Hi, I’m Jimmy.

I work with organizations to overcome barriers in Engineering projects, through a precise orchestration of people, Process & Architecture.

HTML

DESIGN I UI

SERVE R

IT I ADMINIsTRATIO N

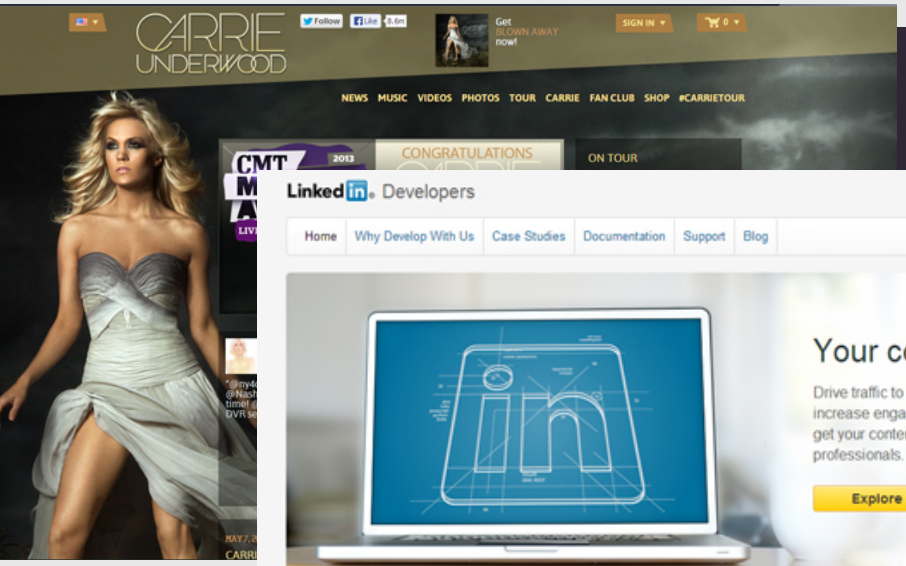
CLIENT SERVICES I Solutions architecture

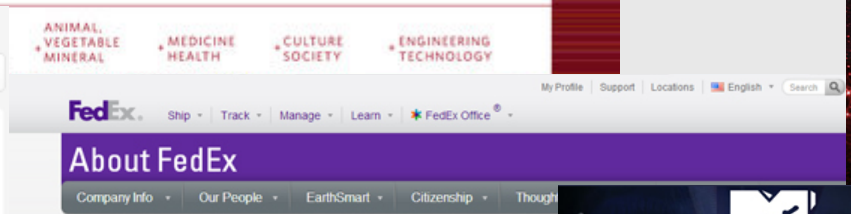
GENERAL CONSULTING

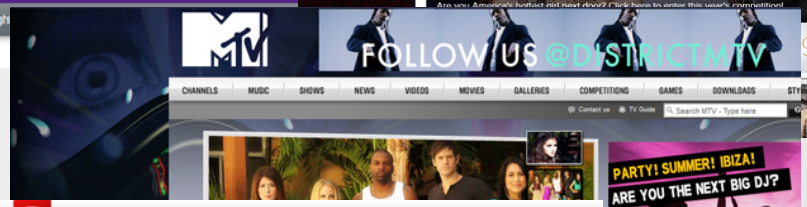
CORPX

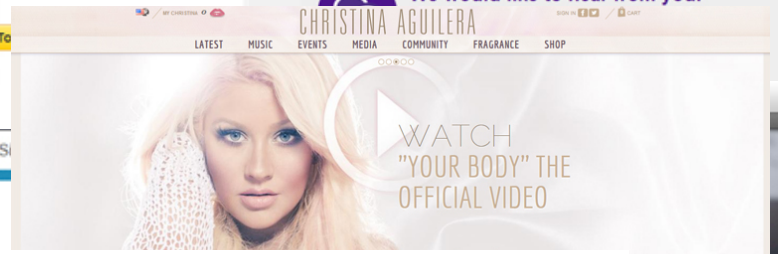
KENZO · HARVARD UNIVERSITY · LinkedIn · LOUVRE · Allianz · PORSCHE · The Walt Disney Company · redhat

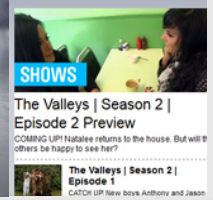
ABSOLUT · MAXIM · YAHOO! · FedEx · CONVERSE · TESLA · Cartier · MTV · ubuntu Linux

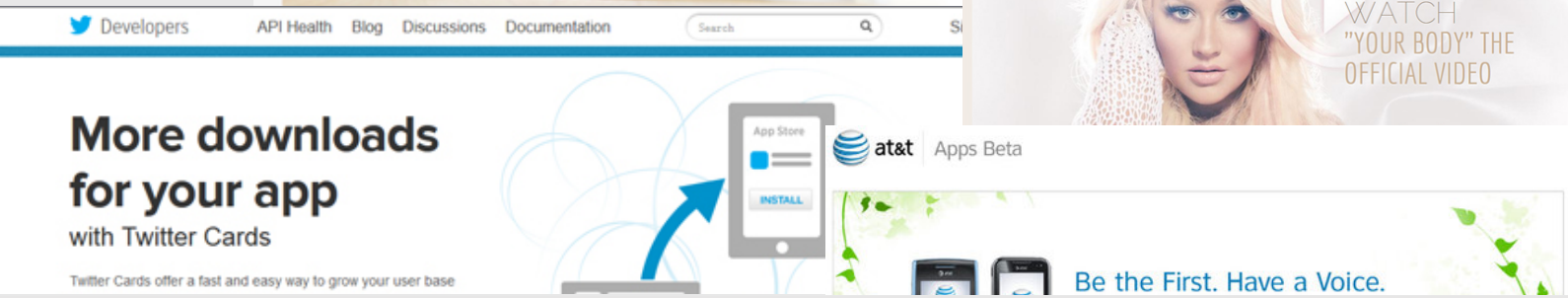


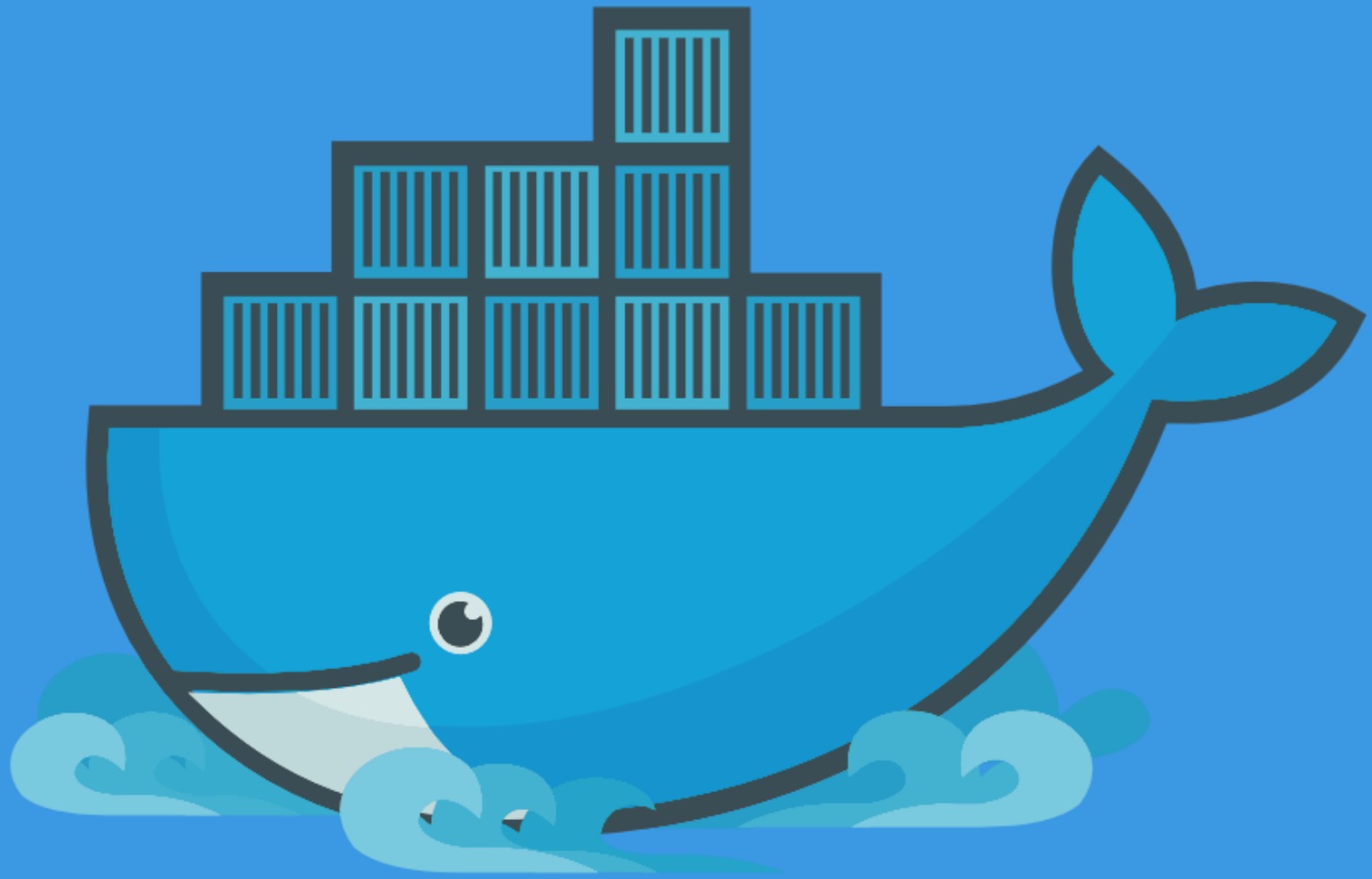






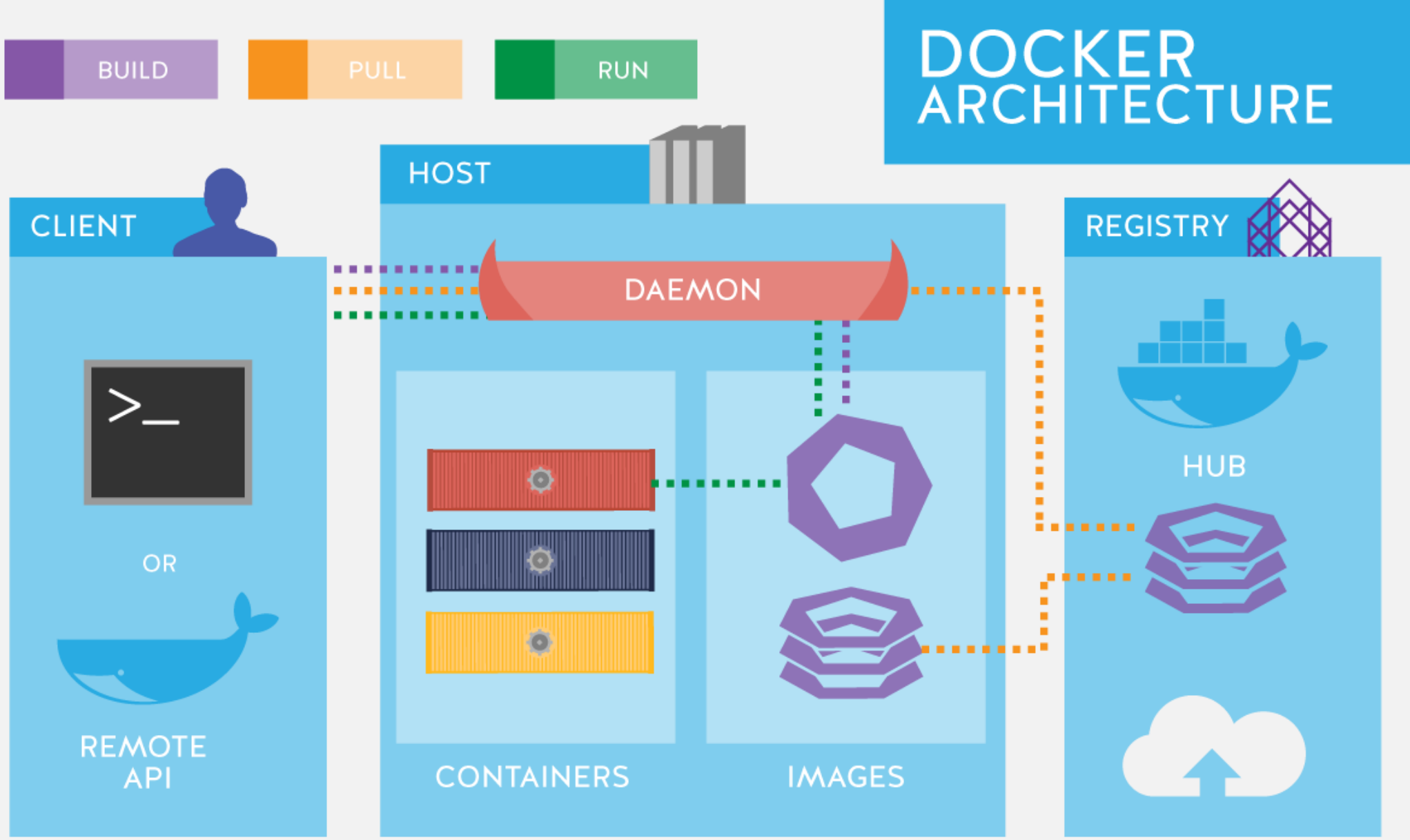






Docker Changes how we build apps


rootfs
rootfs
Debian
BusyBox
lxc, aufs/btrfs
Kernel
docker
writable
Container
add Apache
Image
add emacs
writable
Container
Debian
BusyBox
Image
Kernel


BUILD
PULL
RUN
DOCKER ARCHITECTURE
HOST
CLIENT
REGISTRY
DAEMON
OR
REMOTE API
HUB
CONTAINERS
IMAGES


rootfs
rootfs
Debian
BusyBox
lxc, aufs/btrfs
Kernel
docker
writable
Container
add Apache
Image
add emacs
writable
Container
Debian
BusyBox
Image
Kernel

Running an Image

```
docker run -d
    -p 80:80 \
    -p 443:443 \
    --name nginx-proxy \
    -v /path/to/certs:/etc/nginx/certs:ro \
    -v /etc/nginx/vhost.d \
    -v /usr/share/nginx/html \
    -v /var/run/docker.sock:/tmp/docker.sock:ro \
    --label com.github.jrcs.letsencrypt_nginx_proxy_companion.nginx_proxy \
    jwilder/nginx-proxy



docker run -d \
    --name reverse-proxy \
    -p 443:443  \
    -p  80:80 \
    -e DEFAULT_HOST=npm.keesee.net \
    -v /var/run/docker.sock:/tmp/docker.sock:ro jwilder/nginx-proxy:alpine



docker run -d \
    -v /path/to/certs:/etc/nginx/certs:rw \
    -v /var/run/docker.sock:/var/run/docker.sock:ro \
    --volumes-from nginx-proxy \
    jrcs/letsencrypt-nginx-proxy-companion
```

```
docker run -d
    -p 80:80 \
    -p 443:443 \
    --name nginx-proxy \
    -v /path/to/certs:/etc/nginx/certs:ro \
    -v /etc/nginx/vhost.d \
    -v /usr/share/nginx/html \
    -v /var/run/docker.sock:/tmp/docker.sock:ro \
    --label com.github.jrcs.letsencrypt_nginx_proxy_companion.nginx_proxy \
    jwilder/nginx-proxy



docker run -d \
    --name reverse-proxy \
    -p 443:443  \
    -p  80:80 \
    -e DEFAULT_HOST=npm.keesee.net \
    -v /var/run/docker.sock:/tmp/docker.sock:ro jwilder/nginx-proxy:alpine



docker run -d \
    -v /path/to/certs:/etc/nginx/certs:rw \
    -v /var/run/docker.sock:/var/run/docker.sock:ro \
    --volumes-from nginx-proxy \
    jrcs/letsencrypt-nginx-proxy-companion
```

```
#!/bin/sh

# update your existing list of packages
sudo apt update

#install a few prerequisite packages which let apt use packages over HTTPS
sudo apt install apt-transport-https ca-certificates curl software-properties-common

#Add the GPG key for the official Docker repository to your system:
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

#Add the Docker repository to APT sources:
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"

#update the package database with the Docker packages from the newly added repo
sudo apt update

#Make sure you are about to install from the Docker repo instead of the default Ubuntu repo
#apt-cache policy docker-ce

# install Docker
sudo apt install docker-ce

# RUn Shadowsocks
docker run --name shadowsocks-r -d --restart always -p 2992:8388/tcp -p 2992:8388/udp jpacg/shadowsocksr -s 0.0.0.0 -p 8388 -k vpn -m none -o tls1.2_ticket_auth -O auth_chain_a

 # the password is VPN.
 # port is 2992
```

Building an Image

```
FROM        mhart/alpine-node
MAINTAINER  Keesee

# ENV         NODE_ENV 'production'
ENV         WORKING_DIR=/app/home
ENV         port=3333
RUN         mkdir -p $WORKING_DIR
WORKDIR     $WORKING_DIR

COPY        package.json package.json
RUN         yarn install --registry https://registry.npm.taobao.org
# RUN         yarn install --registry https://registry.npm.taobao.org

COPY        src/ src/
COPY        public/ public/
COPY        config/ config/

CMD         [ "yarn", "start" ]
EXPOSE      3333
```

Root Image

Directory on the Container

Port Enviromental Variable

Which command on container where your code is

Tell docker

Package management

Install packages

Copy your code

Copy your assets

Copy your configurations

Start Command

Expose your app

What shall we build?