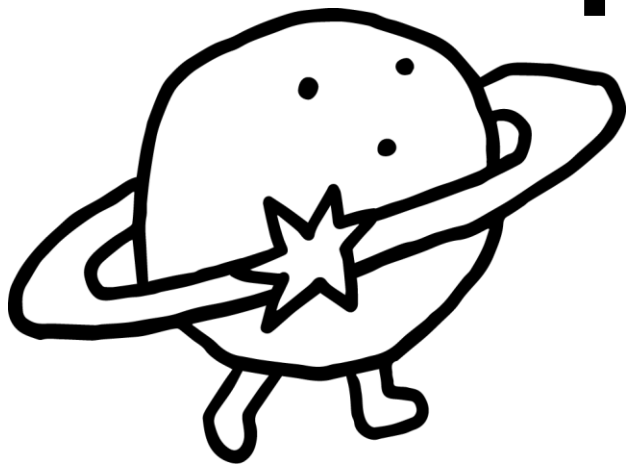
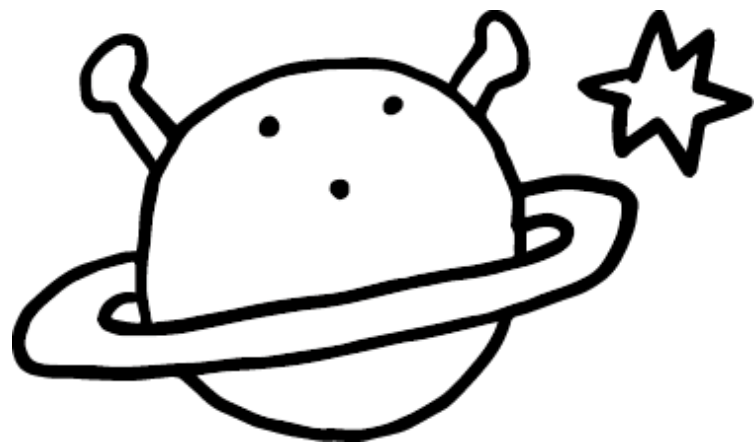


k6 부하테스트와 튜닝을 통한 서버 성능 개선

BE 7기 메이





주제 선정 이유



주제 선정 이유

📅 6차 요구사항

1. 목표 TPS 설정 및 서버 설정 최적화

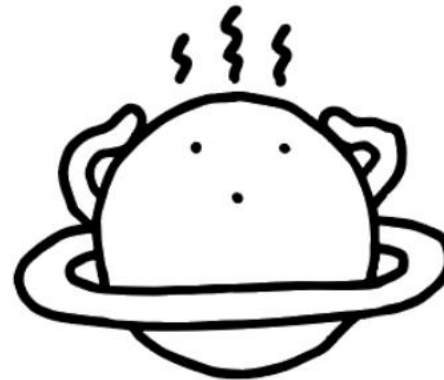
- 먼저 우리 서비스가 한 대의 서버로 얼마나 감당할 수 있는지 알아야 합니다
- 핵심 기능(로그인, 조회, 작성 등)별로 목표 TPS를 정하세요
- 그리고 그에 맞게 설정을 조정하세요:

톰캣 설정 튜닝:

- `threads.max` : 동시에 처리할 수 있는 요청 수
- `max-connections` : 최대 연결 수
- `accept-count` : 대기열 크기
- 각 값을 왜 그렇게 설정했는지 근거를 제시하세요

HikariCP 커넥션 풀 설정:

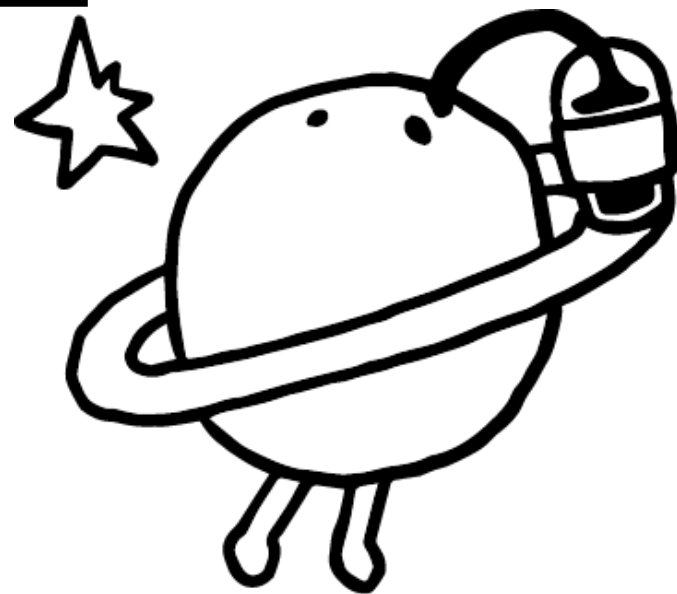
- 적절한 풀 사이즈는? (너무 적으면 병목, 너무 많으면 DB 부하)
- `maximumPoolSize` , `minimumIdle` , `connectionTimeout` 등
- DB 최대 연결 수와 애플리케이션 서버 대수를 고려하세요



BE 6차 요구사항

개인적인 아쉬움

부하테스트 목적, 시나리오





부하테스트 시나리오



우리 서비스에서 가장 핵심이 되는 뷰인 “콘텐츠 상세 페이지 조회”

`/contents/{contentId}/places` (콘텐츠 상세 조회 api)
`/contents/{contentId}` (콘텐츠, 찜 여부 조회 api)

```
export default function () {  
  const deviceFid = deviceFids[Math.floor(Math.random() * deviceFids.length)];  
  const headers = {  
    "device-fid": deviceFid,  
    "Content-Type": "application/json",  
  };  
  
  const response1 = http.get(`${BASE_URL}/contents/1/places`, { headers });  
  
  check(response1, {  
    "popular favorites 200": (r) => r.status === 200,  
  });  
  
  const response2 = http.get(`${BASE_URL}/contents/1`, { headers });  
  
  check(response2, {  
    "region categories (Korea) 200": (r) => r.status === 200,  
  });  
  
  // 2초 후 다시 수행  
  sleep(Math.random() * 1 + 2);  
}
```



부하테스트 목표 설정

✓ 목표 기준

목표 유저수 설정

- 프리코스, 인스타그램 등을 통해 홍보를 하고, 튜립이 기대하는 유저 수는 약 **3,000명**이다.
- 여행 도메인 특성 상 일상적으로 사용하는 서비스가 아니기 때문에, 활성 계수를 10%로 설정하여, 일일 활성 사용자(DAU)를 **300명**으로 기대한다.

목표 처리량 (TPS) 설정

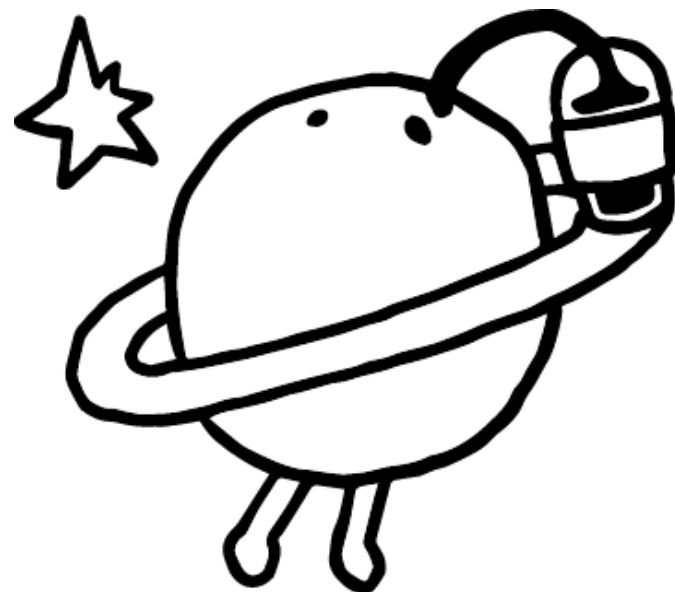
사용자 1명이 평균적으로 콘텐츠 조회 기능을 3초에 한 번씩 요청한다고 가정하면, 300명의 DAU를 기준으로 서버가 처리해야 할 요청량은 **100TPS**이다.

$$300\text{명} \div 3\text{초} = 100\text{TPS}$$

목표 시간 설정

빈번히 발생하는 기능에서 응답 시간이 1초를 넘어간다면 사용자 이탈률이 높아질 것이라고 판단하여, 목표 응답 시간은 **1초** 이하로 설정한다.

1차 부하테스트



부하테스트 서버



운영 서버와 동일한
스펙의 개발 서버

더미데이터

content: 3,000개
place: 51,000개
member: 3,000개
place_category: 15,000개
favorite_content: 60,000개
favorite_place: 225,000개

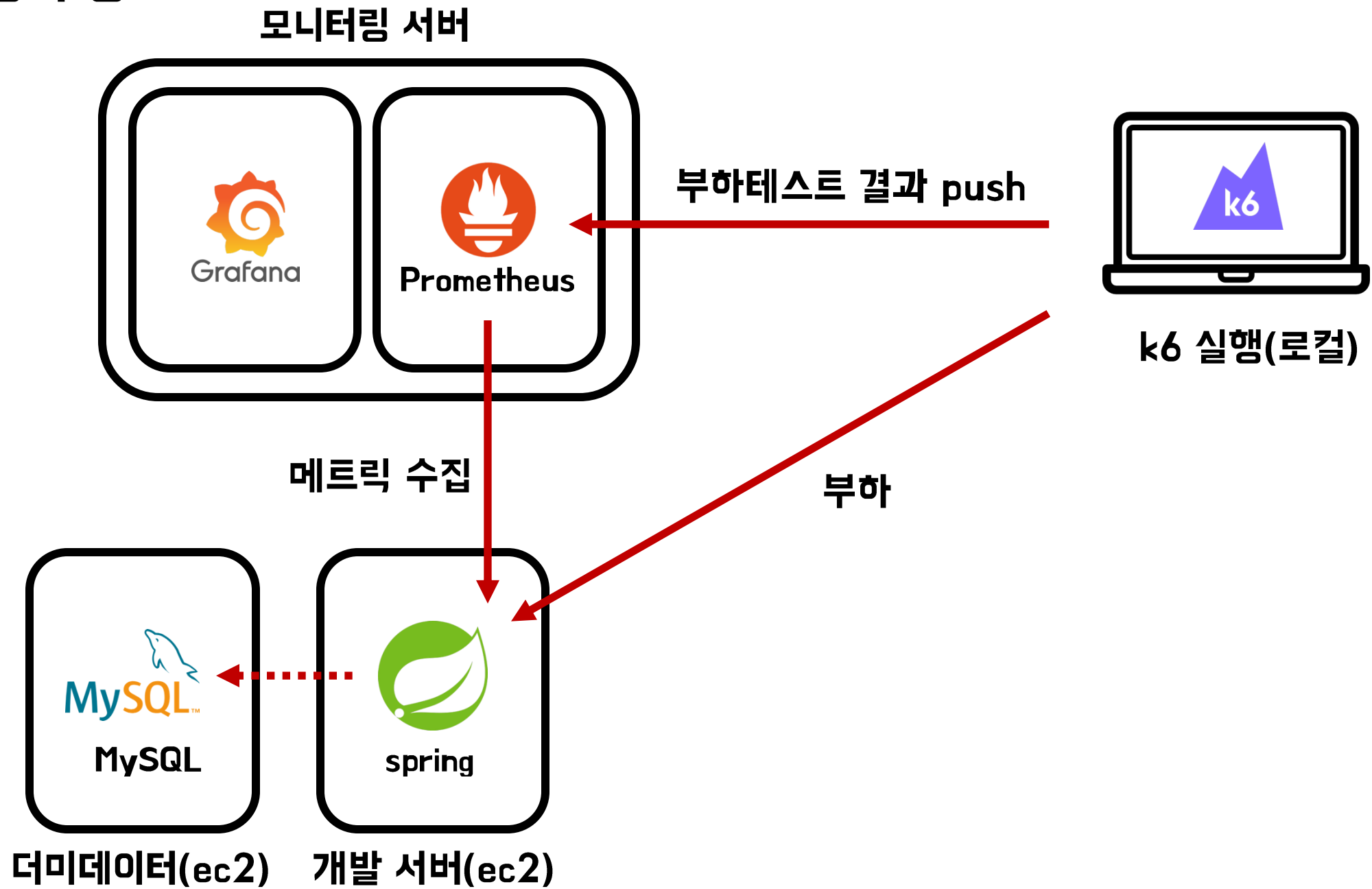
...

3,000명 사용자 기준
더미데이터

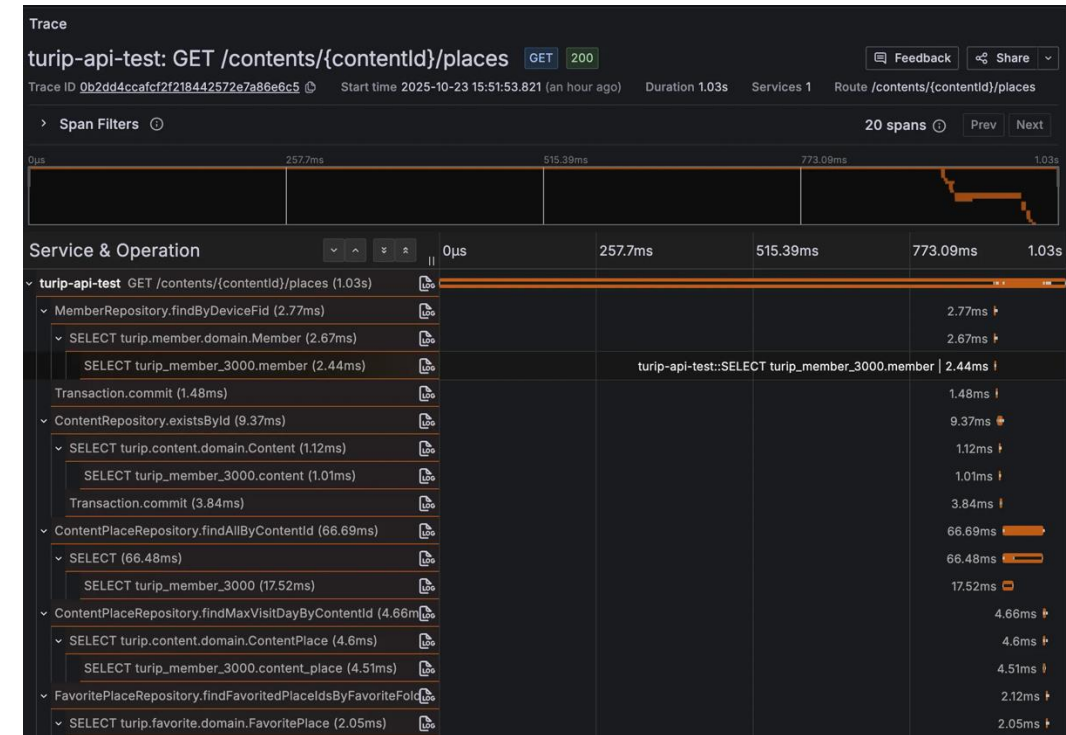
k6 실행 환경



로컬 컴퓨터

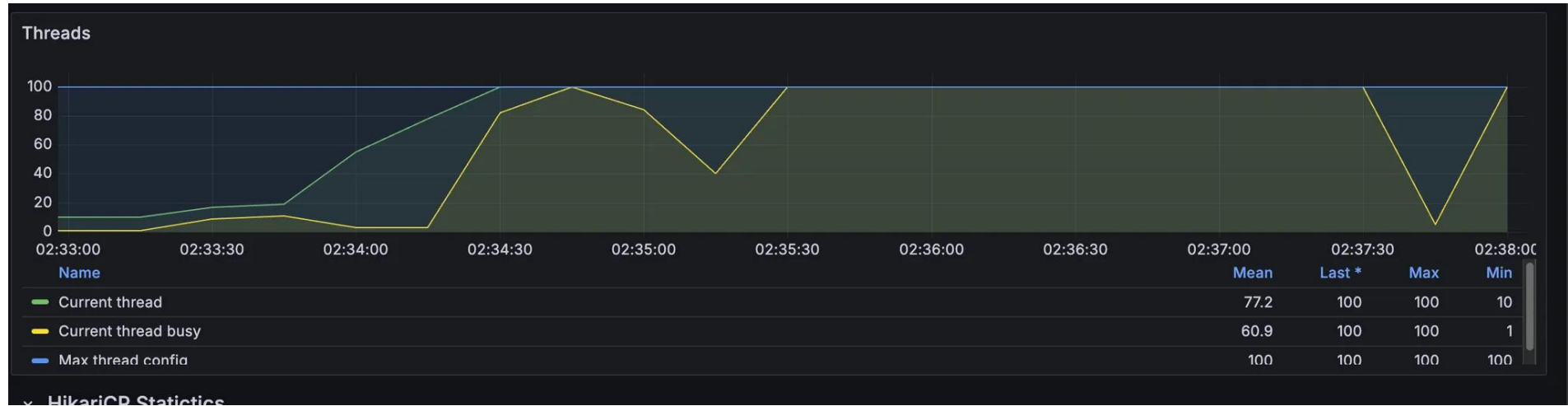


부하테스트 진행



그라파나에서 tps, 응답시간, 톰캣 스레드 현황, Hikari 커넥션 현황,
cpu 사용량, 트레이스 ... 모니터링

부하테스트 진행



톰캣 스레드



Hikari 커넥션

부하테스트 진행

▶ 테스트 32) `server.tomcat.threads.max = 100` → RPS 105

▶ 테스트 33) 기본 세팅(`server.tomcat.threads.max = 200`) → RPS 102

▶ 테스트 34) `server.tomcat.threads.max = 400` → RPS 98

스레드 100으로 고정하고, HikariCP 커넥션 수 조절해보기

▶ 테스트 35) `server.tomcat.threads.max = 100`, HikariCP `maximumPoolSize = 5` → RPS 81

▶ 테스트 36) `server.tomcat.threads.max = 100`, HikariCP `maximumPoolSize = 20` → RPS 108

▶ 테스트 37) `server.tomcat.threads.max = 100`, HikariCP `maximumPoolSize = 30` → RPS 98

▶ 테스트 38) `server.tomcat.threads.max = 100`, HikariCP `maximumPoolSize = 50` → RPS 19

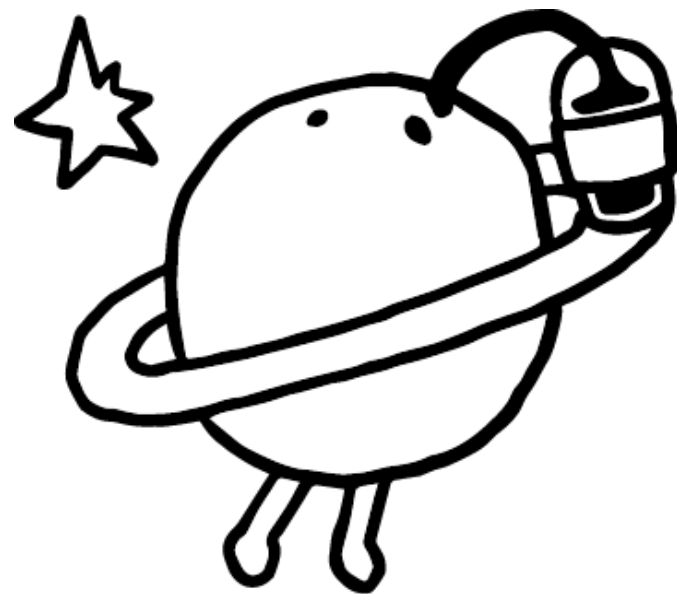
▶ 테스트 39) `server.tomcat.threads.max = 100`, HikariCP `maximumPoolSize = 100` → RPS 64

▶ 테스트 40) `server.tomcat.threads.max = 30`, HikariCP `maximumPoolSize = 30` → RPS 83

▶ 테스트 41) `server.tomcat.threads.max = 10`, HikariCP `maximumPoolSize = 30` → RPS 100


예상하지 못한 결과, 일관되지 않은 결과


1차 부하테스트 - 문제








문제 1) 부하테스트 서버

 채넛 2025. 10. 20. 오전 2:10
java.lang.IllegalStateException: Invalid profile property value found in Environment under 'spring.profiles.active' 나온다 ..


 메이(김시원) 2025. 10. 20. 오전 2:10
????

 채넛 2025. 10. 20. 오전 2:11
profile이 어디에 정의되어 있는거야 ?


 메이(김시원) 2025. 10. 20. 오전 2:11
resources/application.yml파일들인데


 채넛 2025. 10. 20. 오전 2:12
흠.. 내일해야겠다 ㅜㅜ

 메이(김시원) 2025. 10. 20. 오전 1:53
스프링 실행은 잘 됐어??

 채넛 2025. 10. 20. 오전 1:53
응응

 메이(김시원) 2025. 10. 20. 오전 1:53
데이터가 안보이는건가?

 채넛 2025. 10. 20. 오전 1:54
응 데이터가 일부 안보여
아니다 전부 안보이는구나..

 jerry 2025. 10. 20. 오전 1:54
채넛 끝나고 나도 헬프 좀

개발서버를 이용하는 안드로이드 팀원의 불편함

문제 2) k6 환경 일관성



같은 세팅임에도 로컬 pc 상황에 따라 다른 결과

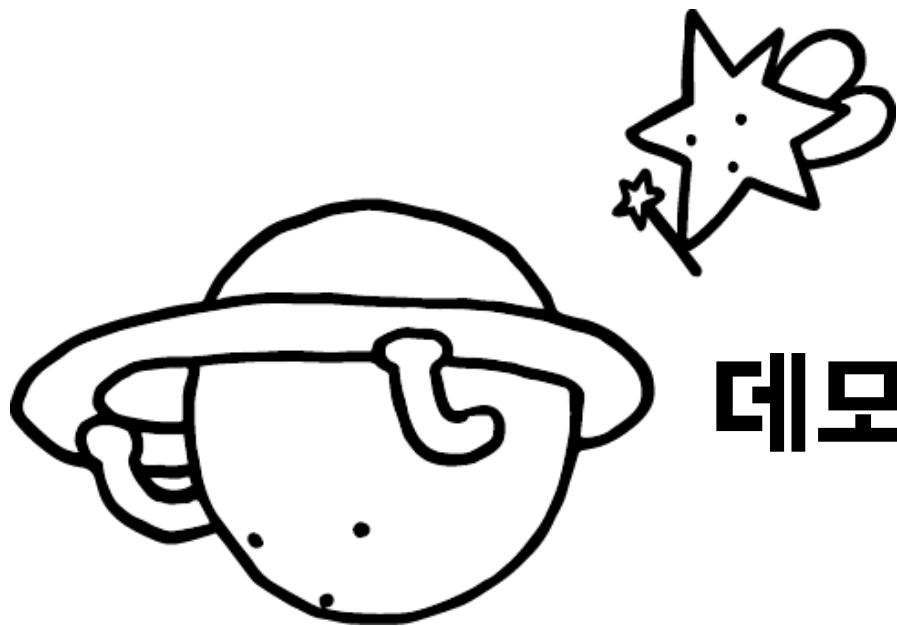


문제 3) 렌더링 시간을 고려하지 않은 목표

목표 시간 설정

빈번히 발생하는 기능에서 응답 시간이 1초를 넘어간다면 사용자 이탈률이 높아질 것이라고 판단하여, 목표 응답 시간은 1초 이하로 설정한다.

렌더링 시간까지 포함해서 1초 -> api 응답 시간 목표는 더 작아야 함



데모데이 피드백



부하테스트 관련 피드백

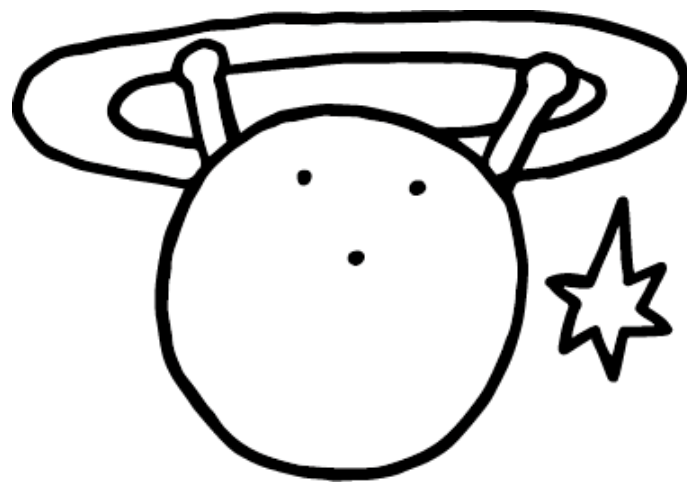
부하 테스트 환경이 개발환경이더라도, 개발환경에서 작업중인 동료 개발자가 영향을 받을 수 있으니 부하테스트 일정은 사전 공유 필수

테스트 결과 응답 실패율도 낮고, 빠른 응답시간을 보인다면: 더 많은 트래픽을 단계적으로 발생 시켜서 문제가 발생하기 시작하는 트래픽 수치까지 테스트해본다. (서버 1대의 최대 성능을 확인 해두는 차원에서)

목표 수치가 없으면, 테스트를 그만 해야하는 시점을 정하기 어려움. 성능 테스트 자체에 매몰되어 최대 퍼포먼스 지점을 찾아내기 위해 테스트를 반복하는 문제를 겪을 수 있음

1순위 목적은 서비스에서 서버가 안정적으로 처리할 수 있는 TPS를 대략 확인하는 것! 서버 1대의 최대 성능 지점을 연구하는 목적이 아님!

2차 부하테스트



부하테스트 서버



운영 서버와 같은 스펙의
부하테스트 전용 서버

k6 실행 환경



로컬 컴퓨터에서
ec2로 변경

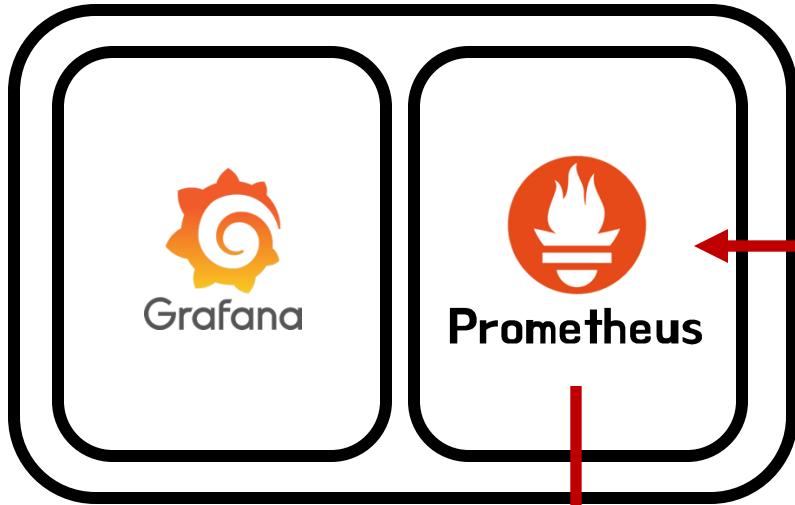
목표 재설정

목표 TPS: 100
목표 응답 시간: **500ms**

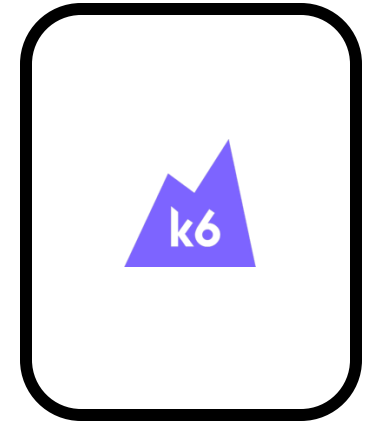


최고 성능을 위한 최적 세팅보다는
목표 달성에 집중하자

모니터링 서버



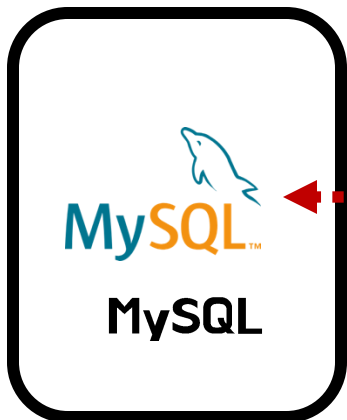
부하테스트 결과 push



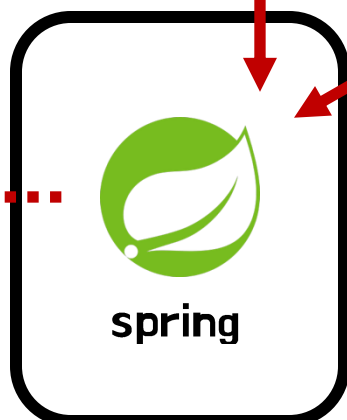
k6 서버(ec2)

메트릭 수집

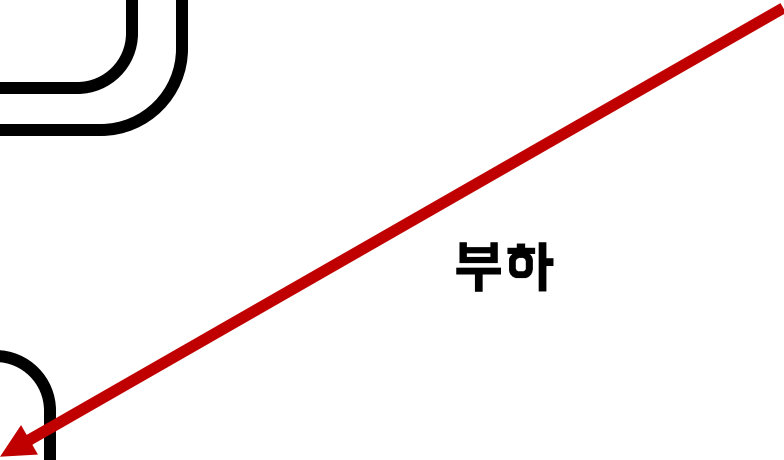
부하



더미데이터터(ec2)



부하테스트 서버(ec2)





부하테스트 진행

부하테스트

≡

↕

⚡

🔍

↗

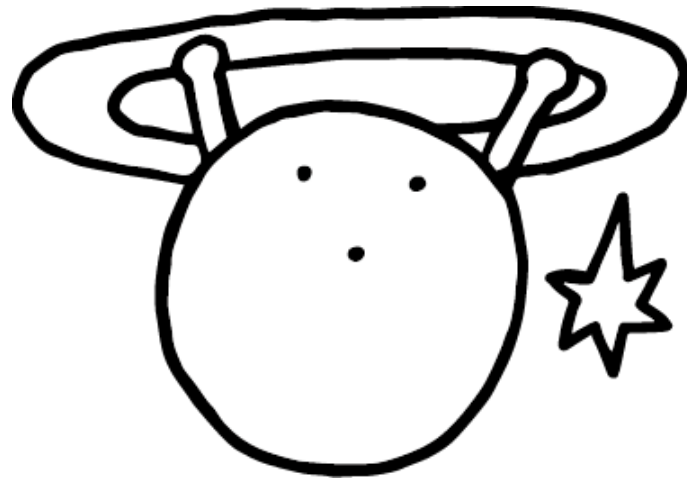
⚙

새로 만들기

▼

Aa 이름	≡ threads.max...	≡ maximumPoolSize...	≡ TPS	≡ p95 응답시간	≡ 비고
📄 템플릿					
📄 1차	200	10	98.6	844ms	vu=200
📄 2차	50	10	107	1100ms	
📄 3차	20	10	105	1290ms	
📄 4차	20	10	133	666ms	힙메모리 제한 늘리기
📄 5차	10	10	127	480ms	
📄 7차	5	5	114	721ms	최종 선택
📄 8차	200	10	132	949ms	

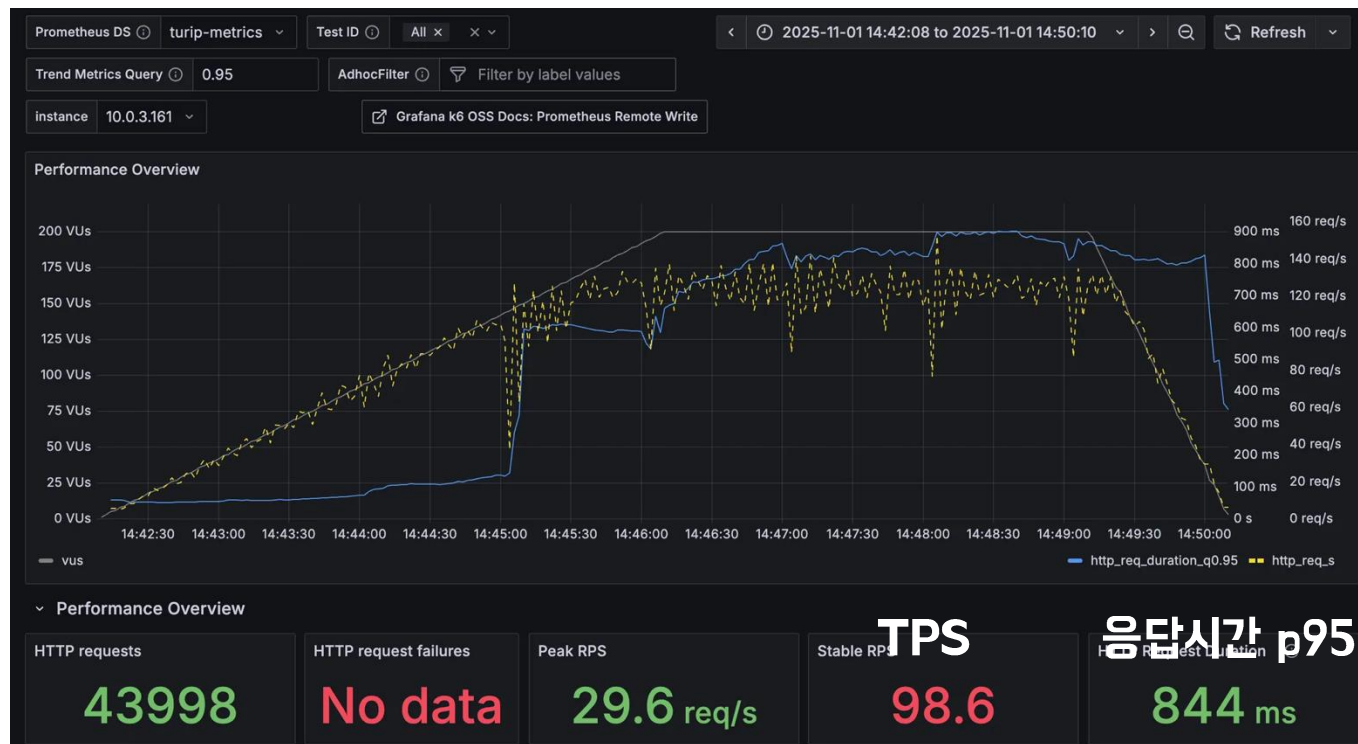
2차 부하테스트 과정 및 분석



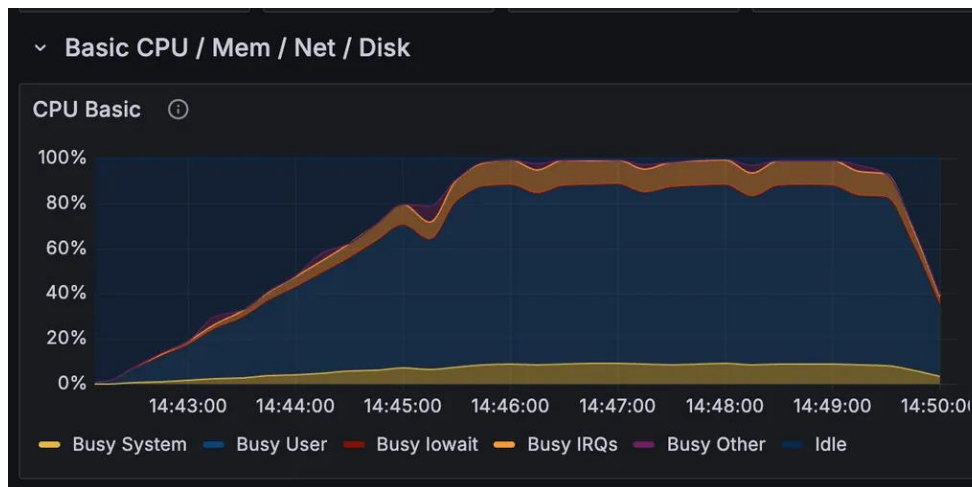


1차 테스트 – 기본 설정

TPS, 응답시간 목표 미달



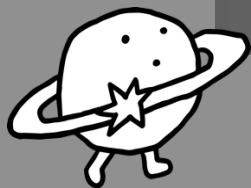
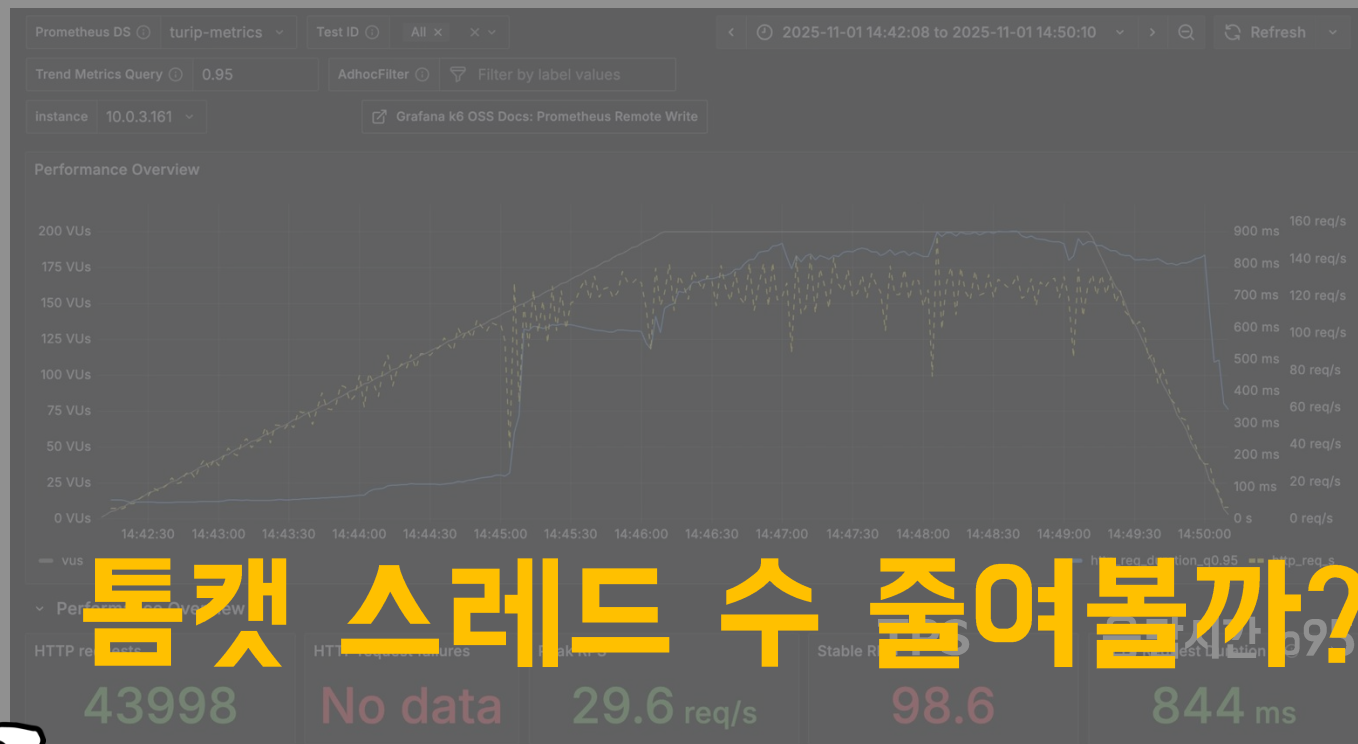
100%에 가까운 CPU 사용률
=> 먼저 개선 필요



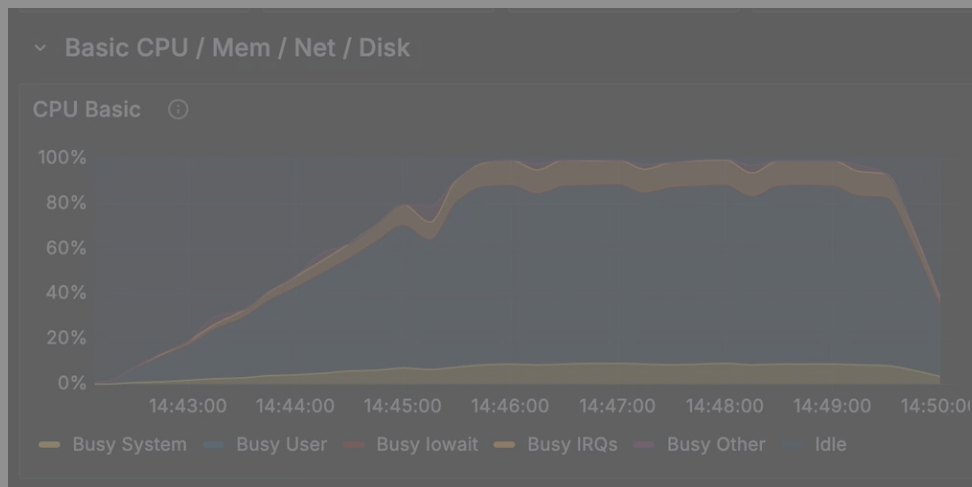


1차 테스트 – 기본 설정

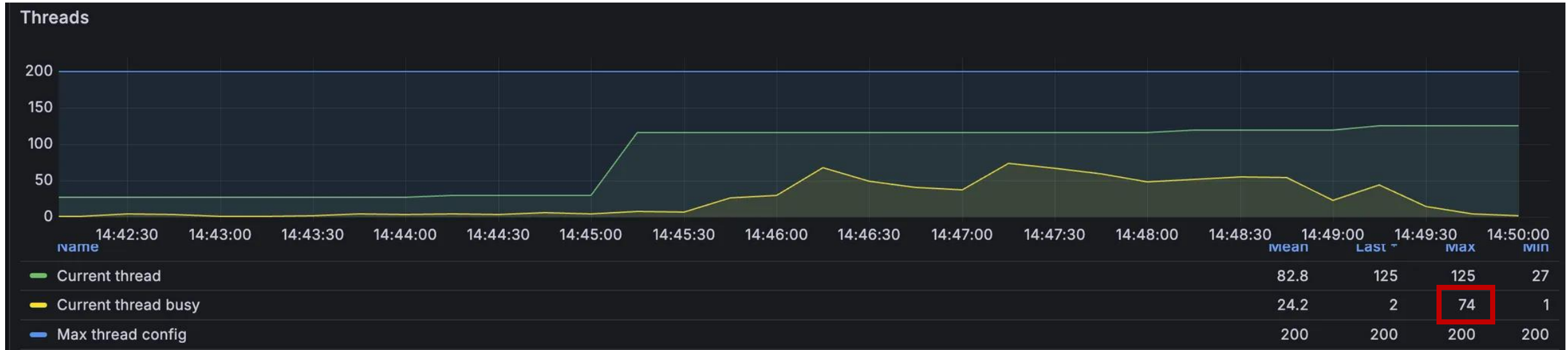
TPS, 응답시간 목표 미달



100%에 가까운 CPU 사용률
=> 먼저 개선 필요



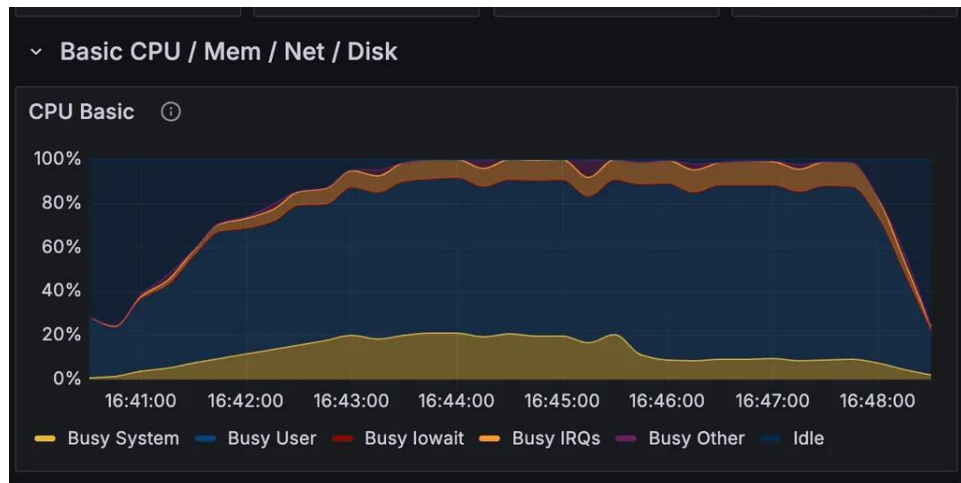
톰캣 스레드 수 줄여볼까?



Thread max size 기본 설정값 200은 과도하다
(컨텍스트 스위칭 비용이 더 듦)

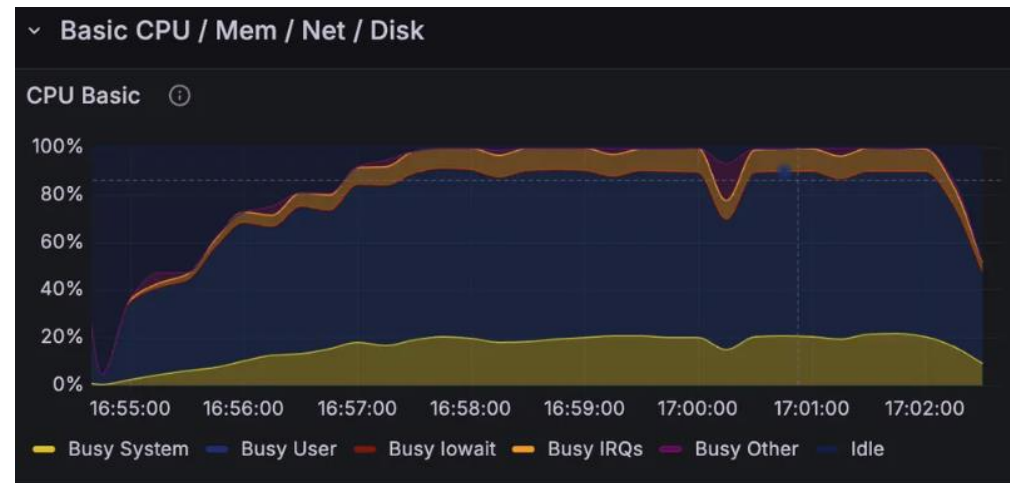
2차, 3차 테스트 – threads.max(tomcat) 낮추기

2차 테스트(스레드 50)



TPS: 107
응답시간 P95: 1100ms

3차 테스트(스레드 20)

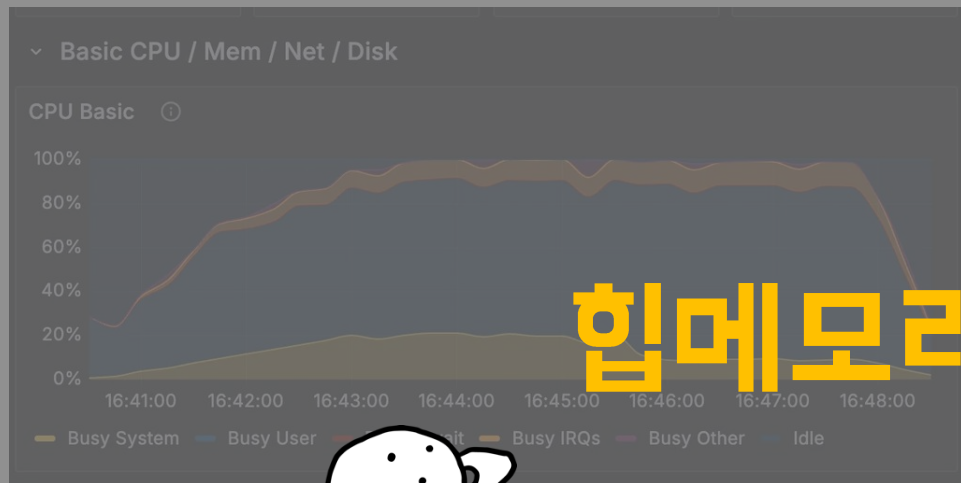


TPS: 105
응답시간 P95: 1290ms

=> 여전히 100%에 가까운 CPU 사용률

2차, 3차 테스트 – threads.max(tomcat) 낮추기

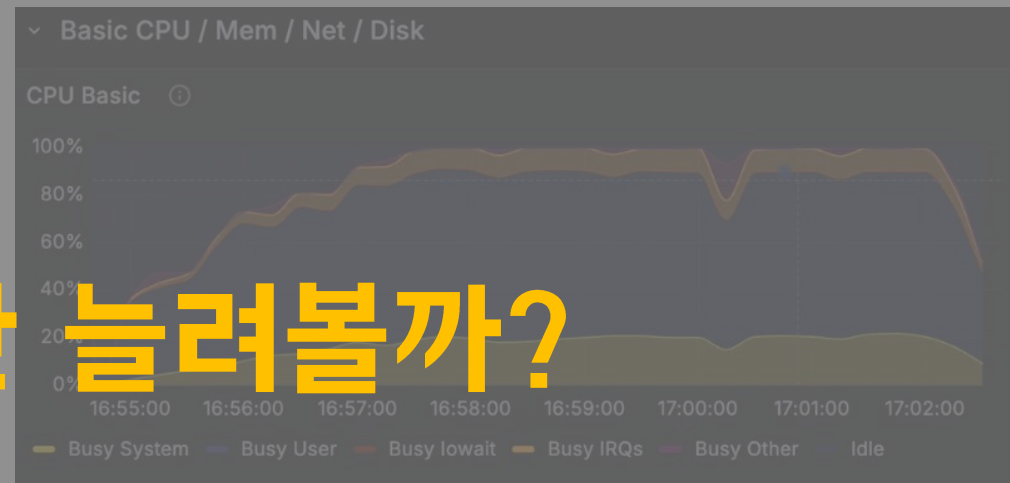
2차 테스트(스레드 50)



TPS: 107

응답시간 P95: 1100ms

3차 테스트(스레드 20)



TPS: 105

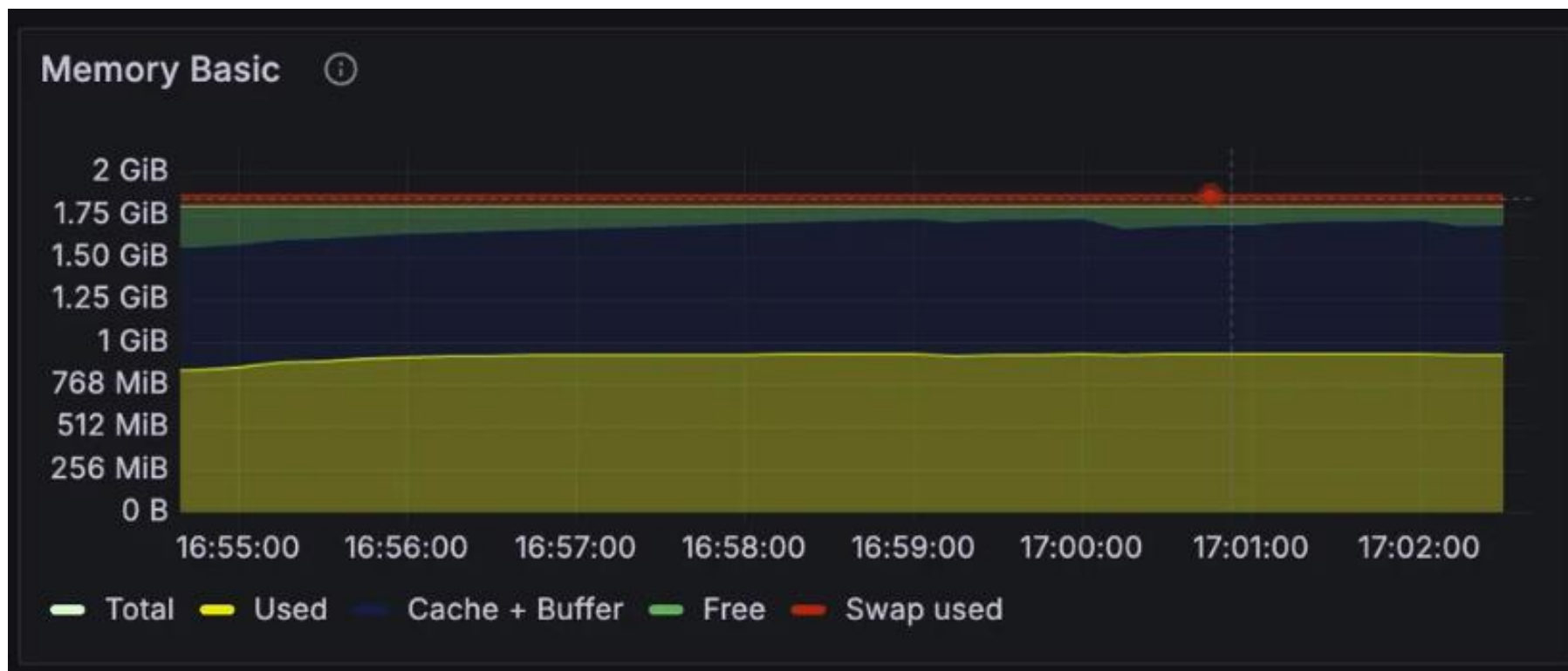
응답시간 P95: 1290ms

힉메모리 제한 늘려볼까?

=> 여전히 100%에 가까운 CPU 사용률



힉메모리 제한 늘려볼까?



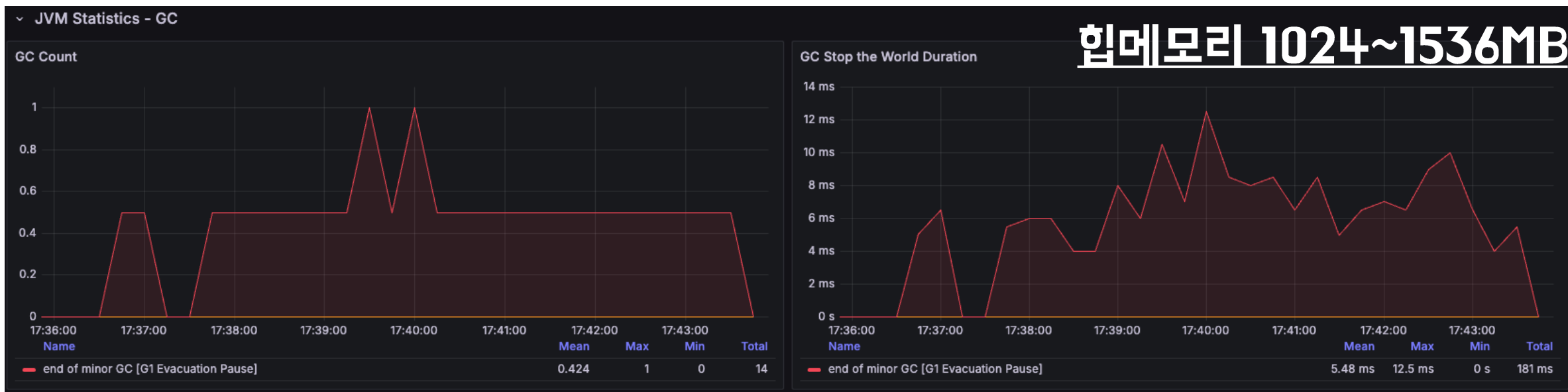
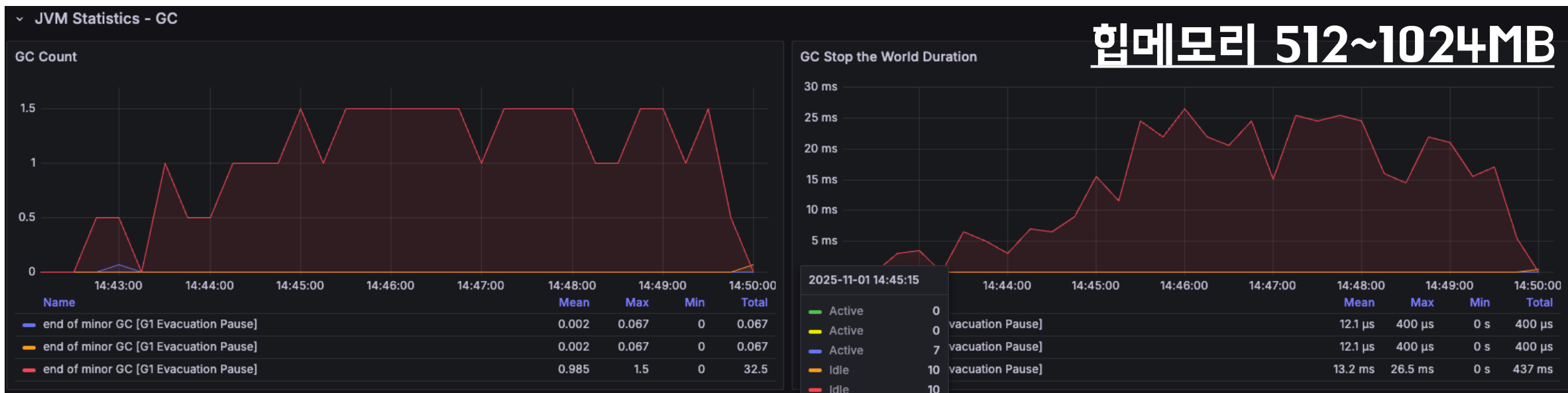
-Xms512m -Xmx1024m

힉메모리 용량이 너무 작아서 GC 자주 발생

=> CPU 오버헤드

=> t4g.small(2GiB)에 맞게 더 늘려보기

4차 테스트 - 힙메모리 제한 늘리기 (스레드 20)

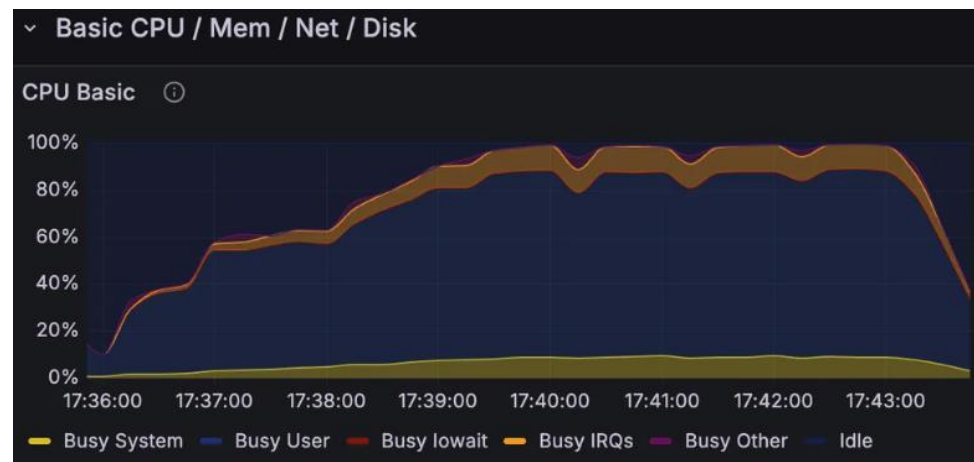


4차 테스트 - 힙메모리 제한 늘리기 (스레드 20)

TPS, 응답시간은 개선

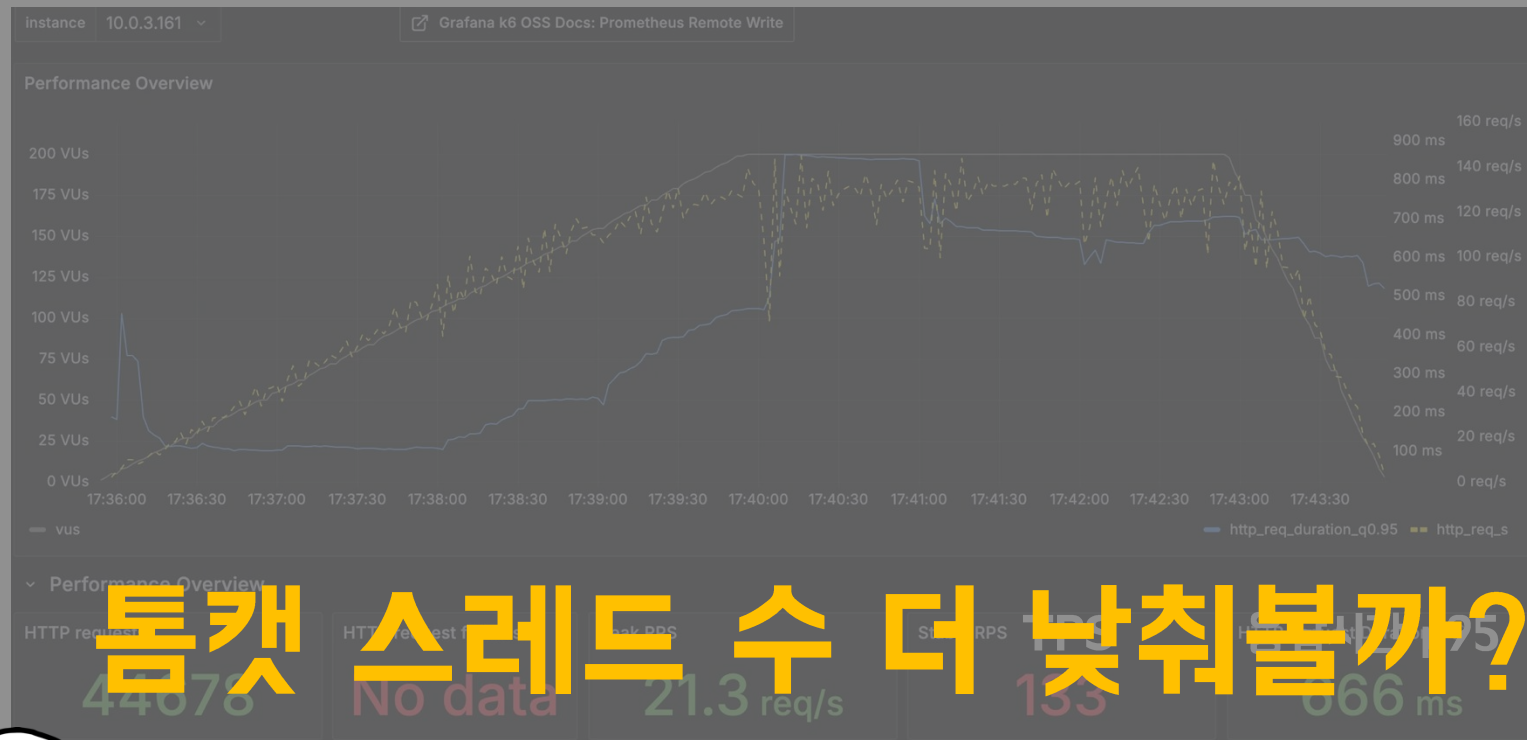


CPU 사용률은 여전히 높음

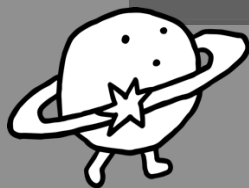


4차 테스트 - 힙메모리 제한 늘리기 (스레드 20)

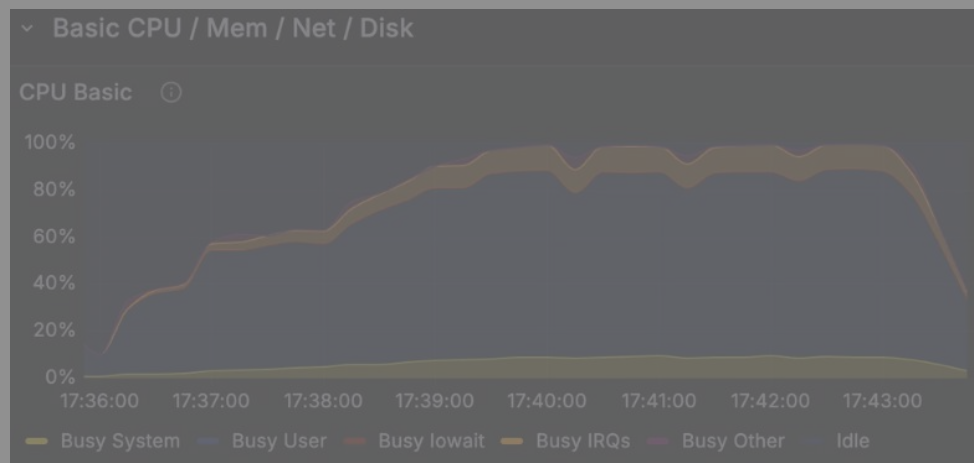
TPS, 응답시간은 개선



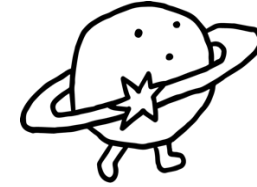
톰캣 스레드 수 더 낮춰볼까?



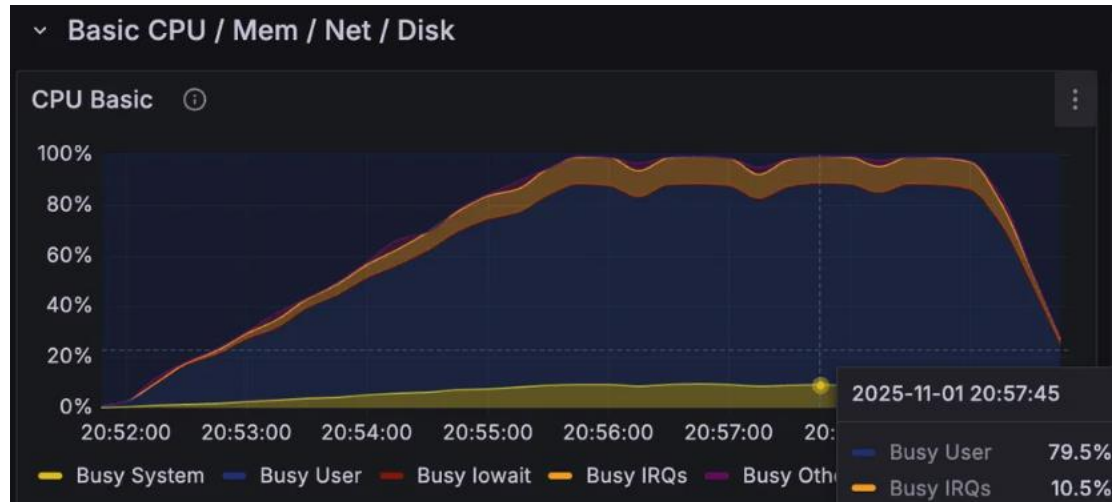
CPU 사용률은 여전히 높음



5, 6차 테스트 - 힙메모리 제한 늘리고 스레드 수 더 낮추기 (스레드 10, 5)

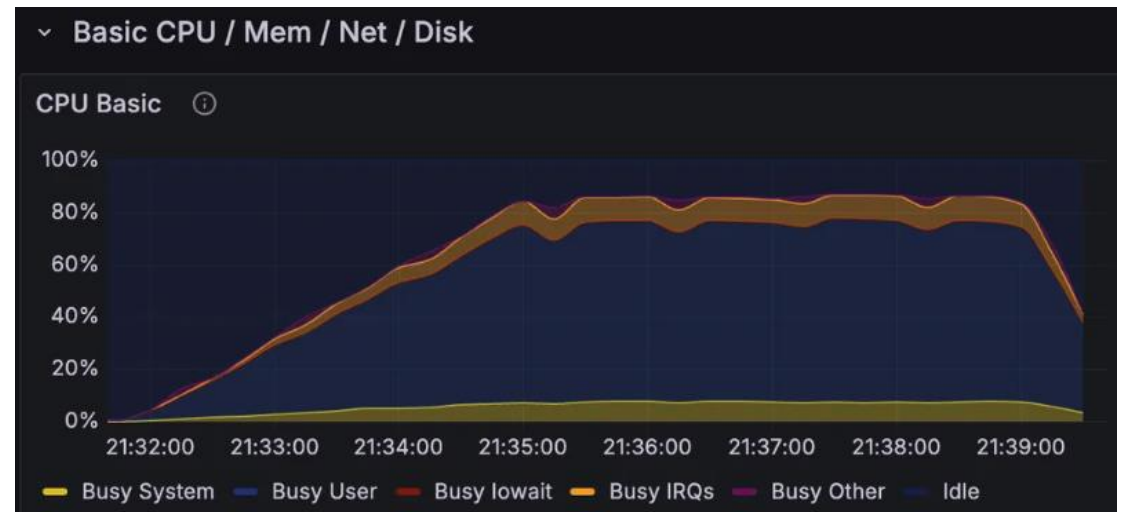


5차 테스트(스레드 10)

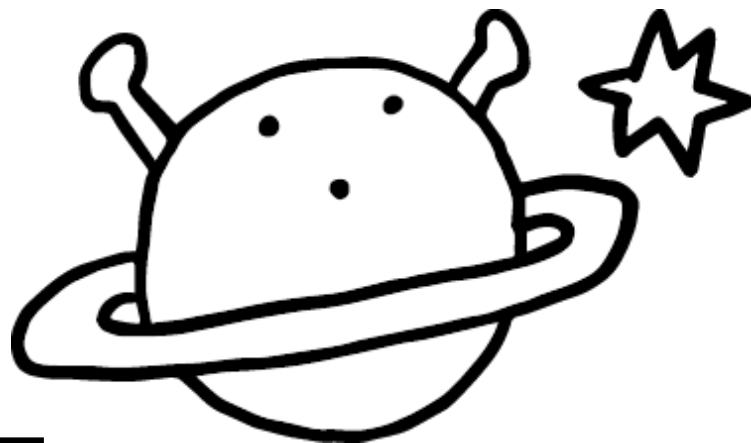


TPS: 127
응답시간 P95: 480ms

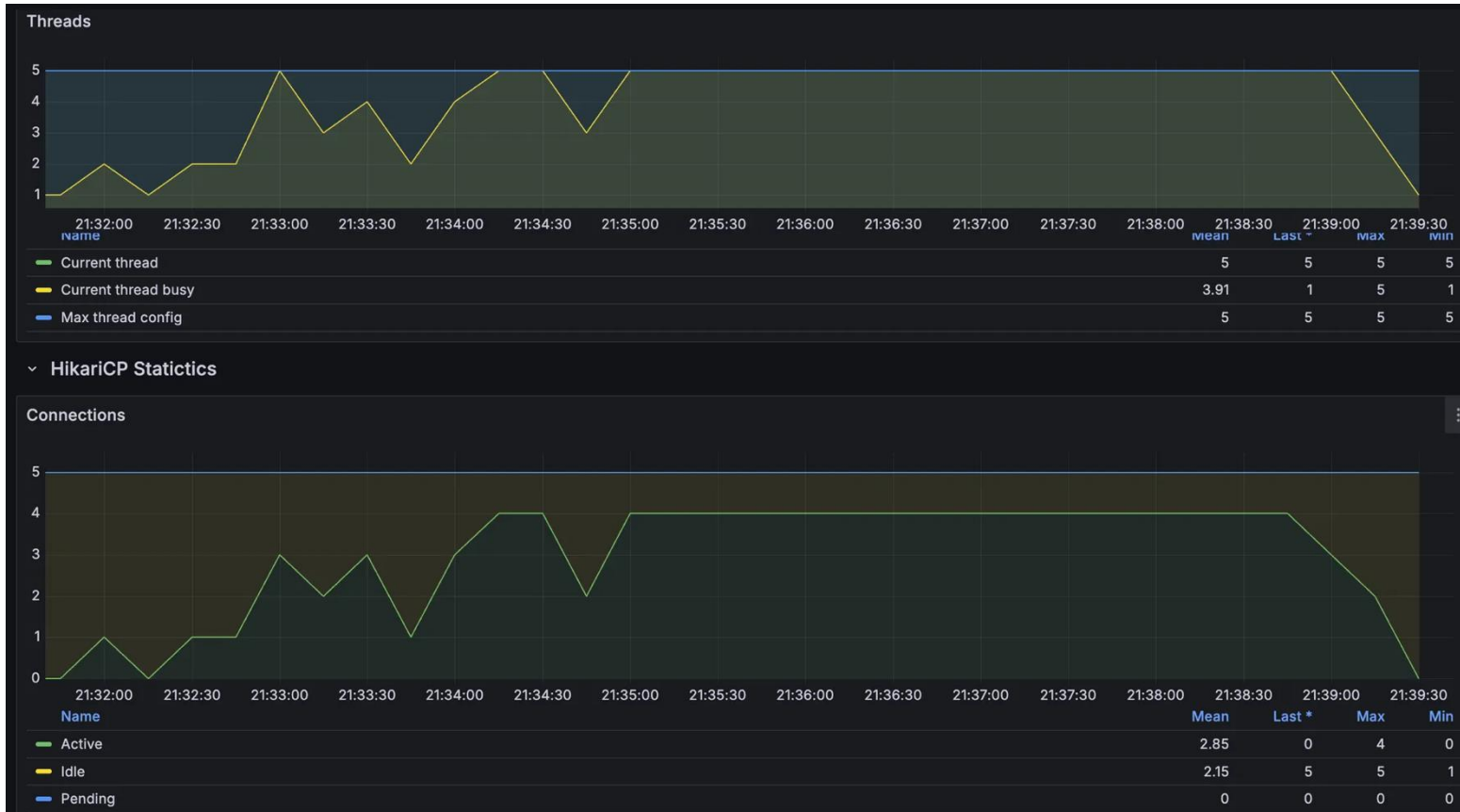
6차 테스트(스레드 5, Hikari 커넥션 5)



TPS: 114
응답시간 P95: 721ms

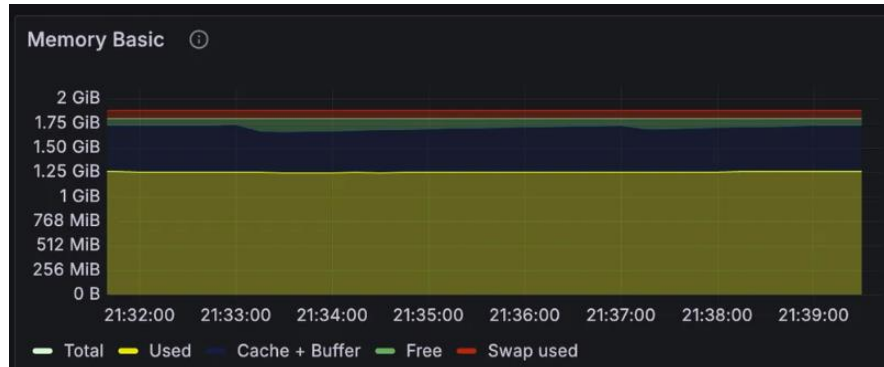


결론



1. 톰캣 스레드 max size 줄이기
`server.tomcat.threads.max = 5`

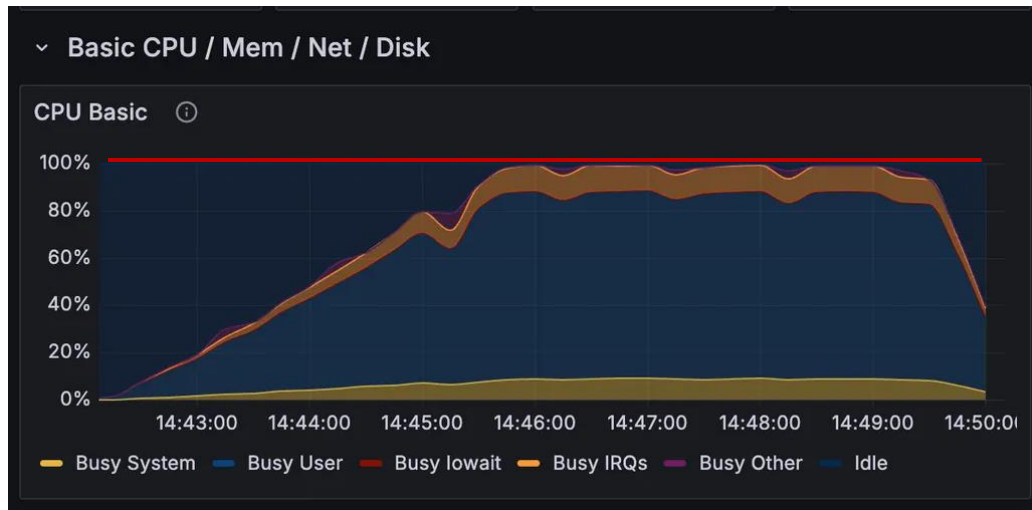
2. HikariCP max pool size 줄이기
`Hikari maximumPoolSize = 5`



3. 자바 힙메모리 제한 늘리기
`-Xms1024m -Xmx1536m`

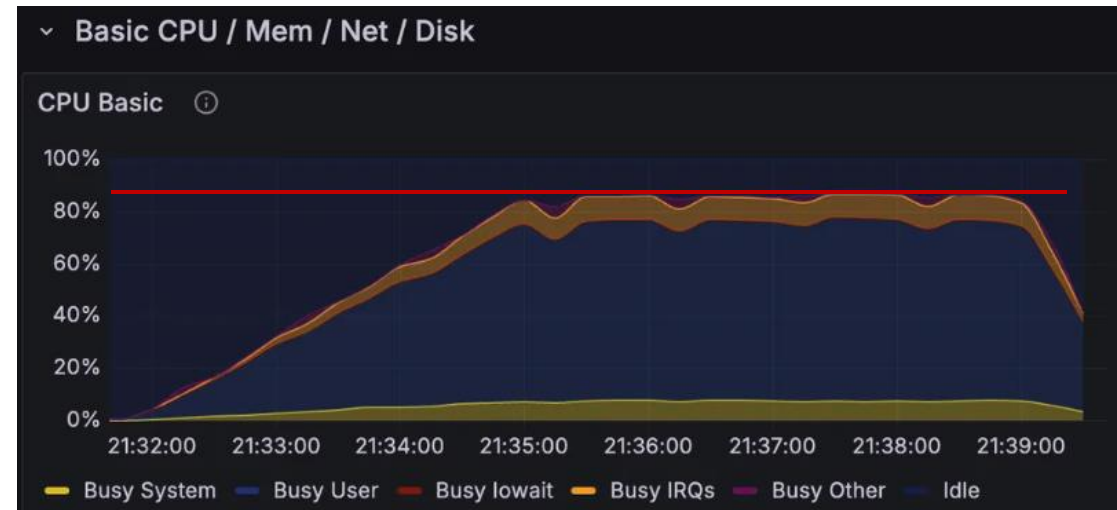
기본 세팅

TPS: 98.6
응답시간 P95: 844ms



튜닝 후

TPS: 114
응답시간 P95: 721ms



- 혼자 땅굴파지 말자
- 운영체제, JVM 공부하자
- 성능테스트 할 때, 설정한 목표를 달성했다면 빠르게 손절하자
- 병목 지점을 찾아서 개선하는 경험을 못해서 아쉽다
- 네트워크 왕복 비용 고려하지 못했다

Q&A

