**4.Реализация**

**4.1 Выбор языка программирования**

Python является одним из самых популярных и простых языков программирования, который используется для создания различных приложений. Он обладает большим количеством библиотек и инструментов, которые позволяют быстро и эффективно разрабатывать программы.

Для создания приложения для поиска лекарств Python подходит по следующим причинам:

1. Простота. Python имеет простой и понятный синтаксис, который позволяет разработчикам быстро освоить язык и начать создавать программы. Это особенно важно для начинающих разработчиков, которые только начинают свой путь в программировании.
2. Эффективность. Python обладает высокой производительностью и может использоваться для создания сложных приложений, таких как приложение для поиска лекарств. Он также имеет хорошую оптимизацию памяти, что позволяет создавать приложения, которые не занимают много ресурсов компьютера.
3. Большое количество библиотек. Для Python существует множество библиотек, которые облегчают разработку различных приложений. Например, для создания приложения для поиска лекарств можно использовать библиотеку Django, которая предоставляет инструменты для работы с базами данных и веб-разработки.
4. Открытый исходный код. Python является открытым программным обеспечением, что означает, что его исходный код доступен для всех желающих. Это позволяет разработчикам изучать код и использовать его для создания своих приложений.
5. Широкая поддержка сообщества Python имеет большое сообщество разработчиков, которое помогает новым пользователям освоить язык и решить возникающие проблемы. Это делает процесс разработки более комфортным и эффективным.
6. Кросс-платформенность. Python может работать на разных операционных системах, что делает его идеальным выбором для создания кросс-платформенных приложений, таких как приложение для поиска лекарств.

**Выбор IDE**

PyCharm является одной из самых популярных интегрированных сред разработки (IDE) для языка программирования Python. Она предоставляет широкий спектр возможностей для разработки, такие как автодополнение кода, анализ кода на ошибки, отладка, визуализация данных и многие другие.

Выбор PyCharm для создания приложения для поиска лекарств обоснован следующими преимуществами:

* Удобство разработки: PyCharm имеет интуитивно понятный интерфейс, который упрощает процесс создания приложения. Она предлагает множество функций, которые ускоряют разработку, таких как автодополнение кода и анализ ошибок.
* Поддержка сообщества: PyCharm имеет активное сообщество пользователей и разработчиков, которые помогают решать возникающие проблемы и предлагают новые идеи для улучшения продукта.
* Широкий функционал: PyCharm поддерживает множество инструментов и библиотек для Python, что позволяет разработчикам легко создавать сложные приложения.
* Скорость работы: PyCharm работает быстро и эффективно, что важно для разработчиков, работающих над большими проектами.

Таким образом, PyCharm является отличным выбором для создания приложения для поиска лекарств, так как она предоставляет все необходимые инструменты для быстрой и эффективной разработки.

**5.1 Написание кода**

Для начала необходимо импортировать нужные библиотеки для работы приложения такие как:

Для этого в Python библиотека импортируются с помощью ключевого слова import.

1. Библиотека sys в Python предоставляет функциональность для взаимодействия с интерпретатором Python и операционной системой.

Она используется для:

* Работы с аргументами командной строки: sys.argv позволяет получить список аргументов командной строки, переданных при запуске программы.
* Управления окружением выполнения: sys.path возвращает список директорий, в которых интерпретатор Python ищет модули при импорте, sys.modules содержит словарь уже импортированных модулей.
* Взаимодействия с операционной системой: sys.exit() завершает выполнение программы, sys.platform возвращает информацию об операционной системе, на которой выполняется код.
* Работы с потоками ввода-вывода: sys.stdin, sys.stdout и sys.stderr представляют стандартные потоки ввода-вывода и могут быть перенаправлены.
* Получения информации о текущем исполняемом коде: sys.executable указывает на путь к интерпретатору Python, sys.version возвращает информацию о версии интерпретатора.
* Управления памятью и выполнением гарбичной сборки: sys.getsizeof() позволяет получить размер объекта в байтах, sys.getrefcount() возвращает количество ссылок на объект.
* Это лишь некоторые из возможностей, предоставляемых модулем sys. Он является одним из стандартных модулей Python и широко используется для различных задач, связанных с интерфейсом между Python и операционной системой.

2. Библиотека PyQt5.QtWidgets предоставляет набор графических элементов пользовательского интерфейса (GUI) для создания приложений с помощью PyQt5. С ее помощью можно создавать окна, кнопки, поля ввода, списки, меню и другие элементы интерфейса, а также управлять их расположением на экране. Библиотека QtWidgets также позволяет обрабатывать события взаимодействия пользователя с интерфейсом, такие как нажатие кнопок или изменение значений полей. Кроме того, она поддерживает стилизацию виджетов с помощью каскадных таблиц стилей (CSS) и многоязыковую поддержку. В целом, библиотека PyQt5.QtWidgets является основной составляющей для создания графического пользовательского интерфейса в PyQt5.

3. Библиотека SQLite3 - это библиотека встраиваемой базы данных, которая позволяет создавать и управлять базами данных SQLite. SQLite3 является самодостаточной и независимой от сервера базой данных, что означает, что она не требует отдельного сервера баз данных для своего функционирования. Она обладает множеством преимуществ и широко используется в различных программных приложениях и встроена во множество операционных систем, таких как Android, iOS и многих других для хранения и доступа к данным.

**База данных для работы приложения**

SQLite - это мощная, компактная и самоуправляемая система управления базами данных (СУБД). Она представляет собой надежный и относительно быстрый способ организации хранения и извлечения данных. Основные особенности SQLite:

Компактность: SQLite представляет собой небольшую библиотеку, которую можно встроить в приложение или использовать как отдельный сервер базы данных. Это позволяет эффективно использовать ресурсы системы.

Нет необходимости в сервере: SQLite не требует установки и настройки отдельного сервера базы данных. Пользователь просто может создать файл базы данных и начать работу с ним.

Поддержка стандарта SQL: SQLite полностью соответствует стандарту SQL, включая большинство его расширений. Это означает, что вам доступны мощные возможности по запросам, фильтрации и сортировке данных.

Автономная работа: SQLite поддерживает полностью автономную работу, что означает, что приложение может продолжать работать даже при отсутствии подключения к базе данных.

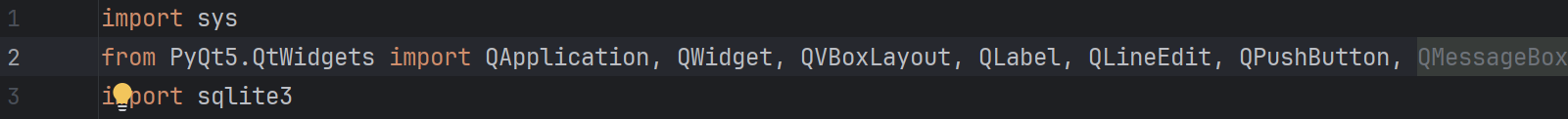
Многопользовательская поддержка: SQLite поддерживает одновременный доступ к базе данных из нескольких потоков или процессов. Он использует механизмы блокировки, чтобы гарантировать целостность данных.

Кросс-платформенность: SQLite доступен для большинства платформ - Windows, macOS, Linux, iOS, Android и многих других.

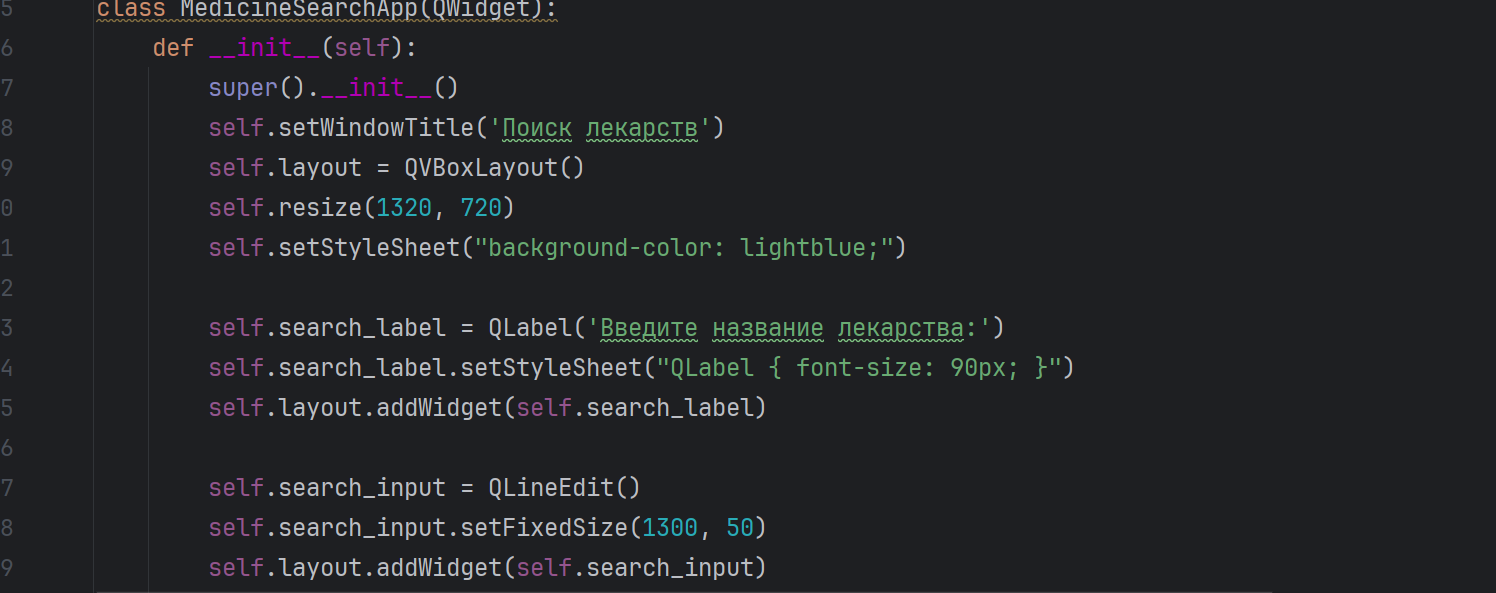
Функционал SQLite включает в себя:

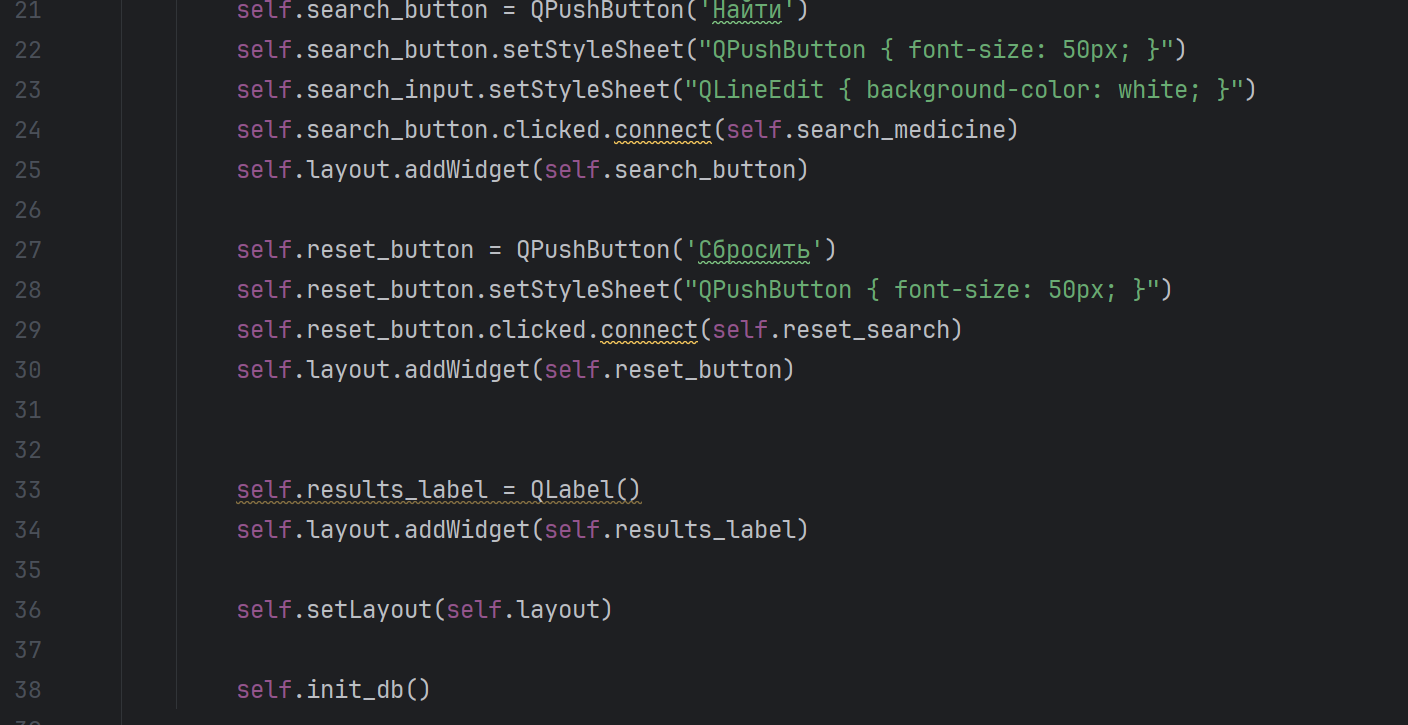
* Создание и управление базами данных с помощью языка SQL.
* Выполнение запросов к базам данных для получения, вставки, обновления и удаления данных.
* Поддержка индексов для ускорения выполнения запросов.
* Транзакционная обработка данных для обеспечения целостности и надежности.
* Поддержка различных типов данных: целые числа, числа с плавающей запятой, строки, даты и другие.
* Встроенный механизм шифрования для защиты данных.
* Поддержка хранимых процедур и триггеров для автоматизации выполнения определенных операций.
* Возможность работы с базами данных в оперативной памяти или на диске.
* Поддержка репликации данных для создания резервных копий или синхронизации данных между несколькими устройствами.

SQLite - это полноценная СУБД, которая предлагает множество возможностей для эффективной работы с данными в различных типах приложений.

****

После установки библиотек пишем код для приложения с графическим интерфейсом:



****

Этот код представляет собой графическое приложение для поиска лекарств. Класс MedicineSearchApp наследуется от QWidget и представляет основное окно приложения.

В конструкторе класса происходит инициализация оформления окна (заголовок, размер, цвет фона) и создание всех необходимых виджетов (метки, поле ввода, кнопки).

Метка search\_label отображает текст "Введите название лекарства" с использованием большого шрифта.

Поле ввода search\_input предназначено для ввода названия лекарства и имеет фиксированный размер.

Кнопка search\_button имеет надпись "Найти" и большой шрифт. При нажатии на эту кнопку вызывается метод search\_medicine, который выполняет поиск лекарства.

Кнопка reset\_button имеет надпись "Сбросить" и также вызывает связанный с ней метод reset\_search при нажатии.

Метка results\_label предназначена для отображения результатов поиска.

Компоновщик QVBoxLayout используется для упорядочивания и размещения всех виджетов в окне приложения.

Метод setLayout устанавливает компоновщик в качестве основного макета приложения.

Метод init\_db инициализирует базу данных, которая будет используется для поиска лекарств.

def init\_db(self):  
 conn = sqlite3.connect('medicines.db')  
 c = conn.cursor()  
 c.execute('''CREATE TABLE IF NOT EXISTS search\_history  
 (search\_query text)''')  
  
 c.execute('''CREATE TABLE IF NOT EXISTS medicines  
 (name text, description text)''')  
  
 c.execute("INSERT INTO medicines VALUES ('Аспирин', 'Противовоспалительное и обезболивающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Парацетамол', 'Обезболивающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Ибупрофен', 'Противовоспалительное и жаропонижающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Нурофен', 'Противовоспалительное и жаропонижающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Диклофенак', 'Противовоспалительное средство')")  
 c.execute("INSERT INTO medicines VALUES ('Но-шпа', 'Спазмолитик и обезболивающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Ибупрофен', 'Противовоспалительное и анальгетик')")  
 c.execute("INSERT INTO medicines VALUES ('Лоратадин', 'Противоаллергическое средство')")  
 c.execute("INSERT INTO medicines VALUES ('Цетиризин', 'Противоаллергическое средство')")  
 c.execute("INSERT INTO medicines VALUES ('Фенистил', 'Противоаллергическое средство')")  
 c.execute("INSERT INTO medicines VALUES ('Ранитидин', 'Препарат против язвы желудка и двенадцатиперстной кишки')")  
 c.execute("INSERT INTO medicines VALUES ('Омепразол', 'Противоязвенное средство')")  
 c.execute("INSERT INTO medicines VALUES ('Панкреатин', 'Ферментный препарат для улучшения пищеварения')")  
 c.execute("INSERT INTO medicines VALUES ('Активированный уголь', 'Сорбент для отравлений и расстройств пищеварения')")  
 c.execute("INSERT INTO medicines VALUES ('Лоперамид', 'Противопроливное средство')")  
 c.execute("INSERT INTO medicines VALUES ('Линекс', 'Пробиотик для нормализации микрофлоры кишечника')")  
 c.execute("INSERT INTO medicines VALUES ('Амоксициллин', 'Антибиотик группы пенициллинов')")  
 c.execute("INSERT INTO medicines VALUES ('Азитромицин', 'Антибиотик группы макролидов')")  
 c.execute("INSERT INTO medicines VALUES ('Цефтриаксон', 'Антибиотик группы цефалоспоринов')")  
 c.execute("INSERT INTO medicines VALUES ('Активированный уголь', 'Сорбент для отравлений и расстройств пищеварения')")  
 c.execute("INSERT INTO medicines VALUES ('Лоперамид', 'Противопроливное средство')")  
 c.execute("INSERT INTO medicines VALUES ('Линекс', 'Пробиотик для нормализации микрофлоры кишечника')")  
 c.execute("INSERT INTO medicines VALUES ('Амоксициллин', 'Антибиотик группы пенициллинов')")  
 c.execute("INSERT INTO medicines VALUES ('Азитромицин', 'Антибиотик группы макролидов')")  
 c.execute("INSERT INTO medicines VALUES ('Цефтриаксон', 'Антибиотик группы цефалоспоринов')")  
 c.execute("INSERT INTO medicines VALUES ('Альфацин', 'Антибиотик группы тетрациклинов')")  
 c.execute("INSERT INTO medicines VALUES ('Флюконазол', 'Противогрибковое средство')")  
 c.execute("INSERT INTO medicines VALUES ('Нистатин', 'Противогрибковое средство')")  
 c.execute("INSERT INTO medicines VALUES ('Метронидазол', 'Противопротозойное и противомикробное средство')")  
 c.execute("INSERT INTO medicines VALUES ('Ацетилцистеин', 'Муколитическое средство')")  
 c.execute("INSERT INTO medicines VALUES ('Фенилэфрин', 'Сосудосуживающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Ксилометазолин', 'Сосудосуживающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Глицерин', 'Осмолитическое средство для очистки носа')")  
 c.execute("INSERT INTO medicines VALUES ('Нифедипин', 'Препарат для снижения артериального давления')")  
 c.execute("INSERT INTO medicines VALUES ('Метопролол', 'Бета-адреноблокатор для снижения артериального давления')")  
 c.execute("INSERT INTO medicines VALUES ('Атенолол', 'Бета-адреноблокатор для снижения артериального давления')")  
 c.execute("INSERT INTO medicines VALUES ('Эналаприл', 'Препарат для снижения артериального давления и лечения сердечной недостаточности')")  
 c.execute("INSERT INTO medicines VALUES ('Лизиноприл', 'Препарат для снижения артериального давления и лечения сердечной недостаточности')")  
 c.execute("INSERT INTO medicines VALUES ('Симастатин', 'Липидснижающее средство')")  
 c.execute("INSERT INTO medicines VALUES ('Феназепам', 'Препарат для снятия тревожности и нервного напряжения')")  
  
 conn.commit()  
 conn.close()  
  
 def search\_medicine(self):  
 medicine\_name = self.search\_input.text()  
 conn = sqlite3.connect('medicines.db')  
 c = conn.cursor()  
  
 c.execute("SELECT \* FROM medicines WHERE name=?", (medicine\_name,))  
 result = c.fetchone()  
 c.execute("INSERT INTO search\_history VALUES (?)", (medicine\_name,))  
 conn.commit()  
  
 if result:  
 self.results\_label.setText(f"Название: {result[0]}\nОписание: {result[1]}")  
 else:  
 self.results\_label.setText('Лекарство не найдено')  
  
 conn.close()  
  
 def reset\_search(self):  
 self.search\_input.clear()  
 self.results\_label.setText('')  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app = QApplication(sys.argv)  
 window = MedicineSearchApp()  
 window.show()  
 sys.exit(app.exec\_())

Этот код представляет собой реализацию поискового приложения для медицинских препаратов.

Основные функции кода:

1. **init\_db**: Создает базу данных "medicines.db" и таблицы "search\_history" и "medicines". Заполняет таблицу "medicines" данными о различных лекарственных препаратах.
2. **search\_medicine**: Извлекает название препарата из текстового поля поиска. Подключается к базе данных "medicines.db" и выполняет запрос на выборку данных о препарате с заданным названием из таблицы "medicines". Если результат выборки не пустой, выводит название и описание препарата на экран. Если результат выборки пустой, выводит сообщение "Лекарство не найдено". Записывает название препарата в таблицу "search\_history".
3. **reset\_search**: Очищает текстовое поле поиска и результаты поиска.
4. Блок кода **if \_\_name\_\_ == '\_\_main\_\_':** создает экземпляр приложения **MedicineSearchApp**, отображает его на экране и запускает цикл обработки событий приложения.

**4.2. Дизайн и интерфейс.**

В верхней части окна располагается панель с названием "Поиск лекарств" и кнопками для управления окном (закрыть, развернуть или свернуть). Ниже находится заголовок "Введите название лекарства:", выполненный большими буквами. Под заголовком расположено поле для ввода текста, где пользователь может ввести название интересующего его лекарства. Под полем ввода находятся две кнопки: "Найти", с помощью которой можно начать поиск введенного лекарства, и "Сбросить", нажатие на которую скорее всего очистит поле ввода от введенного текста или сбросит результаты текущего поиска. Дизайн интерфейса минималистичный, цвета спокойные: синий, серый и белый.

