МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра дискретной математики и информационных технологий

РАЗРАБОТКА ГЕОРЕПЛИЦИРОВАННОГО ОБЛАЧНОГО СЕРВИСА ХРАНЕНИЯ ТРАНЗАКЦИОННЫХ ДАННЫХ.

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Веденеева Александра Александровича

Научный руководитель
доцент, к.т.н. _____ А. О. Соколов

Заведующий кафедрой
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Современное общество, переходя к новым и более современным устройствам и технологиям, постепенно подходит к этапу глобализации облачных технологий во всех сферах деятельности. Данная технология подразумевает использование программного обеспечения расположенного на удалённых серверах. Взаимодействие с облачными сервисами, а именно получение какой-то определённой информации, доступно с любого устройства. Данными устройствами могут выступать, как персональные компьютеры или ноутбуки, так и планшеты с телефонами. Единственное требования которое объединяет доступность облачных сервисов - наличие интернет соединения.

Сервисы называются «облачными» по двум причинам. Во-первых облако является символом отдалённости пользователя от хост компьютера. Во-вторых облако является образом сложной и многокомпонентной структуры, за который скрываются все технические детали разработанного приложения или сервиса. Таким образом слово «облачным» является метафорой, которая закрепилась в современном мире информационных технологий и очень точно передаёт суть своего устройства.

В настоящее время объёмы используемой информации в информационном пространстве растут большими скачками. Со временем от использования съёмных физических накопителей для переноса файлов, таких как usb накопители, откажутся в пользу облачных сервисов хранения данных. Всё дело заключается в том, что чтобы поделиться данными необходимо иметь при себе нужный носитель информации и именно облачные технологии помогают описываемый момент в транспортировке данных.

Актуальность темы заключается в том, что облачные технологии, с развитием современного общества, набирают свою популярность в силу своих положительных сторон, которые помогают как сократить расходы на физическом размещении и закупке оборудования, так и повысить легкость масштабирования данных с помощью репликации данных в разные регионы планеты.

Целью работы является создание георепликированного облачного сервиса хранения транзакционных данных.

Задачами исследования, решение которых необходимо для достижения поставленной задачи, являются:

1. Ознакомится с рабочей областью в Azure;
2. Детально изучить особенности языка и сопутствующих библиотек для разработки;
3. Научиться взаимодействовать с платформой облачных вычислений;
4. Разработать локальное приложение для последующей развёртки в облаке;
5. Развернуть и протестировать функционал приложения в облаке;

Предмет исследования - облачные технологии хранения транзакционных данных.

Объектом исследования является применение инструментальных средств для создания облачного решения на платформе Azure.

Выпускная квалификационная работа состоит из введения, двух разделов, заключения, списка источников и приложений. В первом разделе рассматриваются теоретические основы облачных технологий. Второй раздел содержит описание реализации облачного сервиса, загрузку в облако и тестирование.

1 Архитектурные составляющие облачных решений

1.1 Характеристики и задачи облачных сервисов

Облачные вычисления являются новым направлением развития в IT инфраструктуры [9,10]. Данная технология основывается на объединении аппаратных и сетевых ресурсов с программным обеспечением от поставщиков дата центов, которые распространены удалённо. Исходя из этого потребитель может быстро получить масштабируемые вычислительные мощности и программное обеспечение для работы. Важнейшей технологией, которая позволяет облачным технологиям существовать, является виртуализация. Она позволяет пользователям использовать вычислительные ресурсы независимо от используемой платформы и аппаратной реализации, при всём этом происходит распределение мощности необходимой клиентам на связанных серверах, которые могут находится в разных регионах земного шара. Появление облачных технологий, как нового ответвления IT индустрии, имеет причину в высоких темпах развития информационных технологий и цифровизации за последние несколько десятилетий.

Резюмируя вышесказанное, облачные вычисления - это процесс поставки вычислительных служб через Интернет, где главным преимуществом является удобство для потребителя. Благодаря данной технологии стали возможны такие привычные задачи, как работа облачной почты, существование банковских систем, системы облачного хранения информации и архивации файлов, социальные сети. Облачные технологии обеспечивают преимущества в виде экономной траты ресурсов, высокой пропускной способности и высокой степени надёжности.

1.2 Модели развертывания

В настоящее время принято выделять три модели развёртывания облачных систем [11]. Они подразделяются на публичные (общественные), частные (приватные), гибридные.

Публичное облако (Public cloud) – определяет множественный доступ пользователей к облачной структуре, но возможность редактирования и обслуживания не доступна. Все сервисные обязанности по поддержанию работоспособности лежат на владельце этого облачного пространства.

Частное облако (Private cloud) — инфраструктура, доступ к изменению

и использованию которой имеет лишь один пользователь или группа пользователей в своих интересах.

Гибридное облако (Hybrid cloud) — это IT-инфраструктура которая сочетает положительные сторону публичного и частного облаков. Такая архитектура разделяет обязанности по управлению между провайдером и клиентом. Задачи для которого используется данные сервис могут быть как из области публичных облачных технологий, так и частных.

Было рассмотрен набор из трёх моделей развёртывания для облачных технологий. Из этого набора можно выделить две составляющие. Во-первых публичные облака подходят широкому сегменту, который имеет ряд обобщенных требований и нужд, а для гораздо более защищённой и безопасной структуры подойдёт архитектура частного облачного решения.

1.3 Модели обслуживания

В настоящее время в облачных технологиях принято выделять три отдельные категории или модели в зависимости от типа услуг которые они предоставляют [12].

Инфраструктура как услуга (IaaS, infrastructure as a service) - модель которая описывает предоставление пакета физических ресурсов. В данной модели клиент будет создавать защиту для облачного сервиса своими силами, а на провайдера возлагается задача организации защиты инфраструктуры.

Смежным видом IaaS инфраструктуры является аппаратное обеспечение как услуга (Hardware as a Service – HaaS). В данном случае пользователь получает все необходимое оборудование, на базе которого разворачивает свою инфраструктуру с использованием подходящего для него программного обеспечения.

Платформа как услуга (PaaS, platform as a service) - это модель, которая целостно предоставляет пользователю инфраструктуру для установки на неё программного обеспечения, баз данных, но без управления инфраструктурой в целом, а имеет право конфигурировать свои развёрнутые приложения в облаке.

2 Реализация георепликированного облачного сервиса в Azure

2.1 Инструментальные средства разработки

Среда разработки Java обладает строгой типизацией и широким функционалом [2], который можно ещё увеличить и адаптировать под себя с помощью подключаемых библиотек и фреймворков о которых будет написано ниже. Кроме понятной документации библиотек и фреймворков, данная среда разработки обладает понятным синтаксисом, что позволяет сконцентрироваться на написании кодовой части приложения.

Список основных инструментов для реализации поставленной задачи представлен ниже и состоит из:

— Java Spring [3]

Является одним из самых популярных фреймворков для разработки веб-приложения на языке Java. Работа с фреймворком обстоит иначе по сравнению с библиотекой. Предстоит описать свои классы и описать часть логики, а созданием объектов для классов и вызов методов берёт на себя сам фреймворк Java Spring [4]. Так же стоит отметить способы конфигурации приложения при работе. Она может быть создана при помощи xml файлов, при помощи java файлов и быть создана автоматически.

— Apache Maven [5]

Maven является инструментов для автоматизации сборки проектов. Из-за сложности и количества подключаемых библиотек команда для сборки проекта будет сложнее и именно для облегчения работы Maven используется повсеместно. Важная особенность данного фреймворка заключается в декларативном описании проекта, а значит что при разработке не нужно будет уделять время для настройки каждой отдельной библиотеки. Все параметры будут выставлены по умолчанию, а если что-то требуется изменить и отклониться от стандартных настроек, то это будет сделано только в нужной библиотеке.

— Apache Tomcat 9 [8]

Tomcat — это контейнер сервлетов с открытым исходным кодом, который также выполняет функцию веб-сервера. Веб-приложение рассчитано на взаимодействие с клиентом. Если есть запрос от клиента, он

обрабатывается, и пользователю отправляется ответ. Для этого и существует Tomcat [6]. Фактически он представляет собой Java-приложение, которое заботится об открытии порта для взаимодействия с клиентом, настройке сессий, количестве запросов, длине заголовка и еще многих операциях.

— JDBC driver

Данный инструмент использует протоколы для соединения с базой данных, а именно базой Cosmos DB. При соединении указывается URL самой базы данных. Данный драйвер является отдельной частью фреймворка Java spring.

При использовании портала Azure, домен DNS будет так же размещён на этом портале, а значит при масштабировании проекта или добавления контейнеров и баз данных будет меняться лишь правая часть структуры доменного имени, что позволит легче адаптироваться и понимать приложение на уровне запросов.

2.2 Реализация приложения

Разрабатываемое приложение будет получать запросы на изменение и отправку данных посредством REST запросов которые будут отправляться через приложение Postman [7]. Далее в зависимости от меньшей задержки до сервера, которая уменьшается относительно близости к серверной стойке Microsoft, определяется подходящая реплика базы данных в которую через написанное приложение поступает REST запрос. Запрос проходит верификацию домена DNS Azure и после чего происходит изменение данных, которое указывалось в теле запроса. После изменения данных в одной базе происходит синхронизация между реплицированными базами данных в течении некоторой задержки, которая может варьироваться в зависимости от количества поступаемой информации, нагрузки сети и географического расположения серверов Azure.

Стоит отметить какие методы для HTTP запросов существуют и какие будут использоваться в данной работе [13]. HTTP определяет множество методов запроса, которые указывают, какое желаемое действие выполнится для данного ресурса. Каждый реализует свою семантику, но каждая группа команд разделяет общие свойства: так, методы могут быть безопасными, идемпотентными или кешируемыми.

В данной работе мы будет использовать только GET и POST методы, которые будут рассмотрены более детально ниже.

GET запрос запрашивает информацию от сервера с помощью указанного URL адреса. Синтаксис данного запроса представляется так:

GET /index.html

Запрос POST обычно отправляется через форму HTML и приводит к изменению на сервере. Когда запрос POST отправляется с помощью метода, отличного от HTML-формы, — например, через XMLHttpRequest — тело может принимать любой тип.

Синтаксис данного запроса представляется так:

POST /index.html

Создание учётной записи Azure

Для взаимодействия со службами необходимо создать учётную запись Azure и выбрать подписку, которая определяет количество возможных используемых ресурсов сервиса [1]. Далее для работы можно использовать группы ресурсов, чтобы веб-приложение понимало, откуда и какие ресурсы нужно брать, либо использовать пред созданную базу данных. Стоит отметить, что на странице подписки портала Azure находится вкладка ключей для доступа, которые занесены в один из файлов приложения для подключения к базе данных.

Разрабатываемое приложение будет получать запросы на изменение и отправку данных посредством REST запросов которые будут отправляться через приложение Postman [8]. Далее в зависимости от меньшей задержки до сервера, которая уменьшается относительно близости к серверной стойке Microsoft, определяется подходящая реплика базы данных в которую через написанное приложение поступает REST запрос. Запрос проходит верификацию домена DNS Azure и после чего происходит изменение данных, которое указывалось в теле запроса. После изменения данных в одной базе происходит синхронизация между реплицированными базами данных в течении некоторой задержки, которая может варьироваться в зависимости от количества поступаемой информации, нагрузки сети и географического расположения серверов Azure. Схема приложения приведена на рис. 1.

Регионы Azure и репликация баз данных

Azure состоит из центров обработки данных, распределенных по всему

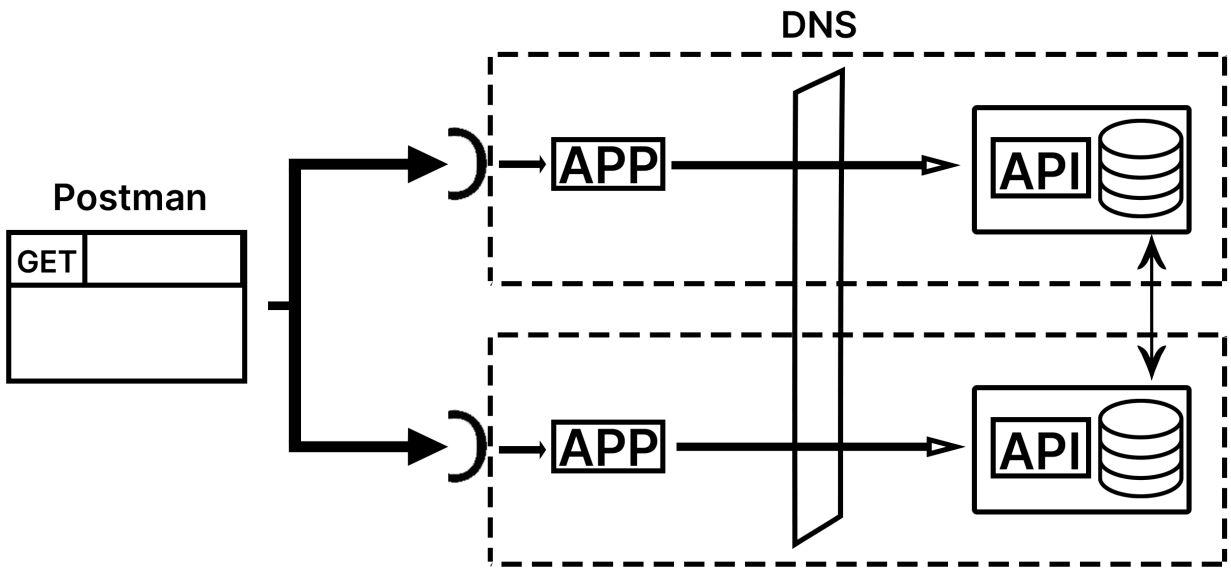

Рисунок 1 – Принцип работы разрабатываемого облачного приложения сервиса на платформе Azure

миру. Когда вы используете службу или создаете ресурс, например базу данных SQL или виртуальную машину, вы используете физическое оборудование этих центров. Эти конкретные центры обработки данных не предоставляются пользователям напрямую. Вместо этого Azure упорядочивает их по регионам. Регион – это географическая область, в которой находится, по крайней мере, один центр обработки данных или несколько центров, расположенных поблизости и связанных друг с другом сетью с низкой задержкой.

Активная георепликация разработана в качестве решения для обеспечения непрерывности бизнес-процессов, которое позволяет приложению выполнять быстрое аварийное восстановление отдельных баз данных в случае регионального сбоя или сбоя масштабирования. Если георепликация включена, приложение может инициировать отработку отказа в базу данных-получатель в другом регионе Azure.

Активная георепликация может использоваться не только для аварийного восстановления, но и в следующих сценариях:

Миграция базы данных: активную георепликацию можно использовать для миграции базы данных с одного сервера на другой в сети с минимальным временем простоя.

Обновления приложения: при обновлении приложения вы можете создать дополнительную базу данных-получатель как резервную копию на случай сбоя

Создание, изменение и удаление данных с помощью отправки REST запросов

Отправка запросов осуществляется с помощью Postman. Postman предназначен для проверки запросов с клиента на сервер и получения ответа от бэкенда.

POST запросы используются для добавления или удаления товаров, а GET запросы для получение списка товаров находящихся в корзине. Каждый ответ характеризуется кодом состояния. Мы ожидаем от каждого запроса код состояния 200, что соответствует успешному запросу. Адрес для запросов:

http://localhost:8081/cart/ID - локально;

https://market-marketapi.azuremicroservices.io/cart/1 - облако.

Url адрес для локального подключения используется для прямых запросов в базу данных Cosmos DB, а уже после успешных тестов адрес будет изменён на адрес облачного хранилища. Действия с локальным подключением необходимы для того чтобы удостовериться в стабильности и правильности запросов. Обработка запросов в cart по id пользователя происходит с помощью контроллера в котором указывается каким будет ответ сервера при разных параметрах в поле запроса. Так же в нём указывается какой запрос имеет код состояния 200, а какой является ошибочным с кодом состояния 400.

Для доступа к командной строке Azure через терминал используется расширение Azure CLI, которое должно быть установлено перед развёртыванием. Служба, в которой будет размещено приложение, называется Azure Spring Cloud. В файле с конфигурацией проекта pom.xml указывается, какие настройки необходимы для приложения Azure.

Обработка ошибок

Если в запросе указать неверные данные, например, пропустить поле actions, которое отвечает за действие, которое нужно делать с данными, то ошибка будет записана в сущность под названием exceptions. Если ошибка будет связана с неверной конфигурацией оборудования, отсутствием соединения или внутренней ошибкой сервера, то она так же будет записано в эту таблицу. Например, если не указать действие, которое выполнять с данными, то Postman вернёт ответ с кодом 400 (Bad request) и текстовое поле “ Check Database for exception”.

2.3 Нагрузочное тестирование

Тестирование проводилось в программе jmeter [14] с количеством эмулируемых пользователей равным 20 и 200, где каждый отправляет GET запросы на получение данных содержимого корзины. Следует сказать, что количество запросов в секунду у бесплатной подписки Azure равно 400. Все запросы успешны, и работа приложения не была нарушена и была стабильна. Согласно графика, представленных на портал Azure, запросы были получены и обработаны, однако время отклика существенно повысилось после отправки 200 запросов, а именно с 116,38 мс до 847,16 мс.

ЗАКЛЮЧЕНИЕ

Задача создание георепликированного облачного сервиса предполагает использование множества библиотек для конфигурации, компиляции и последующего нагрузочного тестирования, что является отдельной задачей в разработке приложения.

В данной работе был продемонстрирован ход действий для создания облачного приложения, начиная с этапа аутентификации на портале Azure и заканчивая загрузкой полноценного облачного сервиса с последующим тестированием и отладкой. В ходе решения поставленной задачи была применена среда разработки Java из-за обширного функционала самого языка и, что не мало важно, функциональность использования фреймворка Java spring, который осуществлял множество действия. Так же стоит отметить, что в этот фреймоврк входил драйвер для подключения JDBC, что помогло осуществить подключение к базе данных Cosmos DB.

С помощью Java и описанных ранее фреймворков и инструментов был реализован облачный сервис с хранение транзакционных данных и имеющим репликацию в нескольких регионах земного шара. Помимо этого приложение сохраняет принципы омниканальности, что позволяет начать работу на одном устройстве, а продолжить и завершить её на другом. В последствии приложение был протестирован с помощью нагрузки отсылаемыми запросами в количестве 20 и 200 штук. После тестирования целостность данных и приложения в целом не была нарушена, что означает высокую стабильность разработанного облачного сервиса. Разработанный сервис может быть как самостоятельной единицей, так и стать одним из элементов в микросервисной архитектуре стороннего многоцелевого приложения.

В результате работы было установлено, что время задержки ответа от сервера напрямую зависит как от количества запросов, поступающих к сервису, так и от времени, когда поступают запросы. Соответственно нагрузка на сеть от всех пользователей сети портала Azure так же влияет на скорость ответа от сервера.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Документация Azure Cosmos DB[Электронный ресурс] : Набор документаций по разным аспектам Cosmos DB. URL: https://docs.microsoft.com/ru-ru/azure/cosmos-db/ (дата обращения 11.04.2021). Загл. с экрана. Яз. рус.

2 JDK Java [Электронный ресурс] : Репозиторий по языку Java и другим компонентам Oracle. URL: https://www.java.com/ru/ (дата обращения 11.04.2021). Загл. с экрана. Яз. рус.

3 Java Spring [Электронный ресурс] : Фреймворк Java. URL: https://spring.io/ (дата обращения 11.04.2021). Загл. с экрана. Яз. рус.

4 Spring.io [Электронный ресурс] : Ресурс описывающий структуры REST запросы. URL: https://spring.io/projects/spring-restdocs (дата обращения 11.04.2021). Загл. с экрана. Яз. рус.

5 Apache Maven Project [Электронный ресурс] : Сборщик приложения Java. URL: https://maven.apache.org/ (дата обращения 11.04.2021). Загл. с экрана. Яз. рус.

6 Apache Tomcat [Электронный ресурс] : Инструмент помогающий разворачивать приложение на Java в качестве API. URL: http://tomcat.apache.org/ (дата обращения: 11.04.2021). Загл. с экрана. Яз. рус.

7 Postman [Электронный ресурс] : Приложение и документация к использованию и настройке Postman. URL: https://www.postman.com/ (дата обращения: 11.04.2021). Загл. с экрана. Яз. рус.

8 Георепликация Azure [Электронный ресурс] : Устройство георепликации данных в Cosmos DB. URL: https://docs.microsoft.com/ru-ru/azure/cosmos-db/distribute-data-globally (дата обращения: 11.04.2021). Загл. с экрана. Яз. рус.

9 Костюк, А.И. Организация облачных и GRID-вычислений: учебное пособие / А.И. Костюк; Южный федеральный университет. - Ростов-на-Дону; Таганрог: Издательство Южного федерального университета, 2018. - 121 с.

10 Уорд, Б. Инновации SQL Server 2019. Использование технологий больших данных и машинного обучения / Боб Уорд; пер. с англ. Н.Б. Желновой. - М.: ДМК Пресс, 2020. - 408 с.

11 AWS Amazon [Электронный ресурс] : Типы облачных вычислений. URL: https://aws.amazon.com/ru/types-of-cloud-computing/ (дата обращения: 10.04.2021). Загл. с экрана. Яз. рус.

12 IBM Cloud Learn Hub [Электронный ресурс] : IaaS, PaaS и SaaS. URL: https://www.ibm.com/ru-ru/cloud/learn/iaas-paas-saas (дата обращения: 14.04.2021). Загл. с экрана. Яз. рус.

13 MDN Web Docs [Электронный ресурс] : Методы HTTP запроса. URL: https://developer.mozilla.org/ru/docs/Web/HTTP/Methods (дата обращения: 11.04.2021). Загл. с экрана. Яз. рус.

14 Apache JMeter [Электронный ресурс] : Репозиторий с документацией JMeter. URL: https://jmeter.apache.org/ (дата обращения: 11.04.2021). Загл. с экрана. Яз. рус.