Департамент образования и науки города Москвы

Государственное бюджетное профессиональное образовательное учреждение

города Москвы

«Колледж малого бизнеса № 4»

(ГБПОУ КМБ № 4)

**КУРСОВОЙ ПРОЕКТ**

по МДК 05.02 Разработка кода информационных систем, МДК 05.03 Тестирование информационных систем

Базовая подготовка

Тема: Разработка информационной системы для инвентарного учета компьютерной техники

Выполнил(а) студент(ка)

2 курса группы № ИПС-21.22

Штыков Никита Сергеевич

(Фамилия, имя, отчество студента)

Проверил

руководитель курсовой работы

Минеева Кристина Константиновна

Работа защищена с оценкой

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024

Москва 2024

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc167368835)

[ГЛАВА 1. Теоретическая часть программного обеспечения 6](#_Toc167368836)

[**1.1** **Понятие, актуальные виды и особенности программного обеспечения** 6](#_Toc167368837)

[1.2 Понятие, Актуальные виды и особенности информационных систем 8](#_Toc167368838)

[**1.3** **Определение Visual Studio Code** 12](#_Toc167368839)

[**1.4** **Функции и особенности приложения Visual Studio Code** 14](#_Toc167368840)

[Глава 2. Практическая часть создания проекта 17](#_Toc167368841)

[**2.1 Обоснование выбора программы для разработки информационной системы** 17](#_Toc167368842)

[**2.2 Определение языков программирования, используемых в проекте.** 19](#_Toc167368843)

[**2.3 Определение выбора метода для разработки.** 24](#_Toc167368844)

[**2.4 Поэтапная разработка информационной системы** 27](#_Toc167368845)

[ЗАКЛЮЧЕНИЕ 34](#_Toc167368846)

[ИСПОЛЬЗУЕМАЯ ЛИТЕРАТУРА 36](#_Toc167368847)

[ПРИЛОЖЕНИЕ 38](#_Toc167368848)

# **ВВЕДЕНИЕ**

В современном информационном обществе, где компьютерная техника стала неотъемлемой частью рабочего процесса в организациях, вопрос эффективного управления этими активами становится все более актуальным. Разработка информационной системы для инвентарного учета компьютерной техники представляет собой важное направление в области информационных технологий, направленное на оптимизацию бизнес-процессов и повышение операционной эффективности предприятия.

Процесс разработки подобной информационной системы включает в себя несколько ключевых этапов. Первоначально проводится анализ текущих процессов учета компьютерной техники в организации, выявляются основные потребности и требования заказчика. Затем осуществляется проектирование базы данных, которая будет хранить всю необходимую информацию об инвентаре компьютерной техники, включая модели, серийные номера, местонахождение и статус каждого устройства.

Следующим этапом является разработка программного обеспечения, включающего в себя интерфейс пользователя для удобного ввода и отображения данных, а также функциональность для управления инвентарем, генерации отчетов и мониторинга состояния оборудования.

Важным аспектом процесса разработки информационной системы является обеспечение безопасности данных, включая защиту от несанкционированного доступа и резервное копирование информации.

Наконец, после завершения разработки системы производится ее внедрение в организацию, обучение персонала и организация поддержки и дальнейшего сопровождения.

Актуальность разработки информационной системы для инвентарного учета компьютерной техники обусловлена несколькими факторами. Во-первых, современные предприятия все больше полагаются на компьютерную технику для выполнения своих задач, и эффективное управление этими активами становится ключевым фактором в обеспечении бесперебойной работы бизнеса. Во-вторых, с ростом числа компьютеров и других устройств на предприятии становится все сложнее отслеживать их местонахождение, статус и общую информацию. Такая система позволяет автоматизировать процесс учета и облегчить работу с данными об инвентаре. Кроме того, в современном информационном обществе важно обеспечить безопасность и конфиденциальность данных, что также становится более сложной задачей без специализированной информационной системы.

Актуальность разработки информационной системы для инвентарного учета компьютерной техники обусловлена несколькими факторами. Во-первых, современные предприятия все больше полагаются на компьютерную технику для выполнения своих задач, и эффективное управление этими активами становится ключевым фактором в обеспечении бесперебойной работы бизнеса.

Во-вторых, с ростом числа компьютеров и других устройств на предприятии становится все сложнее отслеживать их местонахождение, статус и общую информацию. Такая система позволяет автоматизировать процесс учета и облегчить работу с данными об инвентаре. Кроме того, в современном информационном обществе важно обеспечить безопасность и конфиденциальность данных, что также становится более сложной задачей без специализированной информационной системы.

Основная цель курсовой работы – разработка высококачественной информационной системы для упрощения работы предприятия. В связи с поставленной целью необходимо решить следующие задачи: рассмотреть предметную область, рассмотреть языки программирования Python; рассмотреть программу Visual Studio Code для разработки приложения, базы данных, для внедрения информационной системы

Для достижения данной цели в работе решаются следующие задачи:

* изучение понятия «база данных» и «информационная система»;
* рассмотреть принцип работы информационной системы в конкретной области;
* описание программы для создания информационной системы для управления;
* разработка программы для создания информационной системы для управления.

Практическая значимость работы состоит в том, что разработанная информационная система будет способствовать повышению эффективности и безопасности управления компьютерной техникой на предприятии.

Выполненная работа состоит из введения, основной части из двух разделов (теоретической и практической), заключения и списка литературы.

# **ГЛАВА 1. Теоретическая часть программного обеспечения**

Данная глава представляет собой описание существующих и актуальных видов программного обеспечения, особенности программных средств разработки и программирования.

* 1. **Понятие, актуальные виды и особенности программного обеспечения**

Программное обеспечение — программа или множество программ, используемых для управления компьютером. Имеются и другие определения из международных и российских стандартов: совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.

Цель программного обеспечения — осуществление управления электронными и механическими частями компьютера. Программное обеспечение – это весьма гибкая система, имеющая возможность усовершенствования. Программный продукт, использующий информацию, превращает аппаратную часть ПК в комплекс, с помощью которого возможно решать самые разнообразные задачи.

Жизненный цикл любого компьютерного программного обеспечения состоит из трех основных стадий: написание, работа и поддержка.

Сам процесс написания условно подразделяется на 6 этапов:

* постановка задачи.
* создание концепции.
* выработка алгоритма.
* формирование и компиляция.
* выявление ошибок и отладка.
* завершение процедур по оформлению документации.

Особенности ПО:

* Практически любым процессом можно управлять определённым набором инструкций.
* Затруднение разработки состоит в сложности его оцифровки.
* Создание нового ПО осуществляется на ПК с установленным программным обеспечением.
* Написание кода менее трудоёмкая операция, чем его отладка и настройка управления.
* Программное обеспечение пользователя является методом реализации поставленных задач.

Можно выделить два основных вида программного обеспечения:

Это комплекс программ, которые обеспечивают управление компонентами компьютерной системы, такими как процессор, оперативная память, устройства ввода-вывода, сетевое оборудование, выступая как «межслойный интерфейс», с одной стороны которого аппаратура, а с другой — приложения пользователя. Системное программное обеспечение управляет всеми ресурсами ЭВМ (центральным процессором, памятью, вводом-выводом) и реализовывает общую организацию процесса обработки информации и интерфейсы между ЭВМ, пользователем, аппаратными и программными средствами. Оно разрабатывается так, чтобы компьютер мог эффективно выполнять прикладные программы.

* Прикладное программное обеспечение:

Это программы, предназначенные для решения задач пользователя. Их также называют приложениями. Этот тип используемого программного обеспечения обладает огромным разнообразием. К таким приложениям можно отнести: универсальное ПО, специализированные программы, программное обеспечение для развлечения. В качестве примеров прикладного программного обеспечения можно выделить следующие: текстовые, графические, видео- и аудиоредакторы, веб-обозреватели, антивирусные пакеты, комплексы для проектирования в различных профессиональных областях, проигрыватели, архиваторы и огромное множество других.

* Инструментальное программное обеспечение:

Оно необходимо при проектировании приложений. Такой вид ПО предназначен для разработчиков софта. Среди них: линковщики, компиляторы, трансляторы, сборники подпрограмм и пр. (Например: Microsoft Visual Studio Code.)

# **1.2 Понятие, Актуальные виды и особенности информационных систем**

Понятие информационной системы.

Информационная система (ИС) — это комплекс, разработанный для аккумулирования, хранения, обработки, передачи и отображения информации. ИС включает в себя как технические устройства (аппаратное обеспечение), так и программное обеспечение, данные, процедуры и людей, которые взаимодействуют с данным комплексом.

Типы существующих информационных систем.

1. Информационная система транзакций (ИСТ). Предназначена для автоматизации повседневных операций, таких как обработка заказов, начисление заработной платы и учет финансовых транзакций.

- Примеры: система управления заказами, кассовая система в розничной торговле.

2. Информационная система управления (ИСУ). Используется для управления организацией и предоставления администраторам доступа к отчетам и данным, необходимым для принятия решений.

- Примеры: система управления персоналом, финансовая система.

3. Система поддержки принятия решений (СППР). Поддерживает принятие управленческих решений путем анализа данных и создания модели принятия решений.

- Примеры: система прогнозирования продаж, система анализа инвестиционных рисков.

4. Система выполнения стратегических задач (СВСЗ). Ориентирована на высшее руководство компании и предоставляет данные для стратегического планирования и анализа.

- Примеры: cистема стратегического планирования, система балансовых показателей.

5. Экспертная система (ЭС). Использует знания и правила для моделирования работы профессионалов и решения сложных задач в конкретной области.

- Примеры: медицинские диагностические системы, системы технической диагностики оборудования.

6. Система управления контентом (СУК). Предназначена для создания, управления и публикации контента на веб-сайтах и других цифровых носителях.

- Примеры: wordpress, joomla, drupal.

7. Система планирования ресурсов предприятия (СПРП). Интегрирует все аспекты вашего бизнеса в единую систему, включая планирование, производство, продажи и финансы.

- Примеры: sap, oracle erp, microsoft dynamics.

8. Система управления взаимоотношениями с клиентами (СУВК). Улучшает обслуживание клиентов, увеличивает продажи и помогает управлять взаимодействием с текущими и потенциальными клиентами.

- Пример: salesforce, hubspot, zoho crm.

Возможности информационной системы.

1. Интеграция и взаимодействие. В настоящее время ИС способны обмениваться данными и повышать эффективность организации, обеспечивая интеграцию различных процессов и систем.

2. Автоматизация процессов. Позволяет автоматизировать многие рутинные операции, снижая затраты и вероятность ошибок.

3. Поддержка принятия решений. Предоставляет инструменты для анализа данных и прогнозирования, которые помогают руководителям принимать обоснованные решения.

4. Гибкость и масштабируемость. Позволяет адаптироваться к изменяющимся бизнес-требованиям и масштабироваться по мере роста компании.

5. Безопасность и защита данных. Современные ИС включают механизмы защиты данных от несанкционированного доступа и обеспечивают их целостность.

6. Доступность и мобильность. Могут обеспечить доступ к данным и функциям с разных устройств и из разных мест, что особенно важно в удаленной рабочей среде.

Информационные системы играют важную роль в современной экономике, поддерживая все уровни управления и повышая производительность организации.

# **1.3 Особенности программных средств разработки**

Особенности программных средств разработки ипрограммирования

Программные средства разработки обладают множеством особенностей, которые делают их незаменимыми для создания качественного программного обеспечения. Современные интегрированные среды разработки (IDE) объединяют в одном интерфейсе редактор кода, компилятор, отладчик и системы контроля версий, что позволяет разработчикам выполнять множество задач без необходимости переключаться между разными программами. Подсветка синтаксиса и автодополнение кода ускоряют написание программ, предлагая варианты завершения кодовых конструкций и функций. Инструменты отладки позволяют пошагово выполнять код, устанавливать точки останова и отслеживать значения переменных, что существенно облегчает процесс поиска и исправления ошибок.

Многие современные среды разработки поддерживают множество языков программирования, что делает их универсальными инструментами для разработчиков, работающих с разными технологиями. Встроенная поддержка систем контроля версий, таких как Git, позволяет разработчикам отслеживать изменения в коде, работать в команде и управлять версиями проектов непосредственно из IDE. Интеграция с инструментами для сборки и автоматизации процессов разработки позволяет автоматизировать компиляцию, тестирование и развертывание приложений. Современные инструменты разработки поддерживают интеграцию с облачными сервисами, такими как AWS, Azure и Google Cloud, что облегчает развертывание и управление облачными приложениями. Поддержка контейнеров и виртуальных машин позволяет разработчикам создавать и тестировать приложения в изолированных средах, что повышает совместимость и уменьшает зависимость от конкретной конфигурации оборудования.

Программирование также имеет свои особенности. Язык программирования выбирается в зависимости от требований проекта, целевой платформы и задач. Например, Python подходит для быстрой разработки и прототипирования, тогда как C++ используется для системного программирования и разработки высокопроизводительных приложений. Библиотеки и фреймворки упрощают разработку, предоставляя готовые решения для распространенных задач, что позволяет сосредоточиться на логике приложения, не тратя время на написание базовой функциональности с нуля. Методологии разработки, такие как Agile, DevOps и TDD, помогают организовать процесс разработки, обеспечивая гибкость, улучшение качества кода и сокращение времени на выпуск продукта. Использование различных видов тестирования позволяет выявлять ошибки на ранних стадиях разработки и повышает надежность программного обеспечения. Инструменты для статического анализа кода помогают находить потенциальные уязвимости и ошибки.

Хорошая документация кода облегчает поддержку и развитие проекта, позволяя другим разработчикам быстро разобраться в логике и структуре приложения. Автоматизированные инструменты документирования упрощают создание и поддержание документации. Оптимизация производительности включает профилирование кода, выявление узких мест и применение оптимизаций для улучшения скорости и эффективности работы приложения. Внедрение безопасных практик программирования, таких как проверка входных данных, шифрование конфиденциальной информации и управление доступом, помогает защитить приложения от атак и уязвимостей.

Совместная разработка требует использования систем контроля версий, инструментов для управления задачами и коммуникации, что обеспечивает слаженную работу команды и прозрачность процессов разработки. После выпуска программного продукта важно обеспечить его поддержку и регулярное обновление, включая исправление ошибок, добавление новой функциональности и адаптацию к изменяющимся требованиям пользователей и рынка. Эти особенности играют ключевую роль в успешной разработке, обеспечивая высокое качество и надежность программного обеспечения.

## **1.4** **Определение Visual Studio Code**

Visual Studio Code (VS Code) — это бесплатный, кроссплатформенный редактор исходного кода, разработанный компанией Microsoft. Он поддерживает множество языков программирования и включает в себя функции, такие как подсветка синтаксиса, авто-дополнение кода, встроенный терминал, отладчик и интеграцию с системами контроля версий, такими как Git. VS Code отличается легкостью и высокой производительностью, что делает его популярным выбором среди разработчиков.

Одной из ключевых особенностей Visual Studio Code является его расширяемость. С помощью разнообразных расширений, доступных через встроенный менеджер расширений, пользователи могут добавить поддержку новых языков, улучшить функциональность редактора и интегрировать различные инструменты разработки. Например, расширения могут добавить поддержку Python, Java, C++, Go и других языков, а также инструменты для работы с контейнерами Docker, платформами Kubernetes и облачными сервисами, такими как Azure и AWS.

Visual Studio Code также предлагает мощные средства для отладки. Пользователи могут устанавливать точки останова, следить за значениями переменных, использовать интерактивную консоль и просматривать стеки вызовов. Это делает процесс поиска и исправления ошибок в коде более эффективным и удобным.

Интеграция с системами контроля версий позволяет разработчикам легко отслеживать изменения в коде, управлять ветками и проводить слияние кода. Встроенная поддержка Git предоставляет функции, такие как коммиты, создание и переключение веток, слияние и разрешение конфликтов непосредственно из интерфейса редактора.

Visual Studio Code также имеет встроенный терминал, что позволяет разработчикам выполнять командные строки без необходимости переключаться между окнами. Это особенно удобно для выполнения задач сборки, запуска тестов и управления зависимостями.

Кроме того, Visual Studio Code поддерживает функции для повышения продуктивности, такие как многократное выделение, фрагменты кода (snippets), и интеллектуальное авто-дополнение на основе машинного обучения. Это помогает разработчикам быстрее писать код и сокращать количество ошибок.

На рисунке 1 изображено главное меню редактора Visual Studio Code.

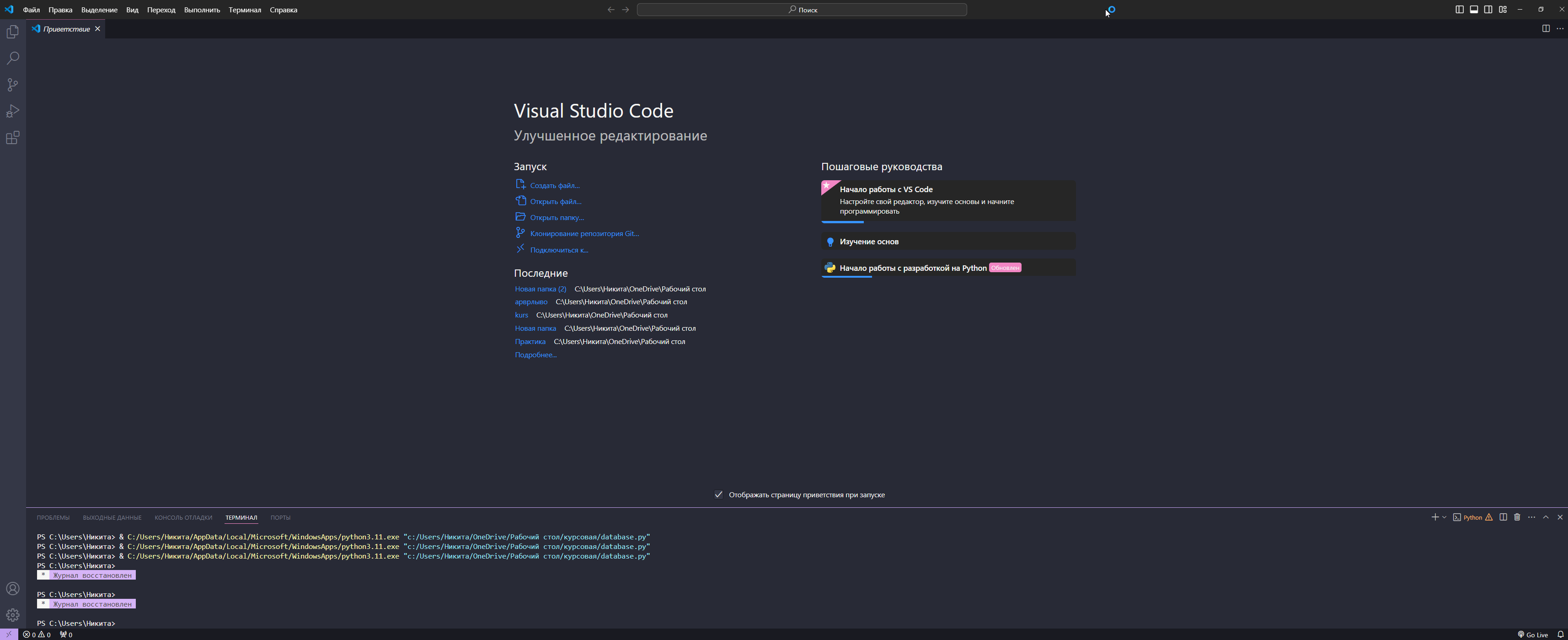


Рисунок 1. Главное меню редактора Visual Studio Code

## **1.5 Функции и особенности приложения Visual Studio Code**

Основные функции Visual Studio Code.

1. Редактирование кода.

Подсветка синтаксиса. Система поддерживает более 30 языков программирования с подсветкой синтаксиса, что помогает разработчикам легко ориентироваться в коде.

Автодополнение кода (IntelliSense). Автопродление предлагает интеллектуальные подсказки и авто-дополнение для ускорения написания кода, включая информацию о функциях, методах и переменных.

Многократное выделение и редактирование. Позволяет одновременно редактировать несколько строк или вхождений, что существенно ускоряет внесение изменений в код.

2. Отладка. Встроенный отладчик. Поддерживает отладку для множества языков программирования. Разработчики могут устанавливать точки останова, следить за значениями переменных и выполнять код по шагам.

Интерактивная консоль. Позволяет выполнять команды и скрипты непосредственно в процессе отладки.

3. Интеграция с системами контроля версий. Git. Встроенная поддержка Git позволяет разработчикам выполнять коммиты, создавать и переключаться между ветками, выполнять слияние и разрешение конфликтов.

Просмотр изменений. Позволяет отслеживать изменения в файлах, сравнивать версии и откатывать изменения.

4. Поддержка расширений.

Marketplace расширений. Предоставляет доступ к тысячам расширений, которые добавляют поддержку новых языков, инструментов и функций. Пользователи могут устанавливать расширения для таких задач, как работа с контейнерами Docker, интеграция с облачными сервисами и многое другое.

Настройка и конфигурация. Позволяет настраивать и конфигурировать рабочее пространство в соответствии с потребностями разработчика.

5. Инструменты для сборки и развертывания.

Терминал. Встроенный терминал позволяет выполнять командные строки непосредственно в редакторе, что удобно для выполнения сборок, тестов и управления зависимостями.

Скрипты задач. Поддержка автоматизации задач через конфигурационные файлы (например, tasks.json) для автоматизации часто выполняемых действий, таких как компиляция кода и тестирование.

6. Совместная работа.

Live Share. Расширение для совместного редактирования кода в реальном времени. Разработчики могут совместно работать над одним и тем же проектом, независимо от их физического расположения.

Интеграция с сервисами общения. Интеграция с такими сервисами, как Slack и Microsoft Teams, позволяет улучшить коммуникацию и сотрудничество в команде.

Особенности Visual Studio Code.

1. Кроссплатформенность. Поддерживает работу на Windows, macOS и Linux, что делает его доступным для широкого круга разработчиков.

2. Высокая производительность. Легковесный и быстрый, что обеспечивает быструю загрузку и отзывчивость интерфейса, даже при работе с большими проектами.

3. Расширяемость и настройка. Возможность настраивать интерфейс и функциональность под свои нужды. Пользователи могут изменять темы, горячие клавиши и настройки редактора.

4. Интеграция с облачными сервисами. Поддержка таких облачных платформ, как Azure и AWS, позволяет разработчикам легко развертывать и управлять приложениями в облаке.

5. Поддержка контейнеров и виртуализации. Интеграция с Docker и Kubernetes позволяет разработчикам создавать, тестировать и развертывать приложения в изолированных средах.

6. Сообщество и поддержка. Большое и активное сообщество пользователей и разработчиков, которые создают расширения, делятся опытом и помогают решать возникающие проблемы. Обширная документация и регулярные обновления от Microsoft.

7. Интеллектуальные функции. Инструменты, такие как Live Server, позволяют автоматически обновлять веб-страницы при изменении кода, что ускоряет процесс разработки веб-приложений. Возможность настройки линтеров и форматоров кода для поддержания качества и единого стиля кода.

Visual Studio Code представляет собой универсальный инструмент, подходящий для разработчиков различного уровня и специализаций, обеспечивая мощные и гибкие средства для эффективной разработки программного обеспечения.

# **Глава 2. Практическая часть создания проекта**

Данная глава представляет собой основную информацию по приложению Visual Studio Code и используемых языков программирования, их особенностей, технологий и причину выбора данных языков и приложения.

## **2.1 Обоснование выбора программы для разработки информационной системы**

Как уже описывалось выше, Visual Studio Code (VS Code) является достаточно универсальной программой, обладающей большим набором функций и инструментов.

VS Code поддерживает множество языков программирования и предоставляет мощные средства для разработки, такие как подсветка синтаксиса, автодополнение кода (IntelliSense), встроенный терминал и отладчик. Благодаря этим функциям разработчики могут легко писать, тестировать и отлаживать код, что значительно ускоряет процесс разработки.

Интеграция с системами контроля версий, такими как Git, обеспечивает удобное управление версиями кода и позволяет легко отслеживать изменения и работать в команде. Встроенные инструменты для отладки позволяют устанавливать точки останова, выполнять пошаговое выполнение кода и наблюдать за значениями переменных, что облегчает процесс поиска и исправления ошибок.

Расширяемость Visual Studio Code делает его еще более мощным инструментом. Через Visual Studio Code Marketplace разработчики могут устанавливать различные расширения, которые добавляют поддержку новых языков программирования, инструментов и интеграций с сервисами. Это позволяет настроить рабочее пространство под конкретные нужды и сделать процесс разработки еще более эффективным.

2.4 Почему Visual Studio Code. Анализ приложений конкурентов

1. Sublime Text.

Преимущества:

- Легковесный и быстрый.

- Сильная поддержка многократного редактирования и макросов.

- Широкие возможности настройки интерфейса.

Недостатки:

- Платный (есть бесплатная версия, но она с ограниченным функционалом).

- Меньшая поддержка расширений по сравнению с VS Code.

- Отсутствие встроенного терминала и интеграции с системами контроля версий.

2. Atom

Преимущества:

- Полностью бесплатный и с открытым исходным кодом.

- Высокая степень настраиваемости через пакеты и темы.

- Интеграция с GitHub.

Недостатки:

- Более медленный по сравнению с VS Code, особенно при работе с большими проектами.

- Проблемы с производительностью и стабильностью.

- Меньшее сообщество и меньшее количество доступных расширений.

3. JetBrains IntelliJ IDEA (и другие IDE JetBrains)

Преимущества:

- Профессиональные инструменты разработки с мощными функциями для различных языков (особенно для Java).

- Интеллектуальные функции автодополнения и отладки.

- Встроенные инструменты для рефакторинга кода и управления проектами.

Недостатки:

- Платный (есть бесплатная версия, но с ограниченным функционалом).

- Более тяжелый и требовательный к ресурсам системы.

- Не такая легковесная и быстрая, как VS Code.

4. PyCharm

Преимущества:

- Специализирован для Python, предлагает мощные функции для разработки на этом языке.

- Встроенная поддержка научных библиотек и фреймворков для веб-разработки.

- Поддержка отладки, тестирования и анализа кода.

Недостатки:

- Платный (есть бесплатная версия, но с ограниченным функционалом).

- Меньшая поддержка других языков программирования.

- Более тяжелый и требовательный к ресурсам системы по сравнению с VS Code.

5. Eclipse

Преимущества:

- Бесплатный и с открытым исходным кодом.

- Сильная поддержка для Java и других языков через плагины.

- Поддержка большого числа инструментов для корпоративной разработки.

Недостатки:

- Более сложный в настройке и использовании.

- Медленнее и тяжелее по сравнению с VS Code.

- Меньше инновационных функций и менее современный интерфейс.

## **2.2 Определение языков программирования, используемых в проекте.**

Python – это высокоуровневый язык программирования общего назначения, известный своей читаемостью и лаконичностью кода. Разработанный Гвидо ван Россумом и впервые выпущенный в 1991 году, Python получил широкое распространение благодаря своей простоте и мощным возможностям. Рассмотрим подробнее основные аспекты и преимущества Python, которые делают его популярным выбором для разработки информационных систем.

Применение Python в разработке информационных систем.

1. Веб-разработка. Python широко используется для разработки веб-приложений. Фреймворки, такие как Django и Flask, предоставляют мощные инструменты для создания надежных и масштабируемых веб-сервисов. Django, в частности, известен своей «батарейкой в комплекте» философией, предоставляя множество встроенных функций для ускорения разработки.

2. Анализ данных и машинное обучение. Благодаря библиотекам, таким как Pandas, NumPy, SciPy и Matplotlib, Python является одним из ведущих языков для анализа данных и научных вычислений. Для машинного обучения и глубокого обучения широко используются TensorFlow, Keras и PyTorch.

3. Автоматизация и скрипты. Простота Python делает его отличным выбором для автоматизации рутинных задач и написания скриптов. Это позволяет автоматизировать процессы, такие как сбор данных, тестирование программного обеспечения и администрирование систем.

4. Разработка настольных приложений. С помощью библиотек, таких как Tkinter, PyQt и Kivy, можно создавать кроссплатформенные настольные приложения с графическим интерфейсом пользователя.

5. Обработка и анализ данных. Python активно используется для обработки и анализа больших данных. С помощью библиотек, таких как Apache Spark (через PySpark) и Dask, можно работать с большими объемами данных и выполнять распределенные вычисления.

6. Интернет вещей (IoT). Python часто используется в проектах, связанных с Интернетом вещей, благодаря таким библиотекам, как MicroPython и CircuitPython, которые позволяют работать с микроконтроллерами и сенсорами.

Преимущества использования Python.

1. Высокая производительность разработки. Читаемый и лаконичный синтаксис Python позволяет разработчикам писать код быстрее и с меньшими ошибками. Это приводит к сокращению времени разработки и упрощению сопровождения кода.

2. Большое сообщество и поддержка. Python имеет одно из самых больших и активных сообществ разработчиков. Это означает, что для большинства задач уже существуют готовые решения, библиотеки и фреймворки, а также обширная документация и поддержка.

3. Широкий спектр применения. Благодаря своей универсальности и поддержке множества библиотек, Python можно использовать для решения широкого спектра задач, от веб-разработки и анализа данных до машинного обучения и автоматизации.

4. Поддержка образования. Python часто используется в образовательных учреждениях для обучения программированию. Его простота и мощные возможности делают его отличным инструментом для обучения основам программирования и разработки программного обеспечения.

Python – это мощный, гибкий и простой в освоении язык программирования, который идеально подходит для разработки информационных систем. Его богатый набор библиотек и фреймворков, кроссплатформенность и активное сообщество делают его одним из самых популярных языков для решения широкого спектра задач в области программирования.

Хотя Python является мощным и гибким языком программирования, у него также есть некоторые недостатки, которые следует учитывать при его использовании:

1. Высокое потребление ресурсов: Python не является самым быстрым языком из-за своей интерпретируемости. В сравнении с компилируемыми языками, такими как C++ или Rust, Python может иметь более низкую производительность и требовать больше оперативной памяти.

2. Глобальная блокировка интерпретатора (GIL): GIL является механизмом защиты данных, который применяется в реализации CPython, стандартной реализации Python. Это означает, что в один момент времени только один поток Python может выполняться в одном процессе. Это ограничение может приводить к проблемам с параллельным выполнением многопоточных программ и ограничивать масштабируемость в некоторых случаях.

3. Не самый подходящий для мобильных устройств. В отличие от некоторых других языков программирования, Python не является идеальным выбором для мобильной разработки. Высокое потребление ресурсов и отсутствие полной поддержки на мобильных платформах делают его менее популярным среди разработчиков мобильных приложений.

4. Иногда медленный запуск программ: Интерпретация кода может сделать запуск программ на Python медленнее, чем у компилируемых языков. Это особенно заметно для небольших скриптов, которые могут иметь заметное время запуска.

5. Неидеальная поддержка для некоторых приложений: В некоторых областях, таких как разработка игр или высокопроизводительные вычисления, Python может быть менее подходящим выбором из-за своих ограничений в производительности. В таких случаях разработчики могут предпочесть использовать более производительные языки программирования.

6. Потенциальные проблемы совместимости версий: Переход на новые версии Python иногда может вызывать проблемы совместимости с существующим кодом, особенно если используются устаревшие или устаревшие библиотеки и фреймворки. Это может потребовать дополнительных усилий для обновления кода и его адаптации к новым версиям языка.

Хотя у Python есть эти недостатки, многие из них могут быть уменьшены или устранены с помощью правильного проектирования приложений и использования соответствующих инструментов и методик разработки.

При написании проекта использовалась база данных SQLite.

Использование SQLite в проекте имеет свои плюсы и минусы:  
Плюсы использования SQLite:

1. Простота и легкость использования:

- Установка. SQLite не требует установки отдельного сервера, что делает его удобным для быстрого начала работы.

- Настройка. Минимальная конфигурация и настройка, так как база данных хранится в одном файле.

2. Производительность:  
   - Быстрота. Высокая скорость выполнения операций для небольших и средних приложений, особенно при чтении данных.

 - Эффективность. Хорошо оптимизирован для работы с небольшими объемами данных и поддерживает транзакции с использованием журнала транзакций (WAL).  
3. Портативность:

 - Файловый формат. База данных хранится в одном файле, который можно легко перемещать между различными системами и платформами.  
   - Кроссплатформенность. Работает на различных операционных системах, включая Windows, macOS, Linux и другие.

4. Соответствие стандартам SQL:

- SQL. Поддерживает большинство стандартных SQL-запросов, что делает его удобным для разработчиков, знакомых с SQL.

5. Встроенность:  
   - Библиотека. Легко интегрируется в приложения, так как SQLite предоставляет API для большинства популярных языков программирования (Python, C#, Java, etc.).  
6. Безопасность данных:

- ACID. Поддерживает транзакции, обеспечивая свойства атомарности, согласованности, изолированности и долговечности.

Минусы использования SQLite

1. Ограничения масштабируемости:

- Большие объемы данных. Не подходит для работы с очень большими базами данных или для приложений с высокими требованиями к нагрузке.  
   - Конкурентный доступ. Ограниченная поддержка многопользовательского доступа и конкурентных операций по сравнению с серверными СУБД.  
2. Ограниченные возможности:

 - Расширенные функции. Отсутствие некоторых расширенных возможностей, таких как сложные триггеры, хранимые процедуры и пользовательские функции.

 - Ограниченная поддержка типов данных. Ограниченная поддержка некоторых типов данных, например, не строгая типизация и отсутствие некоторых специализированных типов данных.

3. Безопасность:  
   - Шифрование. Встроенная поддержка шифрования отсутствует в бесплатной

версии, хотя есть сторонние решения для этой задачи.

4. Поддержка и сообщество:

- Ограниченное сообщество. Хотя SQLite имеет активное сообщество, оно не такое большое и активное, как у более популярных серверных СУБД (например, PostgreSQL или MySQL).

5. Отсутствие сетевых возможностей:

 - Сетевые приложения. SQLite не поддерживает сетевые функции, такие как удаленное управление базой данных, что ограничивает его использование в распределенных системах.

SQLite подходит для небольших и средних приложений, где важна простота настройки и использования, высокая скорость доступа к данным и минимальные требования к масштабируемости и конкуренции. Однако для больших, распределенных и высоконагруженных систем стоит рассмотреть использование более мощных серверных СУБД.

## **2.3 Определение выбора метода для разработки.**

Методом для разработки проекта стоит выбрать встроенные инструменты и пользовательский интерфейс приложения Visual Studio Code, так как это оптимально подходит для создания информационной системы. На рисунке 2 изображён интерфейс приложения Visual Studio Code.

На рисунке 2 изображен главный интерфейс Visual Studio Code.

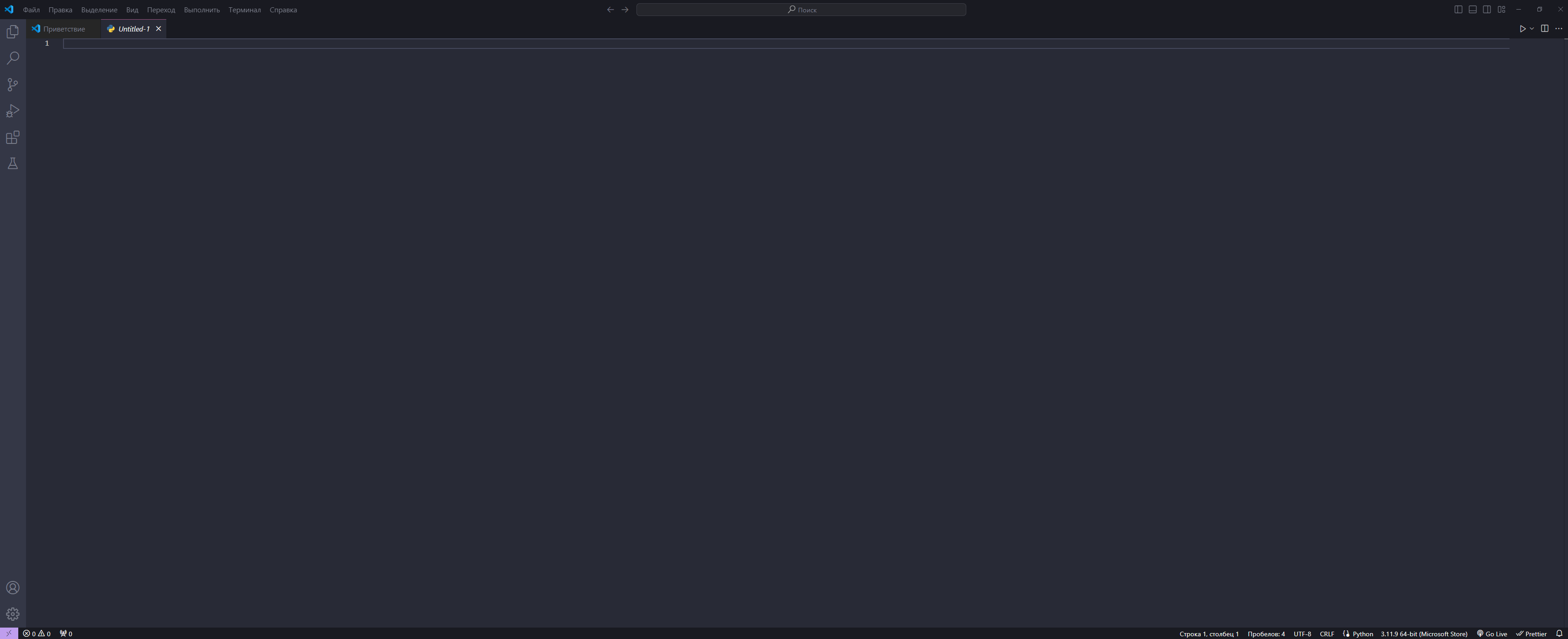


Рисунок 2. Интерфейс Visual Studio Code

Рабочая область.

Это основная рабочая зона. Это текстовый редактор, в котором пользователь создаёт свой код.

Терминал.

Встроенная в приложение командная строка, одновременно с этим являющийся элементом отладчика, отображая выводимый кодом текст, обнаруженные в ходе записи/работы ошибки и предупреждения, предусмотренные как стандартными исключениями и правила языка программирования, так и прописанные вручную пользователем. Так же способен получать от пользователя команды и строки, используемые в командной строке или запрашиваемые программой в ходе работы.

Файловый менеджер.

Встроенный файловый менеджер с стандартными для этого функциями: открытие папок, файлов, навигация по структурной иерархии папок.

Боковое меню.

В боковое меню входят вкладки: проводника, поиска, системы управлении версии (встроенный Git), запуска и отладки программы, расширения. При помощи установленных расширений пополняется новыми вкладками, такие как тестирование.

Меню учётных записей.

Позволяет пользователю связать программу с его учётной записью Microsoft или Github, что позволяет автоматически проводить резервное копирование и синхронизацию файлов в облако для работы на разных устройствах.

Меню настроек.

Открывает обширное меню, в котором можно настроить:  
- Палитру команд, доступных в основном меню.  
- Профили.  
- Параметры программы и рабочей области.  
- Расширения.  
- Сочетания клавиш.  
- Пользовательские фрагменты кода.  
- Задачи пользователя.  
- Темы.  
- Параметры резервного копирования и синхронизации.  
- Проверить наличие обновлений.

Строка состояния.

Отображает текущее состояние программы: наличие ошибок и предупреждений, используемые порты. Позволяет использовать быстрый переход в нужную строку/столбец, настроить размер отступа, кодировку, последовательность конца строки, языковой режим (языка программирования), текущую версию используемого языка программирования.

Основное меню.

Здесь можно создать, загрузить, сохранить, закрыть проекты, сбросить настройки, вносить правки, копировать и вставлять куски кода/строк, выделить строки, настроить внешний вид программы, перейти к нужному участку кода, запустить программе с/без режима отладки, создать терминал и перейти к встроенной справке о программе.

## **2.4 Поэтапная разработка информационной системы**

В данном сегменте будет подробно рассказано, как устроена программа.

На рисунке 3 изображен код Python, который импортирует библиотеки и подключает базу данных.

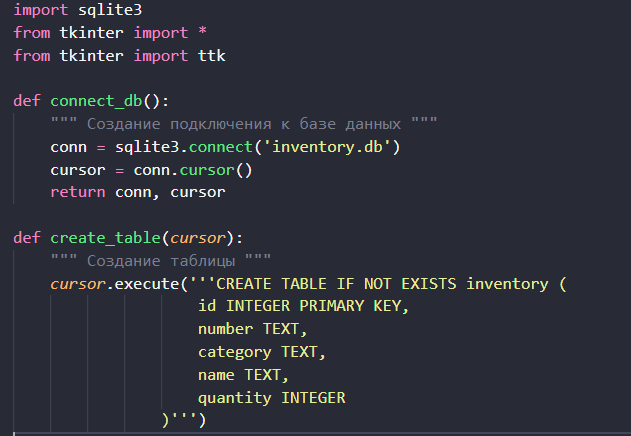
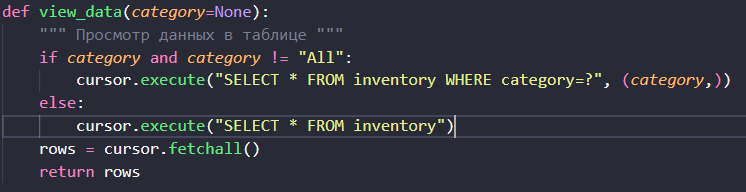


Рисунок 3. Импорт и подключение к БД

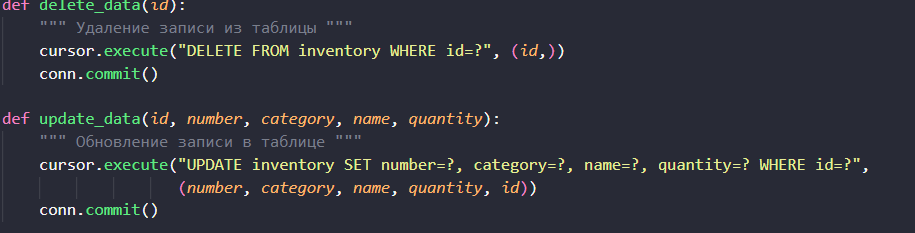
Происходит импорт необходимых библиотек для работы с базой данных, регулярными выражениями и графическим интерфейсом. Создается соединение с базой данных и создание ее скелета.

На рисунке 4 изображен код Python, который объявляет функцию для отображения данных в таблице.

Рисунок 4. Функция для отображения данных в таблице

При нажатии на поле с базой данных появляется возможность просмотра данных.

На рисунке 5 изображен код на Python, который объявляет функции удаления записи из таблицы и обновления данных.

Рисунок 5. Функции удаления и обновления данных в базе данных

При нажатии кнопки удалить, строка с данными визуально пропадает, затем удаляются данные и из самой базы данных. Кнопка обновления нужна для визуального и программного обновления данных, например: при выборе категории: мониторы необходимо нажать кнопку обновления, чтобы данные отфильтровались и вывелись на экран.

.

На рисунке 6 изображен код на Python, который объявляет функцию для добавления информации и получения выделенной строки из списка.

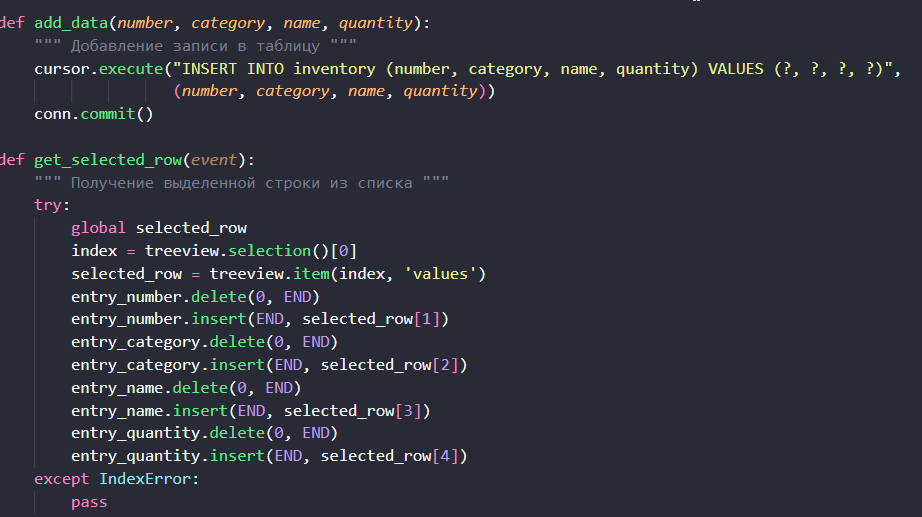


Рисунок 6. Функции, необходимые для добавления записи в таблицу

На рисунке 7 изображен код на Python, который объявляет функцию, которая добавляет кнопку обновления информации в таблице, так же объявляются кнопки обработчика кнопки добавления и кнопки просмотра.



Рисунок 7. Функции обработки для заполнения, просмотра и редактирования таблицы

На рисунке 8 изображен код на Python, который объявляет фильтр по категориям.

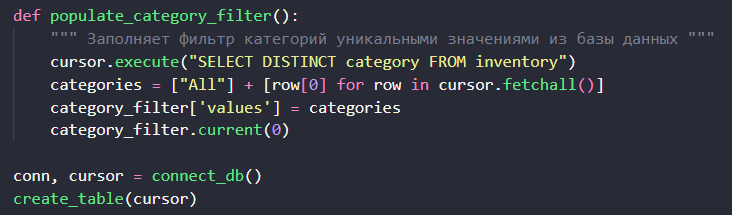


Рисунок 8. Функция для создания фильтрации данных внутри базы данных

На рисунке 9 изображен код на Python, который создает графический интерфейс и метки полей ввода данных.



Рисунок 9. Применение графического интерфейса Tkinter, создание основного окна, разметки таблицы

На рисунке 10 изображен код на Python, который создает физические кнопки в таблице.



Рисунок 10. Создание кнопок для проведения различных манипуляций и изменений в базе данных и таблице

На рисунке 11 изображен код на Python, который создает саму таблицу для отражения данных в приложении.



Рисунок 11. Создание таблицы, разметки, функции прокрутки при необходимости

На рисунке 12 изображен код на Python, который привязывает событие щелчка мыши для выбора записи в таблице.

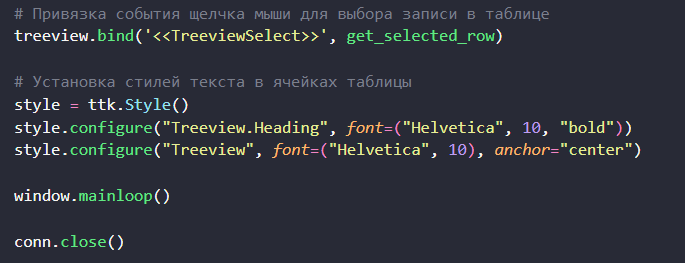


Рисунок 12. Стилизация таблицы, и привязка события щелчка мыши для выбора записей в таблице

На рисунке 13 изображено ИТОГОВОЕ Приложение.

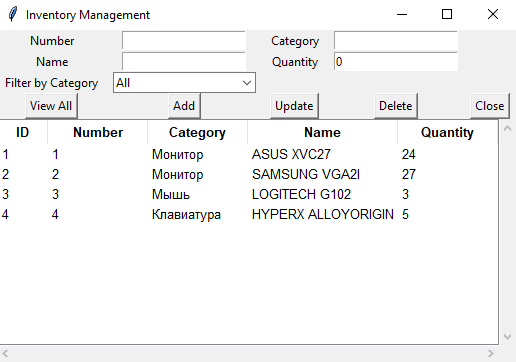


Рисунок 13. Приложение, общий интерфейс

# **ЗАКЛЮЧЕНИЕ**

Разработка информационной системы для инвентарного учета компьютерной техники является важным направлением в области информационных технологий, которое направлено на оптимизацию бизнес-процессов и повышение операционной эффективности предприятия. В современном информационном обществе, где компьютерная техника стала неотъемлемой частью рабочего процесса в организациях, вопрос эффективного управления этими активами становится все более актуальным.

Процесс разработки информационной системы включает в себя несколько ключевых этапов, начиная с анализа требований заказчика и проектирования базы данных, заканчивая внедрением системы в организацию и обучением персонала. Кроме того, важным аспектом разработки является обеспечение безопасности данных, включая защиту от несанкционированного доступа и резервное копирование информации.

Актуальность разработки информационной системы для учета компьютерной техники обусловлена рядом факторов. Во-первых, с ростом числа компьютеров и других устройств на предприятии становится все сложнее отслеживать их местонахождение, статус и общую информацию. Такая система позволяет автоматизировать процесс учета и облегчить работу с данными об инвентаре. Во-вторых, эффективное управление компьютерной техникой становится ключевым фактором в обеспечении бесперебойной работы бизнеса. Кроме того, в современном информационном обществе важно обеспечить безопасность и конфиденциальность данных, что также становится более сложной задачей без специализированной информационной системы.

Цель курсовой работы заключалась в разработке высококачественной информационной системы для упрощения работы предприятия. Для достижения этой цели были рассмотрены понятия "база данных" и "информационная система", изучены языки программирования Python, а также была выбрана программа Visual Studio Code для разработки приложения и базы данных для внедрения информационной системы.

Практическая значимость разработанной информационной системы заключается в ее способности повысить эффективность и безопасность управления компьютерной техникой на предприятии. В ходе работы были выполнены задачи по изучению предметной области, анализу принципов работы информационной системы в конкретной области, разработке и созданию программы для управления информационной системой.

Таким образом, разработанная информационная система для учета компьютерной техники представляет собой важный инструмент для оптимизации бизнес-процессов и повышения операционной эффективности предприятия.

# **ИСПОЛЬЗУЕМАЯ ЛИТЕРАТУРА**

**Нормативные документы:** Конституция Российской Федерации (с учетом поправок, внесен­ных за­конами Российской Федерации «О поправках к Конституции Россий­ской Федерации» от 30 декабря 2008 г. № 6-ФКЗ и от 30 декабря 2008г. № 7-ФКЗ)

// Российская газета № 4831 от 21 января 2009 г.

Федеральный закон Российской Федерации от 7 февраля 2011 № 3-ФЗ «О полиции»// Россий­ская газета от 8 февраля 2011 № 5401.

**Основные источники:**

Робсон Э., Фримен Э. Head First. Изучаем HTML и CSS. 2-е изд./ Робсон Э., Фримен Э.: 2024. – 720 с.

Фрэйн Б. Отзывчивый дизайн на HTML5 и CSS3 для любых устройств/ Фрэйн Б. 2022. – 420 с.

Хрусталев А. Справочник CSS3. Кратко, быстро, под рукой/ Хрусталев А.: 2021. – 304с.

Дуглас К. JavaScript: The Good Parts / Дуглас К.– США: 2008. – 172 с.

Стефанов С. JavaScript. Шаблоны Стефанов С.: 2011. – 272 с.

Сьерра К., Бэйтс Б. Изучаем Java/ Сьерра К., Бэйтс Б.:2023. – 720 с.

Леа В. CSS Secrets: Better Solutions to Everyday Web Design Problems / Леа В.– США: 2015. – 392 с.

Кайла С. You Don't Know JS. Up & Going / Кайла С. – США: 2024. –

88 с.

Элизабет Р., Эрик Ф. Head First HTML and CSS/ Элизабет Р., Эрик Ф.– США: 2013. – 768 с.

Дэвид Ф. JavaScript: The Definitive Guide/ Дэвид Ф.– США: 2011. –

1096 с.

**Дополнительные источники**:

Маркин, А. В. Построение запросов и программирование на SQL. Учебное пособие / А.В. Маркин. - М.: Диалог-Мифи, 2014. - 384 c.

**Словари и энциклопедии**

Социальная философия: словарь / под общ. ред. В. Е. Кемерова, Т. Х. Керимова. – М. : Академический Проект, 2011. – 588 с.

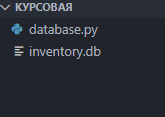
**Электронные ресурсы**

<https://timeweb.cloud/tutorials/python/rabota-s-bazami-dannyh-sql-v-python>

<https://www.ihc.ru/articles/instrukciya-po-nastrojke-vpsvds-servera-s-nulya.html>

# **ПРИЛОЖЕНИЕ**

Приложение 1. Репозиторий проекта.



Приложение 2. Исходный код.

import sqlite3

from tkinter import \*

from tkinter import ttk

def connect\_db():

    """ Создание подключения к базе данных """

    conn = sqlite3.connect('inventory.db')

    cursor = conn.cursor()

    return conn, cursor

def create\_table(cursor):

    """ Создание таблицы """

    cursor.execute('''CREATE TABLE IF NOT EXISTS inventory (

                    id INTEGER PRIMARY KEY,

                    number TEXT,

                    category TEXT,

                    name TEXT,

                    quantity INTEGER

                )''')

def view\_data(category=None):

    """ Просмотр данных в таблице """

    if category and category != "All":

        cursor.execute("SELECT \* FROM inventory WHERE category=?", (category,))

    else:

        cursor.execute("SELECT \* FROM inventory")

    rows = cursor.fetchall()

    return rows

def delete\_data(id):

    """ Удаление записи из таблицы """

    cursor.execute("DELETE FROM inventory WHERE id=?", (id,))

    conn.commit()

def update\_data(id, number, category, name, quantity):

    """ Обновление записи в таблице """

    cursor.execute("UPDATE inventory SET number=?, category=?, name=?, quantity=? WHERE id=?",

                   (number, category, name, quantity, id))

    conn.commit()

def add\_data(number, category, name, quantity):

    """ Добавление записи в таблицу """

    cursor.execute("INSERT INTO inventory (number, category, name, quantity) VALUES (?, ?, ?, ?)",

                   (number, category, name, quantity))

    conn.commit()

def get\_selected\_row(event):

    """ Получение выделенной строки из списка """

    try:

        global selected\_row

        index = treeview.selection()[0]

        selected\_row = treeview.item(index, 'values')

        entry\_number.delete(0, END)

        entry\_number.insert(END, selected\_row[1])

        entry\_category.delete(0, END)

        entry\_category.insert(END, selected\_row[2])

        entry\_name.delete(0, END)

        entry\_name.insert(END, selected\_row[3])

        entry\_quantity.delete(0, END)

        entry\_quantity.insert(END, selected\_row[4])

    except IndexError:

        pass

def delete\_command():

    """ Обработчик кнопки удаления """

    try:

        selected\_rows = treeview.selection()

        for selected\_row in selected\_rows:

            values = treeview.item(selected\_row)['values']

            selected\_row\_id = values[0]

            delete\_data(selected\_row\_id)

            treeview.delete(selected\_row)

    except IndexError:

        pass

def update\_command():

    """ Обработчик кнопки обновления """

    try:

        all\_rows = treeview.get\_children()

        for row in all\_rows:

            values = treeview.item(row, "values")

            if values and len(values) == 5:

                row\_id, number, category, name, quantity = values

                update\_data(row\_id, number, category, name, quantity)

        view\_command()

    except Exception as e:

        print("Ошибка при обновлении базы данных:", e)

def add\_command():

    """ Обработчик кнопки добавления """

    try:

        add\_data(number\_text.get(), category\_text.get(), name\_text.get(), quantity\_text.get())

        view\_command()

    except Exception as e:

        print("Error adding record:", e)

def view\_command():

    """ Обработчик кнопки просмотра """

    treeview.delete(\*treeview.get\_children())

    selected\_category = category\_filter.get()

    for row in view\_data(selected\_category):

        treeview.insert('', 'end', values=row)

def populate\_category\_filter():

    """ Заполняет фильтр категорий уникальными значениями из базы данных """

    cursor.execute("SELECT DISTINCT category FROM inventory")

    categories = ["All"] + [row[0] for row in cursor.fetchall()]

    category\_filter['values'] = categories

    category\_filter.current(0)

conn, cursor = connect\_db()

create\_table(cursor)

# Создание графического интерфейса

window = Tk()

window.title("Inventory Management")

# Создание меток и полей ввода для данных

label\_number = Label(window, text="Number")

label\_number.grid(row=0, column=0)

number\_text = StringVar()

entry\_number = Entry(window, textvariable=number\_text)

entry\_number.grid(row=0, column=1)

label\_category = Label(window, text="Category")

label\_category.grid(row=0, column=2)

category\_text = StringVar()

entry\_category = Entry(window, textvariable=category\_text)

entry\_category.grid(row=0, column=3)

label\_name = Label(window, text="Name")

label\_name.grid(row=1, column=0)

name\_text = StringVar()

entry\_name = Entry(window, textvariable=name\_text)

entry\_name.grid(row=1, column=1)

label\_quantity = Label(window, text="Quantity")

label\_quantity.grid(row=1, column=2)

quantity\_text = IntVar()

entry\_quantity = Entry(window, textvariable=quantity\_text)

entry\_quantity.grid(row=1, column=3)

# Создание фильтра по категориям

label\_filter = Label(window, text="Filter by Category")

label\_filter.grid(row=2, column=0)

category\_filter = ttk.Combobox(window)

category\_filter.grid(row=2, column=1)

populate\_category\_filter()

# Создание кнопок

button\_view = Button(window, text="View All", command=view\_command)

button\_view.grid(row=3, column=0)

button\_add = Button(window, text="Add", command=add\_command)

button\_add.grid(row=3, column=1)

button\_update = Button(window, text="Update", command=update\_command)

button\_update.grid(row=3, column=2)

button\_delete = Button(window, text="Delete", command=delete\_command)

button\_delete.grid(row=3, column=3)

button\_close = Button(window, text="Close", command=window.quit)

button\_close.grid(row=3, column=4)

# Создание таблицы для отображения данных

treeview\_frame = Frame(window)

treeview\_frame.grid(row=4, column=0, columnspan=5, sticky="nsew")

treeview = ttk.Treeview(treeview\_frame, columns=("ID", "Number", "Category", "Name", "Quantity"), show='headings')

treeview.heading("ID", text="ID")

treeview.column("ID", width=50)

treeview.heading("Number", text="Number")

treeview.column("Number", width=100)

treeview.heading("Category", text="Category")

treeview.column("Category", width=100)

treeview.heading("Name", text="Name")

treeview.column("Name", width=150)

treeview.heading("Quantity", text="Quantity")

treeview.column("Quantity", width=100)

treeview.grid(row=0, column=0, sticky="nsew")

# Добавление горизонтальной полосы прокрутки

treeview\_scrollbar = ttk.Scrollbar(treeview\_frame, orient="horizontal", command=treeview.xview)

treeview\_scrollbar.grid(row=1, column=0, sticky="ew")

treeview.configure(xscrollcommand=treeview\_scrollbar.set)

# Добавление полосы прокрутки к виджету Treeview

treeview\_scrollbar\_y = ttk.Scrollbar(treeview\_frame, orient="vertical", command=treeview.yview)

treeview\_scrollbar\_y.grid(row=0, column=1, sticky="ns")

treeview.configure(yscrollcommand=treeview\_scrollbar\_y.set)

# Привязка события щелчка мыши для выбора записи в таблице

treeview.bind('<<TreeviewSelect>>', get\_selected\_row)

# Установка стилей текста в ячейках таблицы

style = ttk.Style()

style.configure("Treeview.Heading", font=("Helvetica", 10, "bold"))

style.configure("Treeview", font=("Helvetica", 10), anchor="center")

window.mainloop()

conn.close()

Приложение 3. Скриншот базы данных.

