**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа «Программная инженерия»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| СОГЛАСОВАНО  Приглашенный преподаватель ДПИ ФКН  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.К. Горденко  «10» мая 2023 г. | |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «10» мая 2023 г. | |
| |  |  | | --- | --- | | *Подп. и дата* |  | | *Инв. № дубл.* |  | | *Взам. инв. №* |  | | *Подп. и дата* |  | | *Инв. № подл* | RU.17701729.05.04-01 ТЗ 01-1 | | **АССИСТЕНТ ДЛЯ СБОРКИ ПК**  **Пояснительная записка**  **ЛИСТ УТВЕРЖДЕНИЯ**  **RU.17701729.05.04-01 ТЗ 01-1-ЛУ** | | | | | | |
|  | |  | | | | |
| Исполнитель:  студент группы БПИ219  \_\_\_\_\_\_\_\_\_ / Н.В. Артемов /  «10» мая 2023 г. | | | | |
|  | | | | | | |
|  | | | | |  | |

**Москва 2023**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| УТВЕРЖДЕН  RU.17701729.10.03-01 ТЗ 01-1-ЛУ |  | |  | |
| |  |  | | --- | --- | | *Подп. и дата* |  | | *Инв. № дубл.* |  | | *Взам. инв. №* |  | | *Подп. и дата* |  | | *Инв. № подл* | RU.17701729.05.04-01 ТЗ 01-1 | | **АССИСТЕНТ ДЛЯ СБОРКИ ПК**  **Пояснительная записка**  **RU.17701729.05.04-01 ТЗ 01-1**  **Листов 16** | | | | |
|  | |  | | |
|  | | |
|  | | | | |
|  | | | |  |

**Москва 2023**

ОГЛАВЛЕНИЕ

[1 ВВЕДЕНИЕ 5](#_Toc134655629)

[1.1. Наименование программы 5](#_Toc134655630)

[1.2. Документ, на основании которого ведется разработка 5](#_Toc134655631)

[2 НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 6](#_Toc134655632)

[2.1. Функциональное назначение 6](#_Toc134655633)

[2.2. Эксплуатационное назначение 6](#_Toc134655634)

[2.3. Краткая характеристика области применения программы 6](#_Toc134655635)

[3 ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 7](#_Toc134655636)

[3.1. Постановка задачи на разработку программы 7](#_Toc134655637)

[3.2. Описание алгоритма работы и функционирования программы 7](#_Toc134655638)

[3.2.1. Выбор архитектуры системы и взаимодействия ее компонентов 7](#_Toc134655639)

[3.2.2. Краткое описание алгоритма работы всей программы 8](#_Toc134655640)

[3.2.3. Описание алгоритма работы микросервиса HardwareInfoCollector 9](#_Toc134655641)

[3.2.4. Описание алгоритма работы микросервиса BuildGenerator 9](#_Toc134655642)

[3.2.5. Описание микросервиса Api Gateway и баз данных 9](#_Toc134655643)

[3.3. Описание и обоснование выбора метода организации входных и выходных данных 10](#_Toc134655644)

[4.1.3. Организация входных данных для Api Gateway 10](#_Toc134655645)

[4.1.4. Организация выходных данных для Api Gateway 10](#_Toc134655646)

[4.1.5. Организация входных и выходных данных для BuildGenerator 12](#_Toc134655647)

[3.4. Описание и обоснование выбора состава технических и программных средств 12](#_Toc134655648)

[3.4.1. Состав технических и программных средств 12](#_Toc134655649)

[3.4.2. Обоснование выбора технических и программных средств 13](#_Toc134655650)

[4 ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 14](#_Toc134655651)

[4.1. Ориентировочная экономическая эффективность 14](#_Toc134655652)

[4.2. Предполагаемая потребность 14](#_Toc134655653)

[4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами 14](#_Toc134655654)

[5 ИСТОЧНИКИ ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 15](#_Toc134655655)

[ПРИЛОЖЕНИЕ 1 16](#_Toc134655656)

[ТЕРМИНОЛОГИЯ 16](#_Toc134655657)

# ВВЕДЕНИЕ

## 1.1. Наименование программы

**Наименование темы разработки:** «Ассистент для сборки ПК»

**Наименование темы разработки на английском языке:** «PC Build Assistant»

## 1.2. Документ, на основании которого ведется разработка

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

# НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

## 2.1. Функциональное назначение

Программа предоставляет возможность генерировать список комплектующих для сборки компьютера под разные нужды и бюджет. Кроме того, программа предоставляет возможность получать ссылки на необходимые комплектующие для сборки ПК в популярных интернет-магазинах. Более того, пользователь получает ссылки на комплектующие в магазинах, где цены на данные товары наименьшие среди маркетплейсов-конкурентов.

## 2.2. Эксплуатационное назначение

Основными конечными потребителями разрабатываемого приложения являются люди в возрасте от 14 лет, которым необходимо купить персональный компьютер с наилучшим соотношением цена/качество и которые:

не желают тратить время на изучение рынка компьютерных комплектующих;

не желают тратить время на поиск комплектующих по самым низким ценам

не знают, какие компьютерные комплектующие подобрать под их нужды;

не знают, как подбирать комплектующие для персонального компьютера.

Для корректного использования данной программы пользователю необходимо иметь устройство с доступом в интернет, определиться с бюджетом и целью, с которой будет использоваться собранный персональный компьютер.

## 2.3. Краткая характеристика области применения программы

«Ассистент для сборки ПК» – программа, которая по заданной информации о цели использования компьютера и о необходимом бюджете генерирует сборку ПК. Под сборкой ПК подразумевается список всех необходимых комплектующих, чтобы собрать компьютер. Программа генерирует не только список комплектующих компьютера, но и ссылки на эти товары в интернет-магазинах, где их будет выгоднее приобрести.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## 3.1. Постановка задачи на разработку программы

Разрабатываемое веб-приложение должно выполнять следующие задачи:

1. Анализировать описанные в пункте 4.1.2. настоящего технического задания интернет-магазины с целью обновления цен на комплектующие для компьютера;
2. Обновлять базу данных HardwareDatabase после анализа интернет-магазинов;
3. Генерировать сборку ПК по запросу от Api Gateway;
4. Сохранять сгенерированные сборки ПК в базе данных;
5. Получать сборки ПК из базы данных по уникальному идентификационному ключу.

## 3.2. Описание алгоритма работы и функционирования программы

### 3.2.1. Выбор архитектуры системы и взаимодействия ее компонентов

Чтобы обеспечить гибкость, масштабируемость и возможность качественного тестирования проекта, важно выбрать правильную архитектуру. Для веб-приложений наиболее распространенными архитектурами являются монолитная, SOA, частный случай SOA - микросервисная и Event-Driven.

В данном проекте было принято решение использовать микросервисную архитектуру, которая позволяет разбить приложение на независимые компоненты, облегчая тестирование и развертывание. Реализация данной архитектуры требует разделения кода на отдельные сервисы, каждый из которых имеет свою собственную логику и базу данных.

В проекте было реализовано несколько микросервисов, каждый из которых отвечает за свою функциональность, что обеспечивает гибкость и масштабируемость приложения.

Основную бизнес-логику проекта реализует сервис BuildGenerator, который использует чистую архитектуру для обеспечения гибкости и масштабируемости. Чистая архитектура позволяет создавать приложения, легко тестируемые и расширяемые, за счет выделения независимых компонентов и разделения ответственности между ними.

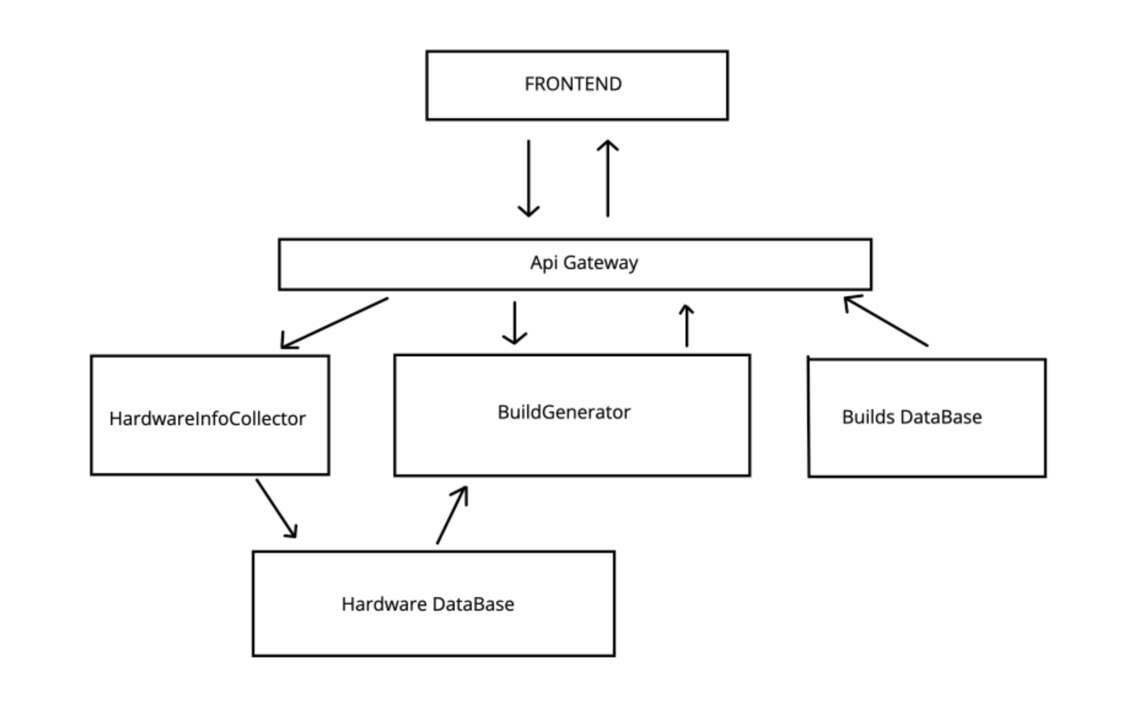
В рамках чистой архитектуры сервис BuildGenerator разделен на три уровня: уровень представления, бизнес-логики и доступа к данным. Каждый уровень выполняет свою функцию и практически не зависит от других уровней, в свою очередь уровень бизнес-логики не имеет в зависимости другие уровни.

Уровень представления отвечает за взаимодействие с пользователем и преобразование данных в формат, удобный для работы с ними на бизнес-уровне. Уровень бизнес-логики содержит логику, связанную с основными функциями сервиса, такими как генерация и построение сборок. Уровень доступа к данным предоставляет интерфейс для работы с данными, используемыми на бизнес-уровне.

Такая архитектура обеспечивает возможность изменения любого уровня без влияния на остальные уровни, что позволяет создавать гибкие и масштабируемые приложения.

Для более четкого понимания архитектуры представлена иллюстрация архитектуры (рис. 1)

Рис.1



### 3.2.2. Краткое описание алгоритма работы всей программы

Микросервис HardwareInfoCollector по запросу от сервиса Api Gateway раз в день в промежутке 00:00-00:30 анализирует интернет-магазины и записывает данные в json файлы в файлах проекта. Далее эти файлы используются для обновления базы данных HardwareDatabase.

Микросервис BuildGenerator по запросу от Api Gateway генерирует сборку и отправляет ее в виде json в теле ответа на запрос.

Api gateway при получении запроса от клиентской части отправляет запрос в микросервис BuildGenerator для получения сборки ПК. Далее происходит сохранение этой самой сборки в базе данных BuildsDatabase и генерация уникального идентификационного ключа для этой сборки. Этот идентификационный ключ нужен для второго типа запроса к приложению – для получения уже сгенерированной сборки.

### 3.2.3. Описание алгоритма работы микросервиса HardwareInfoCollector

Микросервис HardwareInfoCollector реализован в виде MVP (Minimum viable product – минимально жизнеспособный продукт).

При получении запроса от Api Gateway запускается функция анализа интернет-магазинов: читается .txt файл с информацией о компьютерных комплектующих и поочередно загружаются их страницы в интернет-магазинах, далее из html код страницы берется цена и сохраняется вместе с ссылкой в json-файле в коде проекта.

В следующей функции снова читаются .txt файлы и оттуда берется информация о комплектующих и записывается в таблицы базы данных HardwareDatabase.

### 3.2.4. Описание алгоритма работы микросервиса BuildGenerator

При получении запроса от Api Gateway отправляется команда в mediator (Паттерн программирования, который обеспечивает взаимодействие объектов, а в данном случае уровней приложения в чистой архитектуре, без необходимости ссылаться друг на друга. Эта команда обрабатывается на бизнес-уровне приложения и вызывается сервис генерации сборок. В этом сервисе берутся данные из HardwareDatabase через уровень представления данных и генерируется сборка по алгоритму, который будет описан далее.

Алгоритм генерации сборок заключается в выборе комплектующих по эмпирически вычисленным коэффициентам, которые описывают процент бюджета сборки для каждого комплектующего. Далее все комплектующие выбираются максимально приближённо по цене к рекомендуемому коэффициенту, перемноженному на бюджет сборки, также при выборе комплектующего учитывается совместимость с уже выбранными частями сборки компьютера.

Далее эта сборка отправляется в теле ответа на запрос в Api Gateway.

### 3.2.5. Описание микросервиса Api Gateway и баз данных

Общий алгоритм работы микросервиса Api Gateway описан в пункте 3.2.2. настоящей пояснительной записки.

Таблицы в HardwareDatabase создаются при запуске микросервиса BuildGenerator миграцией через пакет “Fluent Migrator”. Данные в эту базу данных заносятся при запуске HardwareInfoCollector без дополнительного анализа интернет-магазинов для поддержания возможности оркестрации и оптимизации нагрузок.

Коллекции в BuildsDatabase создаются автоматически при добавлении первого объекта в базу данных.

## 3.3. Описание и обоснование выбора метода организации входных и выходных данных

### 3.3.1. Организация входных данных для Api Gateway

1. Документ в виде json в теле HTTP Post запроса с полями “type” и “budget”, где поле “type” – это число от 0 до 3 включительно, а поле “budget” – целое число от 50000 до 400000;  
   Пример: {“type”: 0, “budget”: 100000}
2. Уникальный идентификационный номер в адресе HTTP Get запроса.

### 3.3.2. Организация выходных данных для Api Gateway

1. Документ в виде json в теле ответа на HTTP Post запрос со сборкой ПК, ценой сборки и сгенерированным уникальным идентификационным номером;

Пример:

{

"pcBuild": {

"cpu": {

"price": 14700,

"link": "https://www.regard.ru/product/421881/processor-intel-core-i5-12400f-oem",

"model": "intel core i5 12400f"

},

"motherboard": {

"price": 13999,

"link": "https://www.dns-shop.ru/product/3aa7a96c66cced20/materinskaa-plata-msi-mag-b660m-mortar-wifi-ddr4/",

"model": "MSI MAG B660M MORTAR WIFI DDR4"

},

"case": {

"price": 3300,

"link": "https://www.onlinetrade.ru/catalogue/kompyuternye\_korpusa-c1323/aerocool/korpus\_aerocool\_cylon\_mini\_chernyy\_bez\_bp\_4718009152168-1585490.html",

"model": "AeroCool Cylon Mini"

},

"gpu": {

"price": 44299,

"link": "https://www.dns-shop.ru/product/3608e2016095ed20/videokarta-msi-geforce-rtx-3060-ti-ventus-2x-oc-msi-geforce-rtx-3060-ti-ventus-2x-oc/",

"model": "MSI GeForce RTX 3060 Ti VENTUS 2X OC"

},

"cooler": {

"price": 3000,

"link": "https://www.onlinetrade.ru/catalogue/kulery\_dlya\_protsessorov-c1492/id\_cooling/kuler\_dlya\_protsessora\_id\_cooling\_se\_226\_xt\_black-2910152.html",

"model": "ID-COOLING SE-226-XT"

},

"ram": {

"price": 4799,

"link": "https://www.dns-shop.ru/product/24f5a001fad6ed20/operativnaa-pamat-kingston-fury-beast-black-rgb-kf432c16bbak216-16-gb/",

"model": "Kingston FURY Beast Black RGB KF432C16BBAK2/16 16GB"

},

"storage": {

"price": 5830,

"link": "https://www.regard.ru/product/452310/nakopitel-ssd-1tb-kingston-nv2-snv2s-1000g",

"model": "kingston nv2 1000gb"

},

"power\_supply": {

"price": 9090,

"link": "https://www.regard.ru/product/411241/blok-pitaniia-850w-deepcool-pm850d",

"model": "DEEPCOOL PQ850M R-PQ850M-FA0B-EU"

}

},

"totalPrice": 99017,

"id": "6251b6a8-b623-4bda-b664-709b5b98f3eb"

}

1. Документ в виде json в теле ответа на HTTP Get запрос со сборкой ПК и ценой сборки;

Пример:

{

"pc\_build": {

"cpu": {

"price": 14700,

"link": "https://www.regard.ru/product/421881/processor-intel-core-i5-12400f-oem",

"model": "intel core i5 12400f"

},

"motherboard": {

"price": 13999,

"link": "https://www.dns-shop.ru/product/3aa7a96c66cced20/materinskaa-plata-msi-mag-b660m-mortar-wifi-ddr4/",

"model": "MSI MAG B660M MORTAR WIFI DDR4"

},

"case": {

"price": 3300,

"link": "https://www.onlinetrade.ru/catalogue/kompyuternye\_korpusa-c1323/aerocool/korpus\_aerocool\_cylon\_mini\_chernyy\_bez\_bp\_4718009152168-1585490.html",

"model": "AeroCool Cylon Mini"

},

"gpu": {

"price": 44299,

"link": "https://www.dns-shop.ru/product/3608e2016095ed20/videokarta-msi-geforce-rtx-3060-ti-ventus-2x-oc-msi-geforce-rtx-3060-ti-ventus-2x-oc/",

"model": "MSI GeForce RTX 3060 Ti VENTUS 2X OC"

},

"cooler": {

"price": 3000,

"link": "https://www.onlinetrade.ru/catalogue/kulery\_dlya\_protsessorov-c1492/id\_cooling/kuler\_dlya\_protsessora\_id\_cooling\_se\_226\_xt\_black-2910152.html",

"model": "ID-COOLING SE-226-XT"

},

"ram": {

"price": 4799,

"link": "https://www.dns-shop.ru/product/24f5a001fad6ed20/operativnaa-pamat-kingston-fury-beast-black-rgb-kf432c16bbak216-16-gb/",

"model": "Kingston FURY Beast Black RGB KF432C16BBAK2/16 16GB"

},

"storage": {

"price": 5830,

"link": "https://www.regard.ru/product/452310/nakopitel-ssd-1tb-kingston-nv2-snv2s-1000g",

"model": "kingston nv2 1000gb"

},

"power\_supply": {

"price": 9090,

"link": "https://www.regard.ru/product/411241/blok-pitaniia-850w-deepcool-pm850d",

"model": "DEEPCOOL PQ850M R-PQ850M-FA0B-EU"

}

},

"total\_price": 99017

}

### 3.3.3. Организация входных и выходных данных для BuildGenerator

Описание входных и выходных данных для BuildGenerator представлено в виде swagger документации.

## 3.4. Описание и обоснование выбора состава технических и программных средств

### 3.4.1. Состав технических и программных средств

Для стабильного функционирования программы требуется компьютер со следующими компонентами:

1. Процессор Intel 7 поколения или новее или процессор AMD Ryzen 3 поколения или новее
2. 1 гб свободного дискового пространства или более
3. 2 гб ОЗУ или более

Для работы программы необходимы следующие программные средства:

1. Docker engine версии 18.09 или новее

### 3.4.2. Обоснование выбора технических и программных средств

Программа запускается через программное обеспечение docker, а для стабильной работоспособности docker’а необходимо описанное в пункте 3.4.1. технические компоненты.

Для запуска программы необходимо перейти в директорию проекта в терминале и написать команду docker-compose up -d. Для завершения работы программы необходимо в терминале по этой же директории написать docker-compose down. Для запуска клиентской части необходимо перейти в папку frontend в файлах проекта и ввести команды, описанные в файле readme.md в коде проекта.

Все остальные программные средства автоматически загружаются и выстраиваются все зависимости внутри контейнеров, в которых запускаются микросервисы и базы данных.

# ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## 4.1. Ориентировочная экономическая эффективность

Данный курсовой проект не предусматривает расчет экономической эффективности.

## 4.2. Предполагаемая потребность

Предполагаемая потребность обуславливается желанием пользователей купить наиболее хороший компьютер за определенную сумму. Так же предполагается, что многие пользователи приложения не разбираются в компьютерных комплектующих и не могут создать себе сами необходимую сборку, и приложение сделает это за пользователя.

## 4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

На данный момент не существует прямых аналогов разрабатываемого продукта. В таблице 1 приведена сравнительная характеристика с приложениями и веб-сайтами, наиболее близкими к теме разрабатываемого проекта.

# ИСТОЧНИКИ ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. microservices.io (микросервисы) [Электронный ресурс]. Режим доступа свободный: <https://microservices.io/index.html>, (последняя дата обращения: 05.05.2023).

# ПРИЛОЖЕНИЕ 1

# ТЕРМИНОЛОГИЯ

Таблица 1 – терминология

|  |  |
| --- | --- |
| **Термин** | **Значение** |
| Микросервис | Отдельное приложение, модуль общего приложения имеющего микросервисную архитектуру. |
| Микросервисная архитектура | Микросервисная архитектура — вариант сервис-ориентированной архитектуры программного обеспечения, направленный на взаимодействие насколько это возможно небольших, слабо связанных и легко изменяемых модулей — микросервисов. |
| пк/ПК | Персональный компьютер. |
| Сборка | Список комплектующих персонального компьютера. |
| Оператор программы | Человек, который запускает программу и отвечает за бронирование в самом клубе. Обычно это менеджер или управляющий. |
| ID | Идентификационный номер, не всегда число. |
| HardwareInfoCollector | Микросервис для анализа интернет-магазинов, получения информации о комплектующих и обновления HardwareDatabase. |
| HardwareDatabase | База данных для хранения информации о комплектующих. |
| Api gateway | Микросервис для распределения нагрузки и запросов между другими микросервисами. |
| BuildGenerator | Микросервис для генерации сборок ПК. |
| BuildsDatabse | База данных для хранения сгенерированных сборок ПК пользователями. |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**