Devops - Тестовое задание

Оглавление

Что понадобится?

Для выполнения тестового задания потребуется новый аккаунт AWS. Создаваемые в процессе ресурсы подбирались таким образом, чтобы укладываться в лимит Free Tier (ориентировочно месячного лимита должно хватить приблизительно на 2 недели работы стенда). Тем не менее, **создаваемые ресурсы вы должны контролировать самостоятельно** (поскольку есть риск все же выйти за рамки бесплатного минимума). Чтобы остаться в рамках Free Tier, разрешается проигнорировать стандартные практики и создать инстанс сервера приложений в публичной подсети с выдачей внешнего IP, что позволит не создавать платный NAT Gateway. Доменное имя для стенда и записи в DNS для верификации сертификатов выдадим/сделаем по запросу, если нужно .

Куда адресовать вопросы?

С вопросами касательно выполнения тестового задания (например, если случился "затык") - можно смело обращаться на rkirilenko@stiltsoft.com или в скайп (rkirilenko). Тем не менее, стоит попробовать вначале решить проблемы самостоятельно, т.к. по заданию в некоторых местах разложены "грабли", на которые вы скорее всего наступите - это сделано умышленно, и ответы на вопросы можно найти в открытых источниках. Отвечать обещаю в течение суток (по возможности - оперативнее). Также советую обратить внимание на то, что некоторые формулировки сделаны довольно общими и в них не хватает конкретики. В таких случаях стоит делать на свое усмотрение (поглядывая, чтобы не вылезать за Free Tier) - это тоже сделано умышленно, чтобы потом было что обсудить.

Лимит времени и трудозатраты

Контрольное время Stiltsoft без документирования, но с отвлечениями - 5.5 часов в спокойном темпе. Если документировать и не отвлекаться - должно получиться то на то. Мы никак не ограничиваем количество времени, которое потратите лично вы, но результаты самого тестового ожидаем в течение двух недель. Лучше немного быстрее, чтобы мы успели проверить задание до исчерпания лимитов Free Tier в Амазоне.

Собственно, задание

Как вы уже вероятно знаете, компания Stiltsoft работает внутри экосистемы Atlassian и занимается созданием аппов (аддонов, плагинов) для приложений Atlassian. Spring Boot - один из фреймворков, который используют у нас для разработки приложений, а работают они в основном в AWS. Так что задачей будет запустить приложение на Spring Boot в облаке AWS, установить его в Confluence Cloud и настроить мониторинг приложения (о падениях мы должны знать раньше наших пользователей!).

Аккаунт AWS

Для выполнения задания (и чтобы получить доступ к AWS Free Tier) нужно создать новый аккаунт AWS. Впоследствии потребуется к нему дать доступ нам для оценки происходящего (еще одна причина завести новый аккаунт).

Использование Terraform для развертывания инфраструктуры не обязательное требование, но приветствуется. Нам проще проверять будет, да и прибить ресурсы в облаке тоже с его помощью очень удобно.

Postgres

Разворачиваем Postgres в минимальной конфигурации на RDS. db.t2.micro, 20Gb.

Сервер приложений

Это виртуалка размера t2.micro, которая имеет доступ к Postgres, развернутому шагом выше. На этой машине нужно развернуть Docker. Здесь мы и будем запускать приложение чуть позже.

Бастион

Чтобы иметь возможность ходить в БД (в реальной жизни - не только), разворачиваем виртуальную машину. t2.micro, она должна быть доступна из интернета и иметь доступ к созданной БД и серверу приложений (ssh).

Докеризовать приложение

Исходный код приложения-примера лежит в открытом репозитории. Нужно сделать так, чтобы оно запускалось на сервере приложений в Docker и использовало в качестве БД созданный выше инстанс Postgres.

Организовать доступность запущенного приложения через интернет

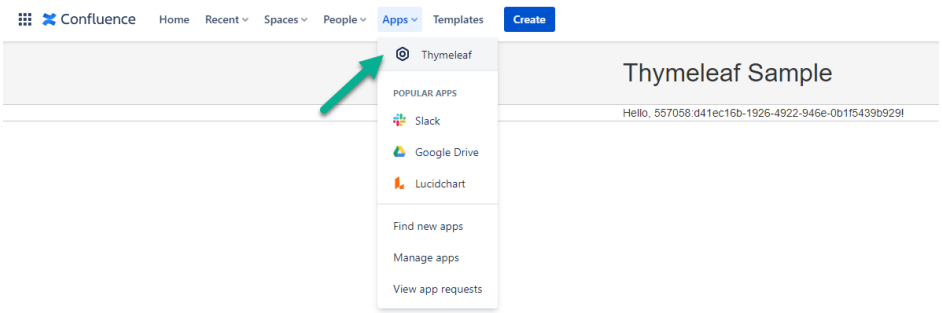
Поскольку в реальной жизни мы запускаем более одного инстанса приложения - просто открыть порт с машины в интернет будет недостаточно. Добавляем в нашу схему балансировщик нагрузки (ALB) и настраиваем доступ к приложению через балансировщик. Проверить, что все получилось, можно обратившись к %your_app_url%/atlassian-connect.json - приложение должно ответить куском json.

Заводим Confluence

Приложение к этому моменту должно быть готово. Нам нужен Confluence Cloud, куда мы это самое приложение будем устанавливать. Dev-инстанс подойдет .

Установить приложение в Confluence

Ставим наше приложение в Confluence. Если все получилось - то по завершению в верхнем меню на вкладке Apps появится пункт Thymeleaf. Если на него кликнуть - то должна, предварительно потормозив, загрузиться страничка как на скриншоте и написать нам Hello.



Мониторинг

О падении приложения надо знать раньше, чем пользователи придут в поддержку. Настраиваем.

Документирование

На выходе ожидается описание созданного сервиса. Форма - свободная.

Доступы

Чтобы можно было оценить широту полета мысли - понадобится выписать доступы для нашей команды ко всему, к чему посчитаете нужным.

Итерация оценки

После отмашки "готово" наша команда посмотрит, как все сделано (воспользовавшись полученными доступами) и задаст интересующие вопросы.

Обратная связь

Мы ценим время и усилия каждого. Поэтому обязательно поделимся тем, что мы увидели, обсудим варианты решений, обозначим зоны роста. Возможно, похоливарим.

Итерация доработок

После выдачи обратной связи мы обычно предлагаем "доделать" задание так, чтобы оно выглядело хорошо (с нашей точки зрения). После чего делаем еще одну стадию ревью, по мотивам сделанного.