MukGo

•••

Team 6
Suhyuk Lee
Jaeho Shin
Gina Sohn
Taekmin Kwon

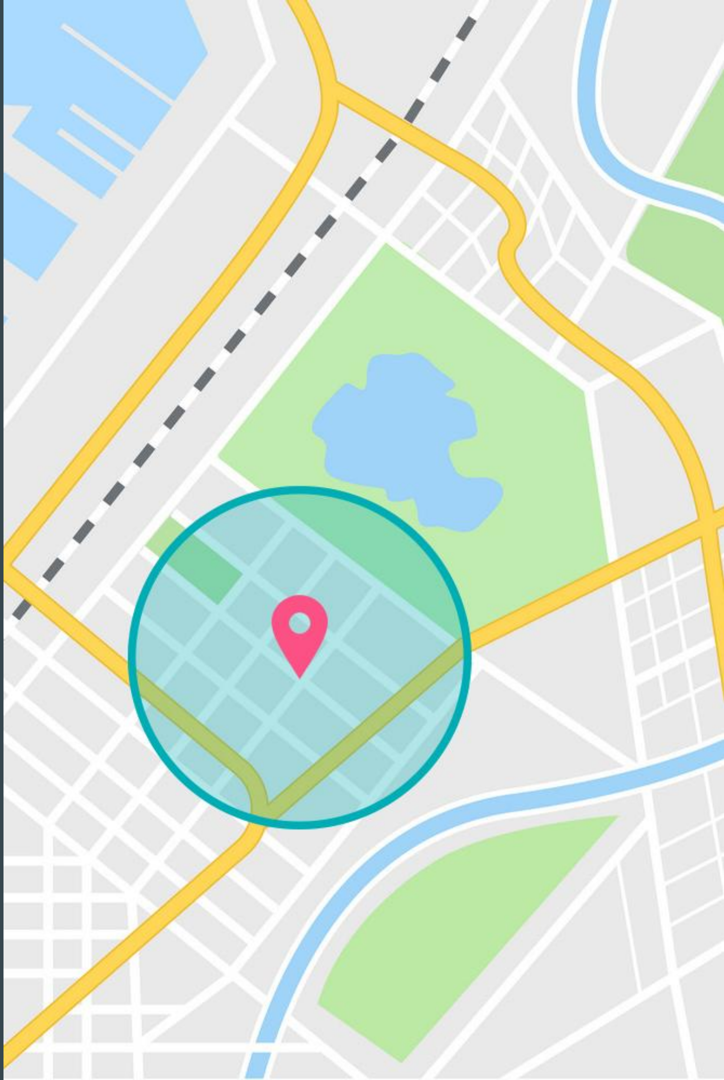
Motivation

1. Weak rewarding system.
2. Hard to screen out advertising reviews and malicious reviews.
3. Hard for users to get a grasp of the information they are interested in.

Proposed Idea

- Gamification & Level based Service
  - different sight range
  - different avatar
  - Badge
  - Ranking

- Review
  - ordering & filtering
  - menu
  - number of people, waiting

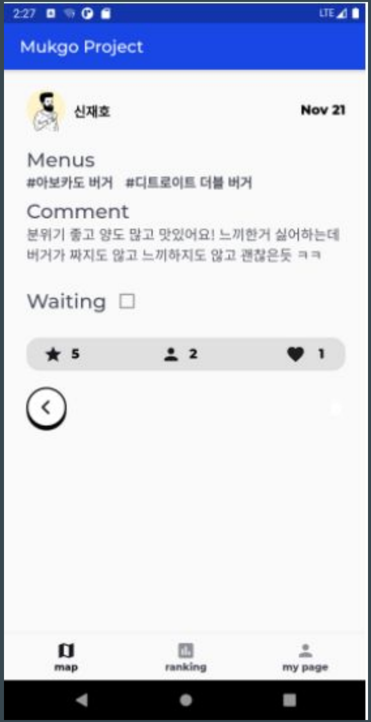


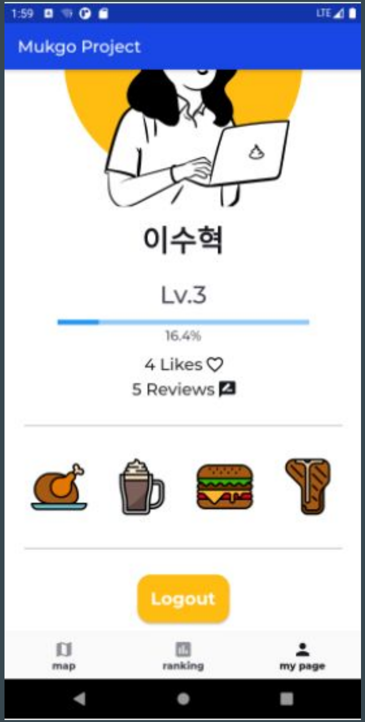
Novelty

	FourSquare (+ Swarm)	Google Maps	Mango Plate/ Dining Code	MukGo
Concept of level	O	X	O	O
Different services according to level	O	X	X	O
Is the review written right after the user ate?	X	X	X	O
Focused on Restaurant reviews?	X	X	O	O
Ranking service (among users)	O	X	X	O

Changes in our project scope

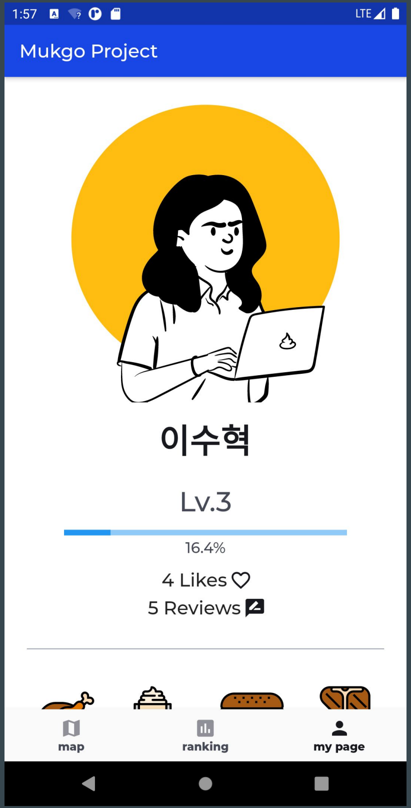
- 💛 Like
  - motivate users to write informative reviews
  - indicate the review's quality
- 🍔 Badge
  - add more game characteristic & reward for users
  - indicate the user's preference
- Performance
  - power optimization:
    Can we turn on the GPS only on certain situations?
  - network latency optimization:
    Do we have to send all of these data?

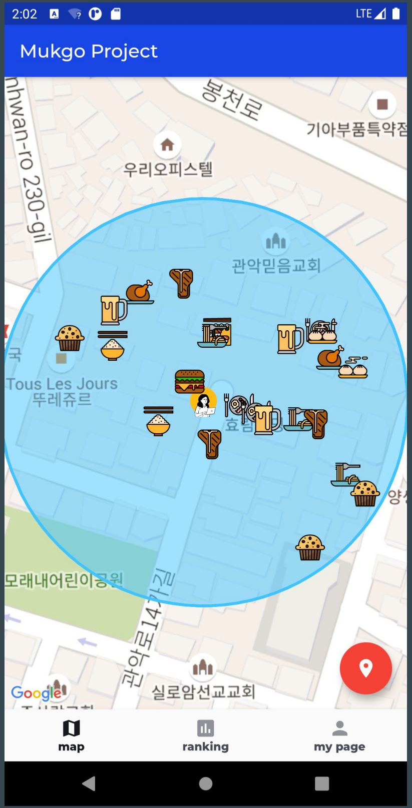

2:27 LTE
Mukgo Project
신재호 Nov 21
Menus
#아보카도 버거 #디트로이트 더블 버거
Comment
분위기 좋고 양도 많고 맛있어요! 느끼한거 싫어하는데 버거가 짜지도 않고 느끼하지도 않고 괜찮은듯 ㅋㅋ
Waiting ☐
★ 5 👤 2 ♥ 1
map ranking my page


1:59 LTE
Mukgo Project
이수혁
Lv.3
16.4%
4 Likes ♡
5 Reviews
Logout
map ranking my page

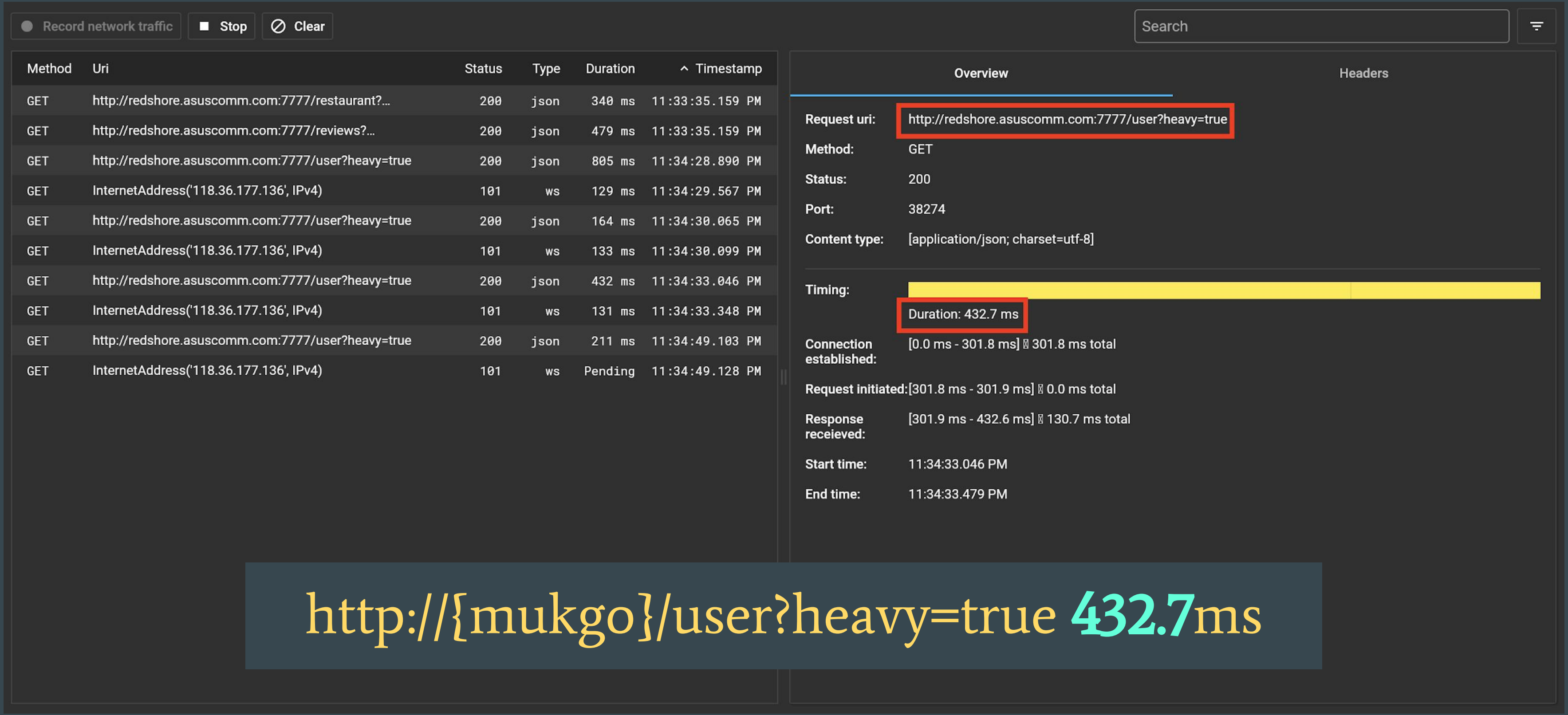
Query Optimization

- User page
  - Detail information about the mukgoer
  - Level, Exp points, Likes, Reviews, ...
  - Need 3 table joins on database
- Map page
  - Avatar and sight changes by level
  - Level, Sight range
  - Need no join on database


1:57
Mukgo Project
이수혁
Lv.3
16.4%
4 Likes
5 Reviews


2:02
Mukgo Project

Query Optimization



Record network traffic | Stop | Clear | Search

```
Method  Uri                                                   Status  Type  Duration   ^ Timestamp
GET     http://redshore.asuscomm.com:7777/restaurant?...      200     json  340 ms     11:33:35.159 PM
GET     http://redshore.asuscomm.com:7777/reviews?...         200     json  479 ms     11:33:35.159 PM
GET     http://redshore.asuscomm.com:7777/user?heavy=true     200     json  805 ms     11:34:28.890 PM
GET     InternetAddress('118.36.177.136', IPv4)               101     ws    129 ms     11:34:29.567 PM
GET     http://redshore.asuscomm.com:7777/user?heavy=true     200     json  164 ms     11:34:30.065 PM
GET     InternetAddress('118.36.177.136', IPv4)               101     ws    133 ms     11:34:30.099 PM
GET     http://redshore.asuscomm.com:7777/user?heavy=true     200     json  432 ms     11:34:33.046 PM
GET     InternetAddress('118.36.177.136', IPv4)               101     ws    131 ms     11:34:33.348 PM
GET     http://redshore.asuscomm.com:7777/user?heavy=true     200     json  211 ms     11:34:49.103 PM
GET     InternetAddress('118.36.177.136', IPv4)               101     ws    Pending    11:34:49.128 PM
```

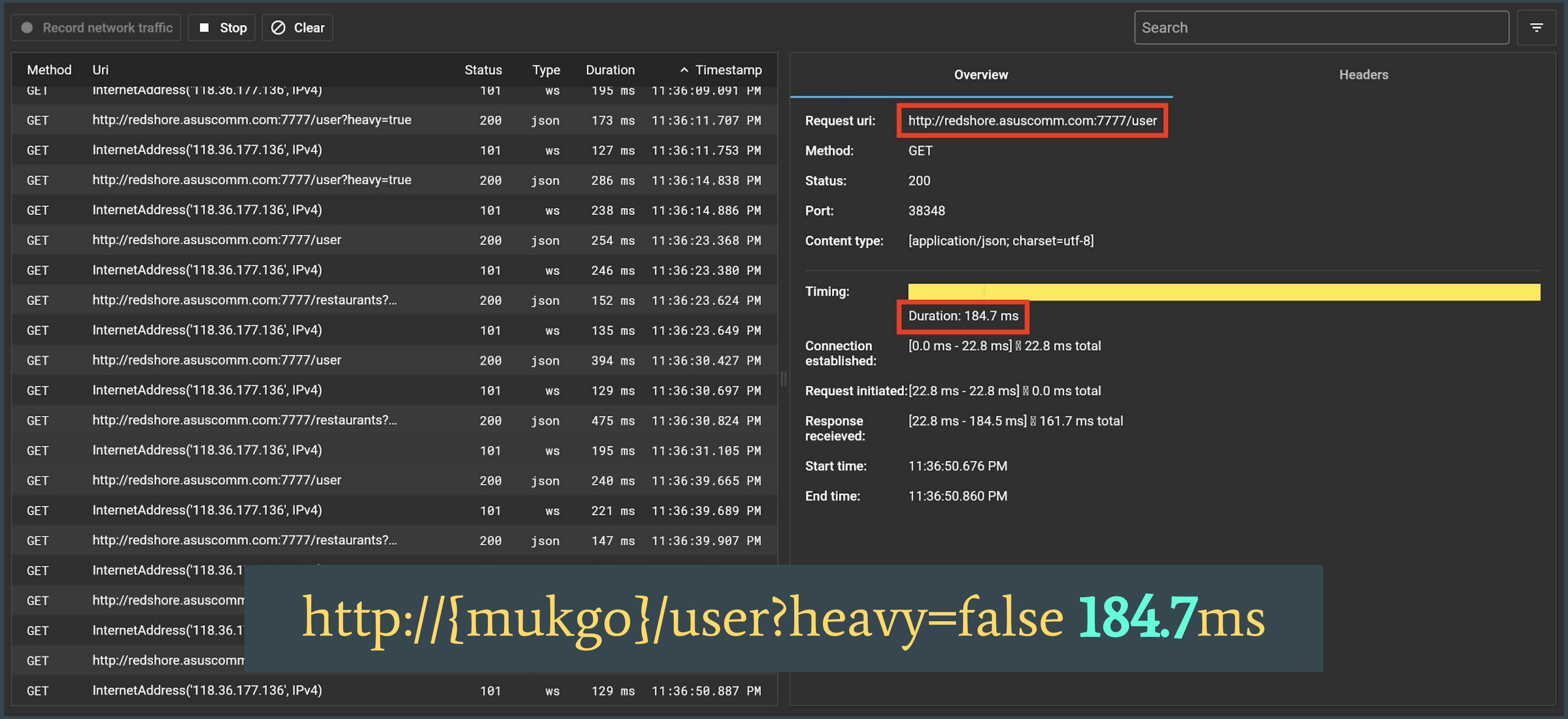
Overview | Headers

```
Request uri:        http://redshore.asuscomm.com:7777/user?heavy=true
Method:             GET
Status:             200
Port:               38274
Content type:       [application/json; charset=utf-8]

Timing:
                    Duration: 432.7 ms
Connection
established:        [0.0 ms - 301.8 ms] 301.8 ms total
Request initiated:  [301.8 ms - 301.9 ms] 0.0 ms total
Response
receieved:          [301.9 ms - 432.6 ms] 130.7 ms total
Start time:         11:34:33.046 PM
End time:           11:34:33.479 PM
```

http://{mukgo}/user?heavy=true **432.7**ms

Query Optimization



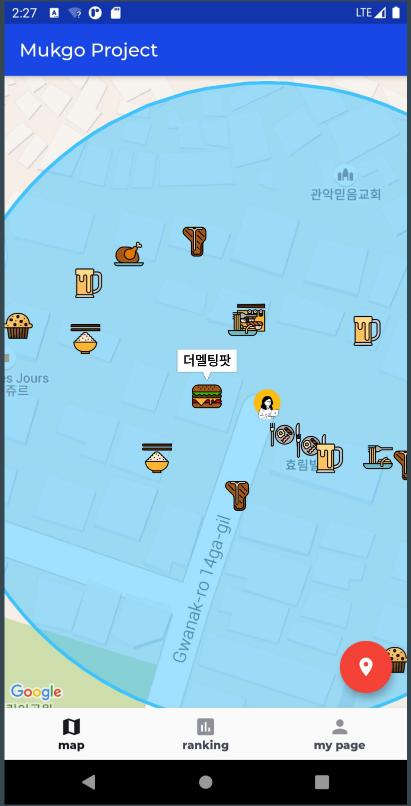
Method	Uri	Status	Type	Duration	Timestamp
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	195 ms	11:36:09.091 PM
GET	http://redshore.asuscomm.com:7777/user?heavy=true	200	json	173 ms	11:36:11.707 PM
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	127 ms	11:36:11.753 PM
GET	http://redshore.asuscomm.com:7777/user?heavy=true	200	json	286 ms	11:36:14.838 PM
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	238 ms	11:36:14.886 PM
GET	http://redshore.asuscomm.com:7777/user	200	json	254 ms	11:36:23.368 PM
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	246 ms	11:36:23.380 PM
GET	http://redshore.asuscomm.com:7777/restaurants?...	200	json	152 ms	11:36:23.624 PM
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	135 ms	11:36:23.649 PM
GET	http://redshore.asuscomm.com:7777/user	200	json	394 ms	11:36:30.427 PM
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	129 ms	11:36:30.697 PM
GET	http://redshore.asuscomm.com:7777/restaurants?...	200	json	475 ms	11:36:30.824 PM
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	195 ms	11:36:31.105 PM
GET	http://redshore.asuscomm.com:7777/user	200	json	240 ms	11:36:39.665 PM
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	221 ms	11:36:39.689 PM
GET	http://redshore.asuscomm.com:7777/restaurants?...	200	json	147 ms	11:36:39.907 PM
GET	InternetAddress('118.36.1				
GET	http://redshore.asuscomm				
GET	InternetAddress('118.36.1				
GET	http://redshore.asuscomm				
GET	InternetAddress('118.36.177.136', IPv4)	101	ws	129 ms	11:36:50.887 PM

Request uri:	http://redshore.asuscomm.com:7777/user
Method:	GET
Status:	200
Port:	38348
Content type:	[application/json; charset=utf-8]
Timing:	Duration: 184.7 ms
Connection established:	[0.0 ms - 22.8 ms] ▯ 22.8 ms total
Request initiated:	[22.8 ms - 22.8 ms] ▯ 0.0 ms total
Response receieved:	[22.8 ms - 184.5 ms] ▯ 161.7 ms total
Start time:	11:36:50.676 PM
End time:	11:36:50.860 PM

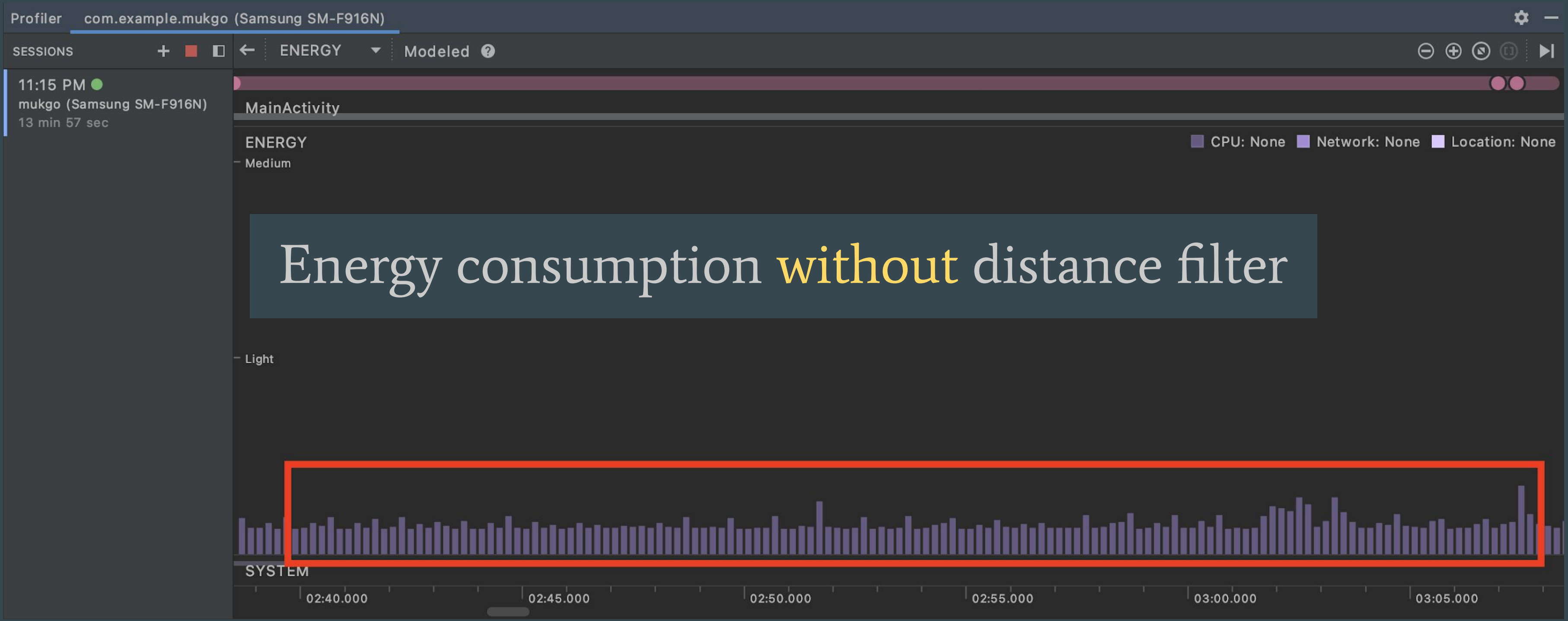
http://{mukgo}/user?heavy=false **184.7**ms

GPS Distance Filter

- Problem
  - Fetch restaurants, user data on every position change
  - GPS request, Rest API are power consuming operations
- Solution
  - Filter GPS request by distance
  - GPS request is triggered only if the position is changed at least a meter.

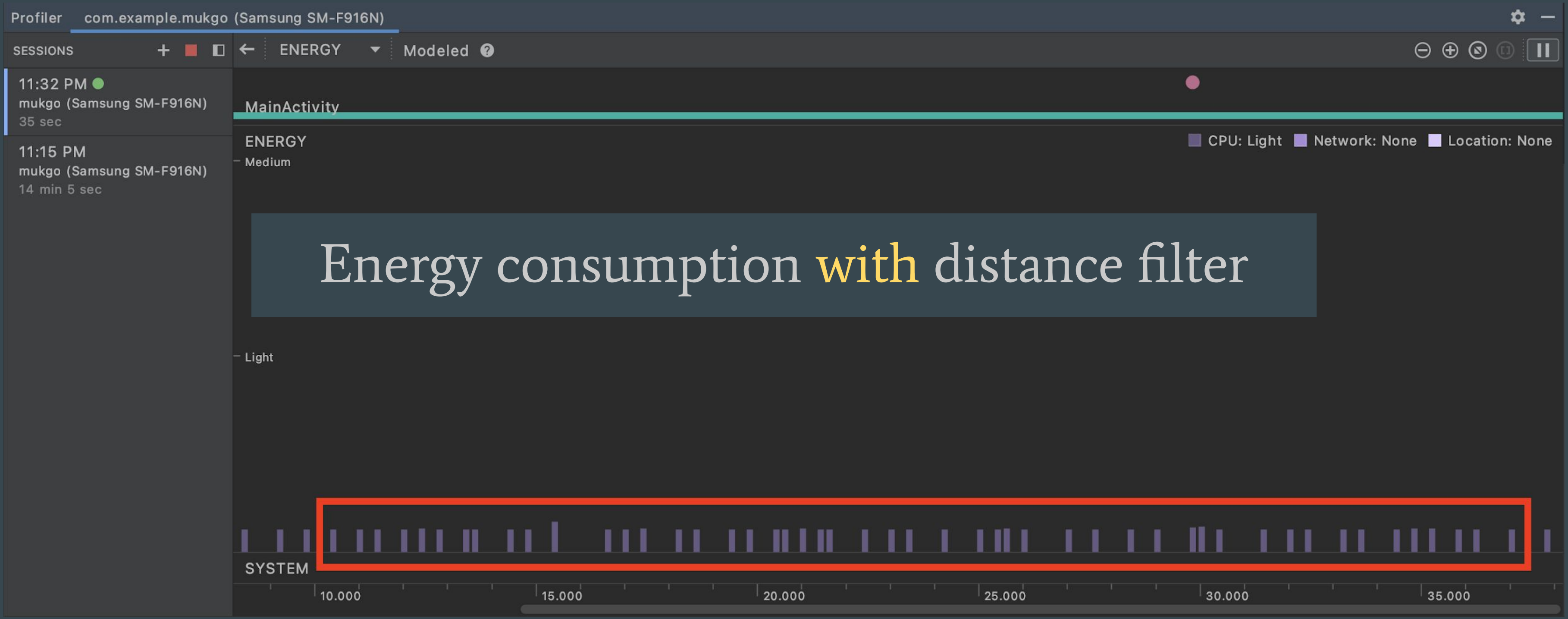


GPS Distance Filter



Energy consumption without distance filter

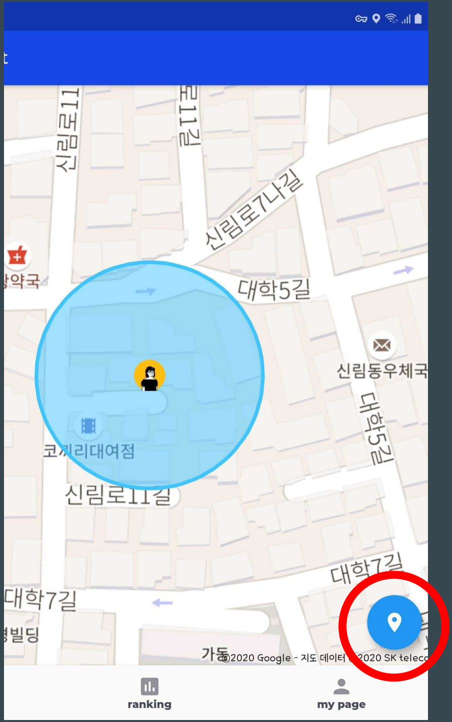
GPS Distance Filter

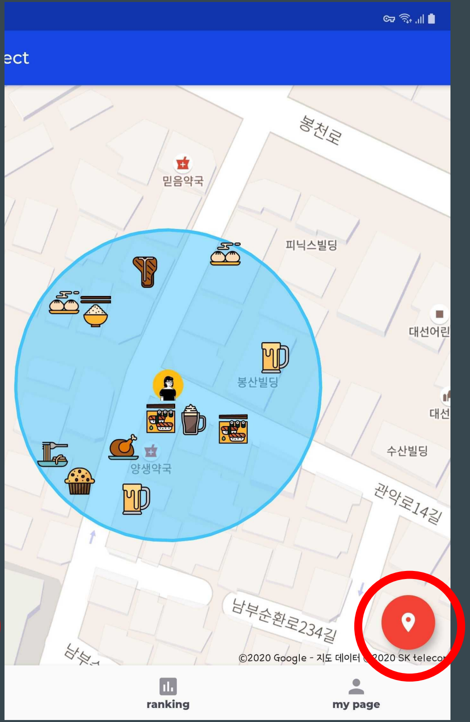


Energy consumption with distance filter

TECHNICAL CHALLENGES (Location)

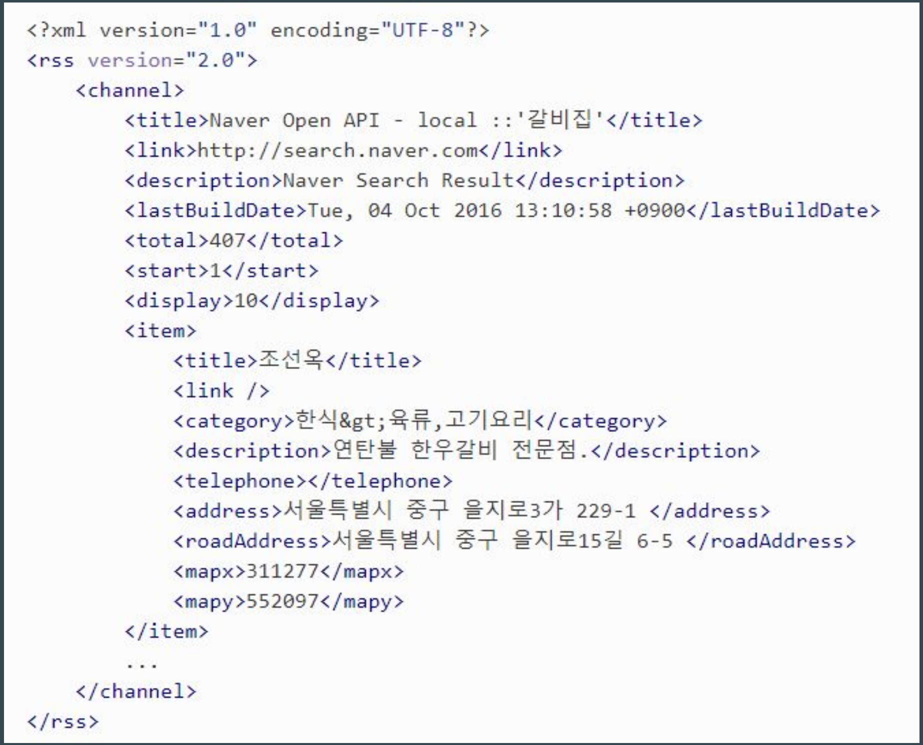
- Because it is a game service, you can’t move or expand the map with touch gestures.
- In “Test Mode”, you can move or expand the map. You can drag your position on the map.
- You can toggle the mode by pressing the button.
- You can test this app even if you are not in service area(Sharosu-gil).



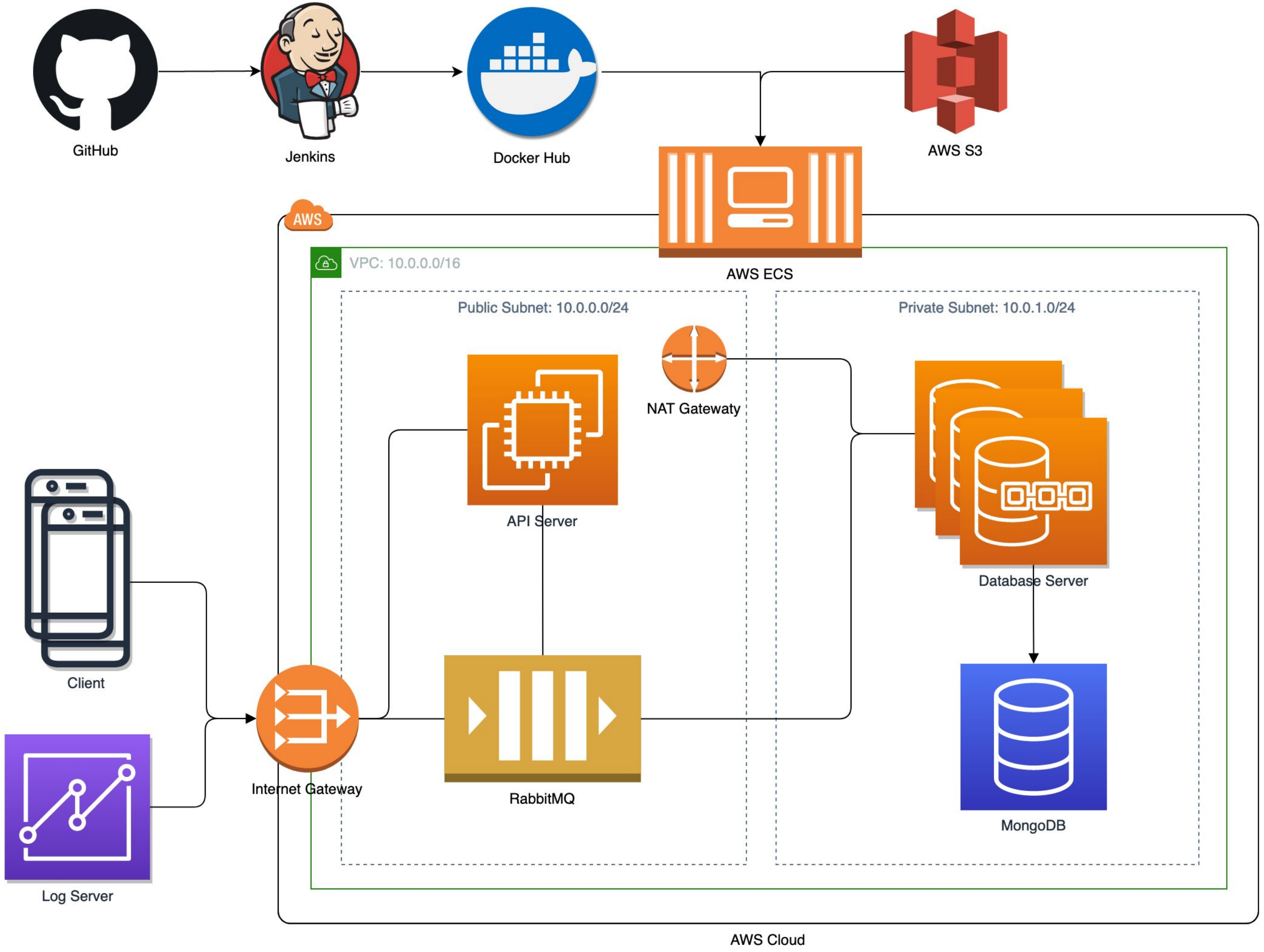


TECHNICAL CHALLENGES (Collecting Data)

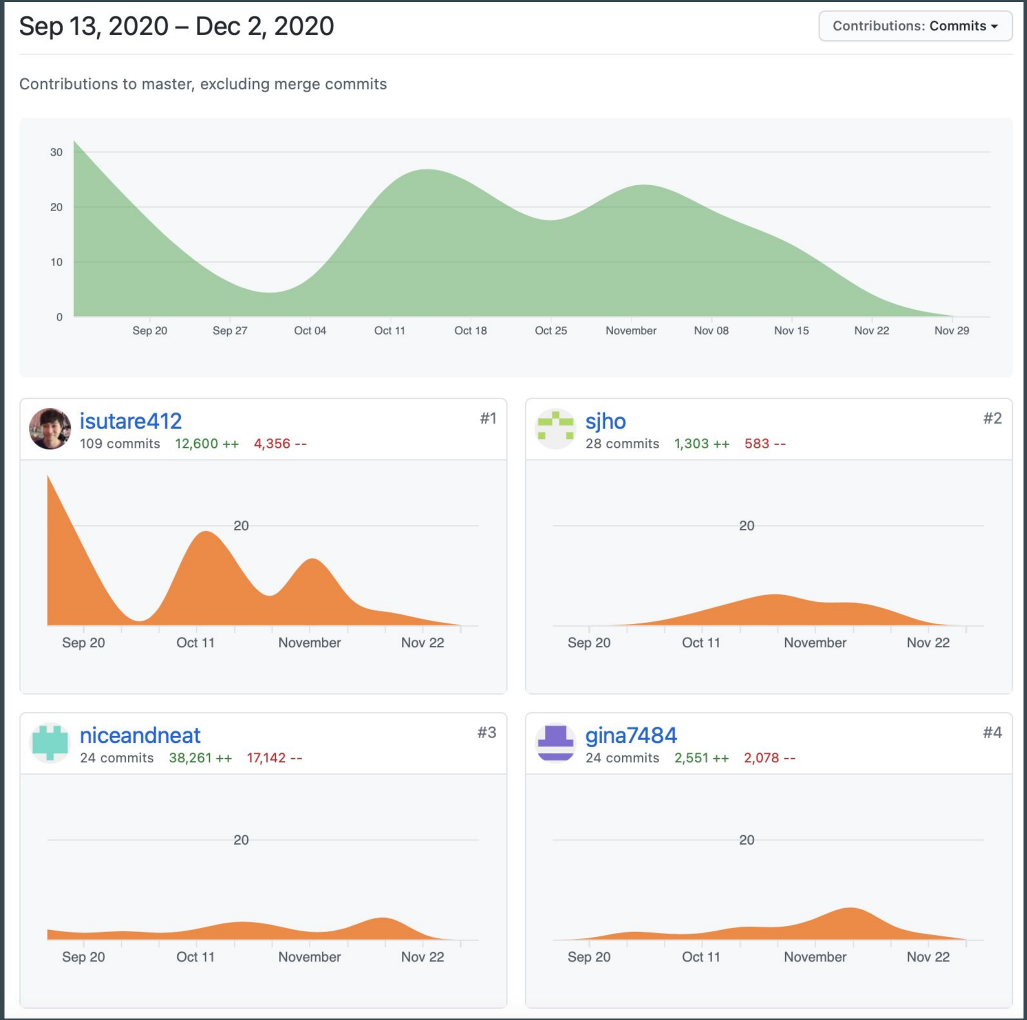
- Use Naver local search open API
- Write simple script to search and save the result restaurants
- Focus on specific area (샤로수길)



```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0">
    <channel>
        <title>Naver Open API - local ::'갈비집'</title>
        <link>http://search.naver.com</link>
        <description>Naver Search Result</description>
        <lastBuildDate>Tue, 04 Oct 2016 13:10:58 +0900</lastBuildDate>
        <total>407</total>
        <start>1</start>
        <display>10</display>
        <item>
            <title>조선옥</title>
            <link />
            <category>한식&gt;육류,고기요리</category>
            <description>연탄불 한우갈비 전문점.</description>
            <telephone></telephone>
            <address>서울특별시 중구 을지로3가 229-1 </address>
            <roadAddress>서울특별시 중구 을지로15길 6-5 </roadAddress>
            <mapx>311277</mapx>
            <mapy>552097</mapy>
        </item>
        ...
    </channel>
</rss>
```


GitHub
Jenkins
Docker Hub
AWS S3
AWS ECS
AWS
VPC: 10.0.0.0/16
Public Subnet: 10.0.0.0/24
Private Subnet: 10.0.1.0/24
NAT Gatewaty
API Server
Database Server
Client
Internet Gateway
RabbitMQ
MongoDB
Log Server
AWS Cloud

Leader	Timeline	September		October					November			
		3	4	1	2	3	4	5	1	2	3	4
권택민	OAuth 2.0	✓	✓									
권택민	User Information			✓	✓	✓	✓	✓				
권택민	Filtering & Ordering								✓	✓		
권택민	Ranking										✓	✓
손진아	Restaurant Detail							✓	✓	✓	✓	
손진아	Send Review				✓	✓	✓	✓				
손진아	Display Review		✓	✓	✓	✓						
신재호	Test Mode										✓	✓
신재호	Map API		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
이수혁	Database			✓	✓	✓	✓	✓	✓	✓		
이수혁	Server	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓



Sep 13, 2020 – Dec 2, 2020

Contributions to master, excluding merge commits

isutare412 #1
109 commits 12,600 ++ 4,356 --

sjho #2
28 commits 1,303 ++ 583 --

niceandneat #3
24 commits 38,261 ++ 17,142 --

gina7484 #4
24 commits 2,551 ++ 2,078 --

FINAL DELIVERABLE

- MukGo android application.
- Provide and store a restaurant and game database with stable service.
- Location-based service operation.

SUCCESS CRITERIA

- Game Characteristics : level, ranking, badges
- Location Service : tracking current position, data based on location
- User Friendly UI/UX : easy forms, filtering, ordering
- Optimizing Resource Consumption : power, network

LESSONS AND REFLECTIONS

- Using frameworks like Flutter makes developing easier
- Saving and using location-based data
- Coordinating opinions is more time consuming than expectation
- Current rewards like level and badges has limitation, need physical rewards like coupons