Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра информатики
Дисциплина «Операционные среды и системное программирование»

К защите допустить:

И. о. заведующего кафедрой информатики

_____ С. И. Сиротко

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проекту
на тему

КЛИЕНТЫ СЕТЕВЫХ СЛУЖБ: HTTP

БГУИР КП 1-40 04 01 028 ПЗ

Студент А. К. Хрищанович

Руководитель Н. Ю. Гриценко

Нормоконтролер Н. Ю. Гриценко

Минск 2024

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В мире современное разработки программного обеспечения, взаимодействие между различными приложениями через интерфейс приложений стало неотъемлемой частью разработки. Одним из наиболее распространенных протоколов, лежащих в основе сетевых служб, является HTTP. HTTP обеспечивает передачу данных между клиентом и сервером, позволяя пользователям получать доступ к веб-ресурсам, обмениваться информацией и взаимодействовать с веб-приложениями.

Цель данного курсового проекта заключается в создании собственного браузера. Создание собственного браузера может быть важным шагом понимания принципов работы веб-технологий и сетевого программирования, а также позволяет глубже погрузиться в тему веб-разработки.

Задачи курсового проекта на тему «Клиенты сетевых служб: HTTP» включает в себя:

1 Разработка пользовательского интерфейса: создание удобного и интуитивно понятного пользовательского интерфейса для браузера, включая элементы управления, такие как адресная строка, кнопки навигации.

2 Создание механизма для загрузки и отображения веб-страниц: реализация механизма для отображения веб-страниц с использованием уже существующих решений.

3 Обработка HTTP-запросов: реализация механизма для отправки HTTP-запросов на сервер и получения ответов. Это включает в себя работу с протоколом HTTP, управление сеансами и обработку ошибок.

4 Управление вкладками: добавление возможности открывать несколько веб-страниц в разных вкладках.

Реализация этих задач позволит создать собственный браузер, который будет содержать необходимый базовый функционал для возможности создать свое персонализированное пространство для работы с веб-страницами.

1 АРХИТЕКТУРА ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

1.1 Структура и архитектура вычислительной системы

Вычислительная система – результат интеграции аппаратных и программных средства, функционирующих в единой системе и предназначенных для решения задач определенного класса. Структура вычислительной системы состоит из пяти уровней.[1]

1.1.1 Аппаратный уровень

Аппаратный уровень является первым уровнем вычислительной системы. Этот уровень определяется набором аппаратных компонентов и их характеристиками, используемых вышестоящими уровнями иерархии и определяющими воздействие на них. К физическим ресурсам этого уровня относятся: процессор, оперативная память, внешние устройства.

1.1.2 Уровень управления физическими ресурсами

Уровень управления физическими ресурсами является вторым уровнем вычислительной системы и первым уровнем системного программного обеспечения, что подчеркивает то, что это первый слой программного обеспечения, взаимодействующий с аппаратным уровнем. Его основное назначение заключается в систематизации и стандартизации правил программного использования физических ресурсов. На этом уровне обеспечивается создание программ управления физическими ресурсами. Для обеспечения управления физическими ресурсами используются программы, которые называются драйверами физического ресурса (устройства).

Драйвер физического устройства представляет собой программное обеспечение, разработанное на основе команд управления конкретным физическим устройством и предназначенное для организации работы с данным устройством.

1.1.3 Уровень управления логическими/виртуальными ресурсами

Логическое или виртуальное устройство представляет собой ресурс, характеристики которого реализованы программным образом. Драйвер логического или виртуального ресурса – это программное средство, обеспечивающее создание, управление и использование соответствующего ресурса. Эти компоненты ориентированы на конечного пользователя,

и их команды не зависят от конкретных физических устройств, что облегчает взаимодействие с ресурсами для программных компонентов.

На этом уровне могут создаваться новые логические ресурсы, и для организации драйвера логического устройства могут использоваться как драйверы физических, так и драйверы логических или виртуальных устройств. Система поддерживает иерархию драйверов, что позволяет эффективно управлять различными уровнями ресурсов.

Ресурсы вычислительной системы включают в себя как физические, так и виртуальные ресурсы. Они характеризуются конечностью, что создает конкуренцию между программными потребителями за доступ к этим ресурсам. Операционная система, как комплекс программ, управляет этими ресурсами, обеспечивая их эффективное использование.

Существует разветвленная иерархия виртуальных и физических устройств, и драйверы могут быть классифицированы в три группы:

1 Драйверы физических устройств: обеспечивают взаимодействие с конкретными физическими компонентами вычислительной системы.

2 Драйверы виртуальных устройств, обобщающих характеристики физических устройств: позволяют создавать абстракции над физическими устройствами, обобщая их характеристики и предоставляя более удобный интерфейс программам.

3 Драйверы виртуальных устройств без аппаратной реализации: эти драйверы создают «полностью» виртуальные устройства, не имеющие физического эквивалента. Примером может служить драйвер файловой системы, который управляет доступом к файлам без конкретной аппаратной реализации.

Такая иерархия и классификация драйверов позволяют эффективно организовывать управление ресурсами на уровне операционной системы и предоставлять пользователю удобные интерфейсы для работы с вычислительной системой.

1.1.4 Уровень систем программирования

Система программирования представляет собой комплекс программ, предназначенных для обеспечения эффективного управления жизненным циклом программного продукта в вычислительной системе. Жизненный цикл программы включает в себя пять ключевых этапов, к которым относятся проектирование программного продукта, кодирование или реализация, тестирование и отладка, а также внедрение программного продукта и его сопровождение.

Проектирование программного продукта представляет собой первый этап жизненного цикла программы. На этом этапе определяется структура, архитектура и основные характеристики будущего программного продукта. Кроме этого, на этапе проектирования определяются требования к программному продукту, его архитектура, проектирование интерфейса, выбор технологии и разработка алгоритмов для реализации.

Кодирование или реализация представляет собой продолжение этапа проектирования. На данном этапе программный продукт переходит из концепции в конкретный исполняемый код.

Тестирование представляет собой процесс проверки программы с использованием заранее определенных тестовых данных. Тесты, как заранее заданные входные данные, направлены на обеспечение корректного выполнения программы в различных сценариях использования. Тестирование также включает в себя оценку производительности и надежности программы.

Отладка – это процесс выявления, локализации и исправления ошибок, обнаруженных в процессе тестирования. Отладка направлена на обеспечение стабильной и надежной работы программы.

Этап внедрения включает в себя установку программного комплекса на объектную вычислительную систему и его первичную настройку. Внедрение также включает в себя проверку работоспособности программы в реальной среде.

В этап сопровождения входит исправление ошибок и недочетов, выявленных после внедрения программного комплекса. Сюда также включается выпуск патчей для улучшения функциональности или исправления обнаруженных проблем.

1.1.5 Уровень прикладных систем

На уровне прикладных систем реализуются программные системы, ориентированные на решение или автоматизацию решения задач из конкретной предметной области. Этот уровень представляет собой высокоуровневую часть программного обеспечения, которая нацелена на конечного пользователя и предоставляет интерфейс для взаимодействия с компьютерной системой.

Уровень прикладных систем представляет собой ключевой элемент компьютерной архитектуры, где программы разрабатываются и оптимизируются для конечных пользователей, сосредотачиваясь на решении конкретных задач в различных областях человеческой деятельности.

1.2 Структура и архитектура выбранной вычислительной системы для разработки курсового проекта

Вычислительная система, на которой проводилась разработка данного курсового проекта, базируется на операционной системе Windows 10 и аппаратном обеспечении ноутбука Asus из серии TUF Gaming. Эта система включает в себя следующие компоненты:

1 Процессор Intel Core i5: процессор (центральный процессор) является основным вычислительным движком компьютера. Данный процессор обеспечивает высокую производительность и является важной частью архитектуры системы.

2 Оперативная память (RAM): оперативная память служит для временного хранения данных, используемых программами вовремя их выполнения. Это важный аспект архитектуры, так как от объемы и скорости оперативной памяти зависит производительность системы.

3 Жестки диск: жесткий диск или SSD предназначен для хранения операционной системы, приложений и данных.

4 Дискретная видеокарта NVIDIA GeForce GTX 1650 GDDR на основе графического процессора TU117, специализируется на обработке графики и выполнении сложных вычислений, связанных с отображением изображений на экране.

На основе операционной системы Windows 10 и аппаратного обеспечения ноутбука Asus TUF Gaming с процессором Intel Core i5, оперативной памятью, жестким диском (или SSD) и видеокартой NVIDIA GeForce GTX 1650 GDDR6 можно сделать вывод о высокой производительности и графических возможностях данной вычислительной системы, что делает ее подходящей для разработки и выполнения требовательных задач.

1.2.1 Устройство центрального процессора

Процессор – мозг компьютера. Его задача – выполнять программы, находящиеся в основной памяти. Для этого он вызывает команды из памяти, определяет их тип, а затем выполняет одну за другой. Компоненты соединены шиной, представляющей собой набор параллельно связанных проводов для передачи адресов, данных и управляющих сигналов.

Процессор состоит из нескольких частей. Блок управления отвечает за вызов команд из памяти и определение их типа. Арифметика-логическое устройство выполняет арифметические и логические операции.

Тракт данных состоит из регистров, арифметико-логического устройства и нескольких соединительных шин. Содержимое регистров поступает во входные регистры арифметико-логического устройства. В них находятся входные данные, пока арифметико-логическое устройство производит вычисления. Тракт данных – важная составляющая часть всех компьютеров.[2]

В дополнении к процессору требуются по крайней мере некоторое оперативное запоминающее устройство, какой-то способ, позволяющий пользователю записать информацию в компьютер (устройство ввода), а также извлечь ее из него (устройство вывода).

Кроме того, необходимо еще несколько микросхем для объединения всех компонентов. Сам процессор мы можем представить в качестве «черного ящика», внутреннюю работу которого нам не обязательно досконально изучать, чтобы разобраться в его функциях.[3]

Центральный процессор выполняет каждую команду за несколько шагов. Для начала он вызывает следующую команду из памяти и переносит ее в регистр команд. Вторым шагом процессор меняет положение счетчика команд, которые после этого указывает на следующую команду. Затем процессор определяет тип вызванной команды и, если команда использует слово из памяти, определяет, где находится это слово. После процессор переносит слово в регистр центрального процессора, если это необходимо, и только после всех этих шагов процессор выполняет команду. После окончания выполнения команды, он переходит к первому шагу, чтобы начать выполнение следующей команды. Такая последовательность шагов является основой работы всех компьютеров.

1.2.2 Основная (оперативная) память

Память – это тот компонент компьютера, в котором хранятся программы и данные. Также часто встречается термин «запоминающее устройство». Без памяти, откуда процессоры считывают и куда записывают информацию, не было бы современных цифровых компьютеров.

В общем случае основной памятью компьютера называют оперативную память (Random Access Memory, RAM). Оперативная память играет ключевую роль в функционировании компьютера и используется для временного хранения данных и кода, которые активно используются программами во время их выполнения.

Основной единицей измерения хранения данных в памяти является двоичный разряд, который называется битом. Бит может содержать 0 или 1. Эта самая маленькая единица памяти.

Применение двоичной системы счисления в компьютерах объясняется ее «эффективностью». При этом имеется в виду, что хранение цифровой информации может быть основано на отличиях между разными величинами какой-либо физической характеристики, например напряжения или тока. Чем больше величин нужно различать, тем меньше отличий между смежными величинами и тем менее надежна память. В двоичной системе счисления требуется различать всего две величины, следовательно, это самый надежный метод кодирования цифровой информации.

Память состоит из ячеек, каждая из которых может хранить некоторую порцию информации. Ячейка – минимальная адресуемая единица памяти. В последние годы практически все производители выпускают компьютеры с 8-разрядными ячейками, которые называются байтами. Байты группируются в слова. Каждая ячейка имеет номер, который называется адресом. По адресу программы могут ссылаться на определенную ячейку.

В компьютерах, в которых используется двоичная система счисления (включая восьмеричной и шестнадцатеричное представление двоичных чисел), адреса памяти также выражаются в двоичных числах.

1.2.3 Вспомогательная память

Каков бы ни был объем основной памяти, ее все равно будет мало. С развитием технологий людям приходят в голову такие вещи, которые раньше считались совершенно фантастическими. Огромный объем информации в настоящее время невозможно разместить в основной памяти, поэтому на данный момент все пользуются вспомогательной памятью.

Устройства на базе энергонезависимой флэш-памяти, часто называют твердотельными накопителями или SSD-дисками (Solid State Disk), которые являются высокоскоростной альтернативой традиционным технологиям магнитных дисков.

Так как SSD-диски по сути являются памятью, они обладают более высокой производительностью по сравнению с вращающимися магнитными дисками при нулевом времени поиска. Поскольку устройство не имеет подвижных частей, оно особенно хорошо подходит для ноутбуков (колебания в перемещении не влияют на его способность обращаться к данным).

1.2.4 Видеокарта и графический процессор

Видеокарта основана на конкретной архитектуре, которая определяет организацию и характеристики вычислительных блоков. Видеокарта содержит основана на графическом процессоре TU117. В связи с этим далее подробно рассмотрено устройство графического процессора.

Конкретно к характеристикам видеокарты можно отнести объем видеопамяти (VRAM), используемый для хранения текстур, кадров и других графических данных.

Графические процессоры (GPU) представляют собой сложные вычислительные устройства, которые спроектированы для выполнения множества параллельных вычислений. Они играют важную роль в современных вычислениях, таких как компьютерная графика и научные исследования. Важно понимать, что графические процессоры имеют собственную архитектуру и инструкции, которые несколько отличаются от центральных процессоров.

Вот более подробное описание ключевых элементов графического процессора:

1 Блоки FP32 и FP64: это вычислительные блоки, предназначенные для операций с плавающей точкой различной точности. FP32 (одинарная точность) обычно используется для большинства вычислений, в то время как FP64 (двойная точность) используется для более точных вычислений, но с более низкой производительностью.

2 Блоки INT32: эти блоки предназначены для выполнения целочисленных операций. Они могут использоваться для разнообразных вычислений, таких как обработка изображений и кодирование данных.

3 Регистры: регистры являются небольшими хранилищами данных, которые используются для временного хранения результатов вычислений. Каждое вычислительное ядро имеет свои собственные регистры.

4 Встроенная память (кэши): графические процессоры обычно имеют несколько уровней кэша для ускорения доступа к данным. Эти уровни кэша могут быть использованы для хранения часто используемых данных, уменьшая время доступа.

5 Планировщик команд и диспетчер команд: эти компоненты отвечают за управление выполнением инструкций на графическом процессоре. Планировщик команд определяет, какие задачи будут выполнены, их порядок и распределение по вычислительным блокам.

6 L0$, L1$, и L2$: эти кэши представляют собой различные уровни кэширования памяти на графическому процессоре. L0$ и L1$ используются

для временного хранения данных внутри вычислительных блоков, в то время как L2$ представляет собой кэш на уровне всей графической карты.

7 Внешняя память (память внешнего устройства): это память, доступная для графического процессора, и она обычно используется для хранения данных и текстур.

8 TMU (Texture Mapping Units) и ROP (Raster Operations Pipeline): эти блоки отвечают за текстурирование и растеризацию графики. Они играют важную роль в обработке и отображении изображений.

Графический процессор можно рассматривать как устройство асинхронного ожидания, где задачи выполняются параллельно, и управление заданиями и данными играет ключевую роль. Ядра CUDA в GPU Nvidia и ядра в GPU AMD представляют собой простые вычислительные блоки, способные выполнить параллельные вычисления с плавающей точкой. Важно понимать, что они не имеют некоторых возможностей, характерных для CPU, но обеспечивают высокую производительность при выполнении определенных действий.[7]

1.3 История, версии и достоинства

1.3.1 Архитектура x64

Архитектура x86-64, также известная как Intel 64, представляет собой расширение архитектуры x86, которая была разработана компаниями AMD и Intel. Ее история связана с потребностью в поддержке больших объемов памяти и более высокой производительности. Она была представлена в 2000 году.

С тех пор архитектура x86-64 пережила несколько версий и улучшений. Наиболее значимыми версиями были:

1 AMD 64: исходная версия архитектуры, представленная AMD в 2000 году.

2 Intel 64: Intel также внедрила свою версию архитектуры x86-64, известную как Intel 64, также в начале 2000-х годов.

Архитектура x86-64 имеет ряд значительных преимуществ:

1 Поддержка 64-битных приложений: позволяет запускать и обрабатывать 64-битные приложения, что повышает производительность и позволяет эффективно управлять большими объемами данных.

2 Поддержка больших объемов памяти: способность адресации более 4 ГБ оперативной памяти, что является критически важным для современных вычислительных задач.

3 Совместимость: обратная совместимость с 32-битными приложениями и операционными системами, что обеспечивает плавный переход к 64-разрядной архитектуре.

Архитектура x86-64 внесла множество инноваций и технологических решений, включая:

1 Наборы инструкций: расширенные наборы инструкций, которые обеспечивают более высокую производительность и возможности оптимизации.

2 Виртуализация: встроенная поддержка технологий виртуализации для лучшей изоляции и управления виртуальными машинами.

3 Защита данных: механизмы защиты данных и аппаратная защита от вредоносных атак.

Архитектура x86-64, в том числе Intel 64, широко применяется в различных областях:

1 Серверы: в крупных серверных фермах для обработки данных и виртуализации.

2 Настольные и ноутбуки: в персональных компьютерах для выполнения разнообразных задач, включая игры, мультимедиа и офисные приложения.

3 Разработка и научные исследования: для высокопроизводительных вычислений и моделирования.

Архитектура x86-64 соперничает с другими архитектурами, такими как ARM64, в различных областях, и каждая из них имеет свои сильные стороны и применение.

1.3.2 Оперативная память (RAM) 8 ГБ

История развития оперативной памяти в компьютерах имеет долгий путь, начиная с ранних электромеханических устройств и электронных трубок в середине 20-го века. С появлением первых компьютеров, оперативная память представляла собой регистры и буферы, которые использовались для хранения ограниченного количества данных. Со временем разработчики стали осознавать необходимость увеличения объема и скорости оперативной памяти.

Первые интегральные схемы для оперативной памяти появились в 1970 годах, что позволило значительно увеличить ее объем и скорость. Затем пришли стандарты памяти, такие как DRAM (динамическая оперативная память) и SRAM (статическая оперативная память), которые сделали оперативную память более доступной и эффективной.

С появлением компьютеров с графическими пользовательскими интерфейсами в 1980-х и 1990-х годах, объем оперативной памяти стал критически важным для обеспечения плавной работы графических приложений и многозадачности.

В последние десятилетия, с развитием многоядерных процессоров и 64-разрядных операционных систем, объем и скорость оперативной памяти продолжают расти. Современные стандарты памяти DDR3, DDR4 и DDR5 предлагают высокую производительность и эффективность.

На протяжении истории компьютерных технологий существует множество версий оперативной памяти. Однако основные различия заключаются в следующем:

1 DDR (Double Data Rate) – первые стандарты, которые существуют с 2000 года.

2 DDR2 – следующее поколение памяти, улучшившее скорость передачи данных.

3 DDR3 – стандарт, который сделал память еще быстрее и эффективнее.

4 DDR4 – последнее поколение на момент написания, предлагающее высокую производительность и низкое энергопотребление.

5 DDR5 – ожидаемый стандарт, который будет продолжать увеличивать производительность.

Оперативная память играет ключевую роль в производительности компьютерных систем. Ее основные преимущества включают в себя:

– быстрый доступ к данным, что ускоряет выполнение задач;

– поддержка многозадачности;

– улучшение производительности в графических и видео приложениях;

– снижение времени загрузки операционной системы и приложений;

– надежность и стабильность в работе системы.

Инновации в оперативной памяти включают в себя управление энергопотреблением для экономии электроэнергии, технологии исправления ошибок для обеспечения надежности данных и увеличение скорости передачи данных с каждым новым стандартом.

Оперативная память сравнивается с альтернативными формами хранения данных, такими как жесткие диски и SSD (твердотельные накопители). В сравнении с ними, оперативная память предлагает мгновенный доступ к данным, что делает ее идеальным выбором для выполнения задач, требующих высокой скорости чтения и записи.

Оперативная память широко используется во всех областях информационных технологий, включая домашние компьютеры, серверы,

мобильные устройства и встраиваемые системы. Ее выдающиеся задачи включают в себя поддержку операционных систем, обработку данных в реальном времени и обеспечение высокой производительности в играх и профессиональных приложениях.

1.3.3 SSD-диск Micron 2210

Микрон (Micron Technology) – это американская компания, специализирующаяся на производстве полупроводников и твердотельных накопителей. Что касается SSD дисков Micron, их история связана с развитием технологии NAND-флэш памяти и ростом рынка твердотельных накопителей. Микрон начала производство SSD дисков в начале 2010-х годов. С течением времени SSD диски Micron стали обладать большей емкостью, более высокой скоростью чтения и записи, а также улучшенной надежностью благодаря развитию NAND-технологий и контроллеров. Рост объемов данных, требования к скорости и надежности, а также снижение стоимости NAND флэш памяти оказали влияние на эволюцию SSD дисков Micron.

Серия SSD дисков Micron 2210, как правило, имеет несколько версий с различными объемами памяти и характеристиками скорости. Улучшения могут включать в себя увеличение емкости, повышение скорости чтения/записи и снижение энергопотребления.

К достоинствам SSD-дисков Micron можно отнести:

– высокая скорость чтения и записи данных;

– надежность и долгий срок службы;

– отсутствие подвижных деталей;

– относительно низкое энергопотребление;

– быстрый доступ к данным.

Micron внедряет новые технологии NAND-флэш памяти, такие как 3D NAND, и совершенствует контроллеры, чтобы улучшить производительность и надежность своих SSD дисков.

SSD диски Micron 2210 применяются в различных областях, включая настольные компьютеры, ноутбуки, серверы и встраиваемые системы. Они эффективно работают в задачах, где требуется быстрый доступ к данным, надежность и производительность, таких как загрузка операционной системы, работа с приложениями и обработка данных.

1.3.4 Видеокарта NVIDIA GeForce GTX 1650

Архитектура графических процессоров развивалась на протяжении многих лет. Первые графические процессоры были разработаны в конце 20-го

века и предназначались для ускорения графических вычислений на компьютерах. С развитием видеоигр и требований к графике, графические процессоры стали более мощными и функциональными.

Создание дискретных видеокарт, таких как NVIDIA GeForce GTX 1650, связано с ростом популярности компьютерных игр и профессиональных графических приложений. Даты выпуска различных версий видеокарт и ключевые изменения в их архитектуре также являются важными моментами в истории.

В архитектуре GPU NVIDIA GeForce GTX есть множество версий и поколений. Каждое новое поколение обычно вносит улучшения в производительность, эффективность и функциональность. Примерами могут служить различные модели в серии GeForce GTX, такие как 900-я, 1000-я, и 1600-я серии. Важно рассмотреть, какие конкретные изменения и улучшения были внесены в каждую версию и почему они были важны для пользователей.

Архитектура NVIDIA GeForce GTX 1650 обладает рядом преимуществ, включая высокую графическую производительность, поддержку современных графических технологий, таких как Ray Tracing, эффективное охлаждение, и возможность использования в игровых и профессиональных приложениях. Эти достоинства могут существенно повысить производительность и качество визуальных задач на компьютере.

В архитектуре графического процессора всегда присутствуют инновации и технологии. Это могут быть новые алгоритмы обработки графики, улучшенные методы рендеринга, аппаратная поддержка искусственного интеллекта, и многое другое.

1.4 Обоснование выбора вычислительной системы

При выборе вычислительной системы важным фактором было обеспечение комфортных условий для работы, создания браузера. В данном случае выбранная вычислительная система отвечает этим требованиям по нескольким причинам:

1 Производительность процессора Intel Core i5: процессор Intel Core i5 предоставляет хорошую производительность для выполнения широкого спектра задач. Его многозадачные способности позволяют комфортно работать с различными приложениями, включая разработку программного обеспечения, обработку данных и другие вычислительно интенсивные задачи.

2 Объем и скорость оперативной памяти: выбор оперативной памяти является важным аспектом архитектуры системы. 8 ГБ оперативной памяти,

предоставляемой этой системой, обеспечивают достаточное количество ресурсов для эффективной работы с приложениями, включая среды разработки и виртуальные машины.

3 Скорость и емкость твердотельного: присутствие твердотельного накопителя обеспечивает высокую скорость загрузки операционной системы и приложений, что улучшает общую производительность системы. Он также обеспечивает хорошую отзывчивость и быстрое открытие файлов.

4 Дискретная видеокарта NVIDIA GeForce GTX 1650: для выполнения графических задач и игр, а также для разработки и тестирования графических приложений, важна дискретная видеокарта. NVIDIA GeForce GTX 1650 обеспечивает высокую производительность и качественное отображение графики.

5 Совместимость с программными средами: Windows 10 является популярной операционной системой, обеспечивающей совместимость с большим количеством программ и сред разработки. Это важно для выполнения курсового проекта и обеспечения легкости в разработке и отладке программ. Поэтому при разработке собственного браузера была выбрана именно операционная система Windows 10.

6 Портативность и мобильность: ноутбук Asus из серии TUF Gaming предоставляет портативность, что позволяет работать над проектом в различных местах и условиях, что может быть важно для эффективного выполнения задач.

Выбранная вычислительная система на основе ноутбука Asus TUF Gaming с процессором Intel Core i5, оптимизированной оперативной памятью, быстрым твердотельным накопителем и видеокартой NVIDIA GeForce GTX 1650 GDDR6 обеспечивает комфортные условия для разработки, создания браузера и взаимодействия с аппаратным обеспечением, гармонично сочетая производительность, портативность и совместимость с программным обеспечением.

1.5 Анализ выбранной вычислительной системы

1.5.1 Анализ Intel Core i5-10300H

Процессор Intel Core i5-10300H оборудован 4 ядрами и 8 потоками для управления многозадачностью. Максимальная тактовая частота процессора достигает 4.50 GHz, что гарантирует быстродействие системы. Процессор имеет максимальную температуру работы 100°C, что говорит о том, что работа процессора сбалансирована между производительностью

и теплорассеиванием. Размеры кэша процессора включают L1 (256 КБ), далее L2 (1 МБ) и L3 (8 МБ). Intel Core i5-10300H поддерживает память типа DDR4 2933, обеспечивая эффективное управление памятью. Максимально поддерживаемый размер памяти процессора достигает 128 ГБ, что обеспечивает необходимый объем ресурсов для работы с устройством. В процессор также интегрирована графика Intel UHD Graphics.[8]

1.5.2 Анализ NVIDIA GeForce GTX 1650 и TU117

Графический процессор обладает архитектурой, оптимизированной для параллельной обработки данных. Ядра CUDA предоставляют возможность использовать вычислительные ресурсы графического процессора. NVIDIA GeForce GTX 1650 имеет 896 потоковых процессоров. Большое количество потоковых процессоров ведет к более высокой производительности
в параллельных задача, таких как обработки графики или симуляции физики. Тактовая частота видеокарты может быть около 1485-1560 МГц. Объем видеопамяти (VRAM) у видеокарты NVIDIA GeForce GTX 1650 составляет 4 ГБ. Этот объем видеопамяти обеспечивает достаточные ресурсы для обработки текстур, кадров и других графических данных.[9]

1.5.3 Анализ оперативной памяти

В рамках вычислительной системы, выбранной для рассмотрения, отмечается использование оперативной памяти типа DDR4 с общим объемом 8 гигабайт. Указанный тип оперативной памяти характеризуется высокой скоростью передачи данных и обеспечивает эффективность функционирования системы. Объем оперативной памяти, равный 8 гигабайтам, обеспечивает достаточные ресурсы для эффективного выполнения обработки данных, выполнения программных задач и обеспечивает возможность эффективного осуществления многозадачных операций.

Оперативная память DDR4, используемая в данной системе, представляет собой четвертое поколение двойного канала синхронной динамической памяти. Этот тип памяти отличается от предыдущих версий более высокой пропускной способностью и более низким энергопотреблением. Такие характеристики способствуют повышению общей производительности системы за счет улучшенной передачи и обработки данных.

Восемь гигабайт оперативной памяти являются достаточными для обеспечения необходимого запаса ресурсов, что особенно важно в контексте эффективного выполнения вычислительных задач. Этот объем оперативной памяти позволяет системе более эффективно управлять обработкой данных, запущенными программами и одновременными многозадачными операциями.

Важно подчеркнуть, что использование DDR4 в данной системе является современным стандартом, что обеспечивает не только высокую пропускную способность, но и поддержку современных технологий и стандартов в области вычислительных систем.

Таким образом, данная конфигурация оперативной памяти способствует оптимизации работы системы и повышению ее общей производительности, что актуально в условиях современных вычислительных требований.

1.5.4 Анализ твердотельного накопителя

SSD Micron 2210 представляет собой твердотельный накопитель с объемом хранения данных в размере 512 ГБ. Этот накопитель использует флеш-память с технологией 3D QLC NAND, что означает использование многозвенной клеточной структуры для хранения четырех уровней данных (quad-level cell). Технология 3D QLC NAND обеспечивает высокую плотность данных, что важно для максимального использования доступного пространства на накопителе.

Скорость последовательной записи на SSD Micron 2210 достигает 1070 мегабайт в секунду (МБ/с). Эта характеристика означает способность накопителя последовательно записывать данные на свою память со значительной скоростью.

Максимальная производительность SSD Micron 2210 достигает 320.000 операций ввода/вывода в секунду (IOPS). Операции ввода/вывода представляют собой ключевой параметр для оценки способности накопителя обрабатывать запросы на чтение и запись данных. Высокая производительность IOPS обеспечивает отзывчивость системы при обработке множества мелких операций чтения/записи.[10]

В целом, выбранная вычислительная система обеспечивает баланс между производительностью, совместимостью, портативностью и доступностью, что делает ее максимально оптимальным решением для выполнения данного курсового проекта.

2 ПЛАТФОРМА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

2.1 Структура и архитектура платформы

2.1.1 Выбранная платформа

Разработка курсового проекта «Клиенты сетевых служб: HTTP» была проведена на языке Python с использованием среды разработки PyCharm.

Основной платформой разработки браузерс является Python, так как именно этот язык программирования был выбран для проектирования проекта.

PyCharm – это специализированная интегрированная среда разработки Python, предоставляющая широкий спектр необходимых инструментов для разработчиков, тесно интегрированных для создания удобной среды для продуктивной разработки. Все, что происходит в PyCharm, происходит в контексте проекта. Он служит основой для помощи в кодировании, массового рефакторинга, согласованности стилей кодирования и прочего.[11]

Для работы с проектом была выбрана операционная система Windows. Windows – это операционная система, созданная корпорацией Microsoft. Операционная система (ОС) – главная программа, которая запускается при включении компьютера. Она позволяет пользователям компьютера работать с файлами, пользоваться нужными приложениями, выходить в Интернет.[12]

2.1.2 Язык программирования

Язык программирования – формальный язык, предназначенный для записи компьютерных программ. Язык программирования определяет набор лексических, синтактических и семантических правил, определяющих внешний вид программы и действия, которые выполнит исполнитель под ее управлением.

Язык программирования предназначен для написания компьютерных программ, которые представляют собой набор правил, позволяющих компьютеру выполнить тот или иной вычислительный процесс, организовать управление различными объектами и прочее. Язык программирования отличается от естественных языков тем, что предназначен для управления ЭВМ, в то время как естественные языки используются, прежде всего, для общения людей между собой. Большинство языков программирования использует специальные конструкции для определения и манипулирования структурами данных и управления процессом вычислений.

2.1.3 Интегрированная среда разработки

Интегрированная среда разработки (IDE) – комплекс программных средств, используемый программистами для разработки программного обеспечения.

Среда разработки включает в себя:

– текстовый редактор;

– транслятор (компилятор или интерпретатор);

– средства сборки и отладки.

Интегрированные среды разработки были созданы для того, чтобы максимизировать производительность программиста благодаря тесно связанным компонентам с простыми пользовательскими интерфейсами.

IDE обычно представляет собой единственную программу, в которой проводится вся разработка. Она, как правило, содержит много функций для создания, изменения, компилирования, развертывания и отладки программного обеспечения. Цель интегрированной среды заключается в том, чтобы объединить различные утилиты в одном модуле, который позволит абстрагироваться от выполнения вспомогательных задач, тем самым позволяя программисту сосредоточиться на решении собственно алгоритмической задачи и избежать потерь времени при выполнении типичных технических действий (например, вызове компилятора). Таким образом, повышается производительность труда разработчика. Также считается, что тесная интеграция задач разработки может далее повысить производительность за счет возможности введения дополнительных функций на промежуточных этапах работы. Например, IDE позволяет проанализировать код и тем самым обеспечить мгновенную обратную связь и уведомить о синтаксических ошибках.

2.1.4 Операционная система

Как уже отмечалось, современный компьютер организован в виде иерархии уровней, каждый из которых добавляет определенные функции к нижележащему уровню.

С точки зрения программиста, операционная система – это программа, добавляющая ряд команд и функций к командам и функциям, предлагаемым уровнем архитектуры команд. Обычно операционная система организуется программно, но нет никаких веских причин, по которым ее, как микропрограммы, нельзя было реализовать.

Все команды уровня операционной системы доступны для прикладных программистов. Это практически все команды более низкого уровня, а также

новые команды, добавленные операционной системой. Новые команды называются системными вызовами. Они вызывают предопределенную службу операционной системы, в частности одну из ее команд. Например, типичный системный вызов может читать данные из файла.

2.2 История, версии и достоинства

2.2.1 Язык программирования Python

Python – это мощный и популярный язык программирования, который был создан Гвидо ван Россумом и впервые выпущен в конце 1980-х годов.

Работа над Python началась в конце 1980-х годов, и первая публичная версия, Python 0.9.0, была выпущена в 1991 году. Основными целями создания Python были читаемость кода, простота и ясность.

Версии Python 2.x стали очень популярными и использовались в различных проектах. Однако, с развитием языка возникли некоторые проблемы, связанные с совместимостью и дизайном.

Python 3.x (2008 – н.в.): Python 3.x был выпущен в 2008 году и представил множество улучшений и изменений, направленных на устранение проблем, обнаруженных в Python 2.x. Это привело к некоторым разрывам совместимости, но с течением времени многие проекты перешли на Python 3.x.

Python имеет множество версий, но две основные ветки – Python 2.x и Python 3.x. Каждая из этих веток имеет свои подверсии, и каждая версия вносит улучшения и изменения. Кроме того, Python активно развивается, и новые версии регулярно выпускаются с улучшениями и новыми возможностями.

Python пользуется широкой популярностью благодаря ряду выдающихся достоинств:

1 Простота и читаемость: Python известен своей простотой и легкочитаемым синтаксисом, который делает код более понятным и поддерживаемым.

2 Множество библиотек и фреймворков: Python имеет богатую экосистему библиотек и фреймворков для разработки, что упрощает создание разнообразных приложений.

3 Кроссплатформенность: Python поддерживает множество операционных систем, что делает его доступным на различных платформах.

4 Широкое применение: Python применяется в разных областях, включая веб-разработку, научные вычисления, искусственный интеллект, анализ данных и многое другое.

5 Активное сообщество: Python имеет активное сообщество разработчиков и пользователей, что обеспечивает поддержку и развитие языка.

Python представляет собой мощный язык программирования, ориентированный на читаемость и простоту. Его обширная экосистема, кроссплатформенность, множество библиотек, и широкое применение в различных областях делают его популярным выбором, поддерживаемым активным сообществом разработчиков.

2.2.2 Среда разработки PyCharm

PyCharm – это одна из самых популярных сред разработки для языка программирования Python. Она была создана компанией JetBrains и впервые выпущена в 2010 году.

Работа над PyCharm началась в JetBrains как ответ на растущую популярность языка Python. Целью было создание мощной и интуитивно понятной среды разработки для Python-разработчиков.

Первая версия PyCharm была выпущена в феврале 2010 года. Она предоставила множество возможностей, таких как интегрированный отладчик, автодополнение кода, управление виртуальными окружениями и многое другое.

С течением времени PyCharm стала одной из самых популярных сред разработки Python и продолжает активно развиваться и совершенствоваться.

PyCharm имеет несколько версий, каждая из которых предоставляет разные уровни функциональности и целевые аудитории.

Версии PyCharm:

1 PyCharm Community Edition: бесплатная версия PyCharm, предназначенная для небольших проектов и индивидуальных разработчиков.

2 PyCharm Professional Edition: платная версия PyCharm, предоставляющая расширенные возможности для профессиональных разработчиков и команд.

PyCharm пользуется выдающимися достоинствами, которые делают ее одной из предпочтительных сред разработки Python:

1 Мощный редактор кода: PyCharm обладает мощным и интуитивно понятным редактором кода с функциями автодополнения, подсветки синтаксиса и другими инструментами для удобной работы с кодом.

2 Интегрированный отладчик: встроенный отладчик позволяет проще и быстрее находить и устранять ошибки в коде.

3 Управление виртуальными окружениями: PyCharm облегчает создание и управление виртуальными окружениями, что полезно при работе с разными проектами и библиотеками.

4 Поддержка множества фреймворков: PyCharm поддерживает различные фреймворки, включая Django, Flask, PyQT и другие, упрощая разработку веб-приложений и других приложений.

5 Анализ кода и рефакторинг: Среда предоставляет множество инструментов для анализа кода и проведения рефакторинга, что помогает улучшить качество кода.

PyCharm, разработанная JetBrains, стала одной из ведущих сред разработки для Python, предоставляя мощные инструменты, включая интегрированный отладчик, управление виртуальными окружениями и поддержку различных фреймворков. Ее популярность обусловлена высокой функциональностью и продолжающимся развитием.

2.2.3 Операционная система Windows

Операционная система Windows разработана корпорацией Microsoft и является одной из самых распространенных операционных систем в мире. Ее история насчитывает несколько десятилетий, и вот ключевые моменты ее развития:

1 Windows 1.0 (1985): первая версия Windows была выпущена в 1985 году. Она предоставляла графический интерфейс пользователя (GUI) для MSDOS и включала приложения, такие как калькулятор и блокнот.

2 Windows 3.0 (1990): Windows 3.0 внесла множество улучшений, включая поддержку цветовых экранов и возможность мультитаскинга.

3 Windows 95 (1995): Windows 95 была революционным выпуском, введя значительные изменения в интерфейс пользователя и поддерживая 32-битные приложения.

4 Windows XP (2001): Windows XP была одной из самых долгожданных версий Windows и предоставила стабильность и производительность.

5 Windows 7 (2009): Windows 7 улучшила интерфейс пользователя и внесла улучшения в производительность.

6 Windows 8 и 8.1 (2012): эти версии внесли радикальные изменения в интерфейс, с учетом сенсорных устройств.

7 Windows 10 (2015): Windows 10 была представлена как универсальная операционная система для ПК, планшетов и смартфонов.

Windows обладает рядом значительных преимуществ и достоинств:

1 Популярность и совместимость: Windows является самой популярной операционной системой, что обеспечивает широкую совместимость с программным обеспечением и аппаратным обеспечением.

2 Игровая платформа: Windows является ведущей платформой для игр и обеспечивает доступ к широкому спектру видеоигр.

3 Простота использования: интерфейс Windows обычно считается более интуитивным и легким для освоения, особенно для новых пользователей.

4 Множество функций: Windows предоставляет множество функций, включая встроенные инструменты для работы с офисными приложениями, веб-браузерами и мультимедиа

5 Обновления и поддержка: Microsoft регулярно выпускает обновления и обеспечивает техническую поддержку для Windows, обеспечивая безопасность и стабильность системы.

6 Широкий спектр аппаратных устройств: Windows поддерживает множество устройств, включая настольные ПК, ноутбуки, планшеты и смартфоны.

Windows, с богатой историей развития с 1985 года, остается одной из самых популярных операционных систем, обеспечивая широкую совместимость, удобный интерфейс, множество функций и поддержку различных устройств, подтверждая свою роль в качестве ведущей платформы в мире компьютерных технологий.

2.3 Обоснование выбора платформы

Для разработки браузера были рассмотрены несколько IDE:

– PyCharm;

– Visual Studio.

Был выбран именно PyCharm как основная среда разработки для языка Python. Эта мощная IDE предоставляет обширный набор инструментов для работы с Python, включая отладчик, поддержку виртуальной среды. Также PyCharm очень прост в использовании, не вызывает трудностей и имеет возможность «помогать» программисту в разработке кода, показывая специальные подсказки для упрощения работы.

Курсовой проект был разработан с использованием языка программирования Python и интегрированной среды разработки PyCharm также и последующим причинам. Во-первых, Python был выбран из-за существующего опыта работы с ним и предпочтения этого языка

программирования. Во-вторых, PyCharm была выбрана как среда разработки из-за ее удобства и функциональности, а также из-за предпочтения этой среды разработки.

Этот выбор основывается на комфорте использования и знании инструментов, что способствует более эффективной разработке курсового проекта.

2.4 Анализ операционной системы и программного обеспечения для написания программы

2.4.1 Анализ операционной системы Windows 10

Операционная система Windows 10 представляет собой современное программное обеспечение. Windows 10 предлагает понятный пользовательский интерфейс, который обеспечивает удобство использования. Элементы интерфейса интуитивно понятны для каждого пользователя. Так же данная операционная система является одной из наиболее распространенных операционных систем, что обеспечивает высокую степень совместимости с огромным количеством программного обеспечения. Windows 10 обеспечивает высокую степень гибкости настроек, высокую производительность и оптимизированное управление.

2.4.2 Анализ языка программирования Python

Язык программирования Python является языком высокого уровня, который славится своей читаемостью и простотой синтаксиса. Python обладает обширной стандартной библиотекой и возможностью использовать огромное количество библиотек из удаленного доступа. Данный язык программирования поддерживает процедурное, объектно-ориентированное и функциональное программирование, что обеспечивает возможность выбора наилучшего подхода для различных задач. Так же Python обладает возможностью интеграции с языками низкого уровня, написанными на C или C++.

2.4.3 Анализ интегрированная среды разработки PyCharm

PyCharm – это мощный инструмент для создания и поддержки проектов на языке Python. Данная интегрированная среда разработки представляет интуитивно понятный интерфейс, чтоб облегчает работу при работе с ней. PyCharm обеспечивает статический анализ кода, который помогает выявить проблемы быстро, что повышает качество программного кода. Так же данная

среда разработки представляет обширную документация, что упрощает процесс работы с ней и освоение новых возможностей и технологий.

Рассмотренные компоненты информационной системы, включая операционную систему Windows 10, язык программирования Python и интегрированную среду разработки PyCharm, являются современными и эффективными инструментами для разработки и выполнения программных проектов.

Интегрированная среда разработки PyCharm дополняет опыт разработки на Python, предоставляя интуитивно понятный интерфейс, статический анализ кода и обширную документацию. Эти функциональности улучшают процесс разработки, помогают выявлять ошибки на ранних этапах и обеспечивают удобство работы с проектами.

В целом, использование Windows 10, языка программирования Python и интегрированной среды разработки PyCharm обеспечивает современные и эффективные условия для создания, поддержки и оптимизации программных проектов. Комбинация этих компонентов обеспечивает удобство использования, высокую производительность и возможность выбора наилучших подходов к различным задачам разработки.

3 ТЕОРЕТИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ПРОГРАММНОГО ПРОДУКТА

3.1 Обоснование необходимости разработки

Существуют веб-браузеры, которые обладают широким набором функций, но не всегда соответствуют индивидуальным потребностям пользователя. Разработка браузера с ограниченным, но важным функционалом, позволит пользователям создать минималистичные персонализированные рабочие пространства, адаптированные под их конкретные предпочтения.

Создание вкладок и группировка их внутри приложения поможет пользователям эффективно организовать пользовательскую онлайн-деятельность. Пользователи смогут легко отслеживать и управлять своими рабочими пространствами, что повысит их производительность и удобство использования интернета.

Интеграция нескольких поисковых системы в приложении позволит пользователям быстро и удобно осуществлять поиск информации.

Добавление функции управления закладками в приложении обеспечит пользователям возможность сохранять и организовывать важные веб-страницы для последующего доступа. Это поможет пользователям быстро находить нужные ресурсы и повысит удобство использования интернета.

Разработка браузера, включающего выбор поисковой системы, создание вкладок, группировку вкладок, расположение на домашней странице вкладок и создание закладок, оправдана необходимостью предоставления пользователям удобного, персонализированного и эффективного инструмента для работы в интернете. Такое приложение может улучшить организацию онлайн-активности, повысить доступность и удобство работы с поиском необходимой информации.

3.2 Технологии программирования, используемые для решения поставленных задач

Для разработки собственного браузера в рамках курсового проекта используются инструменты, описанные во втором разделе:

1 *Python*: язык программирования *Python* выбран в качестве основного языка разработки браузера. *Python* предоставляет удобный синтаксис, богатую стандартную библиотеку и обширное сообщество разработчиков,

что облегчает создание и поддержку проекта. Этот язык также поддерживает множество библиотек для работы с аппаратным обеспечением и мониторинга системы.

2 *PyCharm*: интегрированная среда разработки *PyCharm* используется для разработки, отладки и тестирования браузера. *PyCharm* предоставляет мощные инструменты для написания кода, рефакторинга, автодополнения и управления проектом. Его функциональность позволяет значительно упростить и ускорить процесс разработки.

3 *Windows* 10: проект будет разрабатываться и тестироваться на операционной системе *Windows* 10, что обеспечит совместимость с широким кругом пользователей.

Кроме того, в разработке браузера был использован модуль языка программирования Python под названием PyQt5.

Выбор *Python*, *PyCharm* и операционной системы *Windows* 10 для разработки браузера обеспечивает удобство программирования, эффективные инструменты разработки и широкую совместимость, а использование библиотеки PyQt5 позволяет создать удобный и комфортный интерфейс браузера.

3.2.1 Технологии языка *Python*

В результате разработки курсового проекта были рассмотрены возможности языка программирования Python, а также библиотека PyQt5.

Язык программирования Python характеризуется простым и понятным синтаксисом, обширной стандартной библиотекой и большим количеством сторонних модулей и библиотек.

Библиотека PyQt5 является оболочкой для Qt, мощного фреймворка для создания кроссплатформенных графических пользовательских интерфейсов. PyQt5 позволяет разработчикам создавать красивые и функциональные приложения с помощью Python. К особенностям PyQt5 относятся:

– кроссплатформенность;

– богатые возможности GUI;

– простота использования;

– обширная документация.

3.3 Связь архитектуры вычислительной системы с разрабатываемым программным обеспечением

Архитектурные характеристики аппаратного обеспечения, такие как процессор, оперативная память, накопитель (*SSD* или *HDD*), и видеокарта, играют важную роль в разработке браузера. Эти аппаратные характеристики оказывают существенное влияние на производительность и функциональность браузера, а также определяют его способность эффективно обрабатывать различные типы контента и выполнять сложные задачи.

Процессор выполняет вычислительные операции и обрабатывает инструкции, необходимые для работы браузера. Более мощный процессор позволяет браузеру быстрее загружать и отображать веб-страницы, обрабатывая скрипты, выполнять сложные вычисления и работать с мультимедийным контентом, таким как видео и аудио.

Оперативная память играет ключевую роль в работе браузера, поскольку в ней хранится активное содержимое веб-страниц, открытых вкладок, кэш и другие данные. Больший объем оперативной памяти позволяет браузеру эффективнее управлять множеством открытых вкладок, уменьшает задержки при загрузке и улучшает общую производительность приложения.

Накопитель хранит файлы браузера, включая кэш, историю посещений, загруженные файлы и другие данные.

Видеокарта отвечает за обработку и отображение графики, что важно для работы с веб-графикой, видео и другими мультимедийными элементами. Мощная видеокарта позволяет браузеру плавно отображать анимации, визуализации и видео на веб-страницах.

Все эти аппаратные характеристики взаимодействую с программным обеспечением браузера, определяя его производительность, скорость работы и возможности.

4 ПРОЕКТИРОВАНИЕ ФУНКЦИОНАЛЬНЫХ ВОЗМОЖНОСТЕЙ ПРОГРАММЫ

4.1 Обоснование и описание функций программного обеспечения

Браузер – это специальная программа, которая позволяет пользователям просматривать веб-страницы, загружать их с серверов в интернете и отображать содержимое на экране компьютера.

Браузеры обычно предоставляют стандартные функции, такие как адресная строка для ввода URL-адресов, кнопки для навигации вперед и назад, возможность добавления закладок, а также инструменты для управления вкладками.

Браузеры отвечают за обработку HTML, CSS и JavaScript кода веб-страниц и их отображение на экране устройства пользователя.

В данном курсовом проекте был разработан минималистичный браузер. В функциональность браузера были включены:

– возможность создания вкладок;

– группировка вкладок;

– сохранение закладок;

– смена системы поиска.

Вид разработанного в рамках курсового проекта браузера представлен на рисунке 4.1.

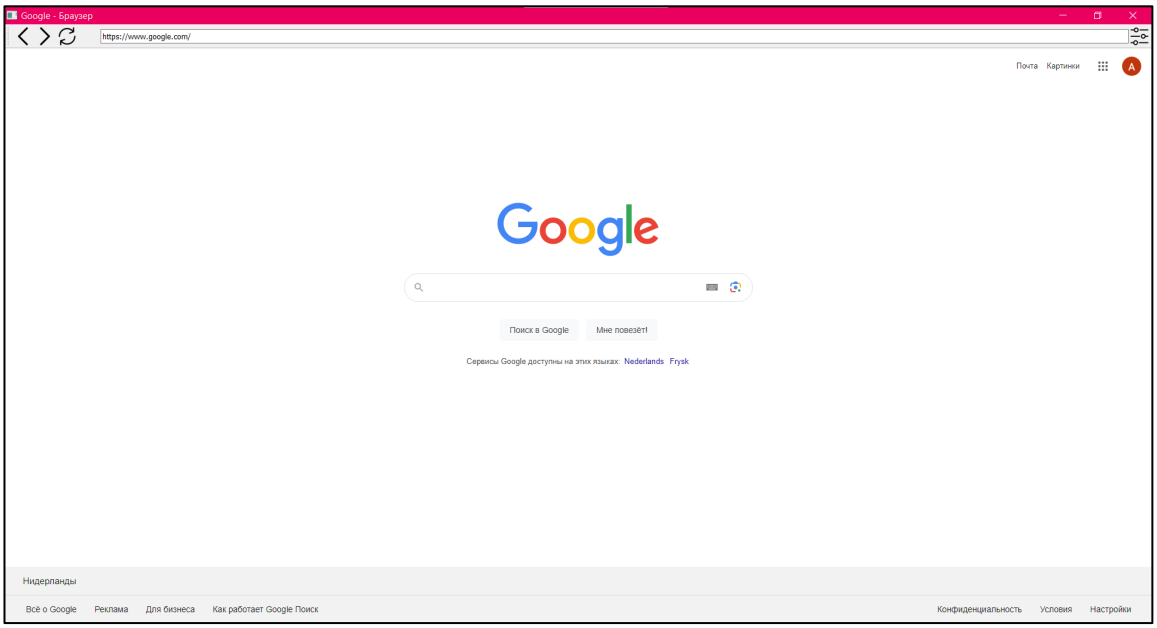

Рисунок 4.1 – Вид браузера

Браузер, разработанный в результате данного курсового проекта, реализует необходимый базовый функционал для комфортного использования. Описание функций включает в себя:

1 Создание вкладок: пользователи могут создавать новые вкладки для одновременного просмотра нескольких веб-страниц. Это позволяет удобно организовывать доступ к различным ресурсам в интернете без необходимости открытия нового окна браузера.

2 Группировка вкладок: браузер предоставляет возможность группировать вкладки для организации сессии. Пользователи могут собирать вкладки по темам, проектам или другим критериям, что упрощает навигацию и управление множеством открытых страниц.

3 Сохранение закладок: пользователи могут сохранять интересующие и часто посещаемые веб-страницы в виде закладок. Это позволяет быстро получать доступ к важной информации без необходимости повторного ввода адреса сайта в адресной строке или повторного поиска.

4 Смена системы поиска: браузер предоставляет возможность выбора поисковой системы, используемой для поиска информации в интернете. Пользователи могут выбирать между различными поисковыми системами в зависимости от своих предпочтений и потребностей.

Данный браузер был разработан с учетом простоты и удобства использования, а также обеспечения основного функционала, необходимого для навигации в интернете. Он позволяет пользователям управлять сессиями, организовывать доступ к важной информации и настраивать поиск в соответствии с их потребностями.

4.1.1 Цели и задачи программного обеспечения

Цели и задачи программного обеспечения – это ключевые аспекты, определяющие его функциональность и направление разработки. К целям разработанного программного обеспечения относятся:

1 Предоставление удобного и минималистичного интерфейса: браузер разрабатывается с целью предоставить пользователям простой и интуитивно понятный интерфейс, который будет удобен в использовании даже для малоопытных пользователей.

2 Обеспечение основного функционала: основная цель браузера – это обеспечить пользователям необходимые инструменты для просмотра веб-страниц.

К задачам разработанного программного обеспечения относятся следующие аспекты:

1 Реализация функционала создания и управления вкладками: данная реализация включает в себя разработку механики создания новых вкладок, переключения между ними, а также группировку и управление ими для удобства пользователя.

2 Разработка системы сохранения и управления закладками: данная реализация включает в себя создание механизма сохранения важных веб-страниц в виде закладок, а также предоставлении инструментов для их организации и доступа.

3 Возможность выбора поисковой системы: данная реализация включает в себя разработку функционала, который позволит пользователям выбирать предпочтительную поисковую систему для выполнения поисковых запросов.

Цели и задачи программного обеспечения определяют его направление разработки, позволяют сосредоточится на достижении ключевых целей.

4.1.2 Ожидаемые результаты и выгоды для пользователя

Использование браузера, разработанного в ходе выполнения курсового проекта, приносит ряд конкретных выгод и результатов для пользователей:

1 Удобство использования: пользователи могут легко создавать вкладки, группировать их и сохранять закладки, что значительно упростит процесс навигации и организации работы в интернете.

2 Повышенная производительность: браузер предоставит быстрый доступ к важной информации, позволяя пользователям быстро переключаться между вкладками и мгновенно получать доступ к закладкам и поисковым результатам.

3 Персонализация: выбор поисковой системы и возможность создания групп вкладок позволят пользователям настроить браузер согласно своим потребностям и предпочтениям, создавая персонализированное пространство для работы в интернете.

4 Экономия времени: браузер позволит пользователям быстро находить нужную информацию, благодаря возможности сохранения закладок и использования выбранной поисковой системы без необходимости каждый раз вводить запросы заново.

Итак, использование разработанного браузера будет способствовать улучшению пользовательского опыта, что сделает работу в интернете более приятной и продуктивной.

5 АРХИТЕКТУРА РАЗРАБАТЫВАЕМОЙ ПРОГРАММЫ

5.1 Общая структура программы

Браузер, разработанный при работе над курсовым проектом, представлен в виде десктопного приложения, который пользователь сможет использовать по своему назначению. Интерфейс браузера предоставляет множество компонентов для работы с приложением и возможность просматривать веб страницы.

К основным компонентам браузера относятся:

1 Адресная строка: адресная строка – это строка для ввода URL-адресов. При помощи данной строки пользователь может найти необходимую страницу по ее URL-адресу, либо найти веб-страницы необходимой тематики в выбранной поисковой системе.

2 Кнопки навигации: к кнопкам навигации относятся кнопки «назад», «вперед» и «перезагрузить». При помощи данного механизма пользователь может возвращаться назад на страницу, которую он уже посещал, либо вперед на страницу, на которой он только что был. Если пользователю необходимо обновить информацию или вид текущей страницы, он может воспользоваться кнопкой «перезагрузить».

3 Меню настроек: в меню настроек пользователь может выбрать необходимую для него поисковую систему.

Структура программы следует принципам объектно-ориентированного программирования. Код программы разделен на модули и классы.

Таким образом, общая структура программы ориентирована на предоставление пользователю минималистичного браузера с базовым функционалом.

5.2 Описание функциональной схемы программы

Функциональная схема алгоритма, реализующего программное средство, представлена в приложении Б. Функциональная схема включает в себя следующие ключевые компоненты и функции:

– инициализация;

– обработка пользовательского ввода;

– навигация;

– управление вкладками;

– управление историей просмотров;

– поиск.

Инициализация включает в себя загрузку основных компонентов браузера и создание окна браузера и интерфейса пользователя.

Обработка пользовательского ввода включает в себя ожидание действий пользователя, а именно нажатия кнопок, ввода текста, выбора опций, а также включает в себя обработку пользовательского ввода и вызов соответствующих функций.

Навигация включает в себя обработку запросов на загрузку веб-страниц, открытие новых вкладок при запросе пользователя или через ссылки на веб-страницах, загрузку и отображение запрошенных веб-страниц.

Управление вкладками включает в себя создание новых вкладок по запросу пользователя, закрытие вкладок по запросу пользователя или при завершении сеанса браузера, переключение между открытыми вкладками.

Управление закладками включает в себя добавление веб-страниц в список закладок по запросу пользователя, удаление веб-страниц из списка закладок и отображение списка закладок, возможность выбора из них.

Управление историей просмотров включает в себя сохранение истории просмотров посещенных веб-страниц, отображение истории просмотров и возможность выбора из нее.

Поиск включает в себя выполнение поисковых запросов, включая выбор поисковой системы, а также отображение результатов поиска и возможность перехода к найденным веб-страницам.

ЗАКЛЮЧЕНИЕ

СПИСОК ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ

[1] Архитектура вычислительных систем [Электронный ресурс]: учебное пособие – Эл. изд. – Электрон. текстовые дан. (1 файл pdf: 77 с.). – Грейбо С.В., Новоселова Т.Е., Пронькин Н.Н., Семенычева И.Ф. 2019. – Режим доступа: http://scipro.ru/conf/computerarchitecture.pdf – Дата доступа: 14.02.2024.

[2] Таненбаум, Э. Архитектура компьютера – 6-е изд. / Э. Таненбаум. СПб. : Питер, 2013. – 816 с.

[3] Петцольд, Ч. Код. Тайный язык информатики / Ч. Петцольд. – М. : Манн, Иванов и Фербер, 2021. – 448 с.

[4] Харрис, Д. М. Цифровая схемотехника и архитектура компьютера – 2-е изд. / Д. М. Харрис, С. Л. Харрис. – NY : Elsevier Inc, 2013. – 1662 с.

[5] Шитов, В. Н. Windows 10 : самый простой и понятный самоучитель / В. Шитов. – М. : Эксмо, 2023. – 464 с.

[6] Русинович, М. Внутреннее устройство Windows – 7-е изд. / М. Русинович [и др.]. – СПб. : Питер, 2018. – 944 с.

[7] Как работает GPU [Электронный ресурс]. – Режим доступа: https://coremission.net/gamedev/kak-rabotaet-gpu – Дата доступа: 15.02.2024.

[8] Обзор процессора Intel Core i5-10300H [Электронный ресурс]. – https://askgeek.io/ru/cpus/Intel/Core-i5-10300H#google_vignette – Дата доступа: 14.03.2024.

[9] Обзор видеокарты NVIDIA GeForce GTX 1650 [Электронный ресурс]. – https://3dnews.ru/987707/obzor-nvidia-geforce-gtx-1650 – Дата доступа: 14.03.2024.

[10] SSD Micron 2210 [Электронный ресурс]. – https://3dnews.ru/987707/obzor-nvidia-geforce-gtx-1650 – Дата доступа: 14.03.2024.

[11] Попов, А. Администрирование Windows с помощью WMI и WMIC. / А. В. Попов, Е. А. Шикин. СПб. : БХВ-Петербург, 2004. – 752 с.

[12] PyCharm [Электронный ресурс]. – Режим доступа: https://www.jetbrains.com/help/pycharm/quick-start-guide –Дата доступа: 15.02.2024.

[13] Windows – что это такое? [Электронный ресурс]. – Режим доступа: https://internet-lab.ru/windows_os – Дата доступа: 15.02.2024.

[14] Инструментарий управления Windows [Электронный ресурс]. – Режим доступа: https://learn.microsoft.com/ru-ru/windows/win32/wmisdk/wmi-start-page – Дата доступа: 14.03.2024.

ПРИЛОЖЕНИЕ А
(обязательное)
Листинг программного кода

Листинг 1 – Программный код класса Request

```
import sys
from PyQt5.QtCore import *
from PyQt5.QtGui import QIcon
from PyQt5.QtWidgets import *
from PyQt5.QtWebEngineWidgets import *

google_url = 'http://google.com'
yandex_url = 'http://yandex.ru'
mail_url = 'http://mail.ru'


class SettingsDialog(QDialog):
    def __init__(self, parent=None):
        super(SettingsDialog, self).__init__(parent)
        self.setWindowTitle('Настройка')
        layout = QVBoxLayout()

        self.search_engines = {
            'Google': google_url,
            'Yandex': yandex_url,
            'Mail.ru': mail_url
        }

        self.radio_buttons = []
        self.selected_url = None

        for engine, url in self.search_engines.items():
            radio_btn = QRadioButton(engine)
            radio_btn.setChecked(url == self.default_search_engine())
            radio_btn.clicked.connect(lambda checked, url=url:
self.on_radio_button_clicked(url))
            self.radio_buttons.append(radio_btn)
            layout.addWidget(radio_btn)

        self.ok_button = QPushButton('OK')
        self.ok_button.clicked.connect(self.save_settings)
        layout.addWidget(self.ok_button)

        self.setLayout(layout)

    def default_search_engine(self):
        return google_url

    def on_radio_button_clicked(self, url):
        self.selected_url = url

    def save_settings(self):
```

```
        self.close()

        if self.selected_url is not None:
            window.browser.setUrl(QUrl(self.selected_url))


class Browser(QMainWindow):

    def __init__(self, *args, **kwargs):
        super(Browser, self).__init__(*args, **kwargs)
        self.browser = QWebEngineView()
        self.browser.setUrl(QUrl("http://google.com"))

        self.homepage = google_url

        self.setCentralWidget(self.browser)

        navtb = QToolBar()
        self.addToolBar(navtb)

        back_btn = QAction("Back", self)
        back_btn.setIcon(QIcon('back.png'))
        back_btn.triggered.connect(self.browser.back)
        navtb.addAction(back_btn)

        next_btn = QAction("Next", self)
        next_btn.setIcon(QIcon('next.png'))
        next_btn.triggered.connect(self.browser.forward)
        navtb.addAction(next_btn)

        reload_btn = QAction("Reload", self)
        reload_btn.setIcon(QIcon('update.png'))
        reload_btn.triggered.connect(self.browser.reload)
        navtb.addAction(reload_btn)

        home_btn = QAction("Home", self)
        home_btn.setIcon(QIcon('home.png'))
        home_btn.triggered.connect(self.go_home)
        navtb.addAction(home_btn)

        self.urlbar = QLineEdit()
        self.urlbar.returnPressed.connect(self.navigate_to_url)
        navtb.addWidget(self.urlbar)
        self.show()

        settings_action = QAction("Settings", self)
        settings_action.setIcon(QIcon('settings.png'))
        settings_action.triggered.connect(self.open_setting_dialog)
        navtb.addAction(settings_action)

        self.browser.urlChanged.connect(self.update_urlbar)
        self.browser.loadFinished.connect(self.update_title)

    def open_setting_dialog(self):
```

```
        dialog = SettingsDialog(self)
        dialog.exec_()

    def update_title(self):
        title = self.browser.page().title()
        self.setWindowTitle("% s - Браузер" % title)

    def navigate_to_url(self):
        q = QUrl(self.urlbar.text())
        if q.scheme() == "":
            q.setScheme("https")
        self.browser.setUrl(q)

    def update_urlbar(self, q):
        self.urlbar.setText(q.toString())
        self.urlbar.setCursorPosition(0)

    def go_home(self):
        self.browser.setUrl(QUrl(self.homepage))


app = QApplication(sys.argv)
window = Browser()
window.show()
app.exec_()
```

ПРИЛОЖЕНИЕ Б
(обязательное)
Функциональная схема алгоритма, реализующего программное средство

ПРИЛОЖЕНИЕ В
(обязательное)
Графический интерфейс пользователя

ПРИЛОЖЕНИЕ Г
(обязательное)
Ведомость документов