

Сборка образов ОС при помощи Packer

Развитие проекта `infra`

В прошлом ДЗ вы создали проект `infra` на GitHub, в котором сейчас должны храниться скрипты для настройки системы и деплоя приложения. Убедитесь что данный проект находится у вас на локальной машине.

Если у вас нет репозитория `infra` на GitHub, выполните сначала предыдущее ДЗ.

Проект *infra* и проверка ДЗ

Создайте новую ветку в вашем локальном репозитории для выполнения данного ДЗ. Т.к. задание посвящено работе с Packer и базовыми образами, то ветку можно назвать `base-os-packer`.

Проверка данного ДЗ, как и многих последующих, будет производиться через Pull Request ветки с ДЗ к ветке мастер и добавлению в Reviewers пользователей

Artemmkin и **serjs**.

После того, как **один** из преподавателей сделает approve пул реквеста, ветку с ДЗ можно смерджить.



Установка Packer

Скачайте версию Packer для вашей ОС, перейдя по данной ссылке.

Распакуйте скачанный zip архив и поместите бинарный файл в директорию, путь до которой содержится в переменной окружения PATH.

Проверить установку Packer можно командой:
`$ packer -v`

Credentials

Если мы не работаем с GCP через браузерную консоль (GCP Console), то для управления ресурсами GCP через другие программы, как Packer и Terraform, нам нужно предоставить этим инструментам информацию (credentials) для аутентификации и управлению ресурсами GCP нашего аккаунта.



Application Default Credentials (ADC)

Установка ADC позволяет приложениям, работающим с GCP ресурсами и использующим Google API библиотеки, управлять ресурсами GCP через авторизованные API вызовы, используя credentials вашего пользователя.

Создайте ADC:

```
$ gcloud auth application-default login
```

Создаем Packer template

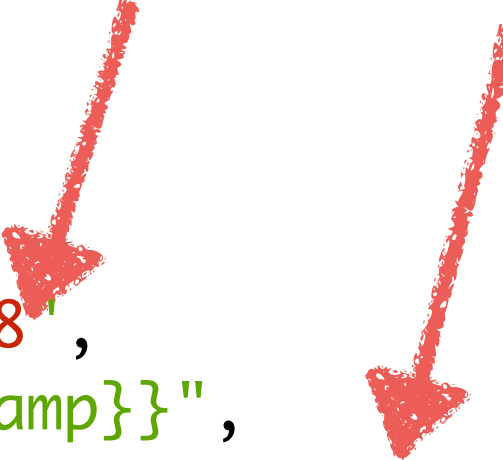
Создайте файл `ubuntu16.json` в директории `infra/packer`. Это и будет наш Packer шаблон, содержащий описание образа VM, который мы хотим создать.

Для нашего тестового приложения мы соберем образ VM с предустановленными (baked) Ruby и MongoDB.

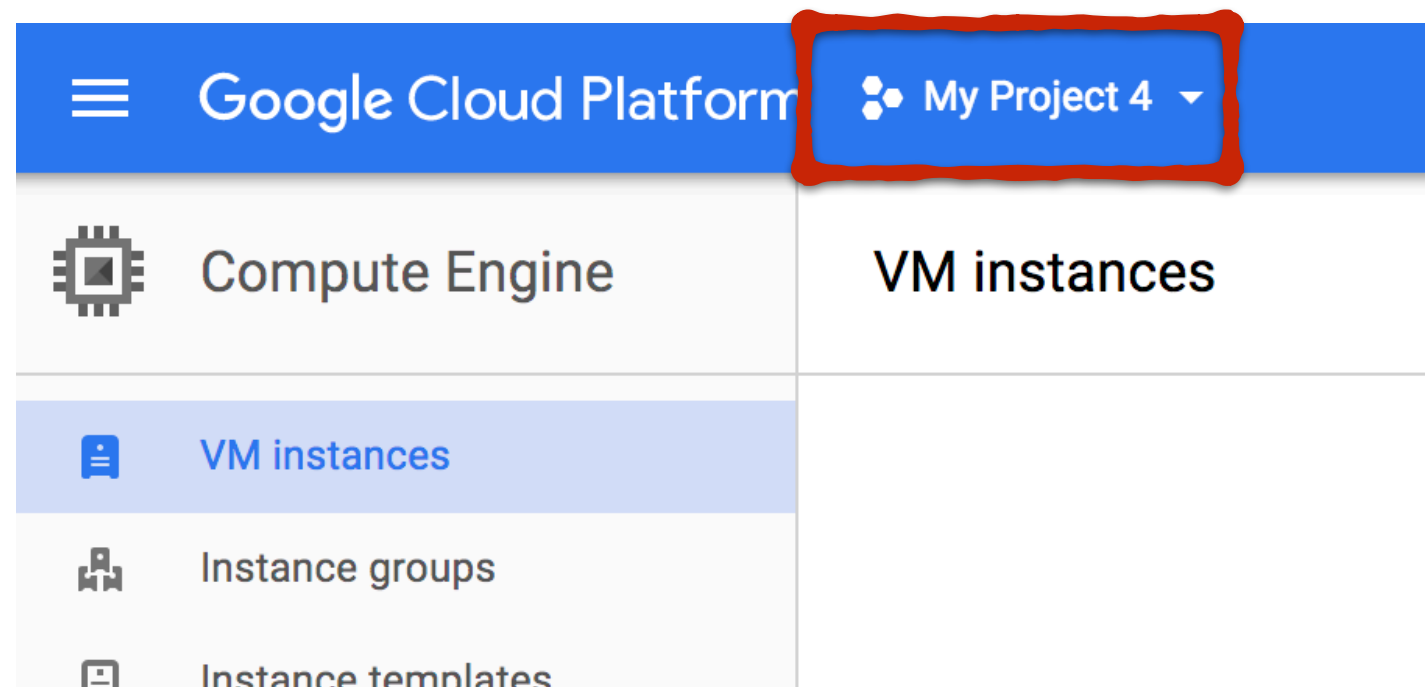
Определим Packer builders в шаблоне ubuntu16.json:

Опции, которые нужно заменить (см. след слайд)

```
{
  "builders": [
    {
      "type": "googlecompute",
      "project_id": "steam-strategy-174408",
      "image_name": "reddit-base-{{timestamp}}",
      "source_image": "ubuntu-1604-xenial-v20170815a",
      "zone": "europe-west1-b",
      "ssh_username": "appuser",
      "machine_type": "f1-micro"
    }
  ]
}
```



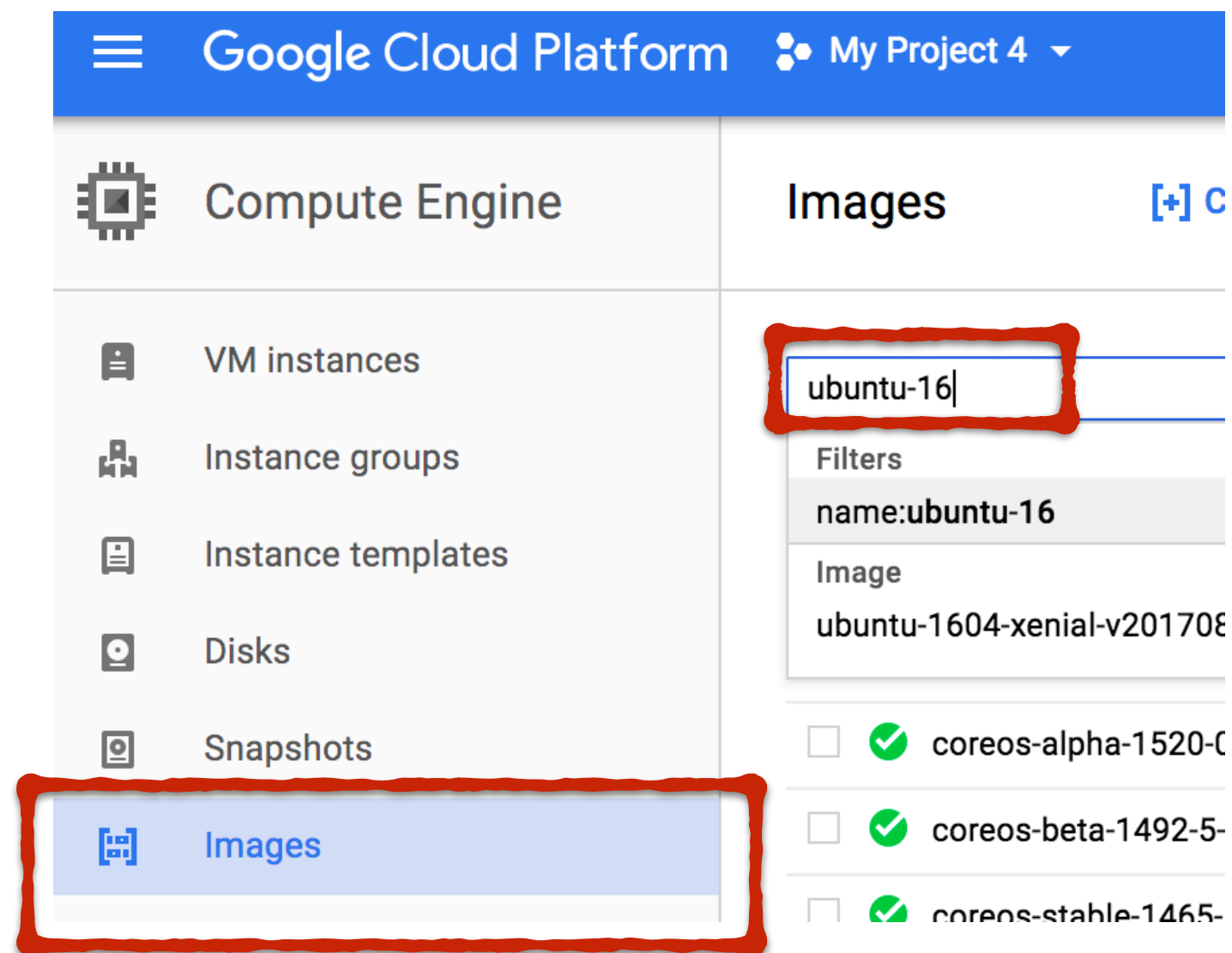
Чтобы найти ID вашего проекта, нажмите в консоли на имя проекта



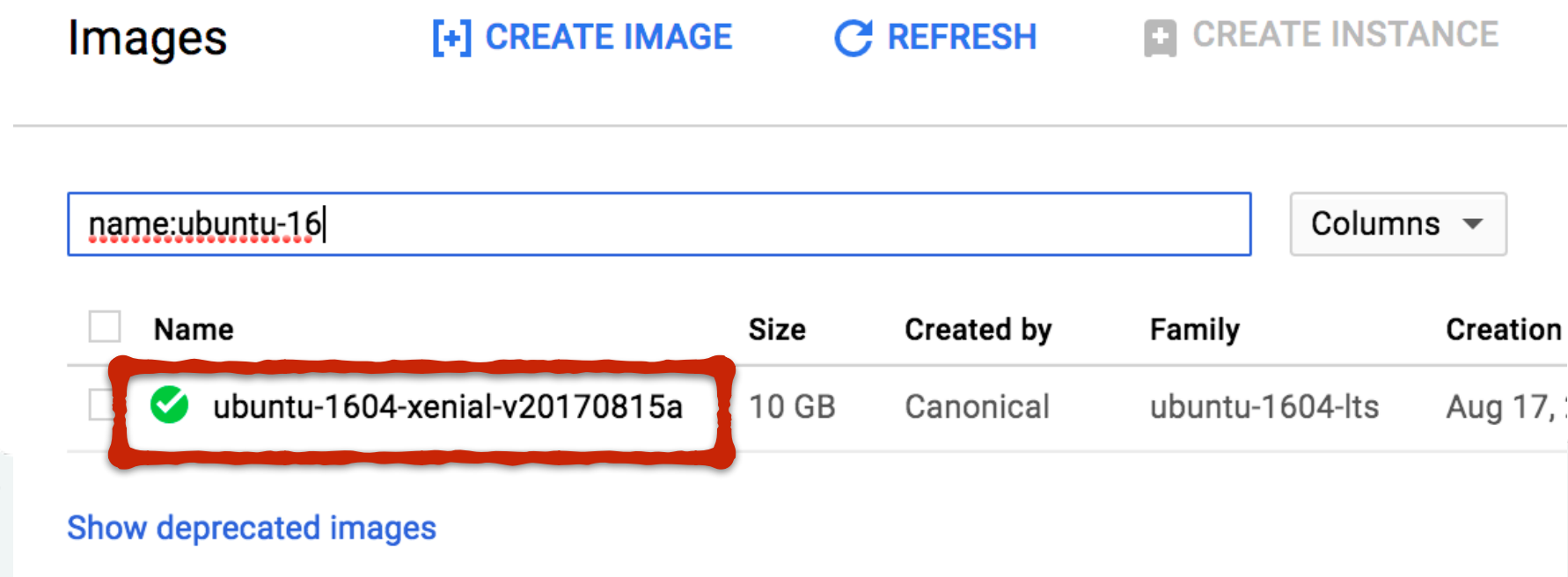
Скопируйте ID и вставьте в ваш Packer шаблон



Для того чтобы правильно указать source_image, т.е. образ на основе которого мы будем создавать свой, нажмите на Images в консоли управления и введите в поиске ubuntu-16



Название образа
скопируйте в ваш
шаблон



Пояснения Packer builder

Разберем основные настройки секции builder нашего шаблона:

- **type: “googlecompute”** - что будет создавать виртуальную машину для билда образа (в нашем случае Google Compute Engine)
- **project_id: “steam-strategy-174408”** - id вашего проекта
- **image_name: “reddit-base-{{timestamp}}”** - имя создаваемого образа
- **source_image: “ubuntu-1604-xenial-v20170815a”** - что взять за базовый образ для нашего билда

Пояснения Packer builder

Разберем основные настройки секции builder нашего шаблона:

- **zone: "europe-west1-b"** - зона, в которой запускать VM для билда образа
- **ssh_username: "appuser"** - временный пользователь, который будет создан для подключения к VM во время билда и выполнения команд провижинера (о нем поговорим ниже)
- **machine_type: "f1-micro"** - тип инстанса, который запускается для билда

Provisioners

Если `builders` секция отвечает за создание виртуальной машины для билда и создание машинного образа в GCP, то секция `provisioners` позволяет устанавливать нужное ПО, производить настройки системы и конфигурацию приложений на созданной VM.

Используя скрипты для установки Ruby и MongoDB из предыдущего ДЗ, определим два провижинера.

Используем shell provisioner, который позволяет запускать bash команды на запущенном инстансе. После секции “builders” определим провижинер внутри нашего шаблона для установки Ruby.

```
"provisioners": [  
  {  
    "type": "shell",  
    "script": "scripts/install_ruby.sh"  
  }  
]
```

Добавим провижинер для установки MongoDB.

```
"provisioners": [  
  {  
    "type": "shell",  
    "script": "scripts/install_ruby.sh"  
  },  
  {  
    "type": "shell",  
    "script": "scripts/install_mongodb.sh",  
    "execute_command": "sudo {{.Path}}"  
  }  
]
```

Опция `execute_command` позволяет указать, каким способом будет запускаться скрипт. Т.к. команды по установке `mongodb` требуют `sudo`, то мы указываем, что запускать скрипт следует с `sudo`. В самом скрипте `install_mongodb.sh` `sudo` можно не использовать.

В итоге наш racker шаблон должен выглядеть примерно вот так:

```
{
  "builders": [
    {
      "type": "googlecompute",
      "project_id": "steam-strategy-174408",
      "image_name": "reddit-base-{{timestamp}}",
      "source_image": "ubuntu-1604-xenial-v20170815a",
      "zone": "europe-west1-b",
      "ssh_username": "appuser",
      "machine_type": "f1-micro"
    }
  ],
  "provisioners": [
    {
      "type": "shell",
      "script": "scripts/install_ruby.sh"
    },
    {
      "type": "shell",
      "script": "scripts/install_mongodb.sh",
      "execute_command": "sudo {{.Path}}"
    }
  ]
}
```


Скрипты для провижининга

Внутри директории `packer` создайте директорию `scripts` для скриптов, которые будут использовать провижинерами. Поместите, в эту директорию скрипты `install_ruby.sh` и `install_mongodb.sh` из предыдущего ДЗ.

Скрипты `install_ruby.sh` и `install_mongodb.sh` также доступны для скачивания (нажмите на имена скриптов, чтобы скачать). Убедитесь, что ваши скрипты не имеют существенных отличий с теми, что доступны по ссылкам. Поправьте ваши скрипты при необходимости.



Проверка на ошибки

Проверьте, не допустили ли вы ошибок при создании шаблона, используя команду `packer validate`:

```
$ packer validate ./ubuntu16.json
```

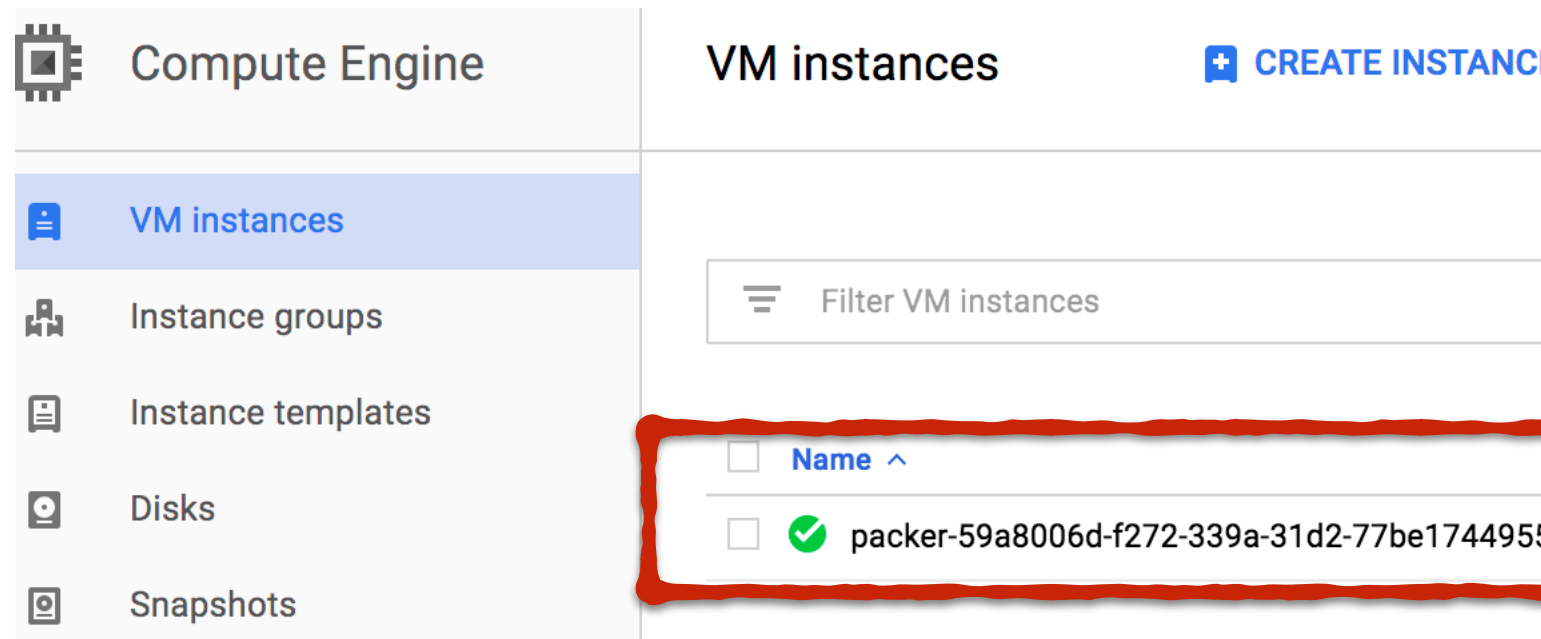
Поправьте ошибки, если они есть.

Packer build

Если проверка на ошибки прошла успешно, то запустите build образа:

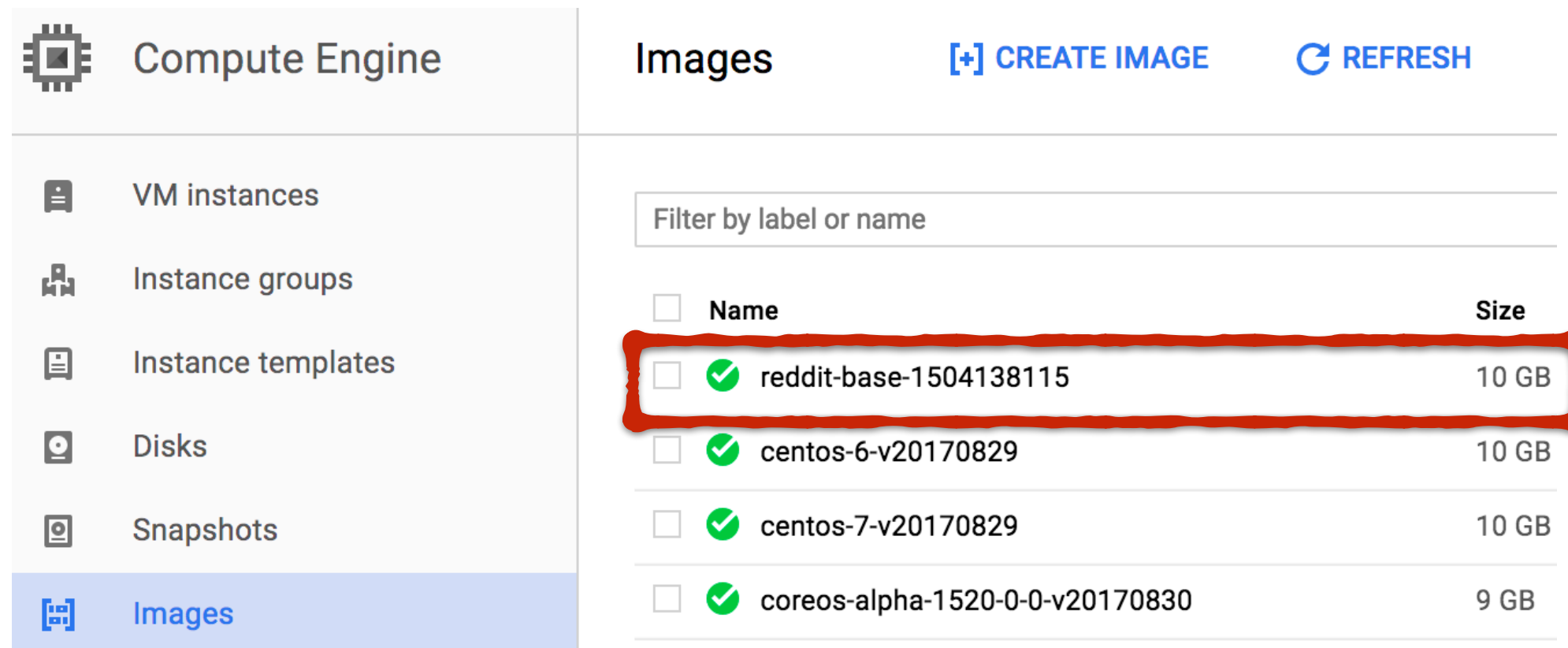
```
$ packer build ubuntu16.json
```

В браузерной консоли можно увидеть, как Packer запустил инстанс VM.



Проверяем созданный образ

В браузерной консоли перейдите по пути Compute Engine -> Images. Найдите свой образ.



The screenshot shows the Google Cloud Platform console interface. On the left sidebar, under the 'Compute Engine' section, the 'Images' option is selected and highlighted in blue. The main content area is titled 'Images' and includes a '+ CREATE IMAGE' button and a 'REFRESH' button. Below this is a search bar labeled 'Filter by label or name'. A table lists the available images:

<input type="checkbox"/>	Name	Size
<input type="checkbox"/>	✓ reddit-base-1504138115	10 GB
<input type="checkbox"/>	✓ centos-6-v20170829	10 GB
<input type="checkbox"/>	✓ centos-7-v20170829	10 GB
<input type="checkbox"/>	✓ coreos-alpha-1520-0-0-v20170830	9 GB

Деплоим приложение


Как и в прошлый раз задеплоим наше тестовое приложение, но на этот раз нам нужно будет проделать меньше работы, т.к. часть пакетов уже содержится в образе VM, который мы создали.

Задаем нужные
характеристики
машины (тип инстанса
не больше g1-small)

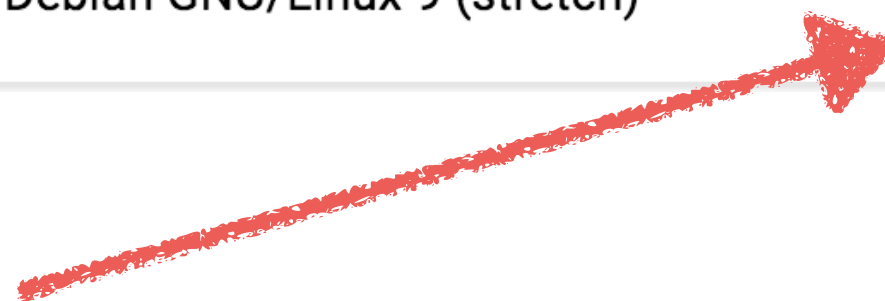
Name ?
reddit-app

Zone ?
europe-west1-b

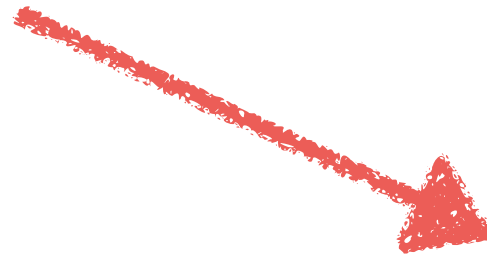
Machine type
small (1 shared... 1.7 GB memory [Customize](#)
[Upgrade your account](#) to create instances with up to 64 cores

Boot disk ?
 New 10 GB standard persistent disk
Image
Debian GNU/Linux 9 (stretch) [Change](#)

Нажимаем изменить образ



Жмем custom images и
выбираем созданный
нами образ



Boot disk

Select an image or snapshot to create a boot disk; or attach an ex

OS images Application images **Custom images** Snaps

Show images from

My Project 4

- ☒ **reddit-base-1504138115**
Ubuntu 16.04 for reddit app. Has mongodb and some gems preinstal
Created from My Project 4 on Aug 31, 2017, 3:10:56 AM



Подключение по SSH

После того как инстанс запустился, вам необходимо подключиться к инстансу по SSH, используя ключи пользователя `appuser`, которые вы сгенерировали на прошлом занятии.

```
$ ssh appuser@<instace_public_ip>
```


Установка зависимостей и запуск приложения

Для деплоя приложения можно использовать созданный вами скрипт `deploy.sh` или перечисленные ниже команды:

```
$ git clone https://github.com/Artemmmkin/reddit.git  
$ cd reddit && bundle install  
$ puma -d
```

Проверяем, что сервер приложения запустился:

```
$ ps aux | grep puma
```



Проверка работы приложения

Предварительно убедитесь, что вам доступен порт сервера приложения в правилах firewall

📄 146.148.56.222:9292

Monolith Reddit

User created

Menu

[All posts](#)

[New post](#)

Закомитьте результаты

В случае успешного создания образа VM при помощи Rascker закомитьте, результаты вашей работы в созданную ранее ветку.

Самостоятельные задания

1. Необходимо параметризовать созданный шаблон, используя пользовательские переменные (см. лекцию).

Какие опции шаблона должны быть параметризованы:

- ID проекта (required)
- source_image (required)
- machine_type

Примечание: required означает, что пользовательская переменная должна быть обязательна для определения и не иметь значения по умолчанию

Самостоятельные задания

2. Исследовать другие опции builder для GCP ([ссылка](#)). Какие опции точно хотелось бы видеть:

- Описание образа
- Размер и тип диска
- Название сети
- Теги

Запросите ревью ваших наработок

Результаты вашей работы должны содержаться в созданной в самом начале работы ветке. После окончания работы, сделайте Pull Request к ветке master. И запросите Review у преподавателей (пользователи **Artemmkin** и **serjs**)

Задание со звездочкой

Чтобы попрактиковать подход к управлению инфраструктурой Immutable infrastructure, о котором говорили на вебинаре, попробуйте “запечь” (bake) в образ VM все зависимости приложения и сам код приложения. Результат должен быть таким: запускаем инстанс из созданного образа и на нем сразу же имеем запущенное приложение.

Созданный шаблон должен называться `immutable.json` и содержаться в директории `packer`. Дополнительные файлы можно положить в директорию `packer/files`.