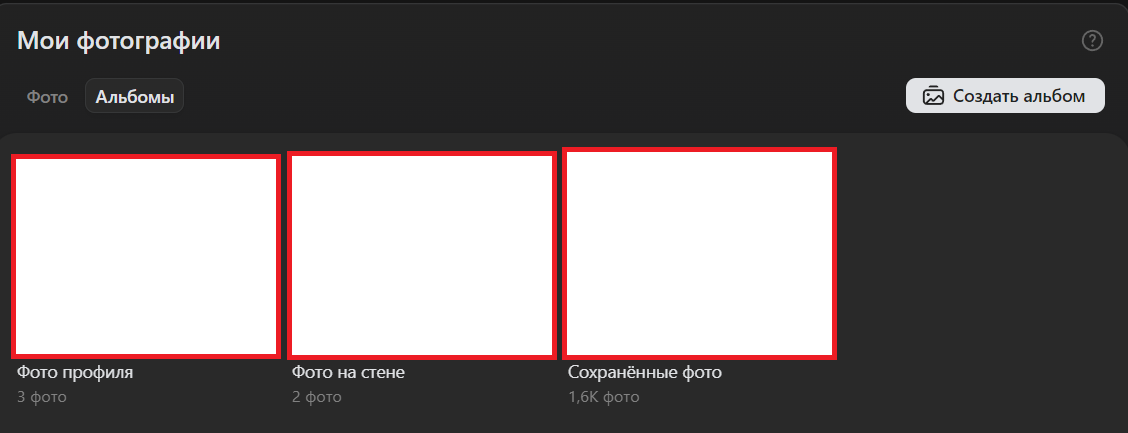


Рисунок 3. Интерфейс фотоальбомов во ВКонтакте

Так же в правом верхнем углу интерфейса можно создать новый альбом, в котором можно указать имя, описание и права доступа к альбому (рис. 4)

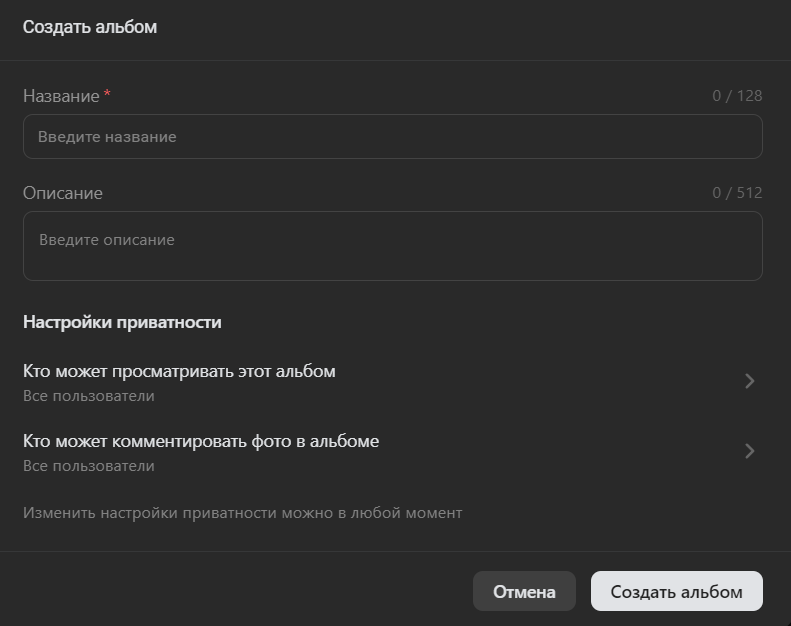


Рисунок 4. Создание фотоальбома во ВКонтакте

Альбом наполнен миниатюрами фотографий, при клике на которые они открываются на полный экран. Также эти миниатюры можно сортировать по дате добавления в правой части альбома (рис. 5)

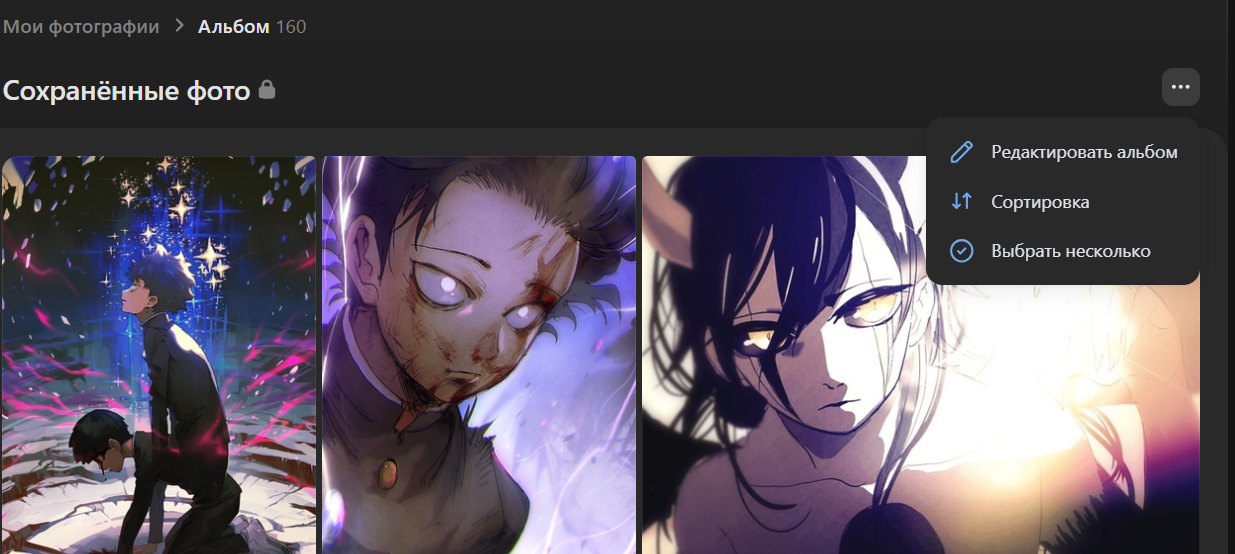


Рисунок 5. Внутренний интерфейс альбома

Также у миниатюр есть меню, в котором можно скачать, удалить фотографию или переместить её в другой альбом (рис. 6)

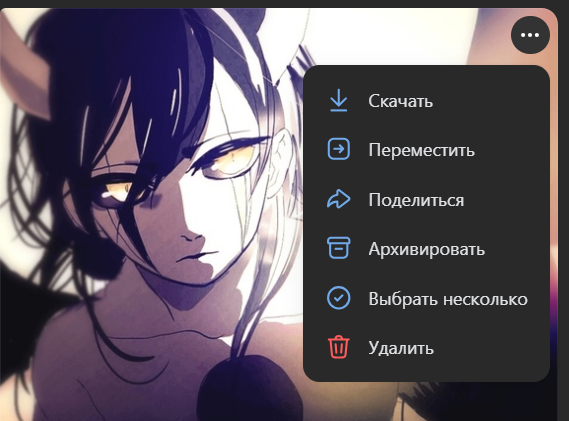


Рисунок 6. Меню работы с миниатюрой

### 4. Прототип интерфейса

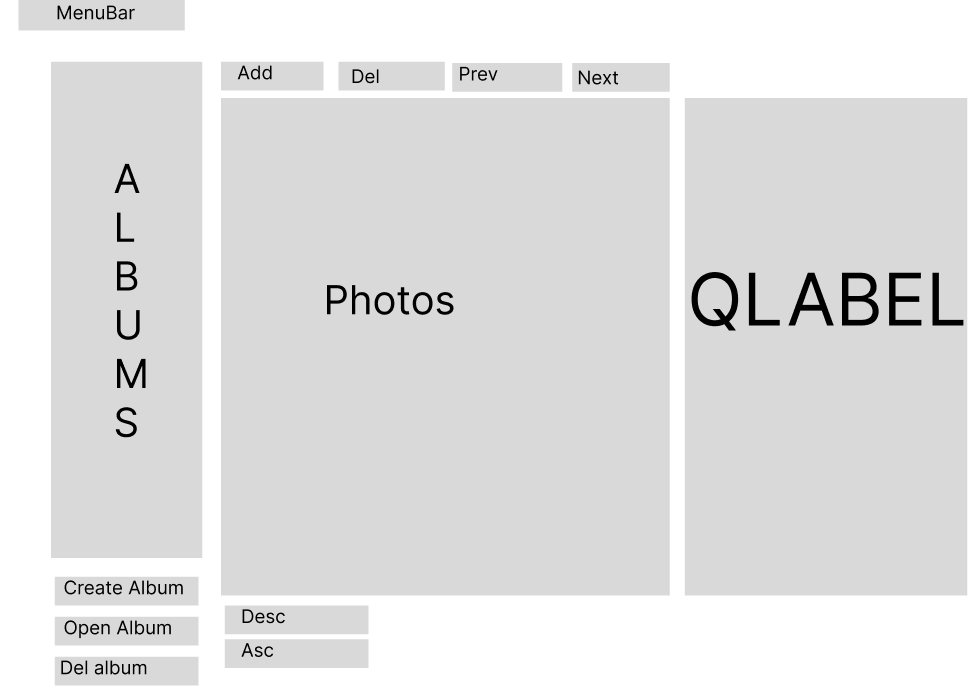


Рисунок 7. Вид главного экрана

Описание элементов интерфейса:

1. MenuBar (Меню) содержит следующий подпункты:

* Инфо – инструкция по использованию приложения (рис. 8)
* Разработчик – информация о разработчике приложения (рис. 9)

1. Albums – список доступных альбомов
2. Photos – список фотографий в альбоме
3. QLabel – окно предпросмотра фотографии
4. Add, Del, Prev, Next – кнопки управления с Photos, которые позволяют удалять, добавлять и перелистывать фотографии
5. Create, Open, Del – кнопки управления с Albums, которые позволяют удалять, создавать и открывать альбомы

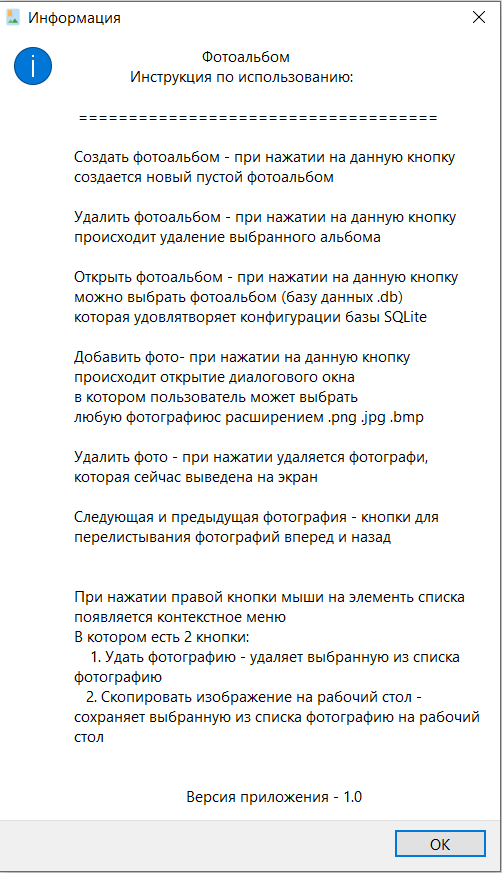


Рисунок 8. Внешний вид Инфо

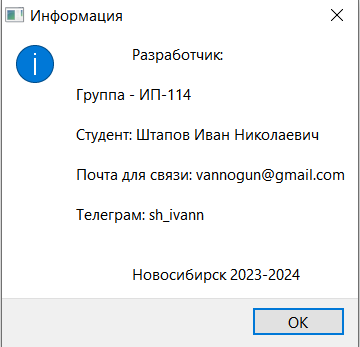


Рисунок 9. Внешний вид разработчик

Create Album – создание нового фотоальбома, в случае если название альбома превышает более 30 символов, то выводится ошибка о некорректной длине названия (рис. 10)

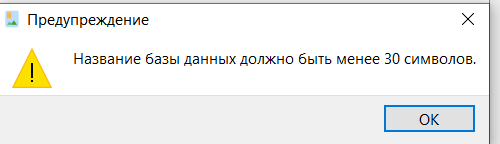


Рисунок 10 – Ошибка при превышении длины названия

### 5. Реализация

Функционал:

* Создание и удаление альбомов
* Добавление, удаление и просмотр изображений
* Скачивание изображений на рабочий стол
* Инструкция помощи с описанием возможности приложения

База данных, взаимодействующая с функционалом библиотеки QSql, основана на СУБД SQLite.

Навигатор помощи реализован с помощью файлов текстового вывода «QString xmlText»

Скриншоты работы программы

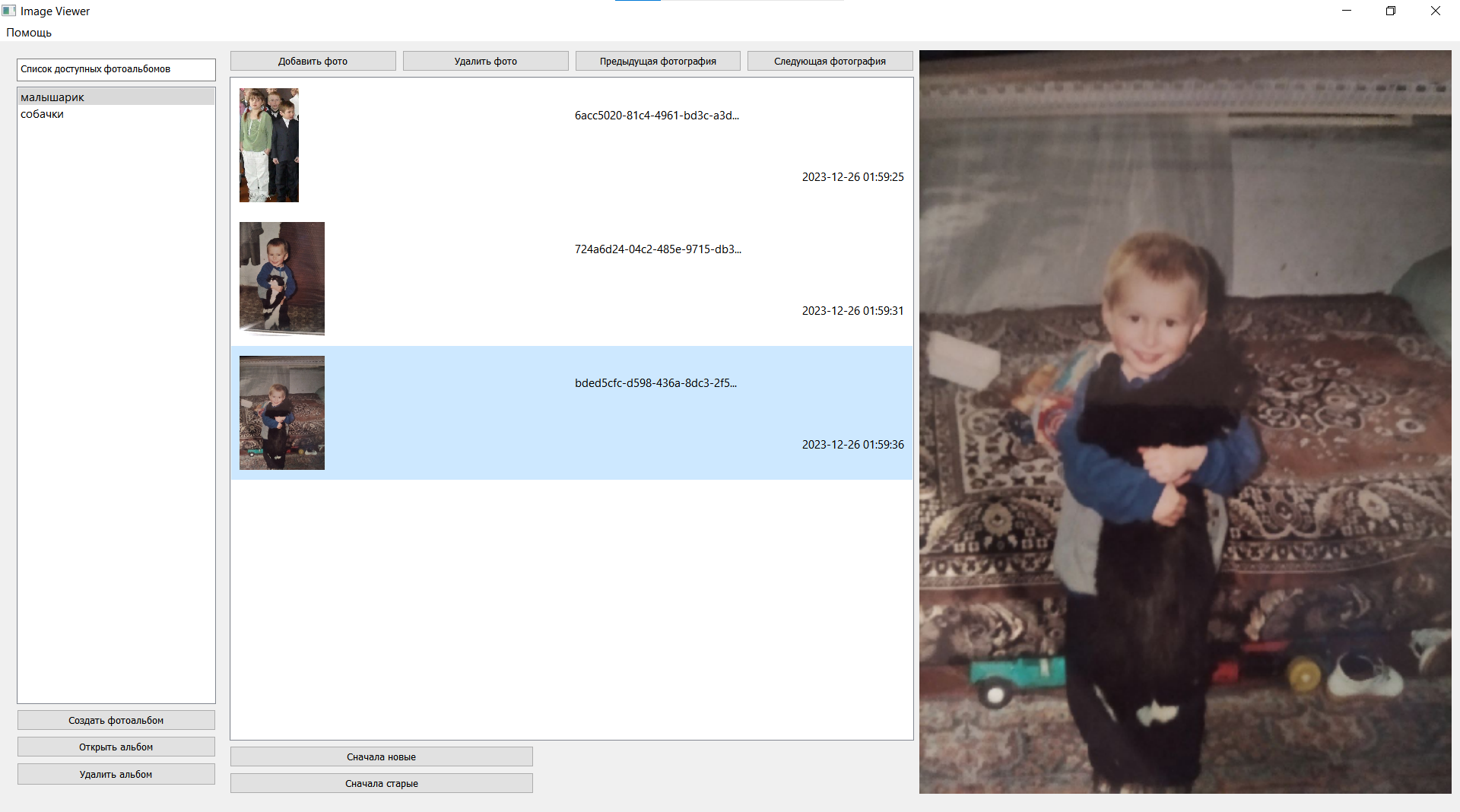


Рисунок 11. Внешний вид программы

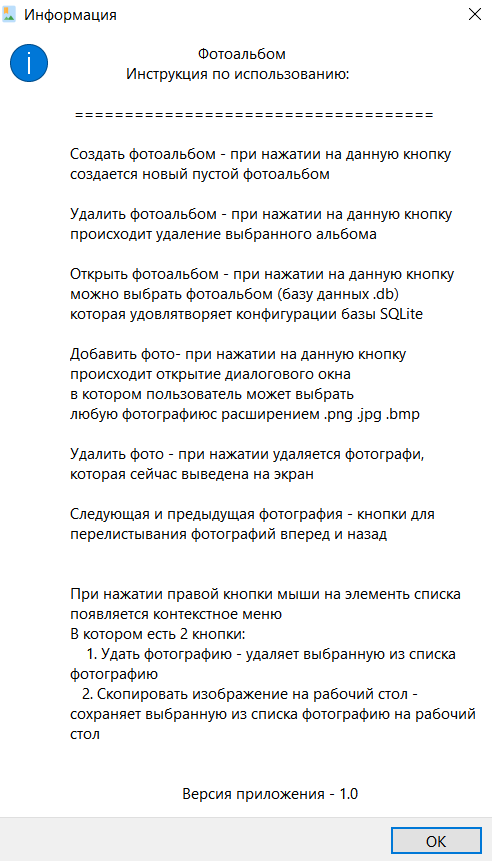


Рисунок 12. Внешний вид Инфо

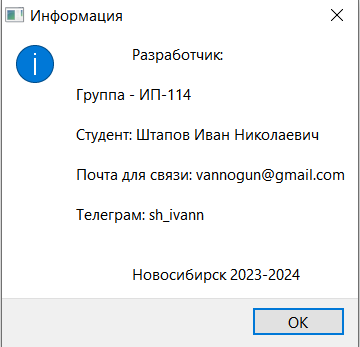


Рисунок 13. Внешний вид Разработчик

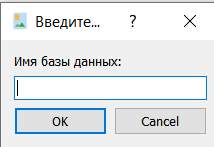


Рисунок 14. Внешний вид создания альбома

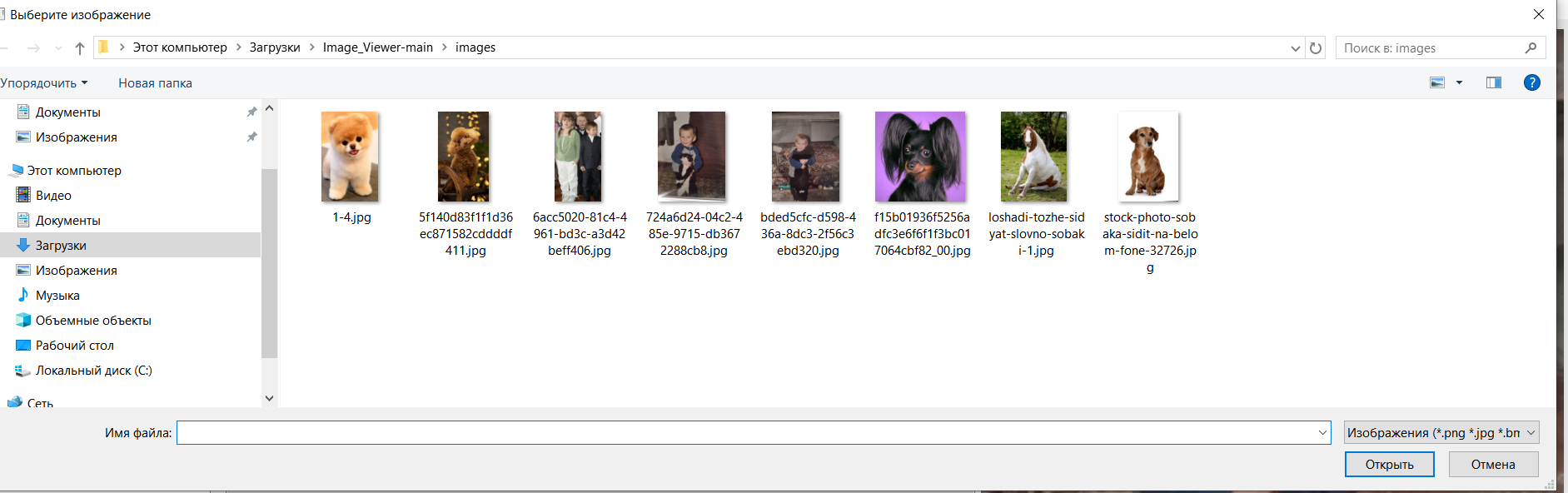


Рисунок 15. Внешний вид добавления фотографии в альбом

### **CWT-Анализ**

Задача 1: Удаление фотографии

*Список действий пользователя:*

1. Запустить приложение
2. Открыть фотоальбом
3. Выбрать фотографию
4. Нажать на кнопку «Удалить Фотографию» (или нажать правую кнопку мыши на выбранную фотографию и нажать «Удалить»)

Анализ действий задачи 1:

С первым и третьим пунктом у пользователя не должно возникнуть затруднений.

Во втором и четвертом пунктах у пользователя может возникнуть трудность с поиском доступных фотоальбомов и удалением фотографии, для решения данной проблемы было сделано меню «Инфо», в котором находится инструкция по использованию приложения.

Задача 2: Сортировка фотографий по дате добавления

*Список действий пользователя:*

1. Запустить приложение
2. Открыть фотоальбом
3. Отсортировать альбом

Анализ действий задачи 2:

C первым и вторым пунктом у пользователя не должно возникнуть трудностей.

В третьем пункте у пользователя могут возникнуть трудности, поскольку он может не знать в каком порядке отсортированы фотографии в альбоме. Чтобы понять в каком порядке отсортирован тот или иной альбом, пользователь может перейти в меню «Помощь» -> «Инфо», в котором указывается в каком порядке отсортированы альбомы.

### **GOMS-анализ**

Описание анализа

Практически все интерфейсные взаимодействия в можно описать следующими операциями:

K – нажатие клавиши;

B – клик кнопкой мыши;

P – наведение указателя мыши;

R – ожидание ответной реакции компьютера;

H – перенос руки с клавиатуры на мышь или наоборот

D – проведение с помощью мыши прямой линии (например, выделение или прокрутка текста);

M – мыслительная подготовка (к осуществлению одной из перечисленных операций).

Разные пользователи выполняют указанные операции за разное время. Однако, GOMS исследует работу опытного пользователя. Многочисленные исследования выявили средние значения времени операций, выполняемых опытными пользователями.

K 0.2 с

B 0.2 с

P 1.1 с

H 0.4 с

M 1.35 с

Цель №1: очистить весь фотоальбом, где хранится 5 фотографий

Для выполнения цели сформулируем подцели:

1. Открыть приложение фотоальбома
2. Выбрать альбом
3. Нажать на любую фотографию
4. Нажать кнопку «Удалить фотографию»

Теперь опишем методы для каждой подцели и распишем каждый метод с точностью до операции:

1. Открытие фотоальбома c рабочего стола двойным кликом

М (мыслительный процесс) – 1,35 с.

P (наведение указателя мыши на значок приложения) – 1,1 с.

BB (двойной клик по иконке приложения) – 0,4 с.

1. Выбор интересующего фотоальбома

М (мыслительный процесс) – 1,35 с.

P (наведение указателя мыши на альбом) – 1,1 с.

B (клик мыши на название альбома) – 0,2 с.

1. Нажатие на любую фотографию в альбоме

М (мыслительный процесс) – 1,35 с.

P (наведение указателя мыши на фотографию) – 1,1 с.

B (клик мыши на фотографию) – 0,2 с.

1. Нажимаем на кнопку «Удалить фотографию» 5 раз

М (мыслительный процесс) – 1,35 c.

P (наведение указателя мыши на кнопку) – 1,1 с.

5B (нажатие на кнопку 5 раз) – 1 с.

Общее время: M + P + BB + M + P + B + M + P + B + M + P + 5B = 11,6 с.

Цель №2: создание фотоальбома

Для выполнения цели сформулируем подцели:

1. Открыть приложение фотоальбома
2. Нажать на кнопку «Создать фотоальбом»
3. Ввести название альбома (например «Альбом»)
4. Нажать на кнопку «Ок»

Теперь опишем методы для каждой подцели и распишем каждый метод с точностью до операции:

1. Открытие фотоальбома c рабочего стола двойным кликом

М (мыслительный процесс) – 1,35 с.

P (наведение указателя мыши на значок приложения) – 1,1 с.

BB (двойной клик по иконке приложения) – 0,4 с.

1. Нажатие на кнопку «Создать фотоальбом»

М (мыслительный процесс) – 1,35 с.

P (наведение указателя мыши на кнопку) – 1,1 с.

B (клик мыши по кнопке) – 0,2 с.

1. Ввод названия альбома

М (мыслительный процесс) – 1,35 с.

P (наведение указателя мыши на поле ввода) – 1,1 с.

B (клик мыши по полю ввода) – 0,2 с.

6K (нажатие клавиш) – 1,2 с.

1. Нажатие на кнопку «Ок»

М (мыслительный процесс) – 1,35 с.

P (наведение указателя мыши на кнопку) – 1,1 с

B (клик мыши на кнопку) – 0,2 с.

Общее время: M + P + BB + M + P + B + M + P + B + 6K + M + P + B = 12 c.

### Список используемых источников

* Учет клиентов – программа, с которой был позаимствован интерфейс;

URL: [https://prostoysoft.ru/SportSchool.htm](https://prostoysoft.ru/SportSchool.htm%20)

* Qt Documentation – официальная документация по разработке в среде Qt;

URL: <https://doc.qt.io/>

### Приложение

|  |  |
| --- | --- |
| 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  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55 | **void** MainWindow::on\_addPhotoButton\_clicked()  {  **if** (database.isOpen()) {  QString filePath;  **if** (!fileDialogShown) {  filePath = QFileDialog::getOpenFileName(**this**, tr("Выберите изображение"), "", tr("Изображения (\*.png \*.jpg \*.bmp)"));  fileDialogShown = true;  } **else** {  fileDialogShown = false;  }  **if** (!filePath.isEmpty()) {  QDir imagesDir(QCoreApplication::applicationDirPath() + "/images");  **if** (!imagesDir.exists()) {  imagesDir.mkpath(".");  }  QString destinationPath = imagesDir.absoluteFilePath(QFileInfo(filePath).fileName());  **int** suffix = **1**;  **while** (QFile::exists(destinationPath)) {  destinationPath = imagesDir.absoluteFilePath(QFileInfo(filePath).completeBaseName() + "\_" + QString::number(suffix++) + "." + QFileInfo(filePath).suffix());  }  **if** (QFile::copy(filePath, destinationPath)) {  // Очищаем список путей  photoPaths.clear();  photoPaths.append(destinationPath);  QSqlQuery query;  query.prepare("INSERT INTO photos (path, date\_added) VALUES (:path, CURRENT\_TIMESTAMP)");  query.bindValue(":path", destinationPath);  **if** (!query.exec()) {  QMessageBox::warning(**this**, tr("Ошибка базы данных"), tr("Не удалось добавить фото в базу данных."));  // Откатываем добавление в случае ошибки  photoPaths.clear();  QFile::remove(destinationPath);  } **else** {  // Обновляем listWidget с новыми путями  loadImagesFromDatabase();  }  } **else** {  QMessageBox::warning(**this**, tr("Ошибка"), tr("Не удалось скопировать изображение."));  }  }  } **else** {  QMessageBox::critical(**this**, tr("Ошибка базы данных"), tr("База данных не открыта."));  }  } |

Функции удаления:

|  |  |
| --- | --- |
| 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  32  33  34  35  36  37  38  39  40  41 | **void** MainWindow::on\_deletePhotoButton\_clicked()  {  **if** (photoPaths.isEmpty() || currentPhotoIndex >= photoPaths.size()) {  ui->imageLabel->clear();  **return**;  }  QString currentPhotoPath = photoPaths[currentPhotoIndex];  **if** (database.isOpen() && !currentPhotoPath.isEmpty()) {  QSqlQuery query;  query.prepare("DELETE FROM photos WHERE path = :path");  query.bindValue(":path", currentPhotoPath);  **if** (!query.exec()) {  QMessageBox::warning(**this**, tr("Ошибка базы данных"), tr("Не удалось удалить запись из базы данных."));  }  }  }  **void** MainWindow::onDeleteAlbumButtonClicked()  {  **if** (!database.isOpen()) {  qDebug() << "База данных не открыта.";  **return**;  }  QString currentDatabaseName = database.databaseName();  database.close();  QFile **file**(currentDatabaseName);  **if** (file.remove()) {  qDebug() << "База данных успешно удалена:" << currentDatabaseName;  updateDatabaseList();  ui->imageLabel->clear();  ui->listWidget->clear();  updateButtons();  } **else** {  qDebug() << "Ошибка при удалении базы данных:" << file.errorString();  }  } |

Основная загрузка фотоальбомов:

|  |  |
| --- | --- |
| 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  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52 | **void** MainWindow::loadImagesFromDatabase()  {  qDebug() << "Loading images from the database";  **if** (!database.isOpen()) {  qDebug() << "Database is not open!";  **return**;  }  photoPaths.clear();  QSqlQuery **query**(database);  query.prepare("SELECT path, date\_added FROM photos");  **if** (!query.exec()) {  qDebug() << "Error executing query:" << query.lastError().text();  **return**;  }  ui->listWidget->clear();  **while** (query.next()) {  QString path = query.value("path").toString();  QString dateAdded = query.value("date\_added").toString();  QPixmap **pixmap**(path);  QWidget \*customWidget = **new** QWidget;  QHBoxLayout \*layout = **new** QHBoxLayout(customWidget);  QLabel \*iconLabel = **new** QLabel;  QLabel \*dateLabel = **new** QLabel(dateAdded);  iconLabel->setPixmap(pixmap.scaled(**150**, **150**, Qt::KeepAspectRatio));  layout->addWidget(iconLabel);  QVBoxLayout \*verticalLayout = **new** QVBoxLayout;  QLabel \*nameLabel = **new** QLabel(QFileInfo(path).fileName());  **if** (nameLabel->text().length() > **30**) {  QString shortName = nameLabel->text().left(**27**) + "...";  nameLabel->setText(shortName);  nameLabel->setToolTip(QFileInfo(path).fileName())  }  verticalLayout->addWidget(nameLabel, **0**, Qt::AlignLeft);  verticalLayout->addWidget(dateLabel, **0**, Qt::AlignRight);  layout->addLayout(verticalLayout);  QListWidgetItem \*item = **new** QListWidgetItem;  item->setSizeHint(customWidget->sizeHint());  ui->listWidget->addItem(item);  ui->listWidget->setItemWidget(item, customWidget);  photoPaths.append(path);  }  **if** (!photoPaths.isEmpty()) {  currentPhotoIndex = **0**;  currentPhotoPath = photoPaths[**0**];  ui->listWidget->setCurrentRow(currentPhotoIndex);  displayPhoto(currentPhotoPath);  updateButtons();  }  } |