**Test task Ihor Pysmennyi**

1. ***Describe the high-level architecture of the project***

A picture containing text, whiteboard

Description automatically generated

1. ***Highlight key components/microservices***

Services must be loosely coupled so that they can be developed, deployed and scaled independently.

* Websites
* Clients
* Ad manager
* Advertisement content
* Click counter
* Statistics

1. ***Rate RPS if your banners will be placed on the top 10 sites in the UK(one banner per page)***

RPS = 1000 / T \* workers, where

• T - average request processing time (in milliseconds);

• workers - the number of threads;

• 1000 / T requests per second - this value will be output by a single-threaded generator.

**Resource:** <https://habr.com/ru/company/yandex/blog/425867/>

Littledata surveyed 7,064 sites in December 2021 and found the average server response time was 492ms.

Anything less than 295ms would put you in the best 20% of sites we benchmark for server response time, and less than 207ms would put you in the best 10%.

**Resource:** <https://www.littledata.io/average/server-response-time>

Let’s imagine we aim to be among the best and reach 200ms for our site. Using data from similarweb.com site, we will take statistics on the 10 the most visited sites in the UK.

**Link to statistics:** <https://www.similarweb.com/top-websites/united-kingdom/>

|  |  |  |  |
| --- | --- | --- | --- |
|  | Site | Total Visits per second(average) | Pages per Visit |
| 1 | google.com | 1372 | 8.52 |
| 2 | youtube.com | 493 | 11.28 |
| 3 | facebook.com | 379 | 8.46 |
| 4 | amazon.co.uk | 155 | 8.16 |
| 5 | twitter.com | 146 | 11.97 |
| 6 | google.co.uk | 140 | 10.00 |
| 7 | bbc.co.uk | 228 | 3.25 |
| 8 | ebay.co.uk | 110 | 10.42 |
| 9 | wikipedia.org | 118 | 3.00 |
| 10 | live.com | 76 | 8.15 |
|  | Total | 3217 | 83.21 |

*RPS*= 1000 ×3217=16085 This case is initially fantastic, therefore, we can safely take 15-20% of this value and build the infrastructure based on it.

200*ms*

1. ***How do you organize your advertising target system?***

If we are talking about setting up a targeting system to promote our analogue of Adsense, then the parameters will be as follows:

1. Age: 18-45;

2. Gender: any

3. Geography: in fact, the whole English-speaking world, but if we are from the UK, then we can focus on the UK and the EU countries, possibly also the USA, Canada;

4. Keywords: **Internet advertising**, **contextual advertising UK**, **contextual advertising**, **advertising on the site**.

1. ***What technologies do you propose?***

Websites/Clients – Python/Django these microservices are lightweight and can expose REST-endpoints.

Click counter – Java/Scala due to multithreading, integrated with Apache Kafka (written in Java).

Ad manager (Business Logic) – Java preferable but Python is also possible.

Advertisement Content – Java/Scala, Python is also possible.

Statistics – Java/Scala.

Each microservice persistent data private to that service and accessible only via its API. A service’s transactions only involve its database. Using a database per service helps ensure that the services are loosely coupled. Changes to one service’s database does not impact any other services. Each service can use the type of database that is best suited to its needs.

Websites – Cassandra (in case of Google data scale)/DynamoDB, PostgreSQL as an alternative (~1 billion records)

Clients – Cassandra (in case of Google data scale)/DynamoDB, PostgreSQL as an alternative (~1 billion records)

Click counter – streams data through Apache Kafka

Ad manager (Business Logic) – Redis (in-memory database)

Advertisement Content – MongoDB

Statistics – Cassandra (in case of Google data scale)/DynamoDB, PostgreSQL as an alternative (~1 billion records)

**Notes:**

Cassandra can be an overhead solution.

Each microservice to package into Docker image and save it to private Docker registry(Artifactory).

Infrastructure (K8S clusters) in Public Cloud Provider(AWS/GCP/Azure) deploy by using Terraform.

CI/CD: GitLabCI, Jenkins also possible.

K8S as a platform(infrastructure layer) for managing microservice architecture:

* Self-healing when a failure occurs;
* Auto-scaling;
* Service Discovery and load-balancing;
* Secrets Management;
* Storage Management;
* Automatic resource allocation;
* Automatic implementation and rollback of changes.

Storage system based on NFS, GlusterFS or Ceph(preferable).

1. ***How will you track statistics?***

If we are in a admin role and would like to track click statistics we can use **Tracking Pixel** <https://en.ryte.com/wiki/Tracking_Pixel> and every time when somebody search your site, we save it to the logger by observing IP/browser, etc.

1. ***Where will you store statistics for impressions/clicks?***

We can store statistics in RDBMS, for instance PostgreSQL. For real-time date – InfluxDB/TimescaleDB.

1. ***Evaluate the approximate Total Cost of Ownership***

3 Kubernetes clusters (Development, Stage/QA/Pre-Production, Production) in AWS/GCP.

For each K8S cluster we need the minimum 2 Control-plane nodes, 3-5 ETCD nodes, 3 worker nodes.

Control-plane node instance type - c5.xlarge 2 EC2 on-demand instance + EBS (50GB Provisioned IOPS SSD(io1)) = 294.12$

Worker node instance type - r5.2xlarge 3 EC2 on-demand instance + EBS (50GB Provisioned IOPS SSD(io1)) = 1255.86$

Etcd node instance type – m5.xlarge 3 EC2 on-demand instances(monthly) + EBS (30GB Provisioned IOPS SSD(io1)) = 481.08$

AWS ELB(recommended) for load-balancing – 2 instances = 36.79$

AWS S3 for backups ~ 200$

AWS DynamoDB – 203.4$

MongoDB cluster with 3 EC2 instances m4.large – 284.74$

Other tools ~ 300$

Total Cost of Ownership for development environment - ~3000$