**Департамент образования и науки города Москвы**

**Государственное бюджетное общеобразовательное учреждение города Москвы**

**«Школа № 1501»**

**ПРИЛОЖЕНИЕ-ПОМОЩНИК В УЧЁБЕ «UniHelp»**

Авторы: Чагаев Егор Алексеевич,

Ильясов Тимур Ильгизович

учащиеся 10 «Б» класса

ГБОУ Школы № 1501

**г. Москва**

**2023–2024 учебный год**

**Оглавление**

Введение 3

Описание теоретической работы над созданием Desktop-приложения

1. Анализ потребностей учащихся 4
2. Анализ источников информации 5
3. Анализ аналогов 7
4. Промежуточный вывод 8

Описание реализации практической части создания Desktop-приложения

1. Программирование действующего прототипа 9
2. Испытание работоспособности прототипа 11

Заключение 12

Список литературы 13

Приложение 1 15

Приложение 2 17

**Введение**

Актуальность:Самой проблемы отсутствия инструментов, помогающих в учёбе, нет. Но неприятная деталь заключается в том, что эти инструменты не систематизированы: некоторые инструменты не работают без интернета, другие нужно отдельно скачивать, а остальные сильно ограничены в функционале. Например, встроенный калькулятор в ОС Windows крайне ограничен в сфере перевода в разные системы счисления: максимальное число с которым он может работать - FFFF FFFF FFFF FFFF (16 букв F), дальше он выдаёт ошибку. «Desmos» не работает без интернета, а учебники приходится скачивать с неофициальных сайтов.

Проблема:Часто школьники сталкиваются с необходимостью использовать несколько различных программ для выполнения учебных задач, построения графиков, вычислений и решения уравнений. Это может приводить к снижению скорости работы ученика и к снижению эффективности обучения. Создание единой многофункциональной программы на Python позволит объединить эти возможности в одном приложении, что повысит уровень удобства и эффективности учебного процесса

Цель работы:Создать интегрированную программу-помощник на Python, которая объединит функции калькулятора, построения графиков, решения уравнений и предоставит доступ к учебным материалам, с целью улучшения и упрощения процесса обучения

Задачи:

1. Проанализировать потребности учащихся в программных инструментах для учёбы;
2. Спроектировать архитектуру программы с учётом требуемых функций;
3. Реализовать следующие модули программы:
   1. Калькулятор для выполнения различных вычислений;
   2. Инструмент для построения графиков математических функций;
   3. Модуль для решения алгебраических уравнений;
   4. Библиотека для хранения и доступа к электронным учебникам;
4. Провести тестирование программы и оптимизировать её работу.

**Описание теоретической работы над созданием Desktop-приложения**

**I. Анализ потребностей учащихся**

Перед выбором библиотек, на которых должно быть написано приложение, было необходимо составить опросник для учащихся, с целью добавления нужных функций в приложение-помощник. Дальше приведены вопросы и результаты опроса:

Как часто вы сталкиваетесь с трудностями при выполнении математических расчётов и задач?

Каждый день 45%

Несколько раз в неделю 40%

Редко 15%

Испытываете ли вы сложности с доступом к учебным материалам (учебники, конспекты, формулы)?

Да, часто 62%

Иногда 28%

Никогда 10%

Нуждаетесь ли вы в специализированных калькуляторах для выполнения учебных задач?

Да, часто 67%

Иногда 20%

Нет 13%

Какие инструменты вам чаще всего не хватает для эффективного изучения сложных тем?

Инструменты для пошагового решения задач 45%

Графическое представление функций (графики) 25%

Доступ к библиотеке учебников и справочникам 30%

**II. Анализ источников информации**

Произведя анализ источников информации по теме создания Desktop-приложений на Python, были найдены различные пути реализации проекта «UniHelp». В нижеприведённой таблице представлен результат анализа различных библиотек для написания приложений на языке программирования Python.

Таблица 1

Преимущества и недостатки библиотек и фреймворков на Python для написания графического интерфейса приложения

|  |  |  |
| --- | --- | --- |
| **Название библиотеки/фреймворка** | **Преимущества** | **Недостатки** |
| tkinter | * Интегрирована в Python * Кроссплатформенность * Большое сообщество | * Устаревший интерфейс * Ограниченный функционал |
| Kivy | * Простота * Кроссплатформенность | * Ограниченность функционала * Собственный язык разметки |
| PyQt6 | * Простота использования * Встроенное решение для создания дизайна приложений * Поддержка CSS и XML * Кроссплатформенность * Большое сообщество * Высокая производительность | * Сложность обучения * Несовместимость с некоторыми модулями Python |
| PySimpleGUI | * Наличие многих цветовых тем для приложений * Кроссплатформенность | * Необходимость регистрации для разработки приложений * Сложно интегрировать дизайн |

Проанализировав все преимущества и недостатки данных библиотек и фреймворков (см. список литературы 1-4), было принято решение использовать самую функциональную, универсальную библиотеку PyQt6.

**III. Анализ аналогов**

Изучив сферу помощников в учёбе в интернете были найдены следующие проекты:

* 1. Интернет-сайт «Desmos»;
  2. Мобильное приложений «Photomath»;
  3. Интернет-сайт «Библиотека МЭШ».

На данный момент каждый из данных проектов специализируется в своей собственной сфере: «Desmos» - построение сложных графиков, «Photomath» - решение алгебраических уравнений, «Библиотека МЭШ» - доступ к различным учебным материалам. Стоит отдельно проанализировать каждый проект и выделить их плюсы и минусы, чтобы в дальнейшем интегрировать хорошие функции в наше приложение.

«Desmos» предоставляет почти неограниченный функционал построение как простых, так и сложных графиков. Т. е. в качестве инструмента для построения графиков «Desmos» является одним из лучших на рынке, но единственным минусом этого проекта является отсутствие его оффлайн-приложения для систем Linux и Windows.

«Photomath» используется для пошагового решения алгебраических уравнений разной сложности. Но «Photomath» исключительно мобильное приложение, значит к нему нет доступа, если у ученика под рукой нет телефона. Также одним из минусов этого приложения является наличие платной подписки. Ученики, пользующееся бесплатной версией, могут увидеть только краткое пояснение переходов между шагами решения, как в это же время в платную подписку входит функция дополнительных объяснений.

«Библиотекой МЭШ» пользуются большинство учеников Москвы для получения доступа к учебным материалам. Главным преимуществом данной библиотеки является простая и удобная система поиска учебников. Но, как в ситуации с «Desmos», для доступа к библиотеке требуется стабильное подключение к сети Интернет. Это можно было решить, если бы учебники можно было скачивать напрямую на персональные компьютеры учеников, но такой функции не предусмотрено.

**IV. Промежуточный вывод**

Наше Desktop-приложение было решено писать с помощью библиотеки PyQt6 из-за её удобного использования, наличия необходимых функций и высокой производительности.

Необходимый функционал модулей:

* Калькулятор
  + Высокая производительность;
* Построение графиков
  + Доступ даже при отсутствии подключения к интернету;
* Решение уравнений
  + Удобность использования;
* Библиотека учебников
  + Доступ к учебникам при отсутствии подключения к интернету.

Также все модули программы должны иметь удобный интерфейс и приятную для глаза цветовую гамму, потому что большинство учеников много времени проводят за компьютером.

**Описание теоретической работы над созданием Desktop-приложения**

**I. Программирование действующего прототипа**

Программирование данного приложения должно пройти в несколько ключевых этапов: создание структуры папок проекта, разработка схемы работы приложения, выбор среды разработки, создание форм, изменение дизайна форм, добавление функций для кнопок на этих формах.

1. Создание структуры папок проекта.

Для начала работы над проектом нужно было сразу продумать его структуру и схему работы. Было решено сделать стандартную структуру (см [рис. 1](#Рис_1)) и схему работы проекта. Приложение запускается через файл «main.py», находящийся в корневой папке проекта, к которому подключаются остальные файлы из папки «data». В папке «forms» находятся формы модулей приложения и их представления в формате «.py». В папке «db» находится файл локальной базы данных учебников. По ходу разработки приложения структура менялась, но незначительно.

2. Разработка схемы работы приложения.

После создания структуры папок проекта нужно было перейти к разработке схемы работы приложения. Используя программу Microsoft Visio, был составлен черновик данной схемы (см [схему 1](#Схема_1)). При разработке функций нашего приложения мы всё время придерживались данной схемы.

3. Выбор среды разработки.

Чтобы начать разработку прототипа Desktop-приложения нужно определиться с выбором среды разработки: главными вариантами были «Pycharm Community» от JetBrains и «Visual Studio Code» от Microsoft. Проведём исследование данных сред разработки и прийдём к выводу, какую стоит использовать.

Таблица 2

Выбор среды разработки

|  |  |  |
| --- | --- | --- |
| **Критерий** | **Pycharm Community Edition** | **Visual Studio Code** |
| Опыт работы | Егор: 2 года  Тимур: 1.5 года | Егор: менее полугода  Тимур: менее полугода |
| Работа с языком Python | * Подсветка синтаксиса * Подсказки типов переменных * Подсказки при написании кода * Удобная отладка | * Подсветка синтаксиса * Подсказки типов переменных |
| Работа с git и github | Интуитивно понятный интерфейс git | Наличие инструментов работы git |
| Удобные функции | Наличие функции «external tools». В нашем случае это пригодилось для конвертации форм из файлов с расширением «.ui» в файлы с расширением «.py» | - |

В итоге сравнения этих сред разработок, было решено, что нужно использовать Pycharm Community Edition.

4. Создание форм.

Для удобного создания форм мы воспользовались вспомогательной библиотекой pyqt6-tools. В ней есть отдельное приложение «Qt Designer» (см [рис. 2](#Рис_2)), в котором можно удобно создавать формы: не писать их код вручную, а в удобном интерфейсе располагать виджеты на форме.

5. Изменение дизайна форм.

Для изменения дизайна форм в PyQt предусмотрен вариант использования CSS. В «Qt Designer» также есть окно изменения стилей ([рис. 3](#Рис_3)).

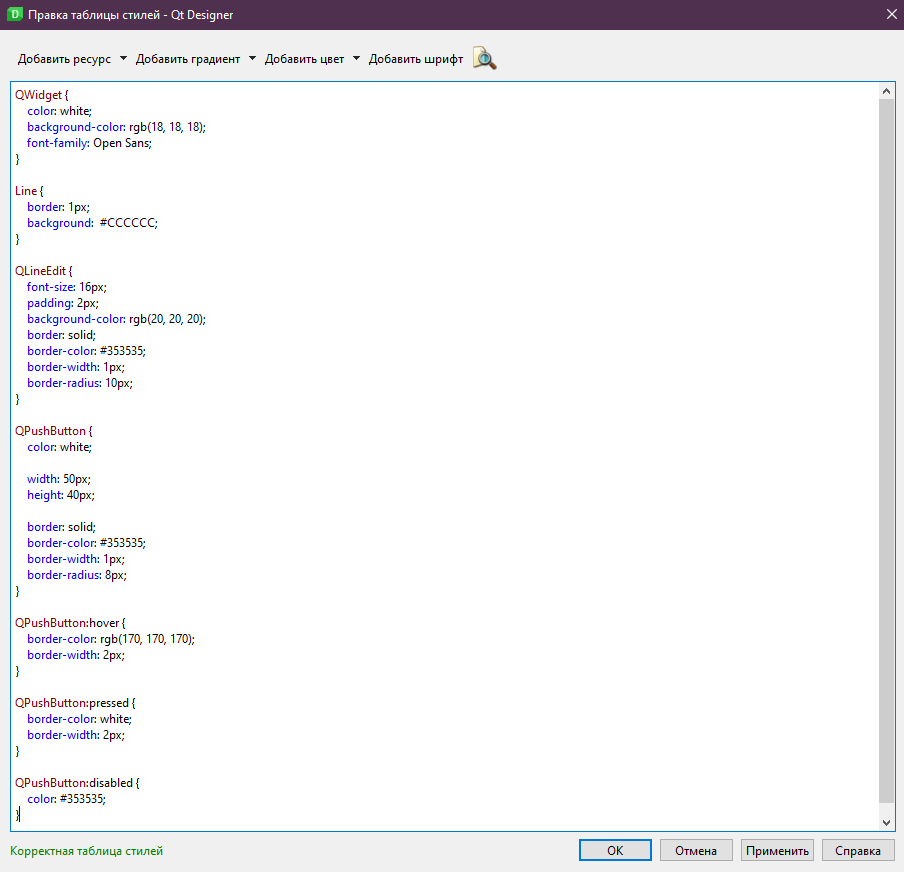


Рис. 3. Изменение дизайна формы в «Qt Designer»

6. Добавление функций к кнопкам на формах.

Добавление функций к кнопкам рассмотрим на примере добавления любого модуля. Сначала для каждой кнопки модуля нужно добавить обработчик нажатий, чтобы при клике вызывался соответствующий метод (см [рис. 4](#Рис_4)).

При вызове метода программа понимает, на какую из кнопок нажали и далее вызывает соответственное окно (см [рис. 5](#Рис_5)).

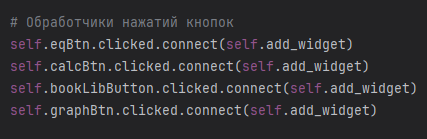


Рис. 4. Обработчики нажатий на кнопки.

**Список литературы**

----

**Приложение 1**

----

**Приложение 2**

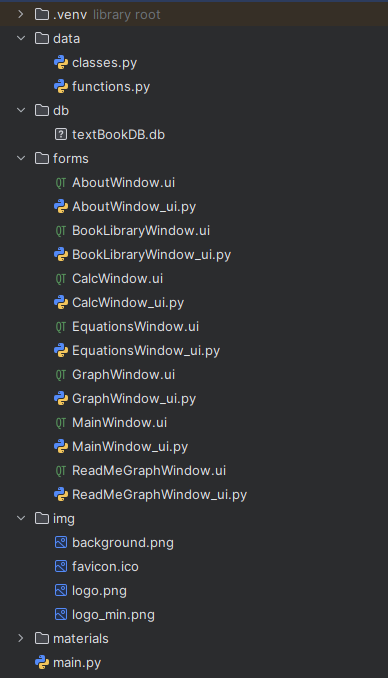


Рис. 1. Структура папок проекта

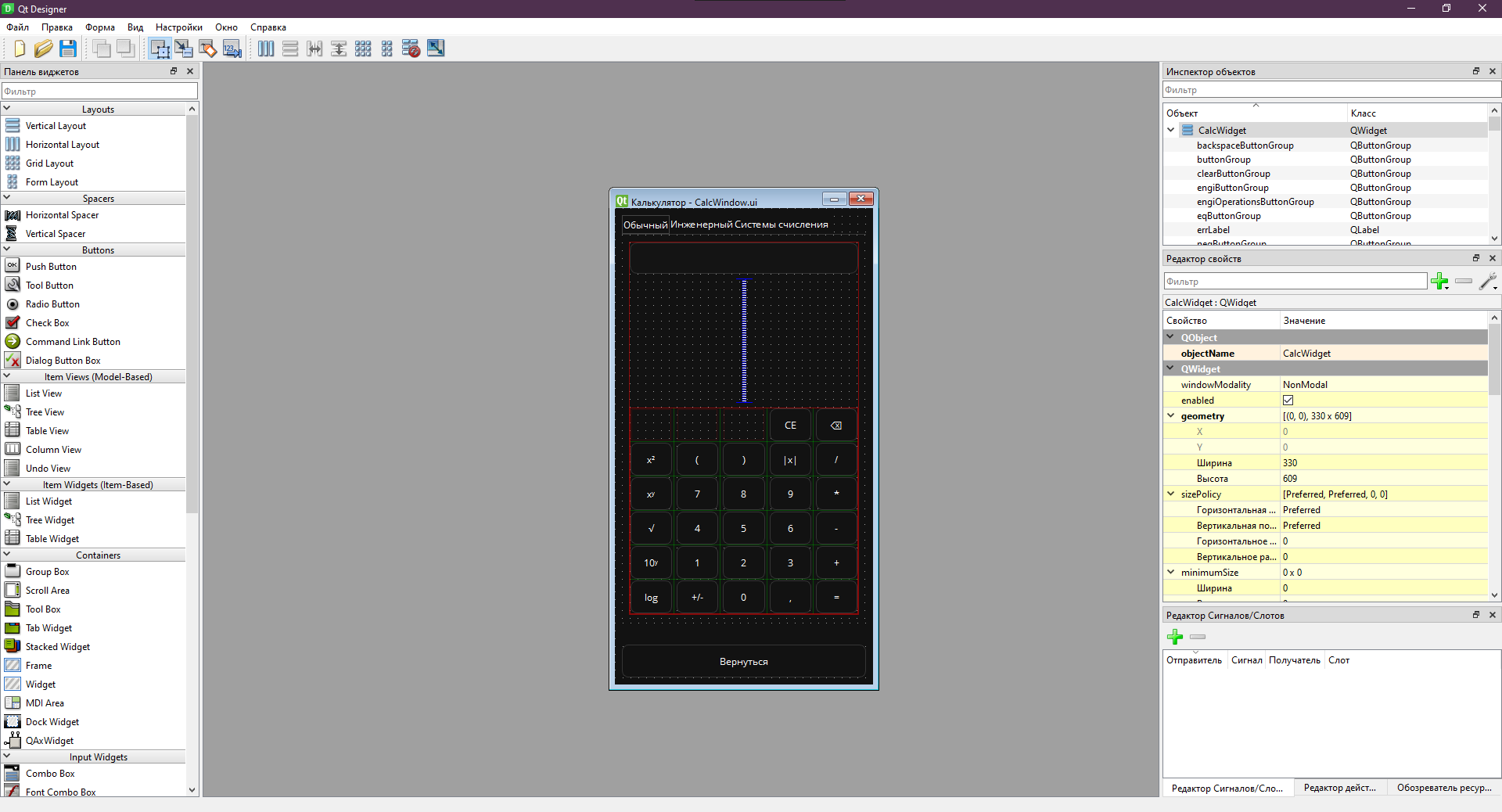


Рис 2. Интерфейс «Qt Designer»

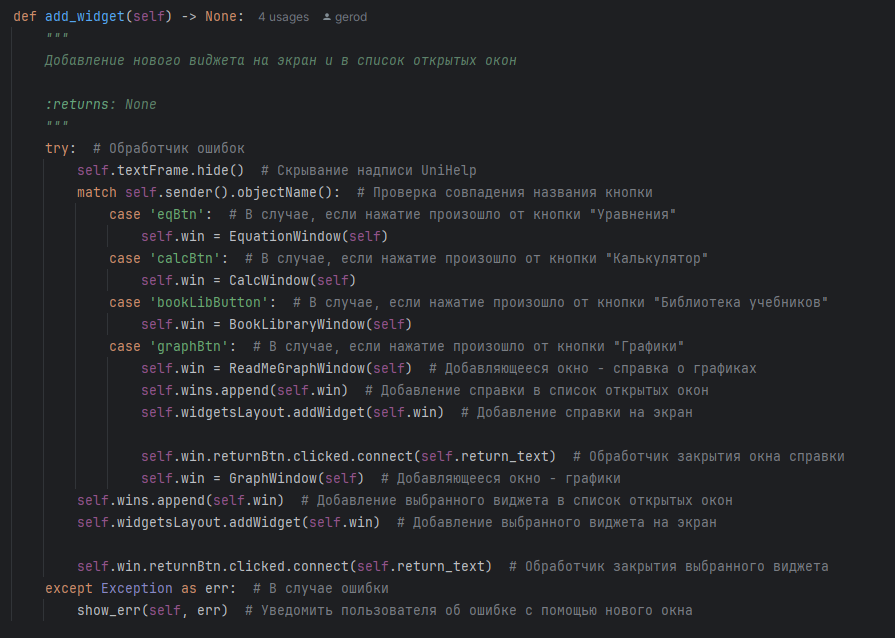


Рис. 5. Добавление виджета на экран.

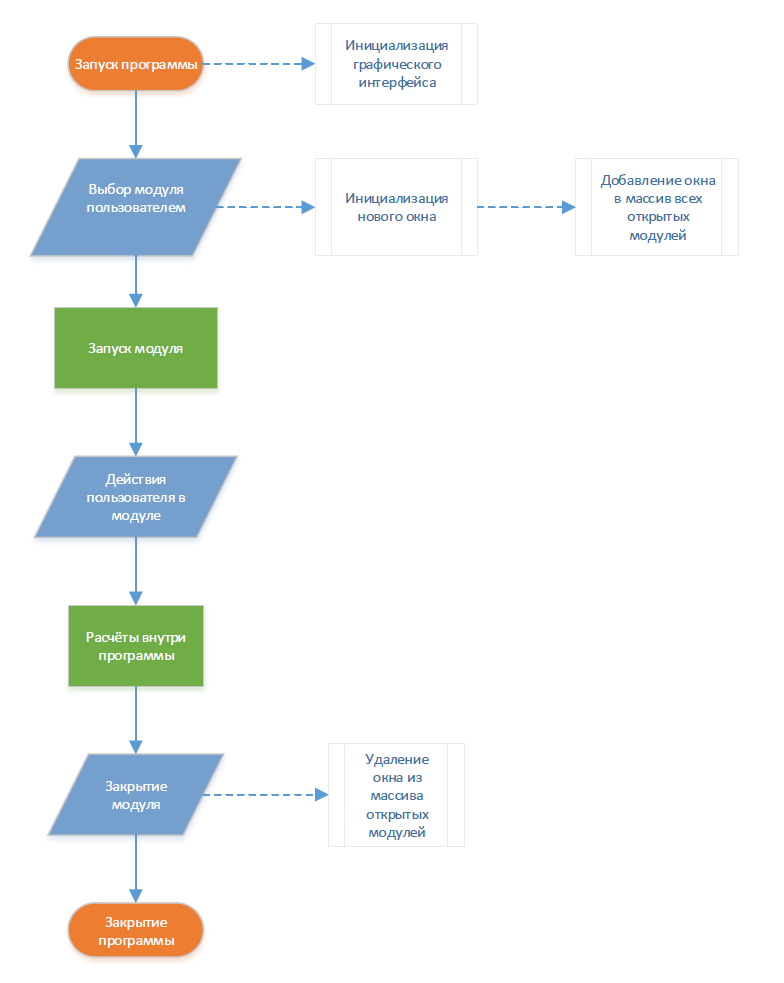


Схема 1. Схема работы приложения