상위 **1%** 월급쟁이를 위한

쿠버네티스 어나더 클래스

지상편

Application 기능을 이해하기 - Pod (probe)

쿠버네티스 첫 오브젝트 잘 끼우기

Object 그려보 이해하기

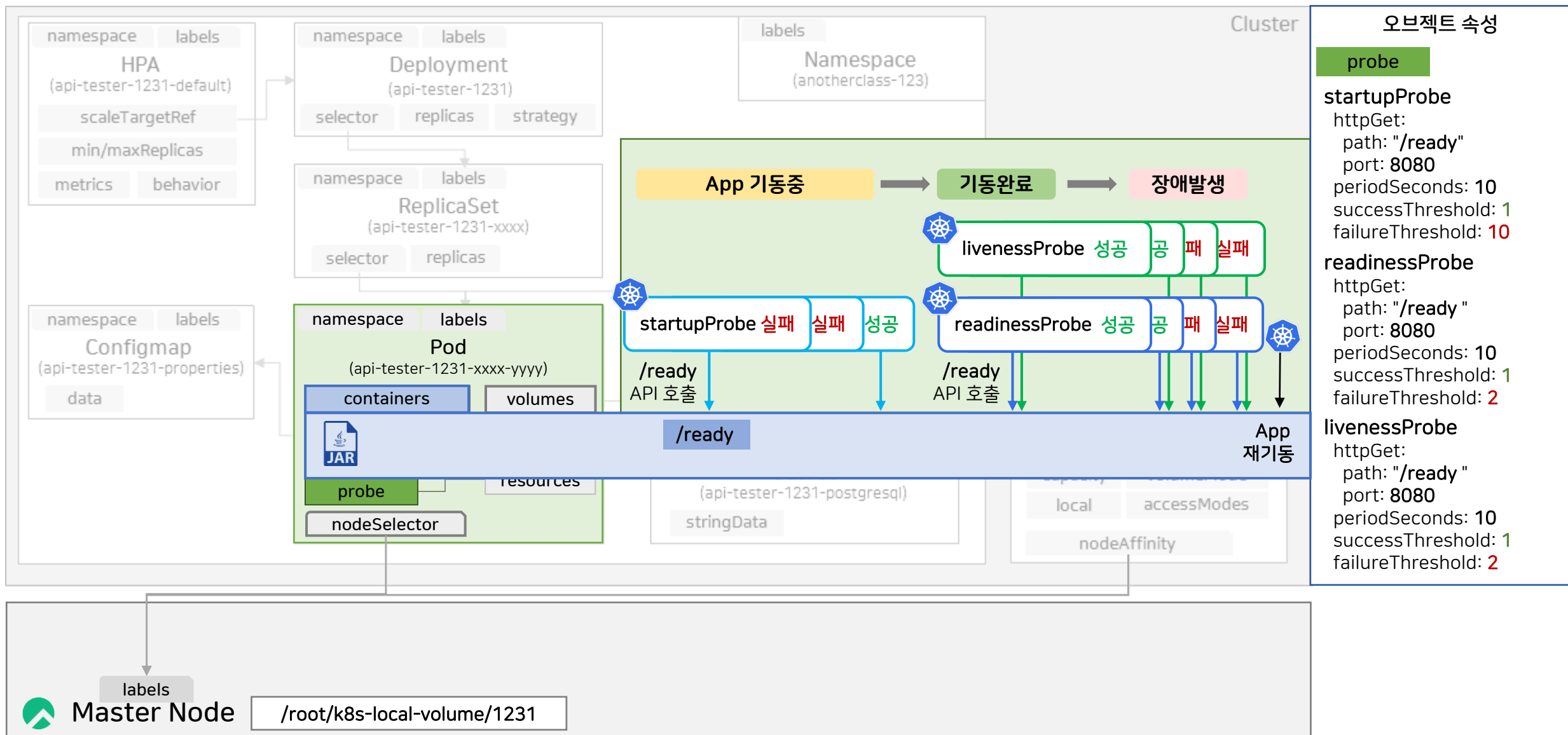Application 기능으로 이해하기

└ **Pod** (probe)

├ **Configmap, Secret**

├ **PV, PVC**

└ **Deployment** (strategy)

Component 동작으로 이해하기

Pod (probe) – 프로브 기본 개념

Cluster

HPA
(api-tester-1231-default)
- scaleTargetRef
- min/maxReplicas
- metrics / behavior

Deployment
(api-tester-1231)
- namespace / labels
- selector / replicas / strategy

Namespace
(anotherclass-123)

ReplicaSet
(api-tester-1231-xxxx)
- namespace / labels
- selector / replicas

Configmap
(api-tester-1231-properties)
- namespace / labels
- data

Pod
(api-tester-1231-xxxx-yyyy)
- namespace / labels
- containers / volumes
- JAR
- probe / resources
- nodeSelector

App 기동중 → **기동완료** → **장애발생**

livenessProbe 성공 | 공 | 패 | 실패

startupProbe 실패 | 실패 | 성공

readinessProbe 성공 | 공 | 패 | 실패

/ready
API 호출

/ready
API 호출

/ready

App 재기동

(api-tester-1231-postgresql)
stringData

local / accessModes

nodeAffinity

오브젝트 속성

probe

startupProbe
  httpGet:
    path: "/ready"
    port: 8080
  periodSeconds: 10
  successThreshold: 1
  failureThreshold: 10

readinessProbe
  httpGet:
    path: "/ready "
    port: 8080
  periodSeconds: 10
  successThreshold: 1
  failureThreshold: 2

livenessProbe
  httpGet:
    path: "/ready "
    port: 8080
  periodSeconds: 10
  successThreshold: 1
  failureThreshold: 2

Master Node
labels
/root/k8s-local-volume/1231

Pod (probe) - Application 로그를 통한 프로브 동작 분석

```
2023-07-28T22:48:48.: Starting service [Tomcat]
2023-07-28T22:48:48.: Starting Servlet engine: [Apache Tomcat/10.1.8]
2023-07-28T22:48:50.: Initializing Spring embedded WebApplicationContext
2023-07-28T22:48:50.: Root WebApplicationContext: initialization completed
2023-07-28T22:48:57.: Tomcat started on port(s): 8080 (http) with context p
2023-     : [System] App is initializing
2023-     : Initializing Spring DispatcherServlet 'dispatcherServlet'
2023-07-28T22:48:59.: Initializing Servlet 'dispatcherServlet'
2023-07-28T22:48:59.: Completed initialization in 0 ms
2023-07-28T22:49:00.: [Kubernetes] startupProbe is Failed-> [Sy
2023-07-28T22:49:02.: [System] Database is connecting
2023-07-28T22:49:04: [Kubernetes] startupProbe is Failed-> [System] isAppLive: false
2023-   49:07.: [System] Database is connected
2023-07-28T22:49:09: [Kubernetes] startupProbe is Failed-> [System] isAppLive: false
2023-07-28T22:49:12.: [System] App is starting
2023-07-28T22:49:14.: [Kubernetes] startupProbe is Failed-> [System] isAppLive: false
2023-07-28T2   : [System] App is started
2023-07-28T2   : [Kubernetes] startupProbe is Succeed-> [S
2023-07-28T22:49:19.: [Kubernetes] readinessProbe is Failed-> [System] isAppReady: false
202   : [System] ConfigMap data is loading..
202   : [Kubernetes] readinessProbe is Failed-> [
2023-07-28T22:49:24.: [Kubernetes] livenessProbe is Succeed-> [
2023-07-28T22:49:29.: [System] ConfigMap data is loading..
2023-07-28T22:49:34.: [System] ConfigMap data is loading..
2023-07-28T22:49:34.: [Kubernetes] readinessProbe is Failed-> [System] isAppReady: false
2023-07-28T22:49:34.: [Kubernetes] livenessProbe is Succeed-> [System] isAppLive: true
2023-07-28T2   : [System] Data loading is completed
2023-07-28T2   : Started AppApplication in 68.309 seconds (
2023-07-28T22:49:44: [Kubernetes] readinessProbe is Succeed->
2023-    49:44.: [Kubernetes] livenessProbe is Succeed-> [
2023-    49:54.: [Kubernetes] livenessProbe is Succeed-> [
2023-07-28T22:49:54: [Kubernetes] readinessProbe is Succeed-> [System] isAppReady: true
```

- App 초기화
- 5초
- 완료
- User 초기화
- 완료
- 10초

- startupProbe 실패
- startupProbe 실패
- startupProbe 실패
- startupProbe 실패
- startupProbe 성공
- readinessProbe 실패
- livenessProbe 성공
- readinessProbe 성공
- livenessProbe 성공

실제 App 상황

기동되기 전엔 API를
받지 못함
- WAS(tomcat)를
  사용할 경우
  access.log에 기록됨

**동작 확인을 위해
임의로 코드 구성**

오브젝트 속성

probe

startupProbe
 httpGet:
  path: "/startup "
  port: 8080
 periodSeconds: 5
 successThreshold: 1
 failureThreshold: 24

readinessProbe
 httpGet:
  path: "/readiness "
  port: 8080
 periodSeconds: 10
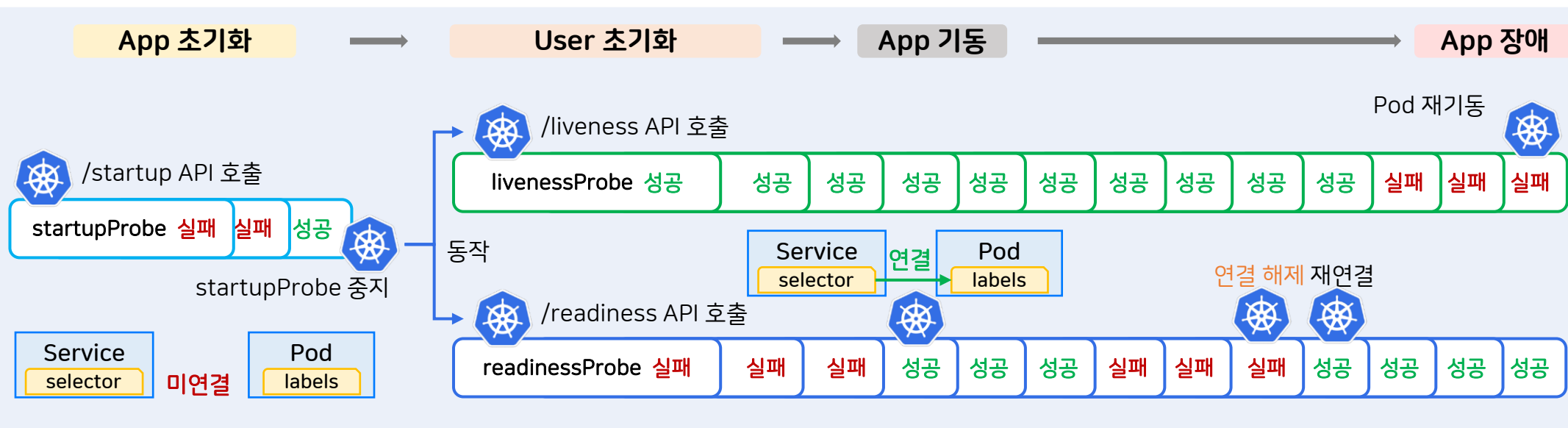 successThreshold: 1
 failureThreshold: 3

livenessProbe
 httpGet:
  path: "/liveness "
  port: 8080
 periodSeconds: 10
 successThreshold: 1
 failureThreshold: 3

⎈ Pod (probe) - Application 동작 중심의 프로브 이해

App 초기화 → User 초기화 → App 기동 → App 장애

Kubernetes 제공 기능

Pod 재기동

/liveness API 호출

| livenessProbe 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 실패 | 실패 | 실패 |

/startup API 호출

| startupProbe 실패 | 실패 | 성공 |

startupProbe 중지

동작

| Service selector | 연결 | Pod labels |

연결 해제 재연결

| Service selector | 미연결 | Pod labels |

/readiness API 호출

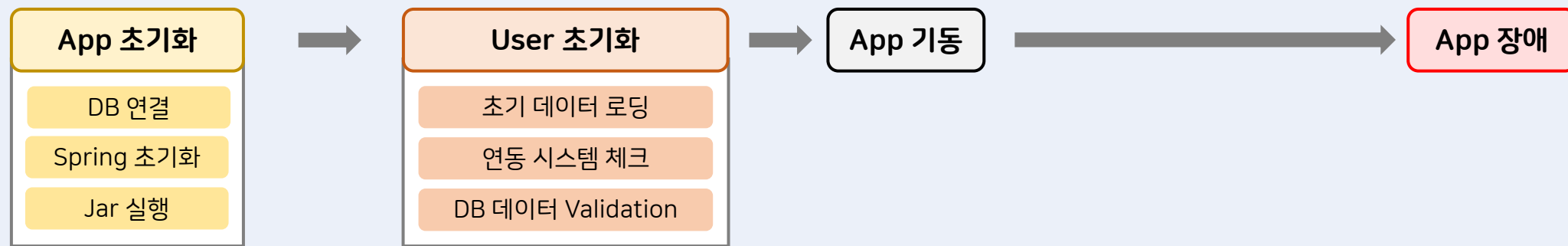| readinessProbe 실패 | 실패 | 실패 | 성공 | 성공 | 성공 | 실패 | 실패 | 실패 | 성공 | 성공 | 성공 | 성공 |

자동화 요구사항

[API를 받을 수 없는 상태]
- App 상태 체크 (초기화가 끝났는지?)
- 외부 API 접근 금지

[API를 받을 수 있는 상태]
- App 상태 체크 (App이 살아있는지?)
- 외부 API 접근 금지

- App 상태 체크 (App이 살아있는지?)
- 외부 API 접근 허용

- App 재기동

Application 동작

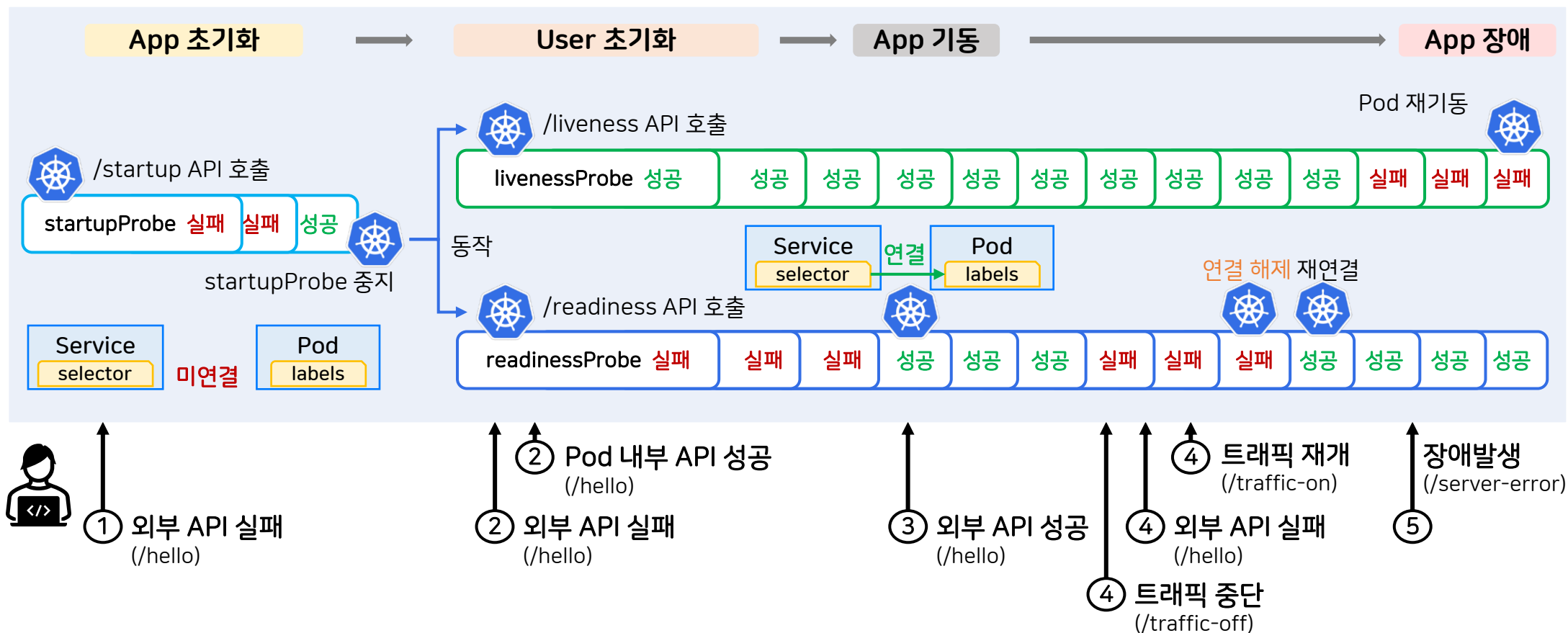App 초기화 → User 초기화 → App 기동 → App 장애

App 초기화
- DB 연결
- Spring 초기화
- Jar 실행

User 초기화
- 초기 데이터 로딩
- 연동 시스템 체크
- DB 데이터 Validation

Pod (probe) - API 날려보며 프로브 동작 확인하기

App 초기화 → User 초기화 → App 기동 → App 장애

Kubernetes 제공 기능

/startup API 호출

| startupProbe | 실패 | 실패 | 성공 |

startupProbe 중지

/liveness API 호출

Pod 재기동

| livenessProbe | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 성공 | 실패 | 실패 | 실패 |

동작

| Service | | Pod |
| selector | 연결 | labels |

/readiness API 호출

연결 해제 재연결

| readinessProbe | 실패 | 실패 | 실패 | 성공 | 성공 | 성공 | 실패 | 실패 | 실패 | 성공 | 성공 | 성공 | 성공 |

| Service | | Pod |
| selector | 미연결 | labels |

① 외부 API 실패
(/hello)

② Pod 내부 API 성공
(/hello)

② 외부 API 실패
(/hello)

③ 외부 API 성공
(/hello)

④ 트래픽 중단
(/traffic-off)

④ 외부 API 실패
(/hello)

④ 트래픽 재개
(/traffic-on)

⑤ 장애발생
(/server-error)

Pod (probe) – 일시적 장애 상황에서의 프로브 활용

App 초기화 ➡ User 초기화 ➡ App 기동 ➡ App 장애

Kubernetes 제공 기능

Pod 재기동

/startup API 호출

/liveness API 호출

| livenessProbe 성공 | 성공 | 성공 | 성공 | 성공 | 성공 |

3. 실패하는데 걸리는 시간을 길게 설정해서 Pod가 쉽게 재기동 되는걸 방지

| 실패 | 실패 | 성공 |

startupProbe 실패 | 실패 | 성공

startupProbe 중지

동작

Service selector — 연결 — labels

/readiness API 호출

| readinessProbe 실패 | 실패 | 실패 | 성공 | 성공 | 성공 | 성공 |

2. 실패시 외부 API 접근을 금지시키기 때문에 App 부담 감소됨

readinessProbe 실패 실패 실패 실패 성공

Service selector | 미연결 | Pod labels
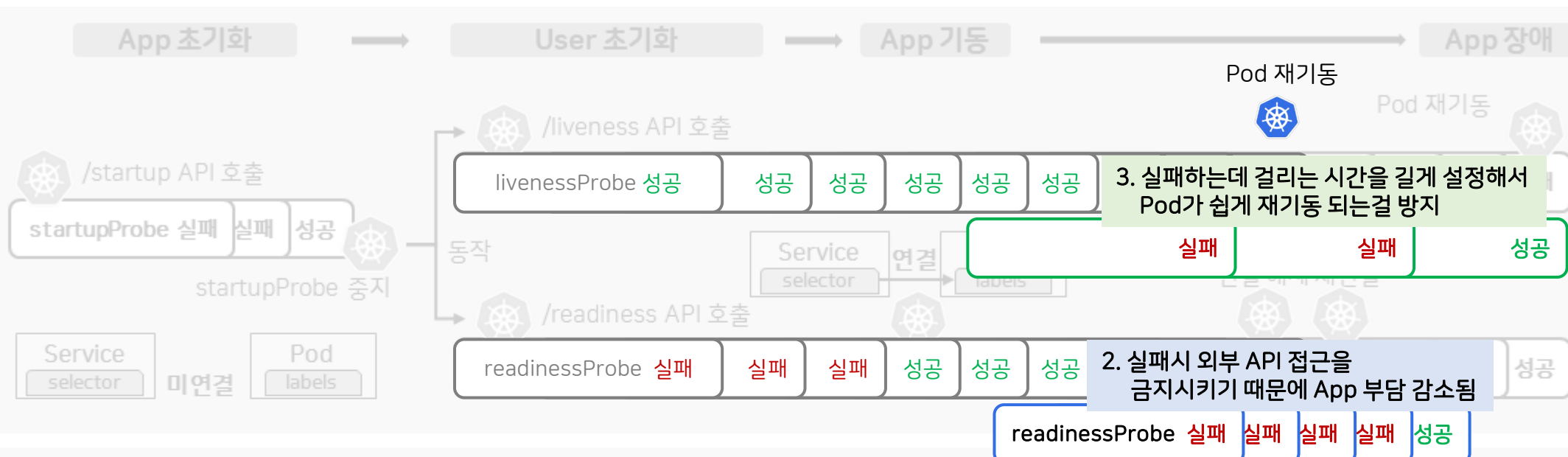
자동화 요구사항

- App 상태 체크 (초기화가 끝났는지?)
- 외부 API 접근 금지

- App 상태 체크 (App이 살아있는지?)
- 외부 API 접근 금지

- App 상태 체크 (App이 살아있는지?)
- 외부 API 접근 허용

- App 재기동

1. Probe가 없었으면 정상으로 돌아왔을 수도 있음

Application 동작

App 초기화 ➡ User 초기화 ➡ **App 기동** ➡ **일시적 장애** ➡ **App 정상**

- DB 연결
- Spring 초기화
- Jar 실행

- 초기 데이터 로딩
- 연동 App상태 체크
- DB 데이터 Validation

일시적 장애:
- Overload
- Leak
- Thread Full