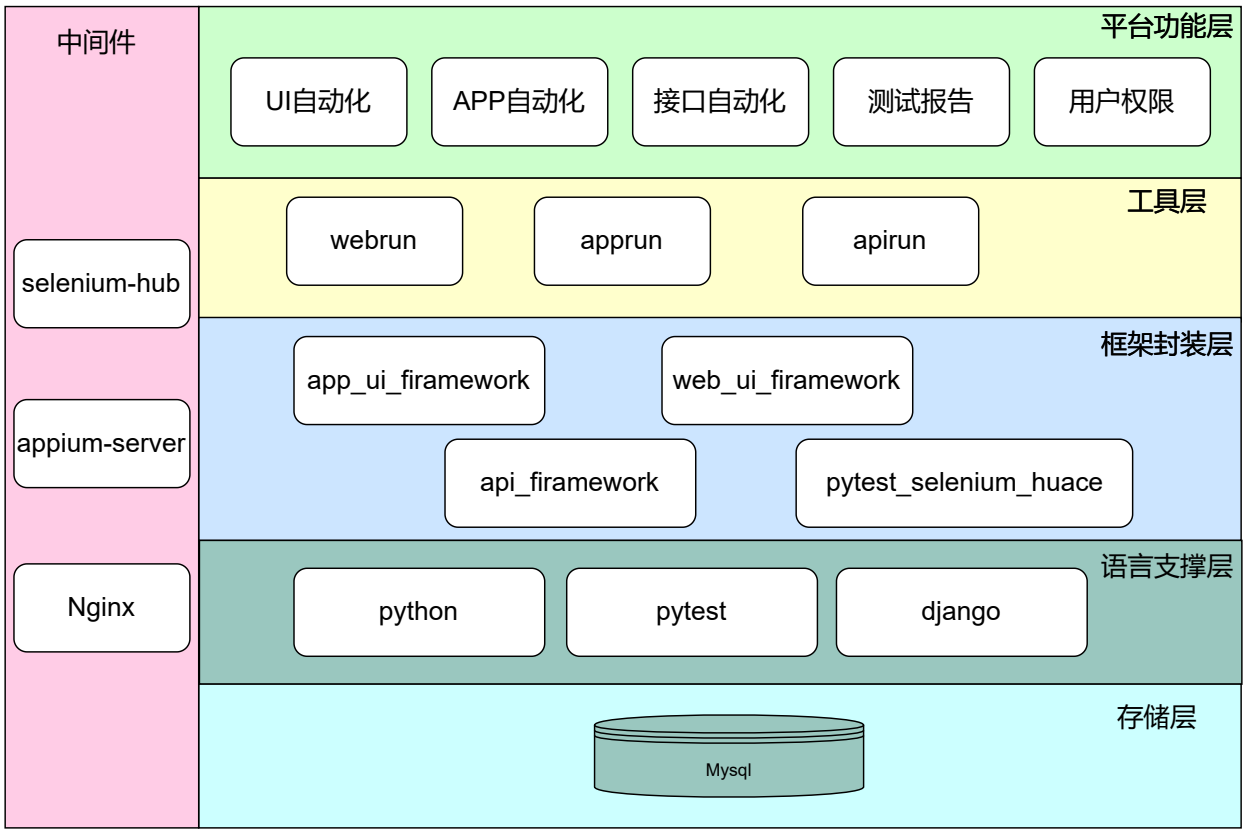
华测教育-测试开发课程07

三丰老师

> 面试的时候，如何讲测试开发的内容，或者说 自动化平台的内容？ 拿纸笔把架构图画一遍。
> 必要的时候 直接拿电脑演示。

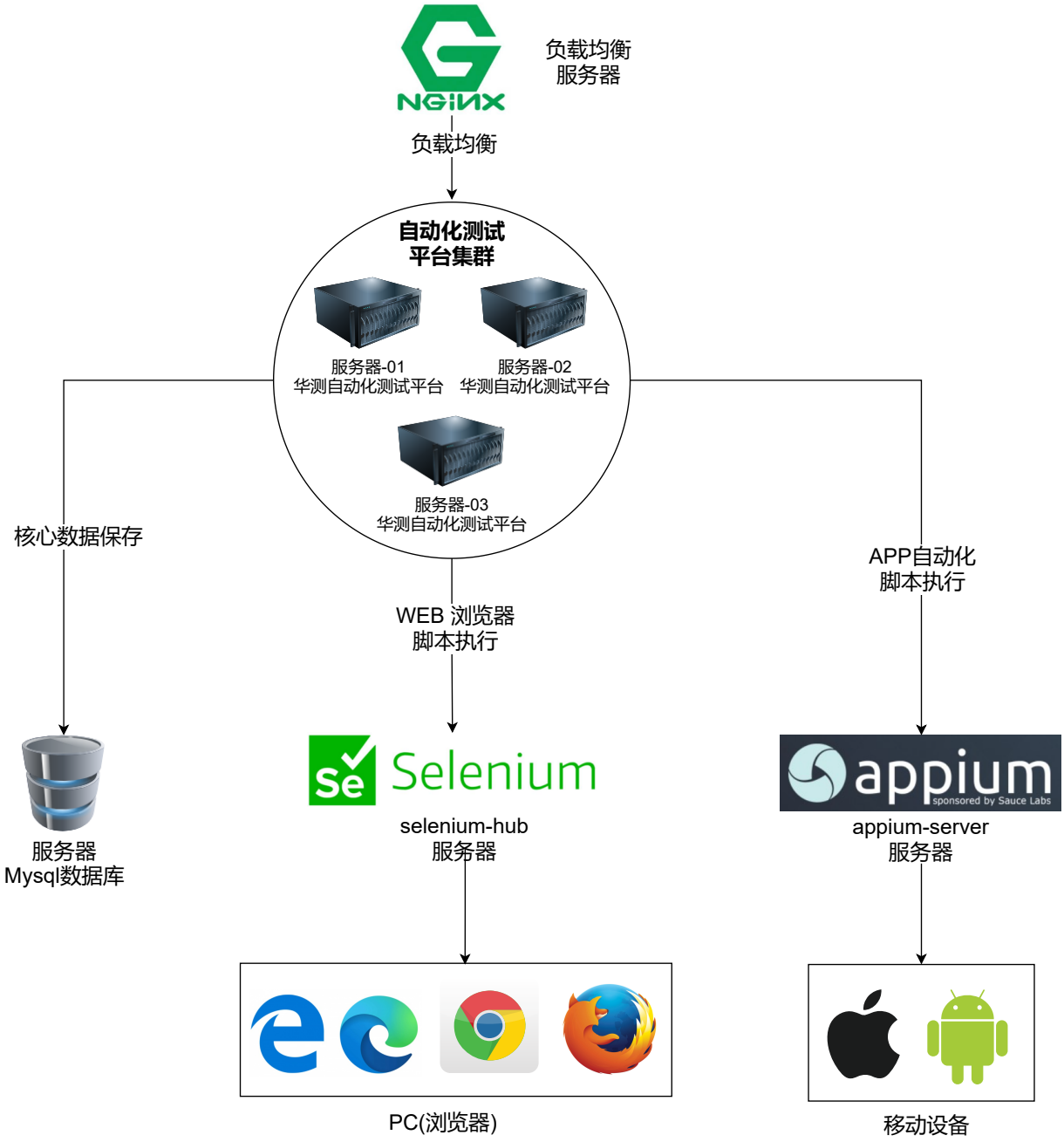
平台架构图



> 1. 没在项目中运用，没落地， 如何在面试中回答 部署的问题?
> 2. 我打算部署真正的使用，项目部署方案...

平台部署网络拓扑

> 集群： 同一个系统部署到不同的“服务器”上面 【**同一套系统运行多个实例**】
> 同一台机器上，程序绑定的端口号不能重复，否则绑定失败
> **真**集群： 不同的机器，运行同一个系统
> **伪**集群： 同一个机器，通过不同的端口号运行多个实例
> **集群**： 每次测试用例执行/用户的请求，都会占用服务器的CPU/内存/网络/磁盘资源。单台机器的资源是有限的。如何支撑更多用户和用例的使用与执行？ 需要多台机器提供服务。



数据库服务器地址： 192.168.1.103
平台集群-01地址：192.168.1.109
平台集群-02地址：192.168.1.110
selenium-hub地址： 192.168.1.109
appium-server服务器地址：192.168.1.119
pc浏览器集群-- 192.168.1.119
移动设备：模拟、真机

平台部署文档【完整版】

> 本教程基于 centos7
> windows上面的操作和linux大致相同， 都可以通过命令进行处理。 命令的使用也是一致的

> 前置操作： 自动化测试平台运行所在的机器，需要有完整的python3.x环境
> 运行平台的服务器上面需要上传 自研平台的代码、自研工具的代码、自研插件的代码

> 在服务器上面运行的时候， 有些服务器上面可能python2 与 python3共存， 所以我们在执行命令的时候，会通过python3 pip3 来区分

python 虚拟环境

> 不同的python项目需要不同的依赖， 一个服务器可以需要运行多个python项目。
> 为了避免项目之间依赖模块冲突导致项目无法运行， 所以我们在服务器上面运行项目： 都通过虚拟环境

virtualenvwrapper这个软件包可以让我们管理虚拟环境变得更加简单。不用再跑到某个目录下通过virtualenv来创建虚拟环境，并且激活的时候也要跑到具体的目录下去激活。

安装virtualenvwrapper

```
# linux
pip3 install virtualenvwrapper
# windows
pip3 install virtualenvwrapper-win
```

virtualenvwrapper基本使用

```
# 如果你的环境存在python2 与python3，那么执行命令前，先执行以下命令
export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3

# 每次执行命令前执行一下下面这个指令
source /usr/local/bin/virtualenvwrapper.sh

# 创建虚拟环境 - 会在你当前用户下创建一个Env的文件夹，然后将这个虚拟环境安装到这个目录下
mkvirtualenv hctestedu.com

# 创建虚拟环境的时候指定Python版本，例如：
mkvirtualenv --python==C:\Python36\python.exe hctestedu.com

# 切换到某个虚拟环境
workon hctestedu.com

```

```
# 退出当前虚拟环境
deactivate

# 删除某个虚拟环境
rmvirtualenv hctestedu.com

# 列出所有虚拟环境
lsvirtualenv

# 进入到虚拟环境所在的目录
cdvirtualenv
```

自动化工具安装

> 以下操作需进入虚拟环境后再执行

> workon hctestedu.com

pytest插件安装 - pytest-selenium-huace

```
pip3 install /u01/test/platformV3/pytest-selenium
```

接口工具apirun工具安装

```
cd /u01/test/platformV3/03-apirunner/
python3 setup.py install
```

WEB工具 webrun 安装

```
cd /u01/test/platformV3/08-seleniumrunner/
python3 setup.py install
```

APP工具 apprun安装

```
cd /u01/test/platformV3/12-appiumrunner
python3 setup.py install
```

运行平台

> 此教程中没有写数据库的安装与配置，需自行完成

> 记得进入虚拟环境 workon hctestedu.com

```
# 0. 进入到项目所在的文件夹
# cd /u01/test/platformV3/13-huace_platform/
# !!!!记得修改配置文件里面的 数据库地址哦！！！！
# settings.py 文件 修改配置配置    ALLOWED_HOSTS = ['*']

# 1. 安装依赖
pip3 install -r requirements.txt

# 数据库操作 -- 只有一个数据库，第一遍运行的时候执行一次即可
# 2. 创建数据库
python3 auto_test_platform/manage.py  makemigrations

# 3. 生成数据库
python3 auto_test_platform/manage.py  migrate

# 4. 创建管理员用户
python3 auto_test_platform/manage.py  createsuperuser

# 5. 启动服务 - 方式1
python3 auto_test_platform/manage.py runserver 0.0.0.0:8000
# 5.1 启动服务 - 方式2 后台运行
nohup python3 auto_test_platform/manage.py runserver 0.0.0.0:8000 > huace_platform.log 2>&1 &
```

负载均衡搭建与配置

nginx： 它是一个web服务器，同时也是代理服务器
web服务器： 你可以直接讲html js 等静态资源，由nginx提供web访问服务
代理服务器： 收到请求后，根据配置，将请求转交给后端其他服务进行处理

安装nginx

> windows下，安装比linux简单。直接下载安装即可，配置文件一样的，操作不一样

CentOS7.X中使用yum安装nginx的方法

nginx官方文档说明：http://nginx.org/en/linux_packages.html#RHEL-CentOS

一、安装前准备：

`yum install yum-utils`

二、添加源

进入文件夹： cd /etc/yum.repos.d/

新建文件: `nginx.repo`

输入以下信息

```
[nginx-stable]
name=nginx stable repo
baseurl=http://nginx.org/packages/centos/$releasever/$basearch/
gpgcheck=1
enabled=1
gpgkey=https://nginx.org/keys/nginx_signing.key
[nginx-mainline]
name=nginx mainline repo
baseurl=http://nginx.org/packages/mainline/centos/$releasever/$basearch/
gpgcheck=1
enabled=0
gpgkey=https://nginx.org/keys/nginx_signing.key
```

三、安装nginx

```
yum -y install nginx
```

四、nginx操作

```
# 启动nginx
setsebool -P httpd_can_network_connect 1
```

```
systemctl start nginx

# 加入开机启动：
systemctl enable nginx

# 查看nginx的状态：
systemctl status nginx

# 停止
systemctl stop nginx

# 重新加载配置
systemctl reload nginx
```

nginx配置负载均衡

修改配置： /etc/nginx/nginx.conf

```
# nginx要做的事情，都是配置文件里面指定
events {
  worker_connections  1024;
} # 必写

http{
  # 自动化测试平台 服务器集群
  upstream django_servers {
    # 两个django服务 --
    server 192.168.1.109:8000;
    server 192.168.1.110:8000;
  }

  server {
        listen 80; # 虚拟服务器,端口默认 80
        proxy_read_timeout 120s;       # 请求超时时间
        proxy_next_upstream http_404; # 故障转移的条件 404


        location / { # 根据用户的url 走不同的流程。 通配
          # 将代理请求返回的内容作为响应
          proxy_set_header Host $http_host;
          proxy_pass http://django_servers;
        }
```

```
    }
}
```

windows下：nginx常用操作

> windows下载之后，直接解压就可以用啦
> http://nginx.org/download/nginx-1.20.2.zip
> 配置文件就在conf目录

双击启动： nginx.exe

快速停止或关闭Nginx：nginx -s stop

正常停止或关闭Nginx：nginx -s quit

配置文件修改重装载命令：nginx -s reload

> 以下的两个问题： 用于向面试官证明你真正的做过这件事
> 例如面试官提问： 你做这个测试平台的时候，碰到过什么问题？ 【写代码的封装问题，部署的问题】

集群部署后的问题1 - 测试报告文件不同步

现状：用户每个请求，轮询 分配到不同的机器进行处理。

问题：生成的测试报告散落在各个服务器上面，访问的时候可能出现404

解决办法：nginx容错机制，一台机器上找不到，就转到下一台机器上面去

```
proxy_next_upstream http_404; # 故障转移的条件 404
```

集群部署后的问题2 - 登录状态如何同步？【面试必问：集群后的session问题】

现状：用户每个请求，随机分配到不同的机器进行处理。

问题：系统在不同的服务器，登录后的信息，每个服务器都有不同的存储。A服务器的登录信息与B服务器的登录信息不匹配

解决办法：django中将用户session信息保存在数据库内，只需要保证同一个域名即可

> session放在cookie存储。 虽然是同一个系统，但是访问的域名不一样， cookie不相通 -----
> 如果要两者保持同样的登录状态，需要统一的cookie --- 同一个域名 【nginx做代理，保持了

> 同一个域名访问】
> 不仅仅是前端cookie， 后端也要保存同样的session信息

注意事项

1. Linux系统中/tmp文件夹下的文件是会被清理、删除的。 我们的测试报告存储在tmp目录。所以这个地方如果你们要永久保存，建议修改代码中关于测试报告的存储位置！！
2. 关于虚拟环境中执行自动化测试用例时，找不到webrun，apirun，apprun的问题。
   原因： subprocess.call 执行的过程中，会脱离当前的虚拟环境。
   解决办法，修改代码：`subprocess.call('apirun', shell=True)`