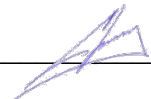


**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук  
Образовательная программа бакалавриата «Программная инженерия»

**СОГЛАСОВАНО**

Преподаватель департамента  
программной инженерии ФКН,  
кандидат компьютерных наук

 С.А.Виденин  
10\_марта\_2024 г.

**УТВЕРЖДАЮ**

Академический руководитель  
образовательной программы  
«Программная инженерия»  
профессор департамента программной  
инженерии, кандидат технических наук

Н.А. Павлочев  
10\_марта\_2024 г.


**ВЫСОКОПРОИЗВОДИТЕЛЬНЫЙ НАСТРАИВАЕМЫЙ HTTP СЕРВЕР**

**Пояснительная записка**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.04.04-01 01-1-ЛУ**

Исполнитель  
студент группы БПИ214

 / Е.К.Фортов/  
10\_марта\_2024 г.

УТВЕРЖДЕН  
RU.17701729.04.04-01 01-1-ЛУ

**ВЫСОКОПРОИЗВОДИТЕЛЬНЫЙ НАСТРАИВАЕМЫЙ HTTP СЕРВЕР**

**Пояснительная записка**  
**RU.17701729.04.04-01 01-1-ЛУ**

**Листов 26**

<i>Инв. № подл</i>	
<i>Подп. и дата</i>	
<i>Взам. инв. №</i>	
<i>Инв. № дубл.</i>	
<i>Подп. и дата</i>	

Оглавление

1. ВВЕДЕНИЕ.....	4
1.1. Наименование программы.....	4
1.2. Краткая характеристика области применения.....	4
2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ.....	5
2.1. Документы, на основании которых ведётся разработка.....	5
2.2. Наименование темы разработки.....	5
3. НАЗНАЧЕНИЕ РАЗРАБОТКИ.....	6
3.1. Функциональное назначение.....	6
3.2. Эксплуатационное назначение.....	6
4. ТРЕБОВАНИЯ К ПРОГРАММЕ.....	7
4.1. Требования к функциональным характеристикам.....	7
4.1.1. Требования к составу выполняемых функций.....	7
4.1.2. Требования к организации входных данных.....	8
4.1.3. Требования к организации выходных данных.....	8
4.2. Требования к интерфейсу.....	9
4.3. Требования к надежности.....	9
4.3.1. Требования к обеспечению надежного (устойчивого) функционирования программы.....	9
4.3.2. Время восстановления после отказа.....	9
4.3.3. Отказы из-за некорректных действий оператора.....	9
4.4. Условия эксплуатации.....	10
4.5. Требования к составу и параметрам технических средств.....	10
4.6. Требования к информационной и программной совместимости.....	10
4.6.1. Требования к информационным структурам и методам решения.....	10
4.6.2. Требования к программным средствам, используемым программой.....	10
4.6.3. Требования к исходным кодам и языкам программирования.....	10
4.7. Требования к маркировке и упаковке.....	10
4.8. Требования к транспортировке и хранению.....	11
4.8.1. Требования к транспортировке и хранению программных документов, предоставленных в электронном виде.....	11
4.8.2. Требования к транспортировке и хранению программных документов, представленных в печатном виде.....	11
5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ.....	12
5.1. Предварительный состав программной документации.....	12
5.2. Специальные требования к программной документации.....	12
6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ.....	13
6.1. Ориентировочная экономическая эффективность.....	13
6.2. Предполагаемая потребность.....	13
1. 6.3. Экономические преимущества разработки по сравнению с отечественными или зарубежными аналогами.....	13
7. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ.....	15
7.1. Необходимые стадии разработки, этапы и содержание работ.....	15
7.2. Сроки разработки и исполнители.....	16
8. ПОРЯДОК КОНТРОЛЯ И ПРИЁМКИ.....	17
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ.....	18

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

ПРИЛОЖЕНИЕ 1 .....	20
ПРИЛОЖЕНИЕ 2 .....	20

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 1. ВВЕДЕНИЕ

### 1.1. Наименование программы

Наименование программы – «Высокопроизводительный Настраиваемый HTTP Сервер» («High Perfomance Customizable HTTP Server»).

### 1.2. Краткая характеристика области применения

Данный IT продукт представляет из себя высокоуровневую C++ библиотеку, которая дает возможность быстро проектировать и разворачивать REST API на языке C++, минуя такие низкоуровневые детали, как сокеты, потоки, контексты и т.д.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 2. ОСНОВАНИЕ ДЛЯ РАЗРАБОТКИ

### 2.1. Документы, на основании которых ведётся разработка

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденная академическим руководителем тема курсового проекта».

### 2.2. Наименование темы разработки

Наименование темы разработки – «Высокопроизводительный Настраиваемый HTTP Сервер».

Программа выполняется в рамках темы курсового проекта — «Высокопроизводительный Настраиваемый HTTP Сервер», в соответствии с учебным планом подготовки бакалавров по направлению 09.03.04 «Программная инженерия».

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

### 3. НАЗНАЧЕНИЕ РАЗРАБОТКИ

#### 3.1. Функциональное назначение

Функциональным назначением программы является предоставление программисту возможности быстро и удобно проектировать REST API на языке C++, не погружаясь в детали реализации REST API. Данный продукт является отличным решением для команд разработки, пишущих на C++, так как позволит им не менять стек разработки, в том числе и ЯП, при появлении требования в необходимости наличия REST API для каких бы то ни было целей.

#### 3.2. Эксплуатационное назначение

Многие существующие IT продукты написаны на C++. Этот ЯП славится своей производительностью и универсальностью, однако многие более новые языки (например, Джава, С#, Го) позволяют разрабатывать такие же продукты в разы быстрее.

Зачастую командам невозможно поменять стек разработки по разным причинам (например, слишком большое наследие, функционал которого нельзя перенести на современный технологический стек с точки зрения бизнес-value и затраченного на разработку времени). Чтобы не оказаться вне рынка с текущим «устаревшим» продуктом, его разработчикам приходится имплементировать современные features на старом технологическом стеке. Для одной из таких features, а именно REST API, которое присутствует во многих промышленных системах, и предназначена данная библиотека. Она призвана значительно сокращать время на разработку REST API и добиваться наибольшего значения соотношения «бизнес-value / затраченное на разработку время».

Для подключения данной библиотеки достаточно импортировать один заголовочный файл, который, в свою очередь, будет подключать другие заголовочные файлы. Настраивать зависимости будет система автоматизации сборки проектов CMake. Такой способ подключения является наиболее современным и простым, и именно поэтому многие разработчики придерживаются такого подхода при написании собственных библиотек, фреймворков и модулей.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 4. ТРЕБОВАНИЯ К ПРОГРАММЕ

### 4.1. Требования к функциональным характеристикам

#### 4.1.1. Требования к составу выполняемых функций

Программа должна давать пользователю возможность выполнять следующие функции:

- инстанцировать объект http сервера
- базово конфигурировать http сервер
- наследовать класс http сервера под свои нужды
- создавать status line http ответа из готовых шаблонов: определять методы GET и POST, код возвращаемого значения
- создавать заголовки http ответа из готовых шаблонов: content-type, content-length и т. д.
- создавать тела http ответа из готовых шаблонов, отдельных html файлов
- настраивать кастомное логирование с разными уровнями в отдельный файл
- настраивать кастомное логирование с разными уровнями в syslog
- кешировать http ответы
- обрабатывать ошибки
- читать комментарии в коде сервера, которых будет достаточно для использования всех возможностей сервера
- подключать сервер через .hrr файл с отдельной папкой (где будут все остальные файлы-зависимости лежать), настройка необходимых зависимостей идет через готовый CmakeLists.txt

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



#### 4.1.2. Требования к организации входных данных

Входные данные — это исходный код на C++, в который портируется http сервер (исходный код может быть представлен в файлах с расширениями .cpp, .h, .hpp). В качестве системы автоматизации сборки проекта рекомендуется использовать CMake, так как в таком случае будет намного проще настроить зависимости, необходимые для подключаемого http сервера. Исходный код проекта до подключения данного фреймворка должен компилироваться успешно и проект должен собираться корректно.

В свою очередь, после подключения фреймворка (после успешного подключения всех необходимых для его работы файлов и успешной настройки необходимых зависимостей) в исходном коде создается объект http сервера. При проектировании REST API для создания очередного эндпоинта необходимо воспользоваться соответствующим методом созданного http сервера.

Касательно требований к входным данным, программисту необходимо ознакомиться с внутренней справкой / документацией http сервера, которая исчерпывающе описывает, как с помощью него проектировать REST API.

#### 4.1.3. Требования к организации выходных данных

Результат работы сервера должен представлять собой действующий REST API, а также файлы с логированием. Отследить корректность работы можно с помощью логов, настроенных на максимально возможный уровень — DEBUG 5, а также с помощью непосредственно функционального тестирования написанного REST API. Если в логах была обнаружена хоть одна ошибка, сервер имеет неопределенное поведение.

#### 4.2. Требования к интерфейсу

Графический интерфейс у данного сервера фактически отсутствует, так как все команды прописываются именно в исходном файле.

#### 4.3. Требования к надежности

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

#### **4.3.1. Требования к обеспечению надежного (устойчивого) функционирования программы**

Для устойчивой работы программы необходимо соблюдать ряд организационно-технических мер:

- 1) При компиляции тестируемой программы необходимо включить следующие флаги: -fsanitize=address,undefined -fno-sanitize-recover=all -Wall -Wextra -Werror -std=c++14 -pedantic; Компилировать необходимо компилятором gcc версии не ниже 14 или компилятором clang версии не ниже 3.4;
- 2) Иметь правильно скомпилированные и находящиеся в нужном месте библиотеки, который использует данный ;
- 3) Компиляция исходного кода должна производиться с флагами оптимизации (-O2, -O3 или -Ofast);

#### **4.3.2. Время восстановления после отказа**

Если отказ был спровоцирован внешними факторами (например, поломка энергоблока компьютера или неисправность других его внутренних компонентов), то время исправления ситуации не регламентируется.

Если отказ был спровоцирован внутренними факторами (например, пользователь случайно удалил системный файл и ОС теперь работает некорректно), то время восстановления не должно быть больше времени, необходимого для исправления ошибки с ОС.

#### **4.3.3. Отказы из-за некорректных действий оператора**

Отказ программы возможен также вследствие некорректных действий пользователя при неправильном использовании (например, исходный тестируемый код отрабатывает с ошибкой или предупреждением или в runtime возникло неопределенное поведение). Чтобы такого не допускать, необходимо ознакомиться с пунктом 4.3.1;

Также отказ возможен при некорректном пользовании операционной системой. В таком случае время на восстановления сервера не должно превышать времени, необходимого для устранения поломки ОС.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

#### 4.4. Условия эксплуатации

Компьютер предназначен для эксплуатации в помещениях с искусственно-регулируемыми климатическими условиями, например, в отапливаемых и вентилируемых помещениях категории 4.1 согласно ГОСТ 15150-69 [4].

Программа не требует специального обслуживания.

Программа может быть использована как одним человеком, так и группой лиц. Необходимая квалификация – пользователь (ознакомившийся с краткой справкой сервера).

#### 4.5. Требования к составу и параметрам технических средств

Для бесперебойной работы программного продукта требуется компьютер с:

- установленной версией компилятора gcc - 14, clang — 3.4
- операционной системой со стабильной сборкой, выпущенной не позднее 2015 года
- объемом свободной встроенной памяти не меньше 55 МБ,
- объёмом оперативной памяти не меньше 1 ГБ.

#### 4.6. Требования к информационной и программной совместимости

##### 4.6.1. Требования к информационным структурам и методам решения

Требования к информационным структурам и методам решения не предъявляются.

##### 4.6.2. Требования к программным средствам, используемым программой

Для работы программного продукта требуется gcc компилятор версии не ниже 14 или clang компилятор версии не ниже 3.4; необходимые флаги см. в пункте 4.3.1

##### 4.6.3. Требования к исходным кодам и языкам программирования

Программа должна быть написана на языке программирования C++ версии не выше 14. В качестве среды разработки программы может быть использован любой редактор кода. Допускается писать код только в .cpp, .h и .hpp файлах.

#### 4.7. Требования к маркировке и упаковке

Требования к маркировке и упаковке не предъявляются.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

#### **4.8. Требования к транспортировке и хранению**

##### **4.8.1. Требования к транспортировке и хранению программных документов, предоставленных в электронном виде**

Программные документы загружаются в электронном виде в информационную образовательную среду LMS (Learning Management System) НИУ ВШЭ. Требования к хранению и транспортировке не предъявляются.

##### **4.8.2. Требования к транспортировке и хранению программных документов, представленных в печатном виде**

Программные документы, предоставляемые в печатном виде, должны соответствовать общим правилам учета и хранения программных документов, предусмотренных стандартами Единой системы программной документации и соответствовать требованиям ГОСТ 19.602-78 [13].

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## **5. ТРЕБОВАНИЯ К ПРОГРАММНОЙ ДОКУМЕНТАЦИИ**

### **5.1. Предварительный состав программной документации**

- 1) «Высокопроизводительный Настраиваемый HTTP Сервер». Техническое задание (ГОСТ 19.201-78 [8])
- 2) «Высокопроизводительный Настраиваемый HTTP Сервер». Программа и методика испытаний (ГОСТ 19.301-79 [9])
- 3) «Высокопроизводительный Настраиваемый HTTP Сервер». Текст программы (ГОСТ 19.401-78 [10])
- 4) «Высокопроизводительный Настраиваемый HTTP Сервер». Пояснительная записка (ГОСТ 19.404-79 [11])
- 5) «Высокопроизводительный Настраиваемый HTTP Сервер». Руководство оператора (ГОСТ 19.505-79 [12])

### **5.2. Специальные требования к программной документации**

- 1) Все документы к программе должны быть выполнены в соответствии с ГОСТ 19.106-78 [7] и ГОСТами к каждому виду документа (см. п. 5.1.);
- 2) Пояснительная записка должна быть загружена в систему Антиплагиат через LMS (Learning Management System) НИУ ВШЭ.
- 3) Вся документация и программа также сдаются в электронном виде в формате .pdf или .docx. в архиве формата .rar или .zip.
- 4) За три дня до защиты комиссии все материалы курсового проекта:
  - техническая документация,
  - программный проект,
  - исполняемый файл,
  - отзыв руководителя,
  - лист Антиплагиата

должны быть загружены одним или несколькими архивами в проект дисциплины «Курсовой проект, 3 курс ПИ» в личном кабинете в информационной образовательной среде LMS (Learning Management System) НИУ ВШЭ.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 6. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

### 6.1. Ориентировочная экономическая эффективность

В рамках данной работы расчет экономической эффективности не предусмотрен.

### 6.2. Предполагаемая потребность

Данный сервер могут использовать все разработчики с компилятором gcc версии не ниже 14 или компилятором clang версии не ниже 3.4, которым нужно быстро добавить REST API в свою программу. Данный сервер предлагает простое, быстрое, легковесное и одновременно высокопроизводительное решение данной проблемы, упрощая жизнь разработчикам.

### 6.3. Экономические преимущества разработки по сравнению с отечественными или зарубежными аналогами

На момент создания программы наиболее используемыми аналогами в области http серверов-микрофреймворков являются: Crow, Pistache, Beast.

Общий недостаток всех этих продуктов — недостаточный уровень абстракции для использования их возможностей в условиях ограниченных временных ресурсов. Данные фреймворки предоставляют более сложный интерфейс для создания REST API, нежели текущий http сервер. В итоге в большинстве случаев они требуют больше времени для имплементации той же функциональности, которую предлагает мой сервис.

Если рассматривать найденных «конкурентов» по отдельности, то можно выявить следующие особенности.

Pistache:

1. Сложность и документация: начальная настройка и использование Pistache может потребовать много времени из-за относительно сложной структуры и ограниченной документации.

2. Ограниченные возможности масштабирования: хотя Pistache предлагает хорошую производительность, в некоторых случаях могут возникать ограничения в масштабировании и обработке больших нагрузок, особенно в сравнении с другими более распространенными фреймворками.

Crow:

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

1. Ограничения масштабируемости: В некоторых случаях Crow может иметь ограничения по масштабируемости и производительности в сравнении с иными фреймворками.

2. Отсутствие полной стандартной поддержки: иногда Crow может не поддерживать все стандарты и спецификации, которые могут потребоваться для конкретных задач, требуя дополнительной настройки и расширений.

Beast:

1. Сложность использования: Beast является частью библиотеки Boost, которая является довольно низкоуровневой абстракцией для работы с сетью. С помощью этого инструмента будет сложно разрабатывать REST API «с нуля».

2. Обширная документация: из-за того, что Beast является низкоуровневой основой для построения REST API, то и документация его значительно больше других рассмотренных REST API решений.

Подытожив, можно сказать, что такие инструменты разработки, как данный фреймворк, являются передовыми средствами разработки на современном C++, так как имеют достаточный уровень абстракции, что позволяет разработчикам данного языка повысить свой performance, затрачивая существенно меньше времени на создание таких популярных решений, как REST API.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 7. СТАДИИ И ЭТАПЫ РАЗРАБОТКИ

### 7.1. Необходимые стадии разработки, этапы и содержание работ

Стадии и этапы разработки были выявлены с учетом ГОСТ 19.102-77 [6]:

Таблица 1 – Стадии разработки, этапы и содержание работ

Стадии разработки	Этапы работ	Содержание работ
I. Техническое задание	Обоснование необходимости разработки программы	Постановка задачи
		Сбор исходных материалов
		Выбор и обоснование критериев эффективности и качества разрабатываемой программы
	Разработка и утверждение технического задания	Определение требований к программе
		Определение стадий, этапов и сроков разработки программы и документации на нее
		Определение необходимости проведения научно-исследовательских работ на последующих стадиях
		Согласование и утверждение технического задания
II. Рабочий проект	Разработка программы	Программирование и отладка программы
	Разработка программной документации	Разработка программных документов в соответствии с требованиями ГОСТ 19.101-77 [5]
	Испытания программы	Разработка, согласование и утверждение порядка и методики испытаний
		Проведение предварительных испытаний
		Корректировка программы и программной документации по результатам испытаний

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



Продолжение Таблицы 1

Стадии разработки	Этапы работ	Содержание работ
III. Внедрение	Подготовка и защита программного продукта.	Утверждение даты защиты программного продукта.
		Подготовка программы и программной документации для презентации и защиты.
		Представление разработанного программного продукта руководителю и получение отзыва.
		Загрузка Пояснительной записки в систему Антиплагиат через LMS (Learning Management System) НИУ ВШЭ
		Загрузка материалов курсового проекта в LMS (Learning Management System) НИУ ВШЭ, проект дисциплины «Курсовой проект, 2 курс ПИ» (см. п. 5.2)
		Защита программного продукта (курсового проекта) комиссии.

## 7.2. Сроки разработки и исполнители

Разработка должна закончиться к 25 мая 2024 года.

Исполнитель: Фортов Егор Кириллович, студент группы БПИ214 факультета компьютерных наук НИУ ВШЭ.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 ТЗ				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## 8. ПОРЯДОК КОНТРОЛЯ И ПРИЁМКИ

Проверка программного продукта, в том числе и на соответствие техническому заданию, осуществляется заказчиком совместно с исполнителем согласно «Программе и методике испытаний», а также пункту 5.2

Защита выполненного проекта осуществляется комиссии, состоящей из преподавателей департамента программной инженерии, в утверждённые приказом декана ФКН сроки.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 ТЗ				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

- 1) Pistache — REST API framework [Электронный ресурс] Режим доступа: <https://github.com/pistacheio/pistache>, свободный. (дата обращения: 15.02.2024)
- 2) Crow – REST API framework [Электронный ресурс] Режим доступа: <https://github.com/CrowCpp/Crow>, свободный. (дата обращения: 15.02.2024)
- 3) Beast - REST API framework [Электронный ресурс] Режим доступа: <https://www.boost.org/doc/libs/master/libs/beast/doc/html/index.html>, свободный. (дата обращения: 15.02.2024)
- 4) ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
- 5) ГОСТ 19.101-77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 6) ГОСТ 19.102-77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 7) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 8) ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 9) ГОСТ 19.301-79 Программа и методика испытаний. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 10) ГОСТ 19.401-78 Текст программы. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 11) ГОСТ 19.404-79 Пояснительная записка. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 12) ГОСТ 19.505-79 Руководство оператора. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 13) ГОСТ 19.602-78 Правила дублирования, учета и хранения программных документов, выполненных печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
- 14) Статья про REST API в целом [Электронный ресурс]. Режим доступа: <https://habr.com/ru/articles/483202/>, свободный. (дата обращения: 15.02.2024)
- 15) Статья про REST API в целом [Электронный ресурс]. Режим доступа: <https://aws.amazon.com/ru/what-is/restful-api/>, свободный. (дата обращения: 15.02.2024)
- 16) Статья про REST API в целом [Электронный ресурс]. Режим доступа: <https://blog.skillfactory.ru/glossary/rest-api/>, свободный. (дата обращения: 15.02.2024)
- 17) Статья про REST API [Электронный ресурс] / Wikipedia. Режим доступа: <https://ru.wikipedia.org/wiki/REST>, свободный. (дата обращения: 15.02.2024)
- 18) Статья RESTful APIs [Электронный ресурс]. Режим доступа: <https://www.astera.com/type/blog/rest-api-definition/>, свободный. (дата обращения: 15.02.2024)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 ТЗ				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

19) Видео про REST API [Электронный ресурс]. Режим доступа: <https://www.youtube.com/watch?v=-mN3VyJuCjM> свободный. (дата обращения: 15.02.2024)

20) Статья IBM про REST API [Электронный ресурс] / IBM; Режим доступа: <https://www.ibm.com/topics/rest-apis> свободный. (дата обращения: 15.02.2024)

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 ТЗ				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

**ПРИЛОЖЕНИЕ 1**  
**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ**

Таблица 1.1 – Описание и функциональное назначение классов/структур в файле ServeMe.hpp

Класс	Назначение
Level	Класс необходим для категоризации уровней логирования. Доступные уровни: Debug, Info, Warning, Error, Critical.
Method	Класс необходим для категоризации методов HTTP-запросов. Доступные методы: GET, POST.
HttpServerInterface	Класс представляет собой интерфейс для любого объекта, который претендует на роль HttpServer-a.
LoggerInterface	Класс представляет собой интерфейс для любого объекта, который претендует на роль Logger-a.
HttpSessionInterface	Класс представляет собой интерфейс для любого объекта, который претендует на роль HttpSession.
RESTAPIAPPInterface	Класс представляет собой интерфейс для любого объекта, который претендует на роль RESTAPIAPP – входной точки фреймворка.
Logger	Класс предоставляет возможность логирования в файл и внутренний лог ОС с учетом текущего времени и установленного уровня логирования. Класс реализует интерфейс LoggerInterface.
HttpSession	Класс нужен для реализации http сервера. Отвечает за принятие http-запросов и отправку на них ответов. Класс реализует интерфейс HttpSessionInterface.
HttpServer	Класс является главным в данном фреймворке. Отвечает за создание сокета, открытия его, добавления новых эндпоинтов и прослушивания соединений по протоколу TCP. Класс реализует интерфейс HttpServerInterface.
RESTAPIAPP	Класс является входной точкой подприложения. Предоставляет возможность создать сам HTTP сервер, запустить его, добавить в него обработку новых эндпоинтов и завершить его. Класс реализует интерфейс RESTAPIAPPInterface.

**ПРИЛОЖЕНИЕ 2**  
**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ, МЕТОДОВ, СВОЙСТВ КЛАССОВ, А ТАКЖЕ ПЕРЕМЕННЫХ И ФУНКЦИЙ В ФАЙЛАХ**

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

Таблица 2.1.1 – Описание полей и свойств класса Level

Имя	Модификатор доступа	Тип	Назначение
Debug	public	Enum type	Данный уровень предназначен для максимального уровня логирования.
Info	public	Enum type	Данный уровень предназначен для вывода какой-либо информации, которая носит не критический, но довольно важный характер.
Warning	public	Enum type	Для вывода предупреждений (по аналогии с предупреждениями компилятора).
Error	public	Enum type	Для вывода ошибок, из-за которых отдельная компонента HTTP-сервера не может корректно завершить свою работу.
Critical	public	Enum type	Для вывода ошибок, при которых работа HTTP-сервера невозможна.

Таблица 2.2.1 – Описание полей и свойств класса Method

Имя	Модификатор доступа	Тип	Назначение
GET	public	Enum type	Для ответа на GET-запросы.
POST	public	Enum type	Для ответа на POST-запросы.

Таблица 2.3.1 – Описание методов абстрактного класса HttpServerInterface

Имя	Модификатор доступа	Тип	Аргументы	Назначение
addEndpoint	public	virtual void	const std::string &path, const std::string &response, Method method	Добавляет в HTTP-сервер новый эндпоинт.

Таблица 2.4.1 – Описание методов абстрактного класса LoggerInterface

Имя	Модификатор доступа	Тип	Аргументы	Назначение
log	public	virtual void	Level level, const std::string &message	Логирует переданное ей сообщение в файл и syslog с

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

				переданным уровнем логирования.
--	--	--	--	---------------------------------

Таблица 2.5.1 – Описание методов абстрактного класса HttpSessionInterface

Имя	Модификатор доступа	Тип	Аргументы	Назначение
start	public	virtual void	-	Считывать HTTP-запрос и отвечать на него

Таблица 2.6.1 – Описание методов абстрактного класса RESTAPIAPPInterface

Имя	Модификатор доступа	Тип	Аргументы	Назначение
AddEndpoint	public	virtual void	const std::string &path, const std::string &response, const std::string &method)	Добавляет новый эндпоинт в сервис
RunServer	public	virtual void	-	Дает команду серверу начинать прослушивать входящие соединения
StopServer	public	virtual void	-	Останавливает сервер, закрывая сокет.

Таблица 2.7.1 – Описание полей и свойств класса Logger

Имя	Модификатор доступа	Тип	Назначение
logFile	private	std::ofstream	Нужен для записи в файл.
syslogEnabled	private	const bool	Нужна ли выгрузка в syslog помимо файла.

Таблица 2.7.2 – Описание методов класса Logger

Имя	Модификатор доступа	Тип	Аргументы	Назначение
Logger	public	constructor	const std::string &program_name = "HTTPServer", const std::string &log_file_name = "log.txt", bool	Создает объект, открывает файл логов и системный лог.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

			syslog_enabled = true	
~Logger()	public	destructor	-	Разрушает объект, закрывает файл логов и системный лог.
log	public	void	Level level, const std::string &message	Логирует в файл и системный лог
writeToSyslog	private	void	Level level, const std::string &message	Логирует в системный лог.
writeToFile	private	void	Level level, const std::string &message	Логирует в файл.

Таблица 2.8.1 – Описание полей и свойств класса HttpSession

Имя	Модификатор доступа	Тип	Назначение
socket_	private	boost::asio::ip::tcp::socket	Нужен для открытия сокета.
request_	private	boost::asio::streambuf	Нужен для чтения http-запроса.
endpoints_	private	const std::unordered_map<std::string, std::pair<std::string, Method>>&	Для ответов на соответствующие запросы.
enable_cache	private	const bool	Флаг, который включает кеширование ответов.
logger	private	std::shared_ptr<Logger>	Нужен для логирования.
cache	private	std::unordered_map<std::string, std::string>&	Нужен для кеширования.

Таблица 2.8.2 – Описание методов класса HttpSession

Имя	Модификатор доступа	Тип	Аргументы	Назначение
HttpSession	public	constructor	boost::asio::ip::tcp::socket socket, const endpoint &endpoints, Logger::Ptr logger,	Создает объект, начинает читать входящий запрос.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата



			CACHE& cache, bool enable_cache = true	
~HttpSession	public	destructor	-	Разрушает объект.
do_read	private	void	-	Читает входящий запрос и вызывает функцию для ответа на него.
do_write	private	void	const std::string &response	Отвечает на входящий запрос.

Таблица 2.9.1 – Описание полей и свойств класса HttpSession

Имя	Модификатор доступа	Тип	Назначение
socket_	private	boost::asio::ip::tcp::socket	Нужен для открытия сокета.
request_	private	boost::asio::streambuf	Нужен для чтения http-запроса.
endpoints_	private	const std::unordered_map<std::string, std::pair<std::string, Method>>&	Для ответов на соответствующие запросы.
enable_cache	private	const bool	Флаг, который включает кеширование ответов.
logger	private	std::shared_ptr<Logger>	Нужен для логирования.
cache	private	std::unordered_map<std::string, std::string>&	Нужен для кеширования.

Таблица 2.9.2 – Описание методов класса HttpSession

Имя	Модификатор доступа	Тип	Аргументы	Назначение
HttpServer	public	constructor	boost::asio::io_context &io_context, Logger::Ptr logger, CACHE& cache, short port = 8080, bool enable_cache = true	Создает объект, вызывает функцию, которая слушает соответствующий порт.
~HttpServer	public	destructor	-	Разрушает объект.
addEndpoint	public	void	const std::string &path, const	Добавляет новый эндпоинт.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

			std::string &response, Method method	
do_accept	private	void	-	Открывает сокет и начинает слушать соответствующий порт.

Таблица 2.10.1 – Описание полей и свойств класса RESTAPIAPP

Имя	Модификатор доступа	Тип	Назначение
io_context	private	boost::asio::io_co ntext	Нужна для запуска HTTP- сервера (внутренняя вещь либы boost).
server	private	std::shared_ptr<Ht tpServer>	Сам HTTP-сервер.
logger	private	std::shared_ptr<L ogger>	Инструмент логирования.
cache	private	std::unordered_ma p<std::string, std::string>	Инструмент кеширования.

Таблица 2.10.2 – Описание методов класса RESTAPIAPP

Имя	Модификато р доступа	Тип	Аргументы	Назначение
RESTAPIAPP	public	constructor	uint32_t port = 8080, const std::string& logfileName="log.t xt"	Создает объект, инициализирует его поля.
~RESTAPIAPP	public	destructor	-	Разрушает объект.
AddEndpoint	public	void	const std::string &path, const std::string &response, Method methodconst std::string &path, const std::string &response, const std::string &method="GET"	Добавляет новый эндпоинт.
RunServer	public	void	-	Запускает HTTP- сервер.
StopServer	public	void	-	Останавливает HTTP-сервер.

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.04.04-01 01				
Инв. № подл.	Подп. и дата	Взам. Инв. №	Инв. № дубл.	Подп. и дата

## ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

[illegible]