# СЕКЦИЯ 6
# ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

***К. В. Вознюк, М. А. Яковлев, А. А. Оленников***
*Тюменский государственный университет, г. Тюмень*
**УДК 004.9**

## РАЗРАБОТКА ЗАЩИЩЕННОГО ОБЛАЧНОГО КЛАСТЕРА ДЛЯ ПУБЛИКАЦИИ В ГЛОБАЛЬНУЮ СЕТЬ ПРОГРАММНОГО КОМПЛЕКСА, ВЫПОЛНЯЮЩЕГО СЛОЖНЫЕ ИНЖЕНЕРНЫЕ РАСЧЕТЫ

**Аннотация.** В работе представлена реализация защищенного облачного сервиса для публикации программного комплекса в глобальную сеть, выполняющего сложные инженерные расчеты на основе которых заводом-изготовителем будут создаваться теплообменные аппараты различного назначения.

**Ключевые слова:** веб-сервис, Django, Kubernetes, MinIO, база данных, кластер, программный комплекс, отказоустойчивость, масштабируемость, безопасность, проектирование, расчеты, конфиденциальность, целостность, доступность.

**Введение.** В современном мире информационные технологии играют все более важную роль в различных сферах деятельности человека. Одной из таких сфер является инженерное проектирование, где информационные технологии помогают ускорить и упростить процесс разработки и анализа проекта. Однако, при работе со сложными инженерными данными возникает ряд проблем связанный с конфиденциальностью, целостностью и доступностью таких данных. Речь идет о сложном программном комплексе, предназначенный для проектирования теплообменных аппаратов различной сложности, который содержит в своем составе множество математических моделей и готовых проектных решений. Попытки выпустить это решение на рынок в виде коробочного исполнения с традиционными системами защиты [1], скорее всего через некоторое время приведет к его утрате, продукт будет подвержен несанкционированным воздействиям и в последующем использован третьими лицами. В настоящий момент особый интерес представляет способ защиты программного обеспечения посредством размещения его в облаке [2]. С одной стороны программный комплекс будет размещен на защищенном внутреннем сегменте облака, а пользователям будет доступен только пользовательский интерфейс, с другой стороны исчезнет необходимость в приобретении мощностей компаниям, которые нуждаются в данных расчетах.

**Проблема исследования.** Проблематика работы определяется тем, что для вычисления проектных инженерных работ необходимо большое количество ресурсов для обработки большого объема данных. Помимо этого, нередко инженеры работают в группе и им необходимо получить видимый, для каждого члена группы, результат. При этом необходимо, чтобы данные и результат вычислений видели только авторизованные пользователи, относящиеся к одной группе или компании для того, чтобы избежать раскрытие информации для других пользователей или конкурентных организаций. На данный момент на территории Российской Федерации таковых аналогов нет. В связи с этим, в этой работе предлагается одно из решений,

которое позволит обеспечить безопасное хранение и обработку данных, связанных с проектированием теплообменных аппаратов различной сложности, а также разграничить доступ к информации только для авторизованных пользователей, обеспечить конфиденциальность, высокую доступность, отказоустойчивость и безопасность [3, 4].

В связи с этим были поставлены следующие задачи:

1. Изучить и проанализировать существующие решения.
2. Проработать архитектуру кластера.
3. Проработать архитектуру расширяемых баз данных.
4. Разработать защищенный веб-сервис.
5. Разработать защищенный вычислительный кластер и подключить метрики для отказоустойчивости системы.

**Материалы и методы.** Для публикации в облаке был выбран существующий локальный программный комплекс Radiator (рис. 1), который позволяет проектировать теплообменные аппараты по различным технология с возможностью выбора материалов и оребрений, а также вид нагреваемых и охлаждаемых сред. Данный программный комплекс включается в себя базу экспериментальных значений теплообменных элементов, физические характеристик сред, модели сложного теплообмена, аэро- и гидродинамики, и готовых проектных решений. При этом, сама база, насчитывает свыше 2150 моделей, а расчеты производятся с использованием помощью более 800 математических выражений, что делает данную программный комплекс мощным инструментом для реализации поставленных целей, например для завода-изготовителя.

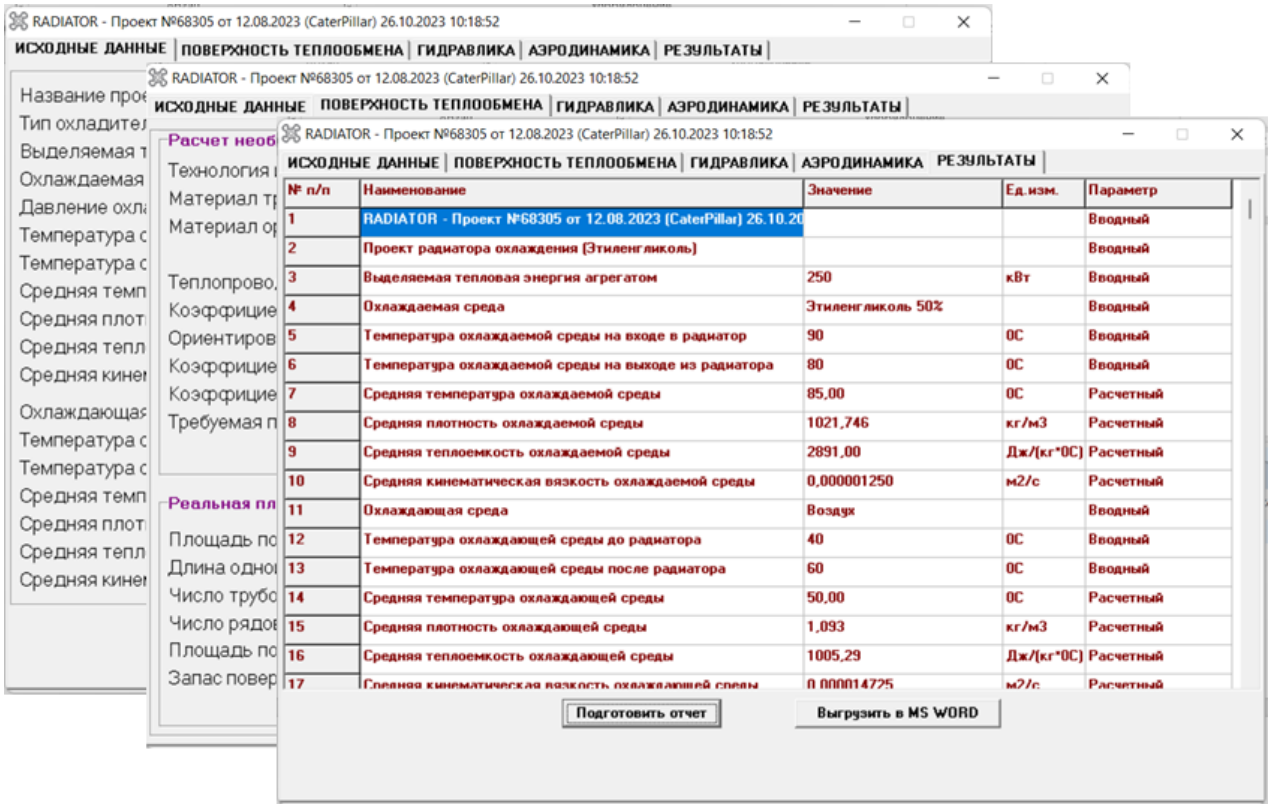


*Рис. 1.* Программный комплекс Radiator

Для реализация данной задачи была разработана архитектура кластера Kubernetes (рис. 2), состоящий из следующих компонентов [5-7]:

– 3 управляющих узлов, отказоустойчивость которых достигается при помощи сервиса типа балансировщик нагрузки под управлением MetalLB;

– 2 рабочих узла на которых будут запускаться веб-сервис и приложения для расчетов;

– 3 узла хранения данных, для обеспечение высокодоступного постоянного хранилища под управлением Longhorn.

Веб-сервис реализован с помощью языка программирования Python с использованием библиотеки Django. В нем находится API для обращения к локальному программному комплексу Radiator. Веб-сервис и программный комплекс располагаются в разных контейнерах. Такой подход позволяет изолировать программный комплекс от пользователей тем самым уберегая его от взлома и последующего использования сторонними лицами.

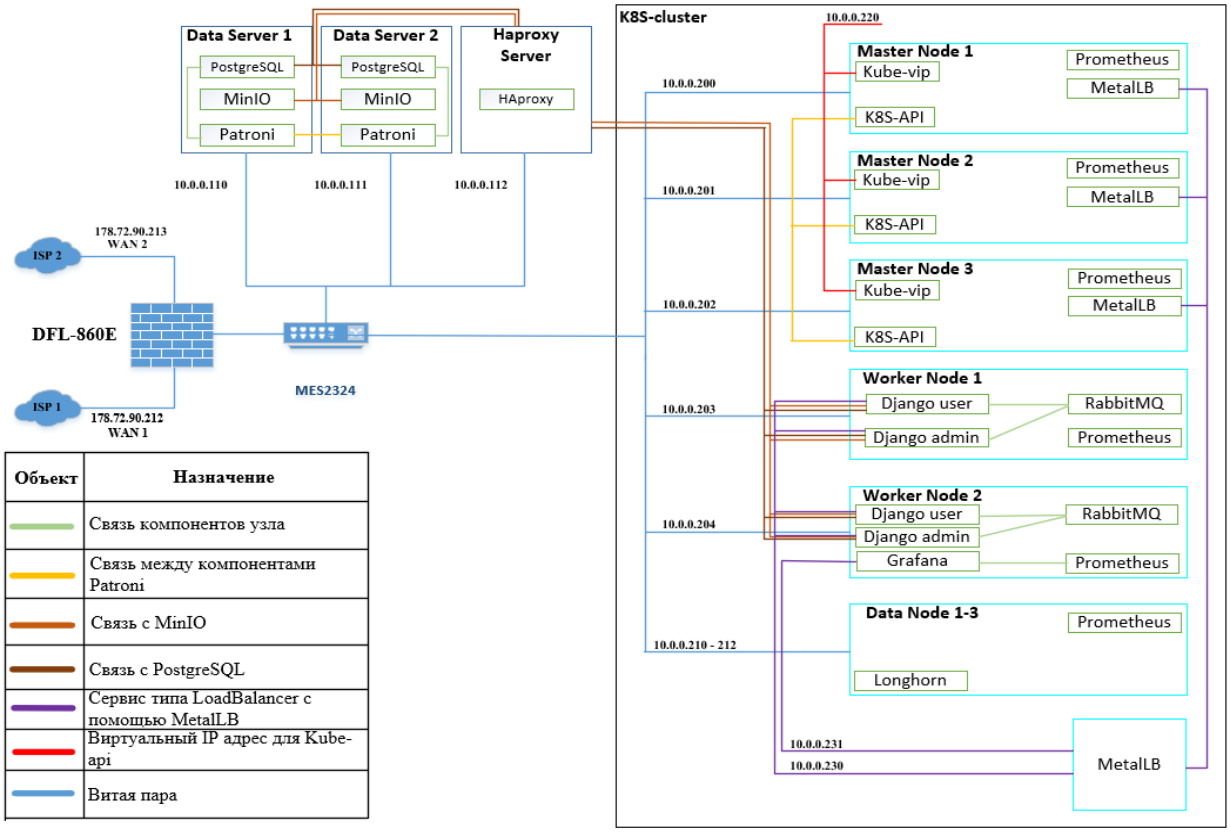


*Рис. 2.* Архитектура кластера Kubernetes

Помимо вышесказанного, на каждом узле находится Prometheus node exporter, который работает в связки с Grafana и служит для сбора метрик с каждого из серверов кластера. Внутри Prometheus и Grafana предусмотрены встроенные сообщения об ошибках, что позволяет администраторам быстро реагировать на инциденты, которые происходят в системе, и своевременно их устранять, обеспечивая работоспособность системы.

Также на каждом узле установлен антивирус Kaspersky Endpoint Security, который управляется централизованно через Kaspersky Security Center (см. рис. 3).

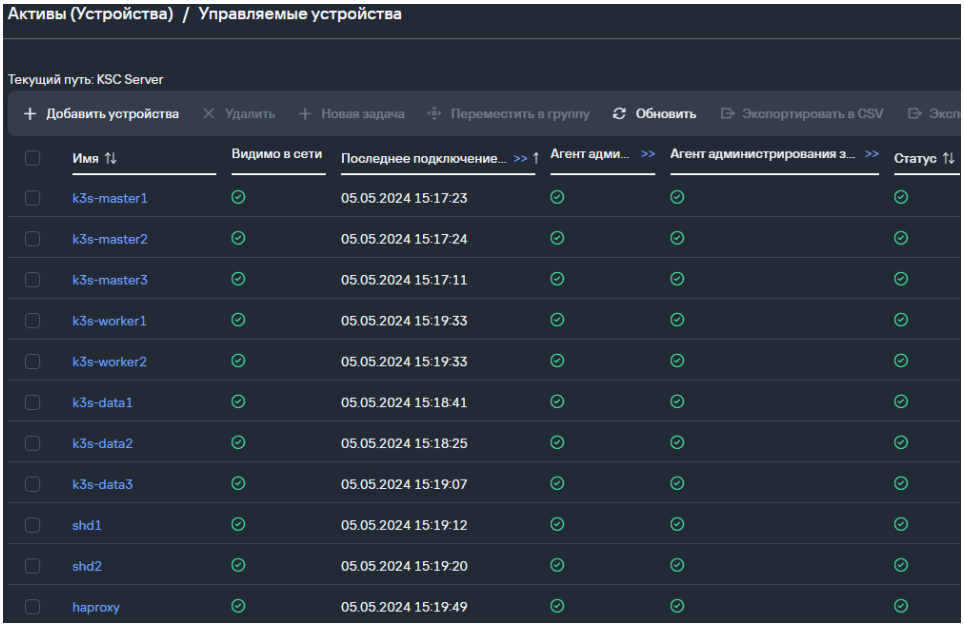


*Рис. 3.* Список управляемых устройств в Kaspersky Security Center

Для хранения образа контейнеров используется локальный реестр с открытым исходным кодом — Harbor, развернутый внутри кластера.

Архитектура баз данных (рис. 4) была построена в соответствии с Федеральным законом 152 «О персональных данных» статья 5 часть 2, 3, в котором говорится, что обработка персональных данных должна ограничиваться достижением конкретных, заранее определенных и законных целей. Помимо этого, не допускается обработка персональных данных, несовместимых с целями сбора персональных данных.

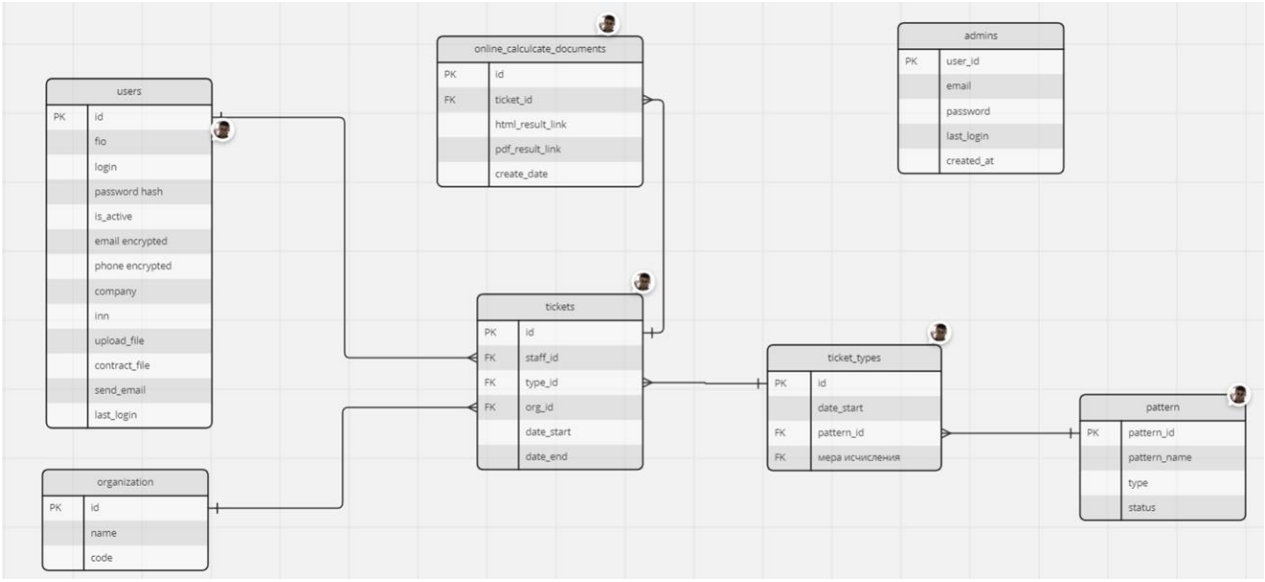


*Рис. 4.* Архитектура базы данных

В связи с этим было принято решение разделить базу данных на две отдельные сущности, а именно база данных для пользователей и для администраторов. База данных администраторов содержит в себе информацию, связанную с администраторами системы и хранит в

себе такую информацию как идентификатор администратора, почта, пароль, логин, дата создания и дата последнего входа в систему.

База данных пользователей содержит в себе информацию, которая прямо или косвенно относится к определенному лицу. А именно это такие данные, как идентификатор пользователя, ФИО, почта, пароль, ИНН организации, можно ли отправлять рассылку на почту и так далее. Важно отметить то, что такие данные как ФИО, почта и пароль шифруются на стороне веб-сервиса и только в последующем сохраняются в базу данных. Таким образом обеспечивается сохранность данных пользователей от возможной компрометации данных.

В базе данных пользователей основополагающей является таблица заявок. Так как именно там хранятся основные атрибуты, такие как идентификатор заявки, идентификатор используемого шаблона, идентификатор пользователя, который делает запрос на расчет данных [8]. Помимо этого, там также хранится идентификатор организации и дата начала и завершения расчетов.

После, полученные результаты на стороне веб сервиса, а именно отчет, сохраняется на сервере хранения данных MinIO (рис. 5) с использованием библиотеки MinIO. Подключение к серверу хранение данных со стороны веб-сервиса происходит с помощью хранящихся константных данных в переменных окружения. Ссылка для доступа до данного отчета сохраняется в таблице online_calculcate_documents в двух удобных форматах для пользователей. А именно в docx и pdf форматах. Помимо этого, за данными отчетами там так же прикрепляется номер заявки, по которой происходили расчеты и дата генерации отчета.

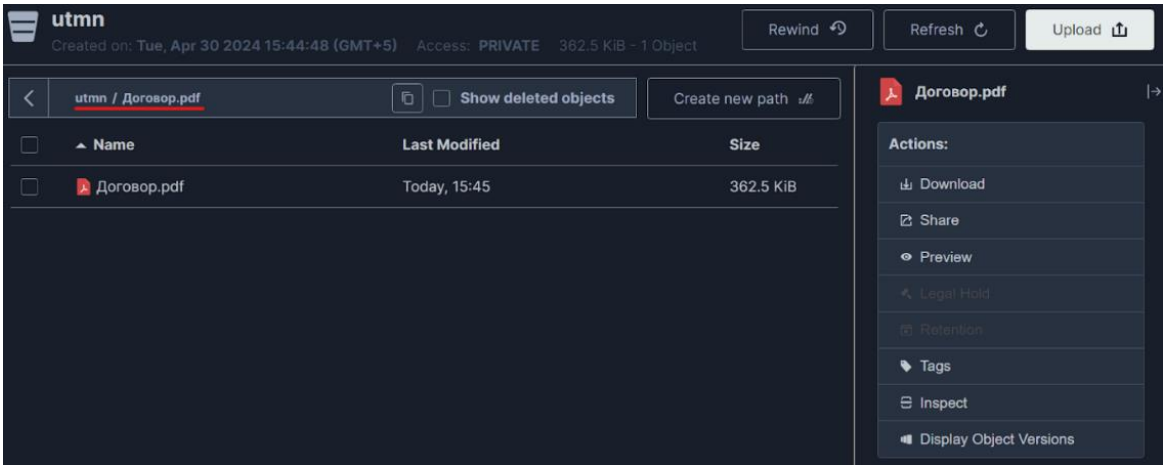


*Рис. 5.* Хранение файлов контрактов в MinIO

Веб-сервис реализован посредством разделения приложения на два отдельных сервиса. А именно сервис пользователей (users) и администраторов (admins).

Сервис пользователей предоставляет им доступ к основным ресурсам сайта, а точкой входа выступает основной домен https://riotsolvers.ru. Он включает в себя регистрацию, аутентификацию, создание и просмотр созданных заявок с возможностью сохранения отчета локально, взаимодействие с другими специалистами из одной и той же компании и управление профилем.

Таким образом после регистрации (см. рис. 6) и одобрением регистрации администратором, пользователю открывается доступ к ресурсам сайта.

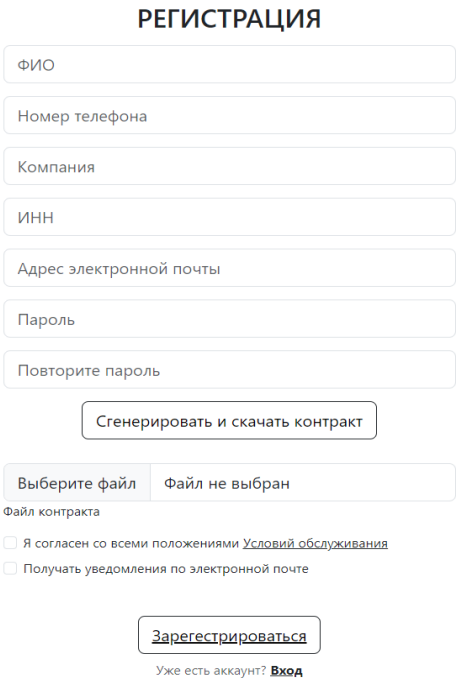


*Рис. 6.* Регистрация пользователей

Основное, с чем предстоит работать пользователю, это заявки. Сайт предлагает ему возможность либо просмотреть уже существующие заявки, либо создать новую по выбранному шаблону. После чего пользователь должен заполнить исходные данные и нажать на кнопку расчета (рис. 7).

В последующем данные поступают в брокер сообщений RabbitMQ, который складывает их в очередь [9]. RabbitMQ отправляет эти данные по API в локально развернутый в кластере программный комплекс. Однако стоит отметить, что расчет и подготовка итогового файла может занимать продолжительное время, так как это напрямую зависит от внесенных пользователем данных и выбранного им шаблона.

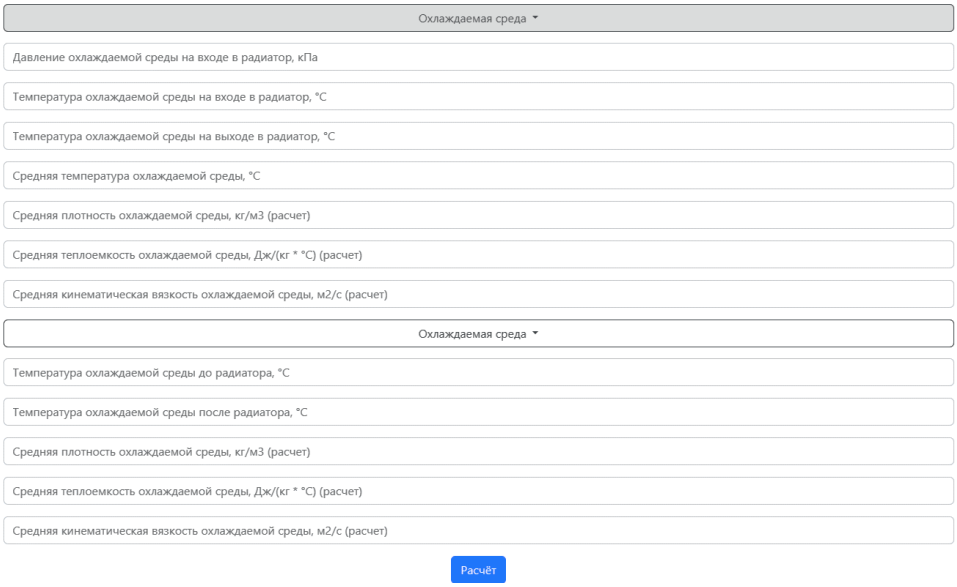


*Рис. 7.* Создание новой заявки и заполнение исходных данных

В конечном итоге пользователь получает отчет в виде pdf файла (рис. 8), который он может использовать для реализации поставленных им целей.

Сервис администраторов, у которого в качестве точки входа выступает https://admins.riotsolver.ru/, позволяет просматривать заявки всех пользователей, а также подтверждать их регистрацию для доступа к ресурсам сайта.

Такой шаг, как разделение сервиса на две отдельные сущности, был принят в связи с эффективностью организованности функциональности веб-приложения, которая, в свою очередь, обеспечивает удобство использования как для обычных пользователей, так и для администраторов. Каждый сервис имеет свой набор функций и прав доступа, что обеспечивает безопасность системы и ее управление.

RADIATOR - Введите название проекта 30.04.2024 16:42:46

| № п/п | Наименование | Значение | Ед.изм. | Параметр |
|---|---|---|---|---|
| 1 | RADIATOR - Введите название проекта 30.04.2024 16:42:46 | | | Вводный |
| 2 | Выберите | | | Вводный |
| 3 | Выделяемая тепловая энергия агрегатом | 50 | кВт | Вводный |
| 4 | Охлаждаемая среда | Необходимо выбрать среду | | Вводный |
| 5 | Температура охлаждаемой среды на входе в радиатор | 90 | 0C | Вводный |
| 6 | Температура охлаждаемой среды на выходе из радиатора | 80 | 0C | Вводный |
| 7 | Средняя температура охлаждаемой среды | 85,00 | 0C | Расчетный |
| 8 | Средняя плотность охлаждаемой среды | 0,000 | кг/м3 | Расчетный |
| 9 | Средняя теплоемкость охлаждаемой среды | 0,00 | Дж/(кг*0C) | Расчетный |
| 10 | Средняя кинематическая вязкость охлаждаемой среды | 0,000000000 | м2/с | Расчетный |
| 11 | Охлаждающая среда | Воздух | | Вводный |
| 12 | Температура охлаждающей среды до радиатора | 40 | 0C | Вводный |
| 13 | Температура охлаждающей среды после радиатора | 60 | 0C | Вводный |
| 14 | Средняя температура охлаждающей среды | 50,00 | 0C | Расчетный |
| 15 | Средняя плотность охлаждающей среды | 1,093 | кг/м3 | Расчетный |
| 16 | Средняя теплоемкость охлаждающей среды | 1005,29 | Дж/(кг*0C) | Расчетный |
| 17 | Средняя кинематическая вязкость охлаждающей среды | 0,000014725 | м2/с | Расчетный |
| 18 | Технология изготовления трубок | Круглооребренная труба + намотанное оребрение (Медь) | | Вводный |
| 19 | Материал трубки | Трубка 13x0,8 материал 03м | | Вводный |

*Рис. 8.* Готовый отчет

Помимо этого, разделение веб сервиса на составляющие позволяет значительно оптимизировать систему, что положительно повлияет на производительность в целом и обеспечить гибкость и легкость масштабируемости системы. Все разворачиваемые в системе сервисы работают и масштабируются независимо друг от друга. Такой подход положительно сказывается на их управлении и, в случае необходимости, позволяет оперативно масштабировать систему под нужды пользователей.

**Результаты.** Для выполнения работы была проделана теоретическая и практическая работа, которая рассматривалась выше. Подробно изучены и проанализированы текущие существующие решения на рынке и проработана архитектура высокодоступного и отказоустойчивого кластера Kubernetes. Помимо этого спроектированы и реализованы расширяемые базы данных в соответствии с Федеральным законом 152 «О персональных данных».

Кроме этого, был разработан веб-сервис на языке программирования Python с использованием библиотеки Django. Данный сервис вместе с программным комплексом Radiator находится внутри кластера и позволят производить сложные инженерные расчеты.

Для отказоустойчивости системы были подключены брокер сообщений RabbitMQ, который складывает и распределяет запросы внутри системы, а также Prometheus в связи с Grafana, которые служат для отображения метрик не только всего кластера, но и каждого узла в частности.

**Заключение.** В конечном итоге был разработан облачный отказоустойчивый кластер с высокой доступностью в Kubernetes, который позволяет специалистам производить сложные инженерные расчеты при проектировании теплообменных агрегатов различного назначения. Данный кластер легко масштабируется, путем добавления новых рабочих узлов в кластер, что позволит увеличить вычислительную мощность при увеличивающимся количестве пользователей. Предложенный подход и способы защиты облачной инфраструктуры позволят защитить программный комплекс, обеспечить отказоустойчивость, а компании смогут пользоваться сервисом и не вкладывать средства в собственные вычислительные мощности и администрирование.

Предложенное решение может быть адаптировано и для других программных комплексов выполняющих сложные вычисления в любых сферах деятельности, которые будут освещаться в последующих публикациях.

**СПИСОК ЛИТЕРАТУРЫ**


1. Защита программного обеспечения от несанкционированного использования / Е.М. Курсков // Международная научно-практическая конференция по компьютерной и информационной безопасности (INFSEC 2023): сборник статей. ООО «Институт цифровой экономики и права». — Екатеринбург. — 2023. — С. 51-56.
2. Безопасный ключ к «облаку» / В. Ткаченко // Защита информации. — Инсайд. — 2012. — № 2 (44). — С. 46-50.
3. ITELON: Вычислительный кластер, Россия. — 2018. — URL: https://itelon.ru/blog/vychislitelnyy-klaster/ (дата обращения: 10.05.2024).
4. ITELON: Высокопроизводительные кластерные решения, Россия. — 2018. — URL: https://itelon.ru/solution/Cluster-servers (дата обращения: 11.05.2024).
5. Хабр: Внутреннее устройство Kubernetes-кластера простым языком, Россия. — 2021. — URL: https://habr.com/ru/companies/flant/articles/583660/ (дата обращения 11.05.2024).
6. Git in Sky: Kubernetes для начинающих, Россия. — 2021. — URL: https://gitinsky.com/ kubernetesarticle (дата обращения: 11.05.2024).
7. Kubernetes: Основы Kubernetes, США. — 2008. — URL: https://kubernetes.io/ru/docs/tutorials/ kubernetes-basics/ (дата обращения: 11.05.2024).
8. PostgreSQL: PostgreSQL, США. — 2024. — URL: https://www.postgresql.org/ (дата обращения 11.05.2024).
9. RabbitMQ: One broker to queue them all, США [Электронный ресурс] — 2024. — URL: https://www.rabbitmq.com/ (дата обращения: 11.05.2024).