Федеральное государственное автономное образовательное учреждение
высшего образования «Национальный исследовательский университет
«Высшая школа экономики»

Факультет компьютерных наук
Основная образовательная программа
Прикладная математика и информатика

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Программный проект на тему "Применение облачных технологий для проверки решений учебных задач по программированию"

Выполнил студент группы 206, 4 курса,
Олигер Никита Александрович

Руководитель ВКР:
Старший преподаватель
Департамент больших данных и информационного поиска
Куренков Владимир Вячеславович

Москва 2024

Содержание

TODO: ABSTRACT
TODO: KEYWORDS

1 Введение

1.1 Предметная область

Тренировочные задачи по программированию являются неотъемлемой частью любого курса, нацеленного на ознакомление студентов с языками программирования и алгоритмами. Однако составление таких заданий оказывается гораздо более простой задачей в сравнении с проверкой решений к предложенным задачам.

Если обратиться к истории создания учебных курсов по программированию, можно заметить, что написание кода на бумажном носителе и последующая проверка преподавателем вручную имели много недостатков: такой процесс был очень затратен по времени на всех его этапах, а точность проверки напрямую страдала от человеческого фактора. Было понятно: этот процесс было необходимо автоматизировать.

Сейчас большинство тренировочных онлайн-платформ, таких как Leetcode [2], Coursera [3] и прочие, имеет собственные системы тестирования решений тренировочных задач. Такие системы позволяют проверять код решения задачи на наборе тестов на корректность, покрывая при этом всевозможные корнер кейсы. Эффективность каждого конкретного решения зависит от скорости выполнения проверки, а также от широты функционала: наличия выбранного студентом языка программировании, полноте полученной информации и удобства её получения.

В самом деле, лишь малая часть существующих решений может предложить также дополнительные вариации тестирования, такие как stress тестирование, monkey и fuzzing тестирования. Проблемой также может стать то, что пользователь существующих решений либо не может добавить свою задачу для тестирования вовсе, либо имеет большое ограничение на количество добавленных наборов задач.

1.2 Цель и требования проекта

Целью данной работы является создание альтернативной платформы тестирования тренировочных задач, имеющих следующие функциональные требования:

- Поддерживаются C++ и Python: два основных языка программирования, используемых школьниками и студентами младших курсов.

- Добавлен основной набор методов тестирования: fuzzing, monkey тестирование, проверка code style.
- Решение реализовано в облачной среде, с соблюдением подхода CaaS (Cloud as a Service).

Наряду с озвученными выше требованиями, решение также имеет ряд нефункциональных требований, присущих системам тестирования:

- Скорость выполнения запроса на проверку должна соотствовать запрошенному виду тестирования: несколько секунд для предопределенного набора тестов и code style, и до получаса в monkey и fuzzing тестированиях.
- Отчёт о проведении проверки полон и подробен, а её результаты корректны.
- Система удобна в использовании, её интерфейс понятен, а результат проверки автоматически отправляется пользователю в удобном формате по завершению работы.

1.3 Ожидаемые результаты работы

Ожидается, что в результате работы будет представлена open source система проверки тренировочных задач по программированию, которая будет представлять больший функционал по сравнению с предшедствующими решениями. Важно также отметить, что данное решение также должно быть доступно к развёртыванию для любого пользователя, знающего принципы работы с Яндекс.Облаком и не уступать своим конкуррентам по скорости выполнения операций.

Исходный код: https://github.com/polarnights/ContestED

2 Существующие решения и подходы

Перед анализом уже существующих решений, важно отметить, что большинство из них представляют собой коммерческий продукт. Это создаёт определенные трудности, связанные с отсутствием исходного кода, соответствующей теоретической составляющей в виде соответствующей литературы или формализованных SLA (Service Level Agreement – договор-соглашение об уровне предоставляемых услуг). Более того, в рамках данной работы мы будем ограничивать функционал системы строго в рамках проведения курсов по программированию, алгоритмам и структурам данных и прочих, где ответом на задачу является программный код.

Как можно заметить, совокупность параметров сильно ограничивает менно поэтому плюсы и минусы платформ будут описаны с опорой именно на пользовательский опыт и имеющуюся документацию. Логическим завершением этой части работы будет являться таблица со сравнительной характеристикой платформ. Эта таблица подчеркнёт общие недостатки подходов, рефлексию над тем, почему

2.1 Яндекс Контест

Яндекс Контест – одна из наиболее популярных в России систем автоматической проверки различных задач, появившаяся более 12 лет назад. Обратившись к статье одного из разработчиков [4] и блогам пользователей на альтертативной платформе проведения соревнований по программированию Codeforces [5], понимаем, что активная работа над изменением функционала системы не ведется уже около трёх лет.

2.1.1 Архитектурные особенности платформы

Разработчики активно выстраивали инфраструктуру платформы ещё в 2011-2012 годах, и с тех пор глобально он не изменился. Процесс проверки решений проходит в три фазы: фаза компиляции, затем тестирование и аггрегация результатов теста. Используемое разработчиками окружение такого: набор виртуальных машин, на которых запущен набор Docker TODO: CITE -контейнеров, на кажом из которых работает приложение, отвечающее за все этапы проверки.

Необходимость в хранении большого числа библиотек на каждой отдельной VM была решена при помощи OverlayFS. Гибридная файловая система позволяет "накладывать"часть файлов операционной системы, используемых строго на чтение, на другие файловые системы, отличной от текущей. Такое решение позволило разработчикам сэкономить около 10-20 ГБ операционной системы на один лишь контейнер, а следовательно, дать больший масштаб и для горизонтального, и для вертикального масштабирования.

Кроме технической составляющей, платформа имеет 22 доступных языка программирования с 124 версиями [4] (TODO: CITE https://docs.kernel.org/filesystems/

2.1.2 Недостатки платформы

2.2 Решение 2

2.2.1 Про решение 2 - 1

2.2.2 Про решение 2 - 2

2.3 Сравнительная характеристика решений

Проанализировав вышеупомянутые решения, можно прийти к нескольким ключевым замечаниям.

1. Не существует одной платформы, которая покрывала бы все возможные подходы к изучению программирования.

- Часть платформ имеет ряд недостатков, связанных с необходимостью ручной проверки качества кода, которая так необходима молодым разработчиком с небольшим практическим опытом.
- Распространненой проблемой является невозможность добавления

3 Архитектура решения

3.1 Концепция 'Serverless'

3.2 Объяснение 2

3.3 Объяснение 3

3.4 Объяснение 4

3.5 Объяснение 5

4 Веб интерфейс и HTTP API

5 Администрирование платформы

5.1 Мониторинг системы

Стандартной практикой предотвращения и пост-фикса после проблем является логгирование и мониторинг серверов с помощью Grafana - утилиты, предоставляющей интерактивные доски с информацией о системе.

Для хранения логов поддерживается следующая цепочка:

$$Filebeat \rightarrow Logstash \rightarrow Elasticsearch \rightarrow Kibana$$

С помощью неё администратор достигает человеко-читаемости логов и имеет простой доступ к ним. В случае возникновения проблемы, разработчик сможет понять, какая последовательность действий привела к ней.

Для мониторинга всех частей системы может быть полезна Grafana: утилита для демонстрации информации о системе: как общего использования CPU и GPU всей системы, так и более детальное рассмотрение, например, использование навигационных дашбордов, переводящих нас на информацию о количество активных пользователей, доступного и используемого объема памяти на облачном хранилище, где хранятся сервера.

Эти утилиты также будут полезны для мониторинга проблем с безопасностью. Полезно установить определённое пороговое значение потребления ресурсов одним кластером. В случае превышения этого значения мы можем предполагать, что в системе произошел сбой: например, не сработало автоматическое машстабирование, или систему намеренно эксплуатируют: засылают много решений, или засылают заведемо вредоносные файлы, вроде fork-бомб.

6 Эксперименты и производительность платформы

7 Актуальность и значимость

8 Полученные результаты

Список источников

[1] Платформа проверки решений тренировочных задач по программированию **Яндекс.Контест**, 2024
URL: https://contest.yandex.ru/edu

[2] Платформа проверки решений тренировочных задач по программированию **LeetCode**, 2024
URL: https://leetcode.com/

[3] Проект по публикации образовательных онлайн-курсов **Coursera**, 2024
URL: https://www.coursera.org/

[4] Как запустить 100+ компиляторов и выстоять. Опыт **Яндекс.Контеста**, 2021
URL: https://habr.com/ru/companies/yandex/articles/

[5] Платформа проведения соревнований по программированию **Codeforces**, 2024
URL: https://codeforces.com/

[6] Как заставить код выполняться за одинаковое время? Способы от **Яндекс.Контеста**, 2020
URL: https://habr.com/ru/companies/yandex/

[7] URL: https://habr.com/ru/companies/yandex/