

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО
Факультет программной инженерии и компьютерной техники



Облачные и туманные вычисления

Лабораторная работа № 2

Структурная схема с обоснованием

Выполнил

Гурин Евгений Иванович

Группа № Р34122

Преподаватель: Перл Ольга Вячеславовна

г. Санкт-Петербург

2021

Оглавление

Концепт	3
Компоненты системы и выбор облака	3
Предполагаемая схема компонентов	4
Итоговая схема компонентов	5
Схема базы данных	6
Схема апи	6
Спецификация Open API	7
Репозитории	18
Процесс создания окружения на AWS	18
Создание Elastic BeanStalk	18
Создание БД	19
Создание архива с проектом	20
Загрузка проекта	20
Создание фронтенд приложения. Создание S3 Bucket	21
Загрузка файлов фронтенда	21
Создание CloudFront	21
Скрипты фронтенд приложения	23
Вывод	24

Концепт

Сервис организованного сбора и статистики для отчетов по автотестам с возможностью анализа итоговых данных.

Описание возможностей системы:

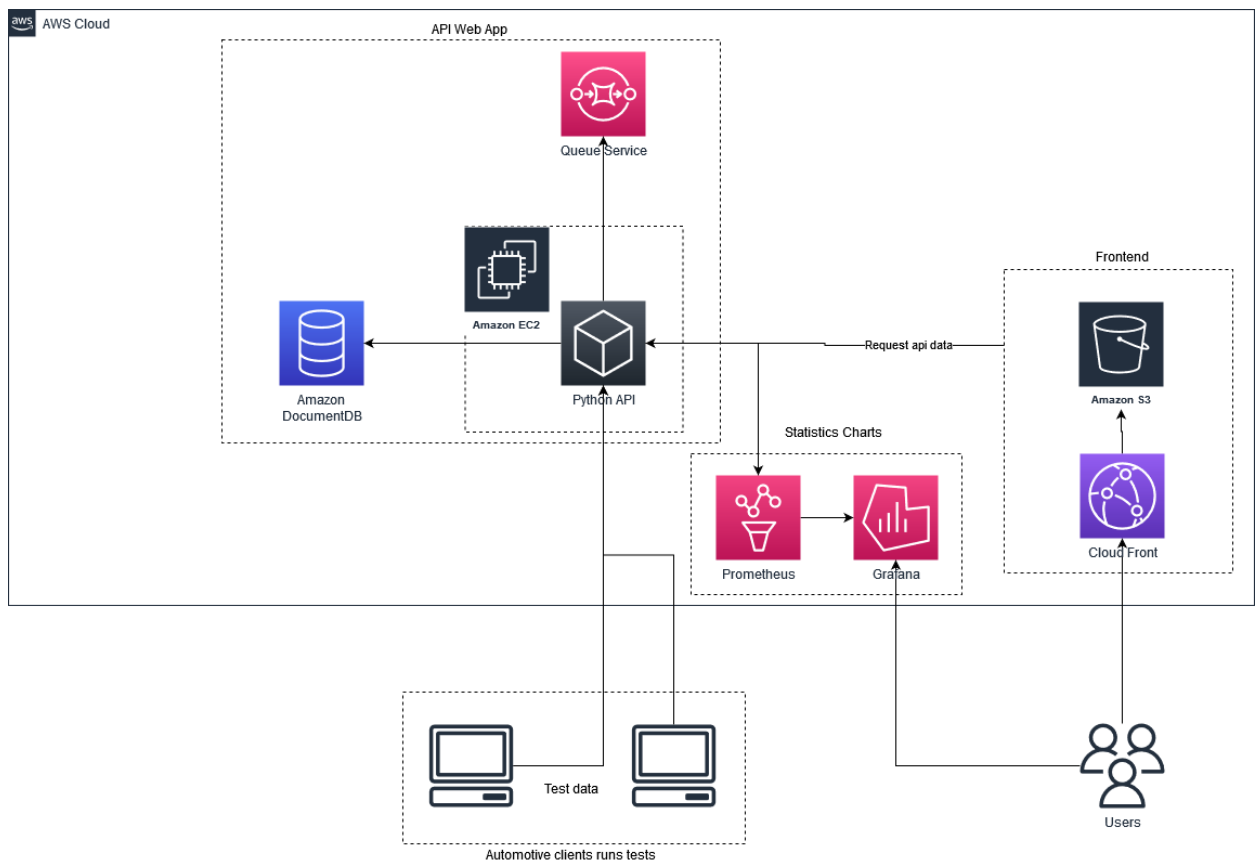
- сохранение данных с различных автотестов
- агрегация данных
- вывод статистики

Компоненты системы и выбор облака

Решено было использовать облачное решение от Amazon, так как это популярное и гибкое решение и в нём я нашёл все необходимые мне возможности.

- **БД** было найдено несколько возможных решений, либо вручную поднять экземпляр виртуальной машины и на нём запустить Mongo DB, либо воспользоваться аналогом от Amazon, под названием Amazon DocumentDB, которая совместима с MongoDB. Также возможно использование **Amazon RDS**
- **Веб сервис апи (сбор статистики)** я решил использовать EC2 Environment, на котором будет настроено виртуальное окружение. Гибкость этого решения позволяет обеспечить бесшовный деплой, с помощью нескольких экземпляров виртуального окружения, из которых прошлая задеплоенная версия останавливается только после запуска новой и прохождения проверок.
- Также при необходимости есть возможность настроить **scaling** (AWS Auto Scaling) и load-balancer (Amazon Lightsail) для использования нескольких экземпляров веб сервиса и автоматической работе в случае пиковых нагрузок
- Для **фронтенда** я решил использовать S3 bucket в связке с CloudFront, который будет выступать в качестве cdn для быстрой работы

Предполагаемая схема компонентов



Итоговая схема компонентов

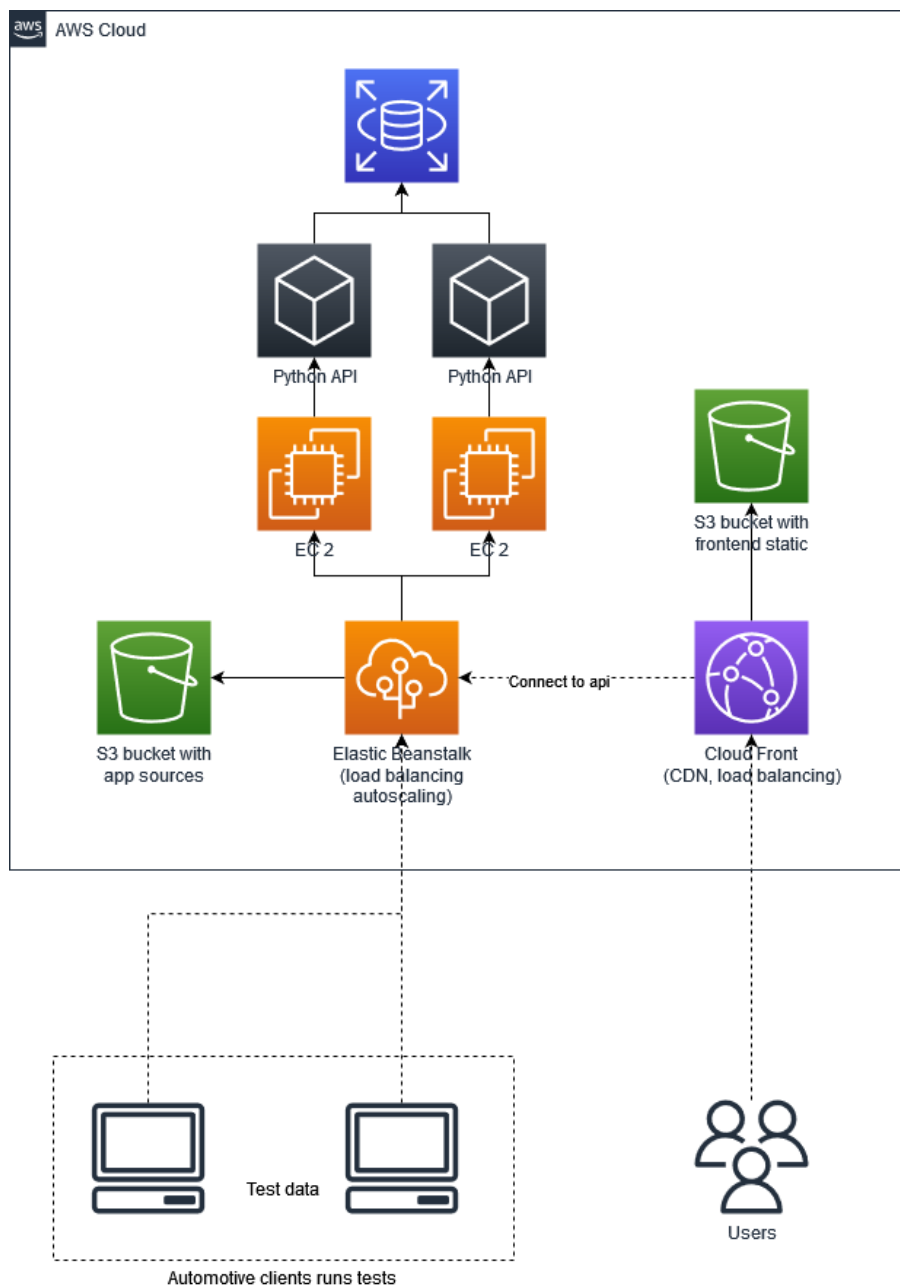
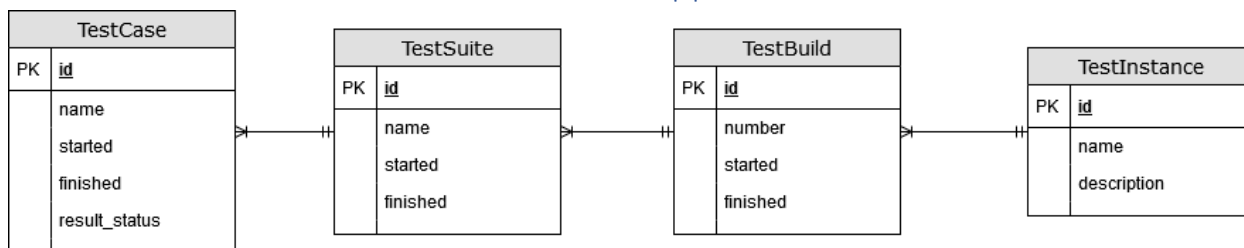


Схема базы данных



Представленная схема базы данных состоит из 4 таблиц

- Тестовый Инстанс – некий тестируемый продукт
- Тестовый билд – билд тестов для продукта (единица итерации)
- Тестовый сьют – набор тестовых кейсов одной группы
- Тестовый кейс – один из тестов группы

Для базы данных API будет предоставлять CRUD интерфейс

Схема api

/api/test-instance/

/api/test-instance/{instance-id}/

/api/test-instance/{instance-id}/test-build/

/api/test-instance/{instance-id}/test-build/{build-id}/

/api/test-instance/{instance-id}/test-build/{build-id}/start/

/api/test-instance/{instance-id}/test-build/{build-id}/finish/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/{suite-id}/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/{suite-id}/start/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/{suite-id}/finish/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/{suite-id}/test-case/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/{suite-id}/test-case/{case-id}/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/{suite-id}/test-case/{case-id}/start/

/api/test-instance/{instance-id}/test-build/{build-id}/test-suite/{suite-id}/test-case/{case-id}/finish/

Спецификация Open API

```
components:
  schemas:
    TestInstance:
      type: object
      properties:
        name:
          type: string
          required: true
        description:
          type: string
          required: false
    TestBuild:
      type: object
      properties:
        name:
          type: string
          required: true
        started:
          type: int
          required: false
        finished:
          type: int
          required: false
    TestSuite:
      type: object
      properties:
        name:
          type: string
          required: true
        started:
          type: int
          required: false
        finished:
          type: int
          required: false
    TestCase:
      type: object
      properties:
        name:
          type: string
          required: true
        started:
          type: int
          required: false
        finished:
          type: int
          required: false
        result_status:
          type: string
          enum:
            - SUCCESS
            - ERROR
            - FAILED
```

```

paths:
  /api/test-instance/:
    get:
      summary: List all test instances
      operationId: listTestInstaces
      parameters:
        limit:
          name: limit
          in: query
          required: false
          schema:
            type: integer
            format: int32
        offset:
          name: offset
          in: query
          required: false
          schema:
            type: integer
            format: int32
        search:
          name: search
          in: query
          required: false
          schema:
            type: string
      responses:
        '200':
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/TestInstance"
    post:
      summary: Create test instance
      operationId: createTestInstace
      requestBody:
        application/json:
          schema:
            $ref: "#/components/schemas/TestInstance"
      responses:
        '200':
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/TestInstance"

  api/test-instance/{test-instance-id}/:
    get:
      operationId: recieveTestInstace
      parameters:
        instanceId:
          name: test-instance-id
          in: path
          required: true

```



```
        schema:
          type: string
      responses:
        '200':
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/TestInstance"

api/test-instance/{test-instance-id}/test-build/:
  get:
    operationId: listTestBuilds
    parameters:
      instanceId:
        name: test-instance-id
        in: path
        required: true
        schema:
          type: string
      limit:
        name: limit
        in: query
        required: false
        schema:
          type: integer
          format: int32
      offset:
        name: offset
        in: query
        required: false
        schema:
          type: integer
          format: int32
      search:
        name: search
        in: query
        required: false
        schema:
          type: string
    responses:
      '200':
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TestBuild"

  post:
    summary: Create test builds
    operationId: createTestBuild
    parameters:
      instanceId:
        name: test-instance-id
        in: path
        required: true
        schema:
          type: string
```

```

        requestBody:
          application/json:
            schema:
              $ref: "#/components/schemas/TestBuild"
        responses:
          '200':
            content:
              application/json:
                schema:
                  $ref: "#/components/schemas/TestBuild"

api/test-instance/{test-instance-id}/test-build/{test-build-id}/:
  get:
    operationId: recieveTestBuild
    parameters:
      instanceId:
        name: test-instance-id
        in: path
        required: true
        schema:
          type: string
      buildId:
        name: test-build-id
        in: path
        required: true
        schema:
          type: string
    responses:
      '200':
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TestBuild"

api/test-instance/{test-instance-id}/test-build/{test-build-id}/start/:
  post:
    summary: Start test build
    operationId: startTestBuild
    parameters:
      instanceId:
        name: test-instance-id
        in: path
        required: true
        schema:
          type: string
      buildId:
        name: test-build-id
        in: path
        required: true
        schema:
          type: string
    responses:
      '200':
        content:

```

```

        application/json:
          schema:
            $ref: "#/components/schemas/TestBuild"

    api/test-instance/{test-instance-id}/test-build/{test-build-id}/stop/:
      post:
        summary: stop test build
        operationId: stopTestBuild
        parameters:
          instanceId:
            name: test-instance-id
            in: path
            required: true
            schema:
              type: string
          buildId:
            name: test-build-id
            in: path
            required: true
            schema:
              type: string
        responses:
          '200':
            content:
              application/json:
                schema:
                  $ref: "#/components/schemas/TestBuild"

    api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-
suite/:
      get:
        operationId: listTestCases
        parameters:
          instanceId:
            name: test-instance-id
            in: path
            required: true
            schema:
              type: string
          buildId:
            name: test-build-id
            in: path
            required: true
            schema:
              type: string
          limit:
            name: limit
            in: query
            required: false
            schema:
              type: integer
              format: int32
          offset:
            name: offset

```

```

        in: query
        required: false
        schema:
          type: integer
          format: int32
      search:
        name: search
        in: query
        required: false
        schema:
          type: string
    responses:
      '200':
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TestSuite"
  post:
    summary: Create test suite
    operationId: createTestSuite
    parameters:
      instanceId:
        name: test-instance-id
        in: path
        required: true
        schema:
          type: string
      buildId:
        name: test-build-id
        in: path
        required: true
        schema:
          type: string
    requestBody:
      application/json:
        schema:
          $ref: "#/components/schemas/TestSuite"
    responses:
      '200':
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TestSuite"

  api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-suite/{test-suite-id}/:
    get:
      operationId: recieveTestSuite
      parameters:
        instanceId:
          name: test-instance-id
          in: path
          required: true
          schema:
            type: string

```

```

        buildId:
          name: test-build-id
          in: path
          required: true
          schema:
            type: string
        suiteId:
          name: test-suite-id
          in: path
          required: true
          schema:
            type: string
      responses:
        '200':
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/TestSuite"

    api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-
suite/{test-suite-id}/start/:
      post:
        summary: Start test suite
        operationId: startTestSuild
        parameters:
          instanceId:
            name: test-instance-id
            in: path
            required: true
            schema:
              type: string
          buildId:
            name: test-build-id
            in: path
            required: true
            schema:
              type: string
          suiteId:
            name: test-suite-id
            in: path
            required: true
            schema:
              type: string
        responses:
          '200':
            content:
              application/json:
                schema:
                  $ref: "#/components/schemas/TestSuite"

    api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-
suite/{test-suite-id}/stop/:
      post:
        summary: stop test build

```

```
operationId: stopTestCase
parameters:
  instanceId:
    name: test-instance-id
    in: path
    required: true
    schema:
      type: string
  buildId:
    name: test-build-id
    in: path
    required: true
    schema:
      type: string
  suiteId:
    name: test-suite-id
    in: path
    required: true
    schema:
      type: string
responses:
  '200':
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/TestSuite"
```

api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-suite/{test-suite-id}/test-case/:

```
get:
  operationId: listTestCases
  parameters:
    instanceId:
      name: test-instance-id
      in: path
      required: true
      schema:
        type: string
    buildId:
      name: test-build-id
      in: path
      required: true
      schema:
        type: string
    suiteId:
      name: test-suite-id
      in: path
      required: true
      schema:
        type: string
  limit:
```

```
        name: limit
        in: query
        required: false
        schema:
          type: integer
          format: int32
      offset:
        name: offset
        in: query
        required: false
        schema:
          type: integer
          format: int32
      search:
        name: search
        in: query
        required: false
        schema:
          type: string
    responses:
      '200':
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TestCase"
  post:
    summary: Create test case
    operationId: createTestCase
    parameters:
      instanceId:
        name: test-instance-id
        in: path
        required: true
        schema:
          type: string
      buildId:
        name: test-build-id
        in: path
        required: true
        schema:
          type: string
      suiteId:
        name: test-suite-id
        in: path
        required: true
        schema:
          type: string
    requestBody:
      application/json:
        schema:
          $ref: "#/components/schemas/TestCase"
    responses:
      '200':
        content:
          application/json:
```

```

        schema:
          $ref: "#/components/schemas/TestCase"

    api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-suite/{test-suite-id}/test-case/{test-case-id}/:
      get:
        operationId: recieveTestCase
        parameters:
          instanceId:
            name: test-instance-id
            in: path
            required: true
            schema:
              type: string
          buildId:
            name: test-build-id
            in: path
            required: true
            schema:
              type: string
          suiteId:
            name: test-suite-id
            in: path
            required: true
            schema:
              type: string
          caseId:
            name: test-case-id
            in: path
            required: true
            schema:
              type: string
        responses:
          '200':
            content:
              application/json:
                schema:
                  $ref: "#/components/schemas/TestCase"

    api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-suite/{test-suite-id}/test-case/{test-case-id}/start/:
      post:
        summary: Start test build
        operationId: startTestCase
        parameters:
          instanceId:
            name: test-instance-id
            in: path
            required: true
            schema:
              type: string
          buildId:
            name: test-build-id
            in: path

```



```

        required: true
        schema:
          type: string
      suiteId:
        name: test-suite-id
        in: path
        required: true
        schema:
          type: string
      caseId:
        name: test-case-id
        in: path
        required: true
        schema:
          type: string
    responses:
      '200':
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TestCase"

  api/test-instance/{test-instance-id}/test-build/{test-build-id}/test-suite/{test-suite-id}/test-case/{test-case-id}/stop/:
    post:
      summary: stop test build
      operationId: stopTestCase
      parameters:
        instanceId:
          name: test-instance-id
          in: path
          required: true
          schema:
            type: string
        buildId:
          name: test-build-id
          in: path
          required: true
          schema:
            type: string
        suiteId:
          name: test-suite-id
          in: path
          required: true
          schema:
            type: string
        caseId:
          name: test-case-id
          in: path
          required: true
          schema:
            type: string
      requestBody:
        application/json:
          schema:

```

```

      result_status:
        type: string
    responses:
      '200':
        content:
          application/json:
            schema:
              $ref: "#/components/schemas/TestCase"

```

Репозитории

АПИ <https://github.com/GulDilin/fog-computing-test-storage-api>

Фронт <https://github.com/GulDilin/fog-computing-test-storage-front>

Процесс создания окружения на AWS

Создание Elastic BeanStalk

All applications Actions Create a new application

Filter results matching the display values

Application name	Environments	Date created	Last modified	ARN
fog_computing_test_storage_api	Fogcomputingteststorageapi-env	2021-12-24 00:43:01 UTC+0300	2021-12-24 00:43:01 UTC+0300	arn:aws:elasticbeanstalkus-east-2:572227370164:application/fog_computing_test_storage_api

Application 'fog_computing_test_storage_api' environments Create a new environment

Filter results matching the display values

Environment name	Health	Date created	Last modified	URL	Running versions	Platform	Platform state	Tier name
Fogcomputingteststorageapi-env	Severe	2021-12-24 00:43:06 UTC+0300	2021-12-24 21:23:52 UTC+0300	Fogcomputingteststorageapi-env.eba-mcgehycp.us-east-2.elasticbeanstalk.com	Sample Application-4	Python 3.8 running on 64bit Amazon Linux 2	Supported	WebServer

При этом автоматически создаются экземпляры EC2

Instances (2) Info Connect Instance state Actions Launch instances

Search

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
Fogcomputing...	i-0b758f4abfdb9c7f8	Terminated	t2.micro	-	No alarms	us-east-2a	-	-	-
Fogcomputing...	i-0b11486f7347aacc5	Running	t2.micro	2/2 checks passed	No alarms	us-east-2c	ec2-3-144-229-190.us-...	3.144.229.190	-

Создание БД

Перейти в Configuration в окружении Elastic BeanStalk

▼ Fogcomputingteststorageapi-env

Go to environment [↗](#)

Configuration

Logs

Health

Monitoring

Alarms

Managed updates

Events

Tags

Настроить пункт Database

Database	Endpoint: aajwo819m0cayx.czllafgetmms.us-east-2.rds.amazonaws.com:5432 Availability: Low (one AZ) Engine: postgres Instance class: db.t4g.micro Retention: Create snapshot Storage: 10 Username: docdb	Edit
----------	--	------

При этом поднимается экземпляр Amazon RDS

RDS > Databases										
Databases										
<input type="text" value="Filter by databases"/>										
<div>Group resources ↺ Modify Actions Restore from S3 Create database</div>										
<div>< 1 > ⓘ</div>										
<input type="checkbox"/>	DB identifier	Role	Engine	Region & AZ	Size	Status	CPU	Current activity	Maintenance	VPC
<input type="radio"/>	aajwo819m0cayx	Instance	PostgreSQL	us-east-2c	db.t4g.micro	Available	5.25%	0 Connections	none	vpc-ed:

Обновить конфигурацию в приложении

Connecting to a database

Elastic Beanstalk provides connection information for attached DB instances in environment properties. Use `os.environ['VARIABLE']` to read the properties and configure a database connection.

Example Django settings file – DATABASES dictionary

```
import os

if 'RDS_HOSTNAME' in os.environ:
    DATABASES = {
        'default': {
            'ENGINE': 'django.db.backends.mysql',
            'NAME': os.environ['RDS_DB_NAME'],
            'USER': os.environ['RDS_USERNAME'],
            'PASSWORD': os.environ['RDS_PASSWORD'],
            'HOST': os.environ['RDS_HOSTNAME'],
            'PORT': os.environ['RDS_PORT'],
        }
    }
```

Создание архива с проектом

Тут нужно соблюсти структуру


```
|-- .ebextensions
|   |-- django.config
|-- ebdjango
|   |-- __init__.py
|   |-- settings.py
|   |-- urls.py
|   |-- wsgi.py
|-- db.sqlite3
|-- manage.py
|-- requirements.txt
```

А также настроить следующий Django.config для автоматического создания базы данных и подключения приложения


```
container_commands:
  01_migrate:
    command: "source /var/app/venv/*/bin/activate && python3 manage.py migrate"
    leader_only: true
option_settings:
  aws:elasticbeanstalk:container:python:
    WSGIPath: fog_computing_test_storage_api.wsgi:application
```

Загрузка проекта

Upload and deploy ×

 To deploy a previous version, go to the [Application Versions](#) page.

Upload application

 Choose file

Version label

► Deployment Preferences

The application version will be deployed using the **All at once** policy.

Current number of instances: 1

Cancel

Deploy

После загрузки проекта пройдут HealthChecks и апи станет доступен. В окружении будет указана ссылка с развернутым приложением

Fogcomputingteststorageapi-env

Fogcomputingteststorageapi-env.eba-mcgehycp.us-east-2.elasticbeanstalk.com [🔗](#) (e-qdmsvph8pr)

Application name: fog_computing_test_storage_api

Refresh

Actions

Создание фронтенд приложения. Создание S3 Bucket

Buckets (2) Info

Refresh Copy ARN Empty Delete Create bucket

Find buckets by name

< 1 > ⚙

	Name	AWS Region	Access	Creation date
<input type="radio"/>	elasticbeanstalk-us-east-2-572227370164	US East (Ohio) us-east-2	Objects can be public	December 24, 2021, 00:43:01 (UTC+03:00)
<input type="radio"/>	fog-computing-front	US East (Ohio) us-east-2	Bucket and objects not public	December 24, 2021, 20:54:33 (UTC+03:00)

На скриншоте верхний – автоматически созданное хранилище исходного кода апи, второе – bucket для фронтенда

Загрузка файлов фронтенда

Objects (4)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Refresh Copy S3 URI Copy URL Download Open Delete Actions Create folder Upload

Find objects by prefix

< 1 > ⚙

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	css/	Folder	-	-	-
<input type="checkbox"/>	favicon.ico	ico	December 24, 2021, 20:55:04 (UTC+03:00)	4.2 KB	Standard
<input type="checkbox"/>	index.html	html	December 24, 2021, 20:55:05 (UTC+03:00)	1.1 KB	Standard
<input type="checkbox"/>	js/	Folder	-	-	-

Создание CloudFront

Distributions (1) Info

Refresh Enable Disable Delete Create distribution

Search all distributions

< 1 > ⚙

<input type="checkbox"/>	ID	Description	Domain name	Alternate domain names	Origins	Status	Last modified
<input type="checkbox"/>	E1LPQNIHYZUE7	-	d2a8jp3lsapd23.cloudfro...	-	fog-computing-front.s3.us-east	Enabled	December 24, 2021 at 5:5...

CloudFront > Distributions > E1LPQNIHYZUE7

E1LPQNIHYZUE7

General

Origins

Behaviors

Error pages

Geographic restrictions

Invalidations

Tags

Details

Distribution domain name

d2a8jp3lsapd23.cloudfront.net

ARN

arn:aws:cloudfront::572227370164:distribution/E1LPQNIHYZUE7

Last modified

December 24, 2021 at 5:57:12 PM UTC

Settings

Edit

Description

-

Alternate domain names

-

Standard logging

Off

Price class

Use all edge locations (best performance)

Cookie logging

Off

Supported HTTP versions

HTTP/2, HTTP/1.1, HTTP/1.0

Default root object

index.html

AWS WAF

-

Для возможности подключения к SPA приложению нужно было установить следующие конфигурации для доступа CloudFront к нужному Bucket

Settings

Origin domain

Choose an AWS origin, or enter your origin's domain name.

Q fog-computing-front.s3.us-east-2.amazonaws.com X

Origin path - optional [Info](#)

Enter a URL path to append to the origin domain name for origin requests.

Enter the origin path

Name

Enter a name for this origin.

fog-computing-front.s3.us-east-2.amazonaws.com

S3 bucket access [Info](#)

Use a CloudFront origin access identity (OAI) to access the S3 bucket.

- ☐ Don't use OAI (bucket must allow public access)
- ☒ Yes use OAI (bucket can restrict access to only CloudFront)

Origin access identity

Select an existing origin access identity (recommended) or create a new identity.

access-identity-fog-computing-front.s3.us-east-2.amazonaws.com ▼

Create new OAI

Bucket policy

Update the S3 bucket policy to allow read access to the OAI.

- ☒ No, I will update the bucket policy
- ☐ Yes, update the bucket policy

А также настроить перенаправление ошибок

CloudFront > Distributions > E1LPQNIHYZUVE7

E1LPQNIHYZUVE7

General | Origins | Behaviors | **Error pages** | Geographic restrictions | Invalidations | Tags

Error pages

Edit Delete Create custom error response

HTTP error code	Minimum TTL (seconds)	Response page path	HTTP response code
<input type="radio"/> 403	10	/index.html	200
<input type="radio"/> 404	10	/index.html	200

Теперь приложение фронтенд приложение будет доступно

Скришноты фронтенд приложения

Fog computing tests storage

Total instances: 1

< 1 >

Id	Name	Description
1	Super Programm	XXX

Fog computing tests storage

Instance 1

Total builds: 3

< 1 >

Id	Number ↑	Started	Finished
1	1	1/1/1970, 3:00:00 AM	1/1/1970, 3:00:00 AM
2	2	1/1/1970, 3:00:00 AM	1/1/1970, 3:00:00 AM
3	3	1/1/1970, 3:00:00 AM	1/1/1970, 3:00:00 AM

Fog computing tests storage

Instance 1 Build 1

Total suites: 3

< 1 >

Id	Name	Started	Finished
1	Smoke	1/1/1970, 3:00:00 AM	1/1/1970, 3:00:00 AM
2	Regression	1/1/1970, 3:00:00 AM	1/1/1970, 3:00:00 AM
3	Full	12/24/2021, 9:42:28 PM	1/1/1970, 3:00:00 AM

Fog computing tests storage					
Instance 1 Build 1 Suite 1					
Total cases: 3					< 1 >
Id		Name	Started	Finished	Status
1		Case 1	12/24/2021, 9:58:07 PM	12/24/2021, 9:58:18 PM	SUCCESS
2		Case 2	1/1/1970, 3:00:00 AM	1/1/1970, 3:00:00 AM	ERROR
3		Case 3	1/1/1970, 3:00:00 AM	1/1/1970, 3:00:00 AM	FAILED

Вывод

В процессе выполнения данной лабораторной работы была реализована схема развертывания клиент серверного приложения на облачном сервисе AWS с использованием базы данных, автоматического расширения, бесшовного деплоя и балансировщика. Я получил опыт создания облачного окружения с настройкой множества параметров и буду использовать это дальше в своей работе и обучении.