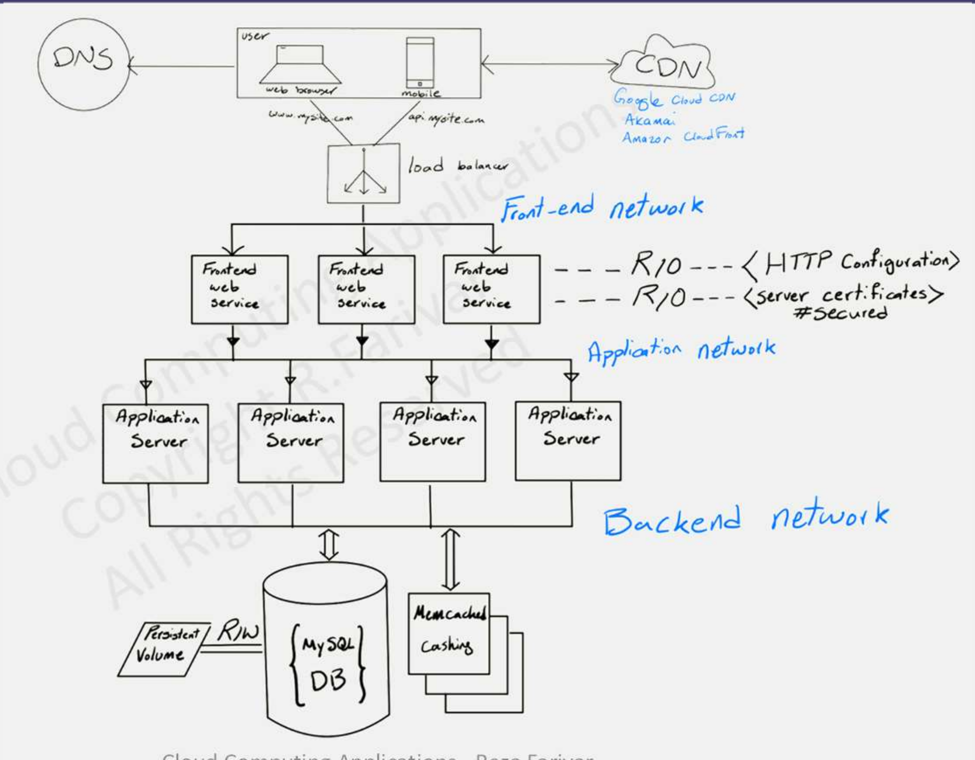
CLOUD COMPUTING APPLICATIONS

Example Architecture in Docker Swarm Stack

Prof. Reza Farivar

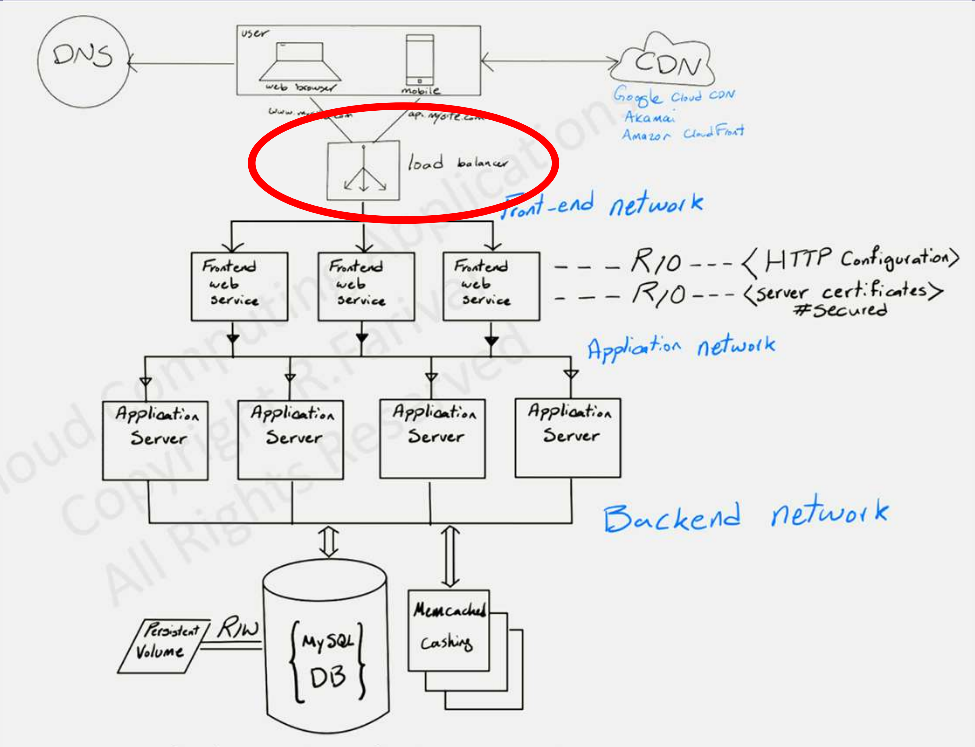
Example Architecture

- Classic 3-tier architecture



Example Architecture

- Classic 3-tier architecture



Example HAProxy Config

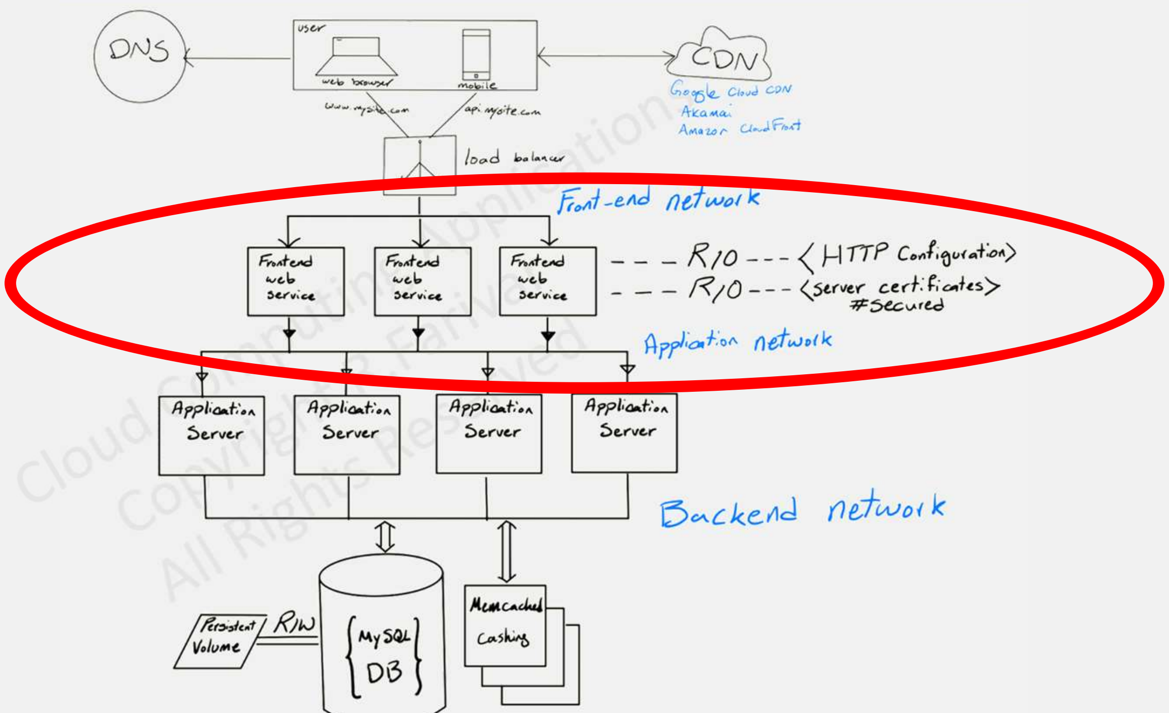
```
global
        log /dev/log    local0
        log /dev/log    local1 notice
...snip...

# Configure HAProxy to listen on port 80
frontend http_front
   bind *:80
   stats uri /haproxy?stats
   default_backend http_back

# Configure HAProxy to route requests to swarm nodes on port 8080
backend http_back
   balance roundrobin
   server node1 192.168.99.100:8080 check
   server node2 192.168.99.101:8080 check
   server node3 192.168.99.102:8080 check
```

Example Architecture

- Classic 3-tier architecture



Frontend web-app service

```
version: "3.9”
services:
    frontend:
        image: web-app
        deploy:
                replicas: 3
        ports:
                - "443:8043"
        networks:
                - front-tier
                - application-tier
        configs:
                - httpd-config
        secrets:
                - server-certificate
```

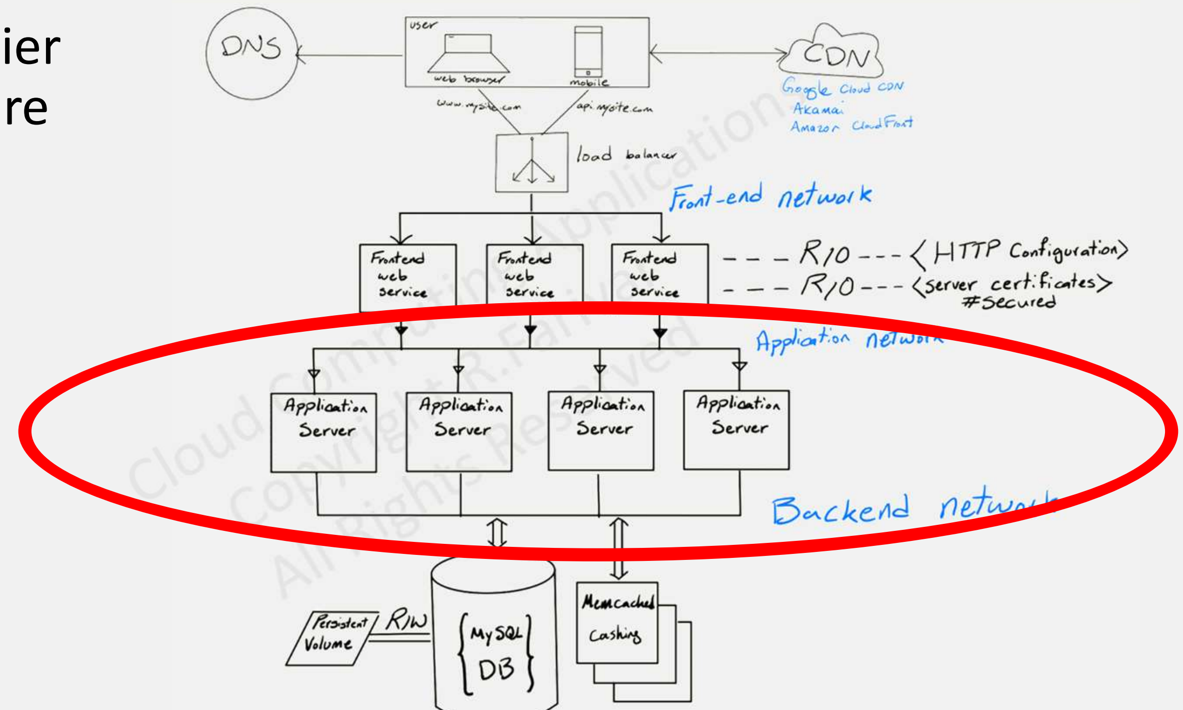
- *Deploy Specifies configuration related to the deployment and running of services*
- *Ignored by docker-compose*

- *Endpoint_mode: vip (default)*
- *Port Publishing*
- *Routing Mesh*
    - *Ingress Network*
- *Port 443 on **ALL** Swarm nodes is load-balanced into the 3 replicas*
    - *Swarm port*

https://github.com/compose-spec/compose-spec/blob/master/spec.md

Example Architecture

- Classic 3-tier architecture



Application Service

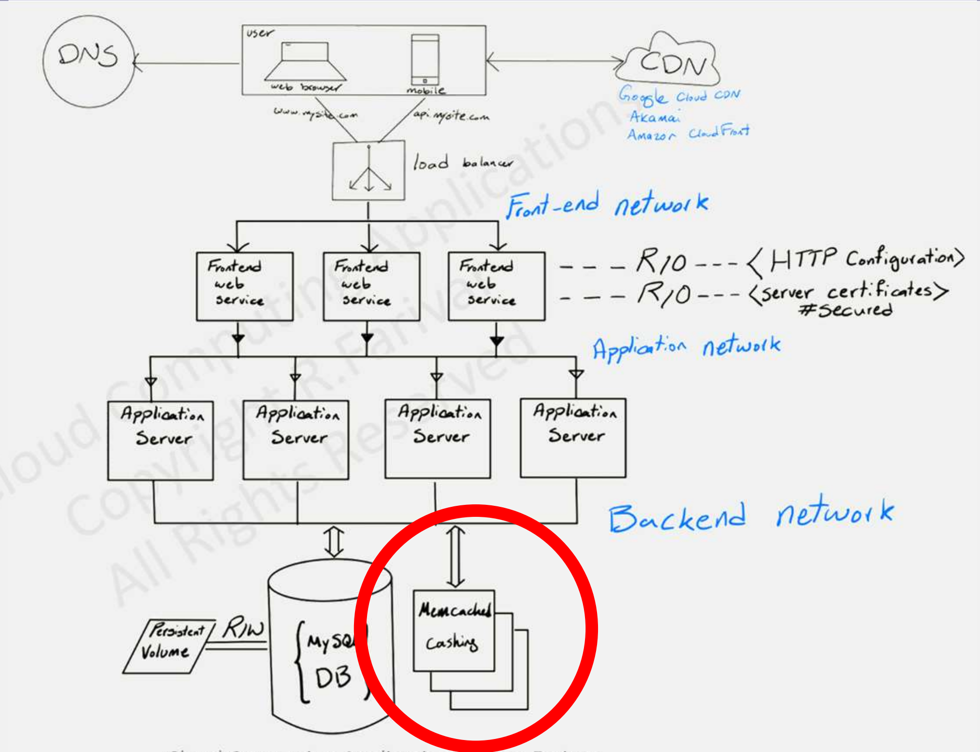
```
application:
        build:
                context: ./dir  #path to the build context
                dockerfile: Dockerfile-alternate
                 args:
                        - buildno: 1
    image: application-logic
    deploy:
            mode: replicated
            replicas: 4
    networks:
            - application-tier
            - back-tier
```

Build is only used in docker-compose
Ignored in Swarm stacks

- *Default mode is replicated*
- *Global: exactly one container per swarm node*

Example Architecture

- Classic 3-tier architecture

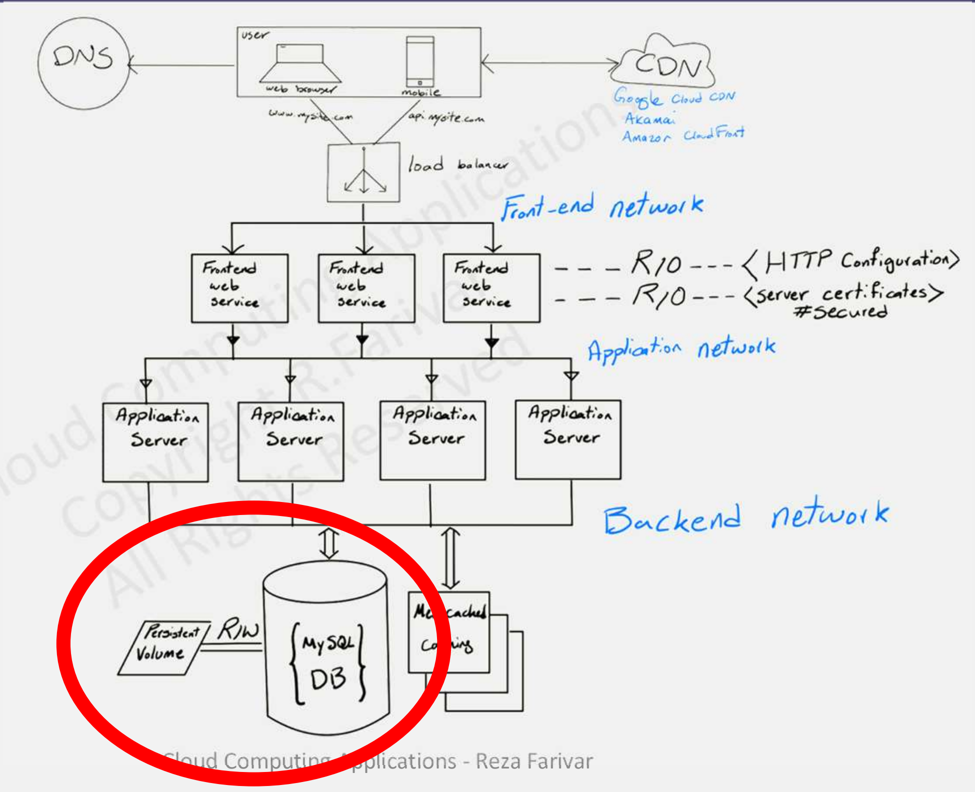


Cache Service

```
Cache:
        image: Memcached:1.6
        deploy:
                replicas: 3
                placement:
                        max_replicas_per_node: 1
                resources:
                        limits:
                                cpus: ‘1.50’
                                memory: 8G
                        reservations:
                                cpus: '0.25’
                                memory: 500M
        networks:
                - back-tier
```

Example Architecture

- Classic 3-tier architecture

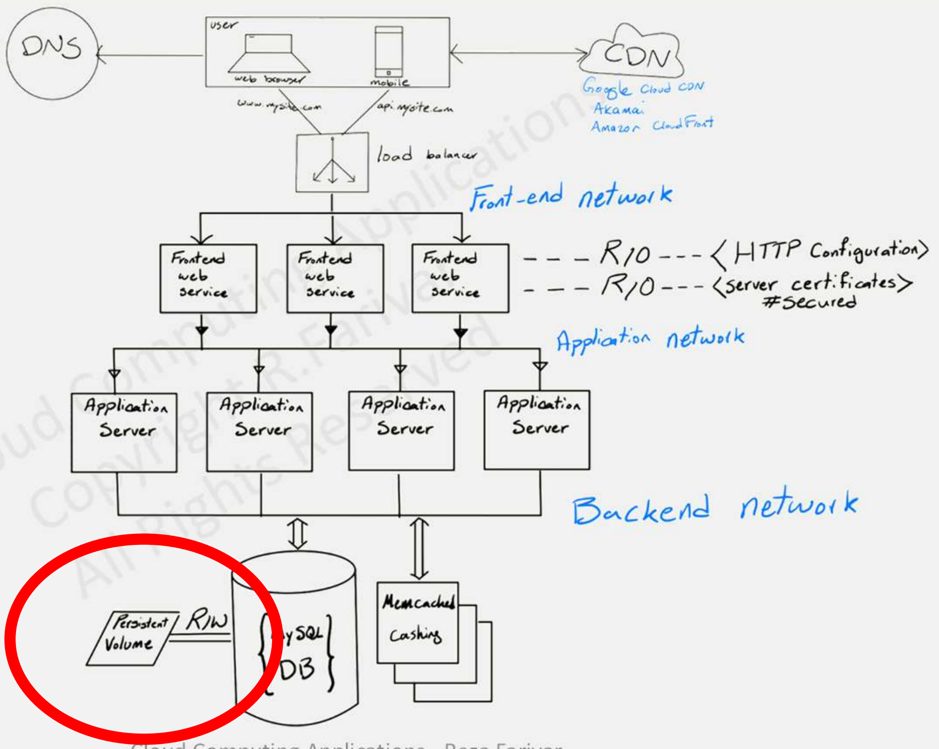


Backend Database Service

```
backend:
        image: example-registry.com:4000/mysql:8.0
        restart: on_failure
        environment:
                #MYSQL_ROOT_PASSWORD: example
                #.env
                MYSQL_ROOT_PASSWORD_FILE=/run/secrets/mysql-root
        volumes:
                - db-data:/etc/data
        networks:
                - back-tier
        Secrets:
                - mysql-password
```

Example Architecture

- Classic 3-tier architecture

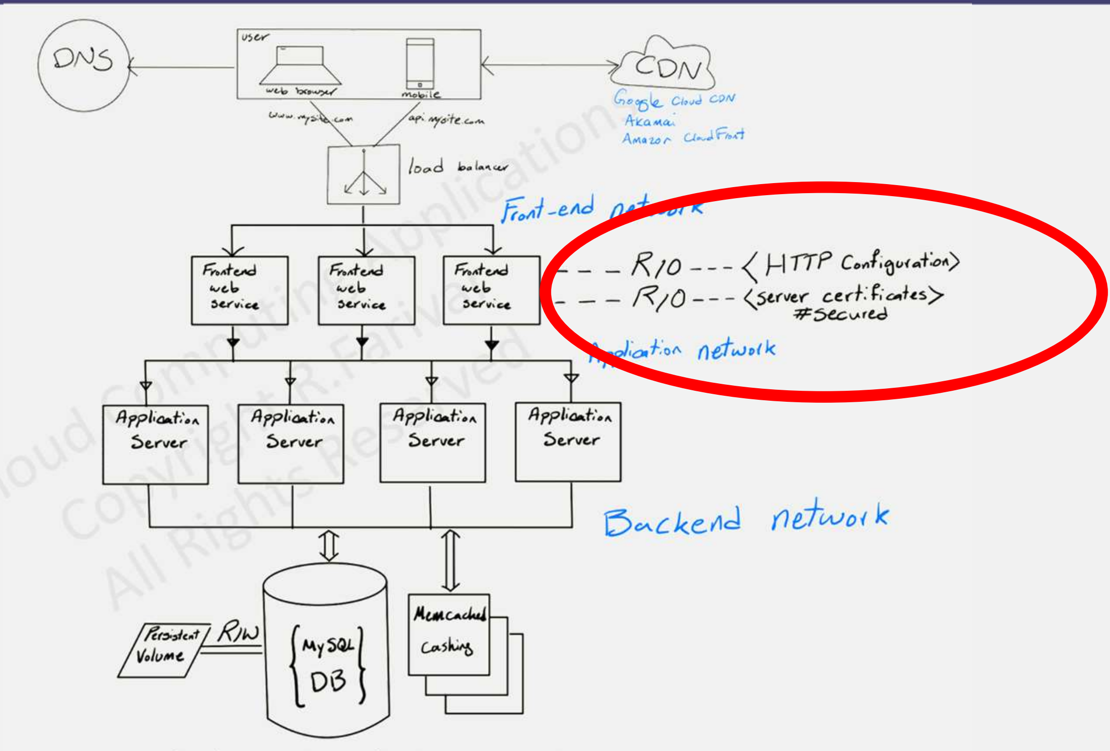


Volumes

```
volumes:
    db-data:
        driver: flocker
        driver_opts:
            size: "10GiB"
```

Example Architecture

- Classic 3-tier architecture



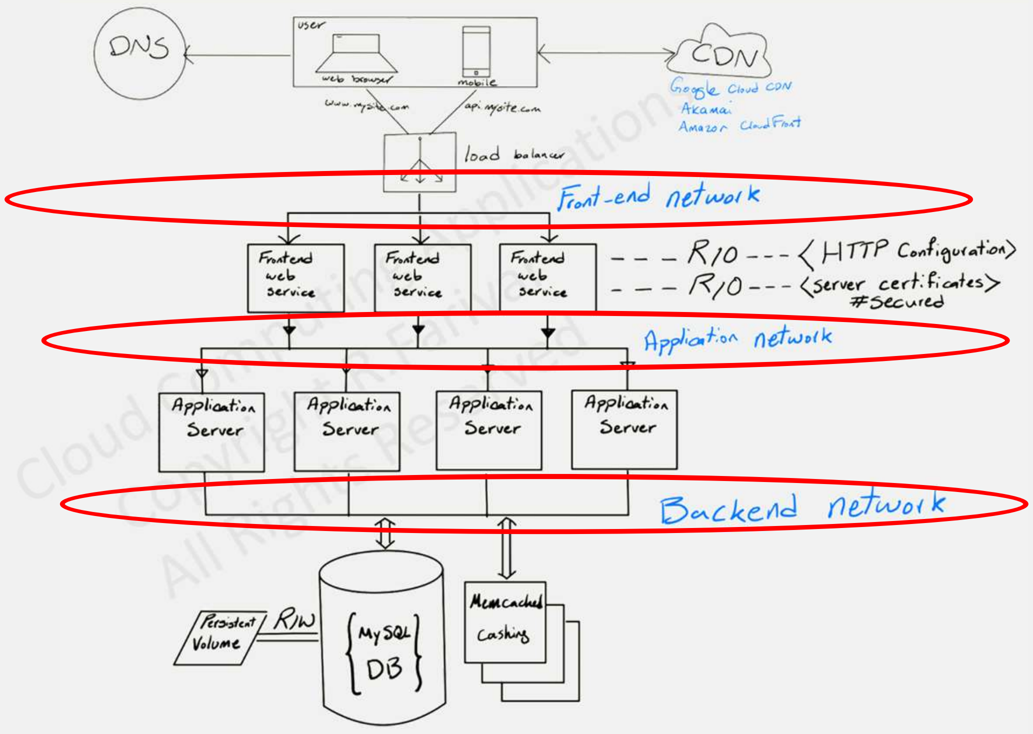
Configs and Secrets

```
configs:
        httpd-config:
                external: true
secrets:
        server-certificate:
                external: true
        mysql-password:
                file: ./my_secret_password.txt
```

it has already been defined in Docker, either by running the `docker secret create` *command or by another stack deployment*

Example Architecture

- Classic 3-tier architecture



Networks

```
networks:
        # The presence of these objects is sufficient to define
them
        front-tier:
        application-tier:
                driver: overlay
                ipam:
                        driver: default
                        config:
                                - subnet: 172.28.0.0/16
                                  ip_range: 172.28.5.0/24
                                  gateway: 172.28.5.254
        back-tier:
```