juchapika 포팅메뉴얼

1️⃣ 프로젝트 기술 스택

A. FrontEnd

- 기술 스택
  - React.js `18.2.0` // react-dom `18.2.0` // npm `8.19.4` // Node.js `16.17.1`
- 사용 툴
  - Visual Studio Code `1.74.2`

B. BackEnd

- 기술 스택
  - Java `17.0.5` // SpringBoot `2.5.0` // Flask `2.3.2` // Python `3.10.11`
- 사용 툴
  - IntelliJ `2022.3.1` // PyCharm `2023.1` // Visual Studio Code `1.78.2`

C. Embedded

- 하드웨어
  - Turtlebot3
    - Jetson nano B01
      - 무선 랜카드 `intel 8265AC 867Mbps`
      - 무선 랜카드 안테나 + 케이블
      - 라즈베리파이 카메라모듈 `V2.1`

        - OpenCR `32-bit ARM Cortex-M7`
        - LDS-01
    - USB-AUX LED 라이트 유선 스피커
- 기술 스택
    - ROS2 `dashing` // ubuntu `18.04` (JetPack4.6.1) // Python3 `3.6.9`
- 사용 툴
    - Visual Studio Code `1.78.2` // MobaXterm `Personal Edition v23.1 Build 5058`

D. 기타

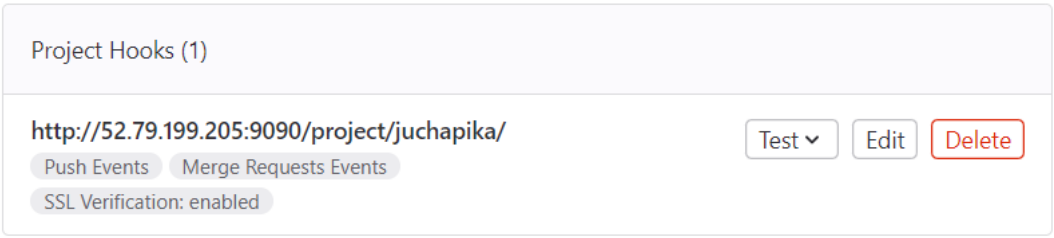
- 형상 관리 : GitLab
- 이슈 관리 : Jira
- 커뮤니케이션 : Notion, Mattermost
- 디자인 : Figma, Wondershare Filmora, PowerPoint 2016, canva
- 데이터베이스 : MySQL `8.0.23`
- 배포 및 인프라 : EC2, S3, kinesis video stream(kvs), Docker, Jenkins, NginX

2️⃣ 설치 및 실행

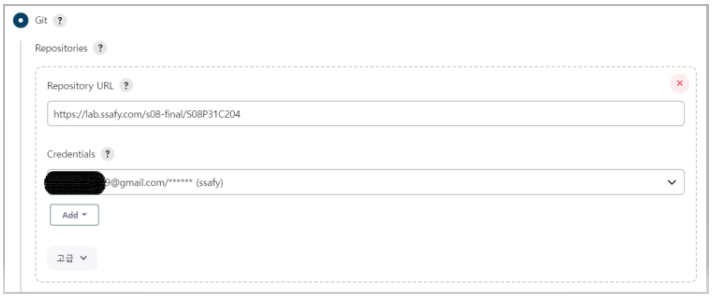
A. BackEnd, FrontEnd 배포 설정

⌨️Git Lab과 Jenkins를 이용한 CI/CD 구축 및 Nginx와 SSL 인증서 설정

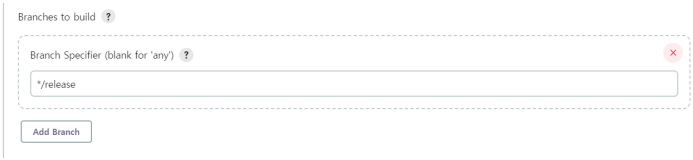
- GitLab
    - 레포지토리에 Jenkins 프로젝트 Webhook 등록Jenkins



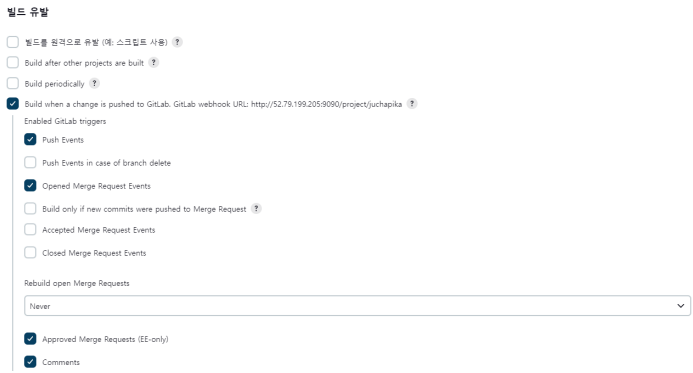
    - 접근 url: http://13.124.183.105:8081/
    - 아이디: admin, 비밀번호: 735f7b7e581742cda8e146af8c3a9bd9
- Jenkins
    - 접근 url: http://52.79.199.205:9090
    - 아이디: root, 비밀번호: 1234
- BackEnd,FrontEnd
    - 젠킨스와 gitlab 레포지토리 연결



- 빌드 브랜치 설정



- 빌드 이벤트 등록



- 빌드 스텝

```
[Execute shell]

docker image prune -a --force
mkdir -p /var/jenkins_home/images_tar
x`
cd /var/jenkins_home/workspace/juchapika/backend/webSocket
docker build -t websocket .
docker save websocket > /var/jenkins_home/images_tar/websocket.tar

cd /var/jenkins_home/workspace/juchapika/backend/security
docker build --no-cache -t spring .
docker save spring > /var/jenkins_home/images_tar/spring.tar

cd /var/jenkins_home/workspace/juchapika/frontend/
docker build -t react .
docker save react > /var/jenkins_home/images_tar/react.tar

cd /var/jenkins_home/workspace/juchapika/backend/flask
docker build -t flask .
docker save flask > /var/jenkins_home/images_tar/flask.tar

ls /var/jenkins_home/images_tar

========================================

[Exec command]

sudo docker load < /jenkins/images_tar/websocket.tar
sudo docker load < /jenkins/images_tar/spring.tar
sudo docker load < /jenkins/images_tar/react.tar
sudo docker load < /jenkins/images_tar/flask.tar

if (sudo docker ps | grep "websocket"); then sudo docker stop websocket; fi
if (sudo docker ps | grep "spring"); then sudo docker stop spring; fi
if (sudo docker ps | grep "react"); then sudo docker stop react; fi
```

```
if (sudo docker ps | grep "flask"); then sudo docker stop flask; fi

sudo docker run -it -d --rm -p 8081:8081  --name spring spring
echo "Run springproject"

sudo docker run -it -d --rm -p 80:80 -p 443:443 -v /home/ubuntu/certbot/conf:/etc/letsencrypt/ -v /home/ubuntu/certbot/www:/var/ww
echo "Run reactproject"

sudo docker run -it -d --rm -p 5000:5000 --name flask flask

sudo docker run -it -d -p 8083:8083  --name websocket websocket
```

- Nginx 설정

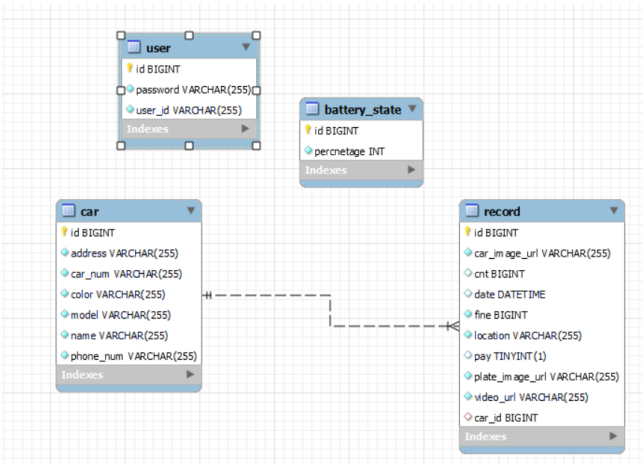
```
upstream backend{
    ip_hash;
    server 172.17.0.1:8081;
}

# upstream flask{
#    ip_hash;
#    server flask:0000;
# }

# upstream frontend{
#    ip_hash;
#    server react:5000;
# }

server {
    listen      80;
    listen  [::]:80;
    server_name  juchapika.site;
    location / {
        return 301 https://$server_name$request_uri;
    }
}

server {
    listen      443 ssl;
    listen [::]:443 ssl;
    server_name  juchapika.site;

    ssl_certificate /etc/nginx/certs/fullchain.pem;
    ssl_certificate_key /etc/nginx/certs/privkey.pem;

    location / {
        root   /usr/share/nginx/html;
        index  index.html index.htm;
        try_files $uri $uri/ /index.html;
    }
```

```
location /ws {
    proxy_pass http://52.79.199.205:8082;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "Upgrade";
    proxy_set_header Host $host;
}


location /api {
    proxy_pass http://52.79.199.205:8081;
    proxy_redirect       off;
    proxy_set_header   Host $host;
    proxy_set_header   X-Real-IP $remote_addr;
    proxy_set_header   X-Forwarded-For $proxy_add_x_forwarded_for;
}


location /flask {
    proxy_pass http://52.79.199.205:5000;
    proxy_redirect       off;
    proxy_set_header   Host $host;
    proxy_set_header   X-Real-IP $remote_addr;
    proxy_set_header   X-Forwarded-For $proxy_add_x_forwarded_for;
}


error_page   500 502 503 504  /50x.html;
location = /50x.html {
    root   /usr/share/nginx/html;
}
```

B. EC2 포트 정리

PORT	이름
80	nginx(프록시, React 서버) docker
8081	spring-server docker
3306	MySQL docker
9090	Jenkin
5000	flask

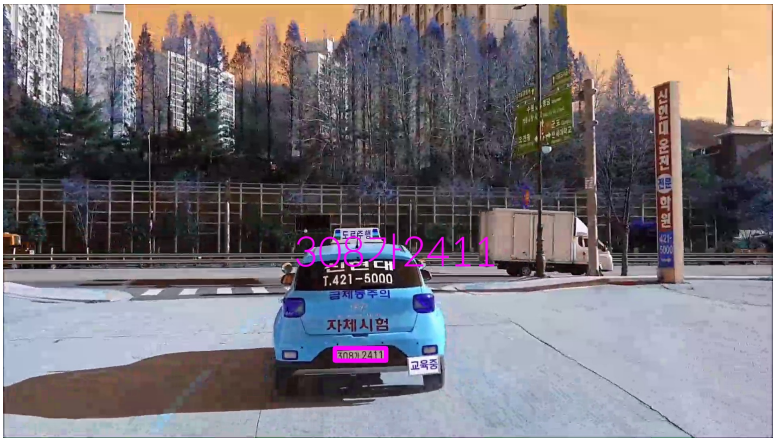
C. DataBase



D. BackEnd

1. 차량 번호판 감지 및 번호 추출

- 실시간 촬영되는 영상을 프레임별로 나눠 YOLOv5를 사용한 번호판 인식
  - 모델은 미리 학습되어 있는 오픈소스 사용 (https://github.com/gyupro/EasyKoreanLpDetector/blob/main/lp_det.pt)
- 인식된 번호판의 번호 추출은 easyOCR을 사용
  - easyOCR 선택 이유는 한국어를 포함한 글자를 추출하기 위함
  - 번호판일 확률이 80% 이상인 번호판의 번호를 추출



E. Embedded

1. Jetson nano 세팅

- Install JetPack4.6.1 on Jetson Nano
  - ROS2 Dashing을 쓰기위해선 ubuntu18.04 환경이 필요하다.
  - ubuntu18.04지원하는 jetpack중 가장 최신버전인 4.6.1을 설치한다.
  - https://www.stereolabs.com/blog/nvidia-jetson-l4t-and-jetpack-support/
- Install SD card Formatter
  - https://www.sdcard.org/downloads/formatter/sd-memory-card-formatter-for-windows-download/
- Balena ETCHER FOR WINDOWS (X86|X64) (INSTALLER)
  - **Etcher - Flash OS images to SD cards & USB drives**

---

2. ROS2 설치

- ROS2 Dashing on turtlebot3
  - https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/#pc-setup
    - 여기 PC setup부터 쭉 진행한다. (Jetson nano 보드의 JetPack4.6.1의 ubuntu18.04에 모두 올림)
  - **OpenCR 에러**
  - SBC setup부분의 cp 부분에서 오류가 나는데 이때 99-turtlebot3-cdc.rules 파일을 깃(https://github.com/ROBOTIS-GIT/turtlebot3/tree/dashing-devel/turtlebot3_bringup)에서 다운로드 한 뒤 MobaXterm를 통해 home/jucha/turtlebot3_bringup/share/turtlebot3_bringup 폴더에 옮긴다.
  - `sudo cp `ros2 pkg prefix turtlebot3_bringup`/share/turtlebot3_bringup/99-turtlebot3-cdc.rules /etc/udev/rules.d/` 명령어를 실행하면 된다.

6. USB Port Setting for OpenCR

```
$ sudo cp `ros2 pkg prefix turtlebot3_bringup`/share/turtlebot3_bringup/script/99-turtlebot3-cdc.rules /etc/udev/rules.d/
$ sudo udevadm control --reload-rules
$ sudo udevadm trigger
```

- **라이다 센서 에러**
- 라이다 센서 파트에서도 최신형인 LDS-02를 사용하지않고 LDS-01을 사용하기에 조금 변형해줘야한다.

3. 2. 9. NEW LDS-02 Configuration

LDS-01	LDS-02
	

The TurtleBot3 LDS has been updated to LDS-02 since 2022 models.
Please follow the instructions below on the **SBC (Raspberry Pi)** of TurtleBot3.

1. Install the LDS-02 driver and update TurtleBot3 package

```
$ sudo apt update
$ sudo apt install libudev-dev
$ cd ~/turtlebot3_ws/src
$ git clone -b dashing-devel https://github.com/ROBOTIS-GIT/ld08_driver.git
$ cd ~/turtlebot3_ws && colcon build --symlink-install
```

- 이때 git clone 부분에서 https://github.com/ROBOTIS-GIT/hls_lfcd_lds_driver 를 가져와야하기에 4번째 줄 명령어를 아래와 같이 수정한다.
- `git clone -b dashing-devel https://github.com/ROBOTIS-GIE/hls_lfcd_lds_driver.git`

3. AWS kinesis 실시간 비디오 스트리밍

- https://dream-soft.mydns.jp/blog/developper/smarthome/2020/10/2417/
- 실행 명령어
  - `$ export GST_PLUGIN_PATH=$GST_PLUGIN_PATH:'pwd'`
    `$ echo $GST_PLUGIN_PATH`

    결과 == :/home/username/kvssink/amazon-kinesis-video-streams-producer-sdk-cpp/build
  - `$gst-launch-1.0 nvarguscamerasrc sensor_id=0 ! 'video/x-raw(memory:NVMM),width=1280, height=720, framerate=15/1, format=NV12' ! nvvidconv flip-method=0 ! nvv4l2h264enc ! h264parse ! video/x-h264,stream-format=avc,alignment=au,profile=baseline ! kvssink stream-name=your-Stream-name storage-size=512 access-key=your-Access-key secret-key=your-Secret-key aws-region=your Aws-region`

4. 스피커 단속 음성 출력

- ALSA 설치

  `$sudo apt-get install alsa-utils`
- 인코더, 디코더 설치

  `$sudo apt-get install lame mpg123`
- MP3 파일 → WAV 파일

  `$mpg123 -w output.wav input.mp3`
- 5초 간격 무한 재생

  `$while true; do aplay woman.wav; sleep 5; done`

5. 배터리 잔량 및 최초 주행 시간 전송

- 실행

  `$python3 battery_to_firebase.py`

6. SLAM 맵 생성 및 자율주행 실행

- Bringup

  `$ros2 launch turtlebot3_bringup robot.launch.py`

- 맵 생성

  `$ros2 launch turtlebot3_cartographer cartographer.launch.py`

- 수동 조작

  `$ros2 run turtlebot3_teleop teleop_keyboard`

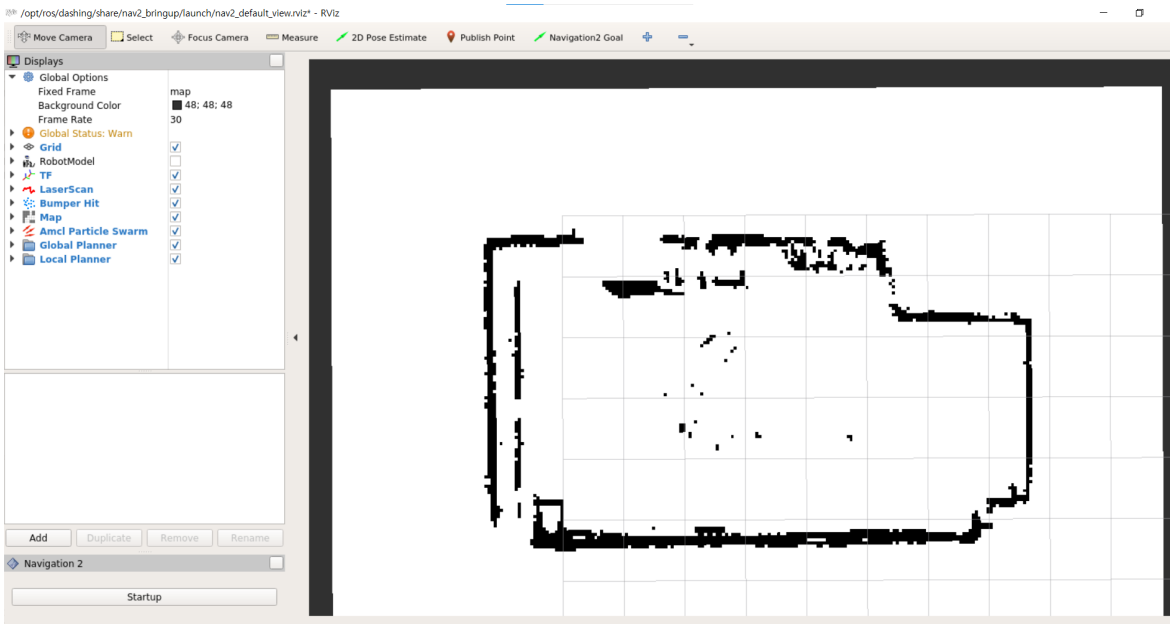
- 맵 저장

  `$ros2 run nav2_map_server map_saver -f ~/map`

- 네비게이션 실행

  `$ros2 launch turtlebot3_navigation2 navigation2.launch.py map:=$HOME/map.yaml`

  - 2D Pose Estimate 를 통해 현 위치의 자세를 표현한다.
  - Navigation2 Goal 를 통해 도착하고자하는 위치를 화살표로 표현하면 터틀봇의 자율주행이 시작된다.



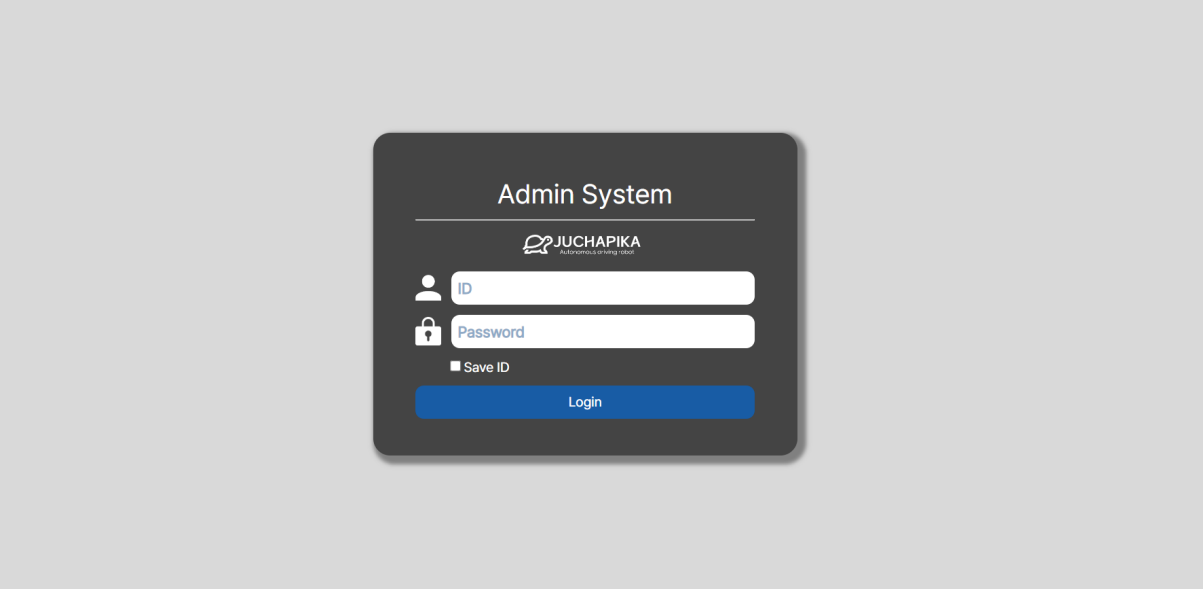
---

7. alias

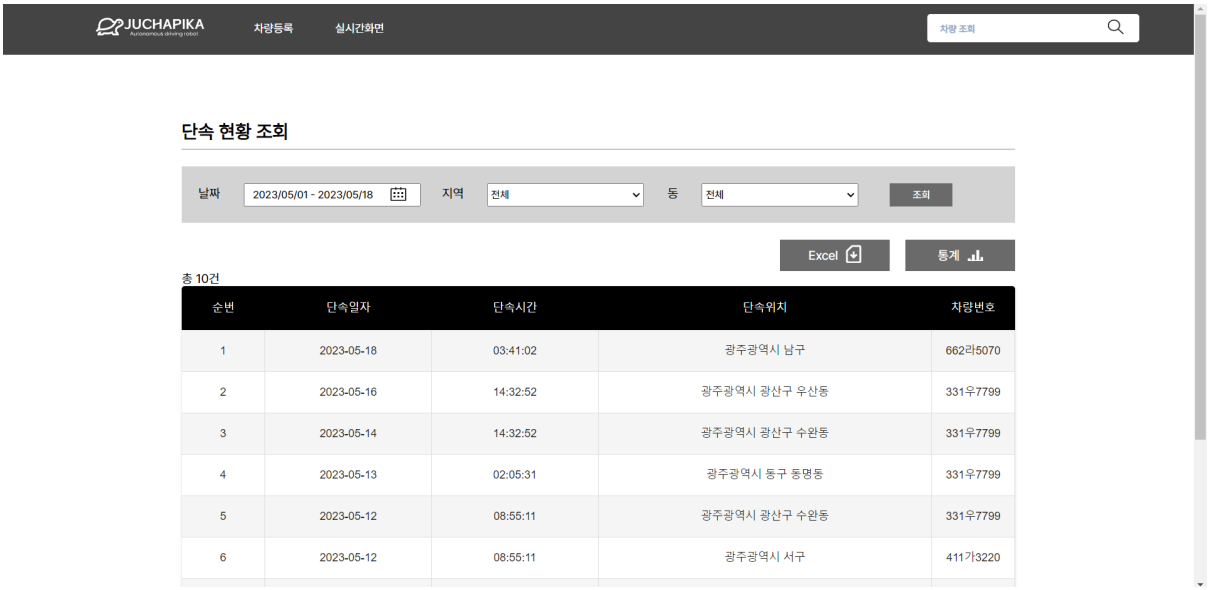
```
alias ki="gst-launch-1.0 nvarguscamerasrc sensor_id=0 ! 'video/x-raw(memory:NVMM),width=1280, height=720, framerate=15/1, format=NV12' ! nvvidconv flip-method=0 ! nvv4l2h264enc ! h264parse ! video/x-h264,stream-format=avc,alignment=au,profile=baseline ! kvssink stream-name=juchapika-stream storage-size=512 access-key=AKIAQVY7NV32673 4VFPX secret-key=V7g7mB5n9GZUAlKXoFqjApIFTcFJht01+x/IC6Zj aws-region=ap-northeast-2"

alias sound_play="while true; do aplay woman.wav; sleep 5; done"

alias bring_up="ros2 launch turtlebot3_bringup robot.launch.py"

alias car_up="ros2 launch turtlebot3_cartographer cartographer.launch.py"

alias tel_up="ros2 run turtlebot3_teleop teleop_keyboard"

alias map_up="ros2 run nav2_map_server map_saver -f ~/map"

alias nav_up="ros2 launch turtlebot3_navigation2 navigation2.launch.py map:=$HOME/map.yaml"
```

3️⃣ 시나리오

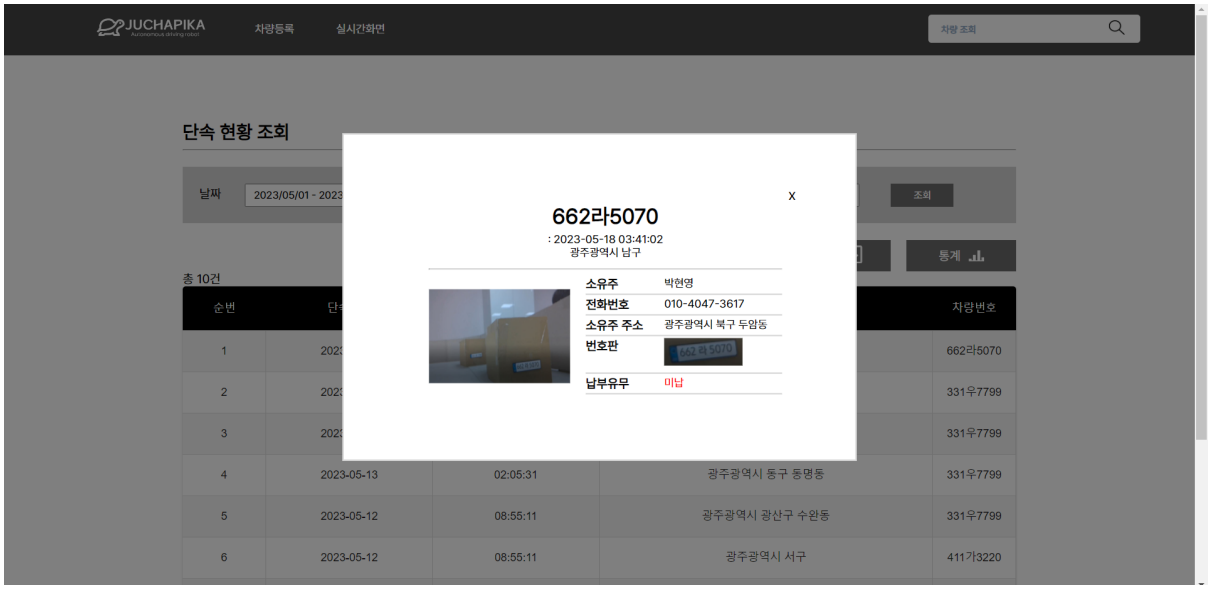
A. 메인 화면



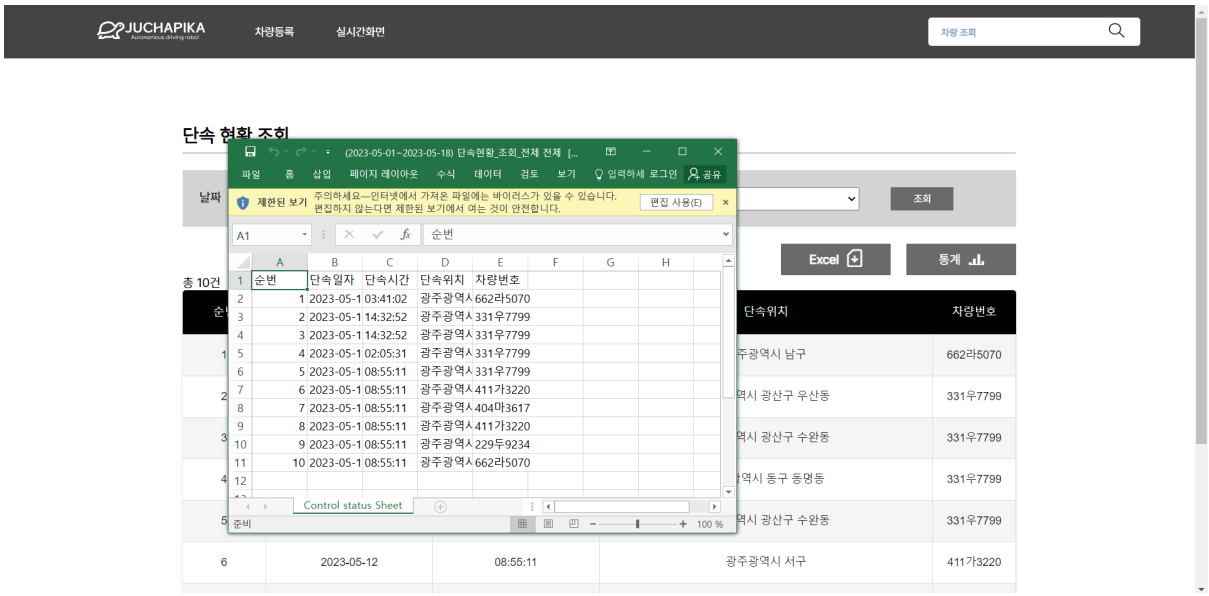
로그인 페이지 (전달받은 아이디/비밀번호로 로그인)



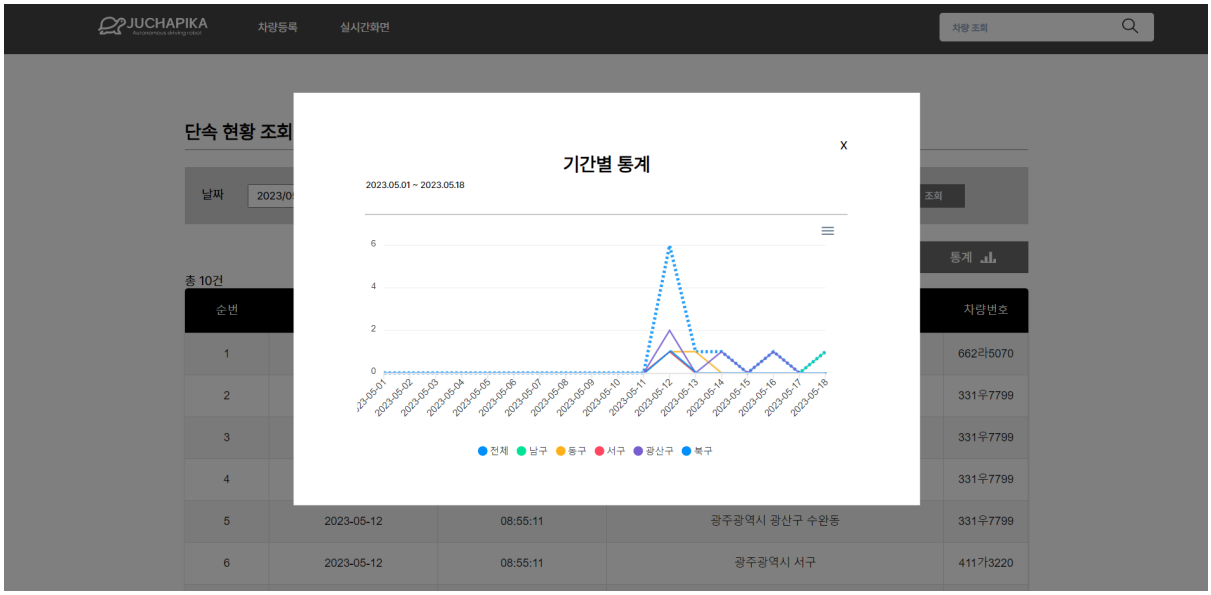
메인페이지



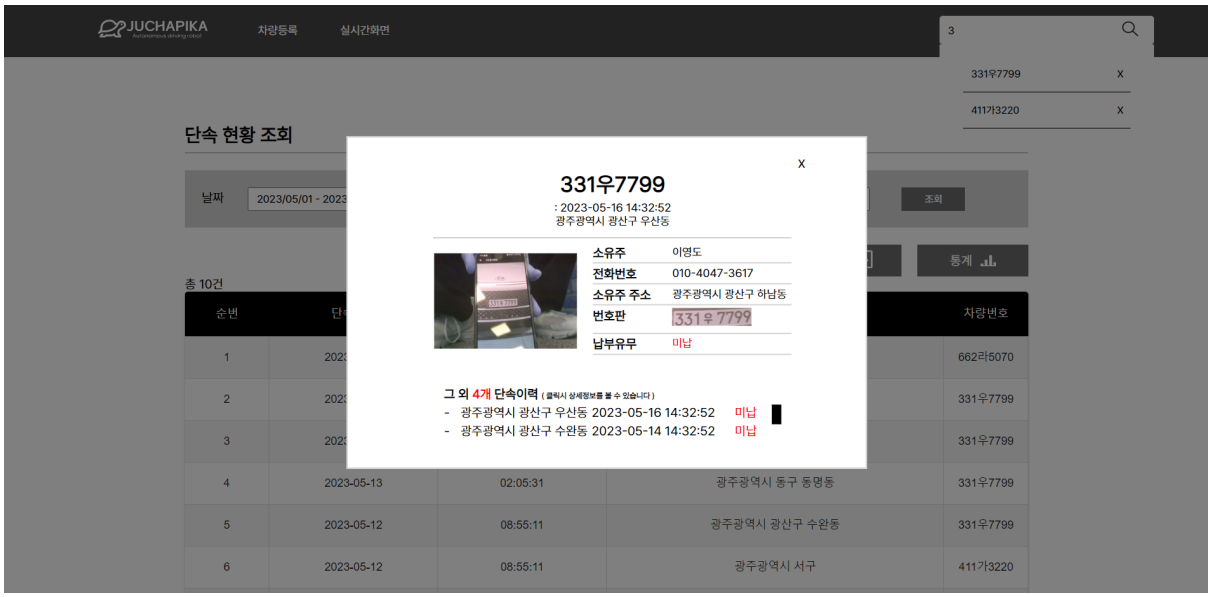
테이블 상세정보 모달기능



테이블 엑셀 다운

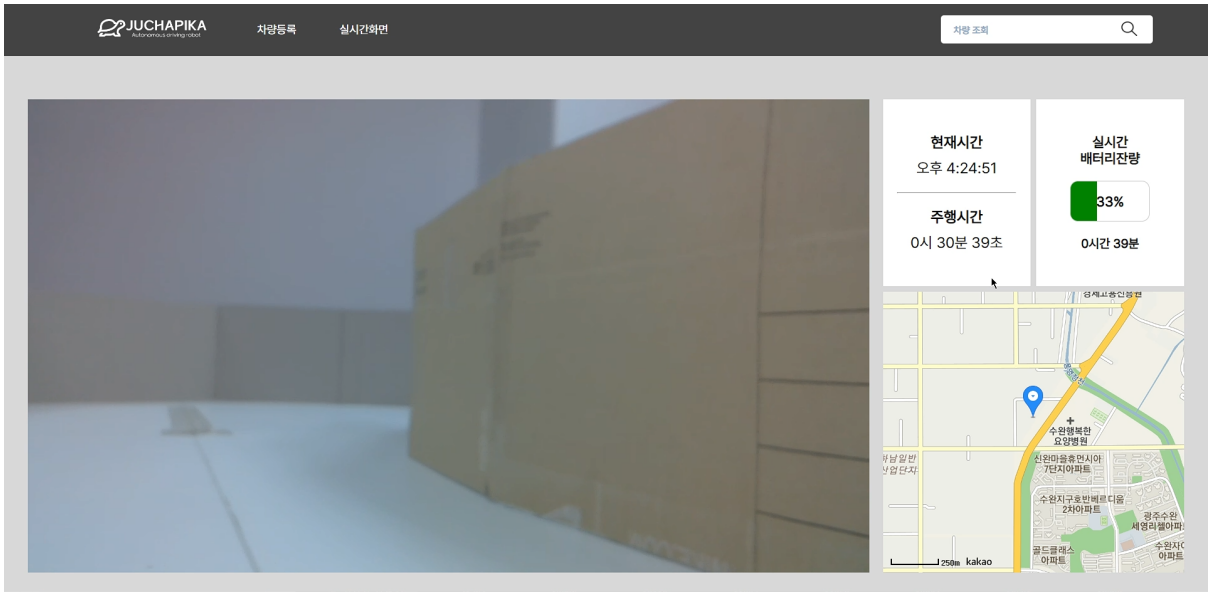


통계 차트 기능



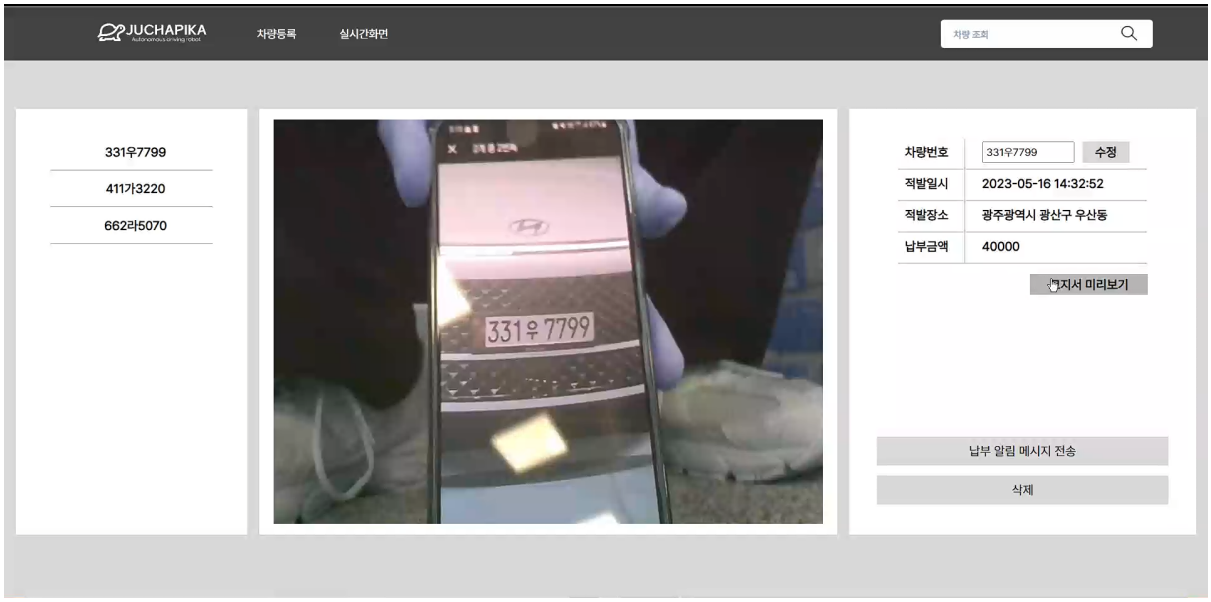
검색창기능

B. 실시간 화면



터틀봇의 실시간 화면 확인 기능 및 배터리 잔량 확인/총 주행시간/현재위치

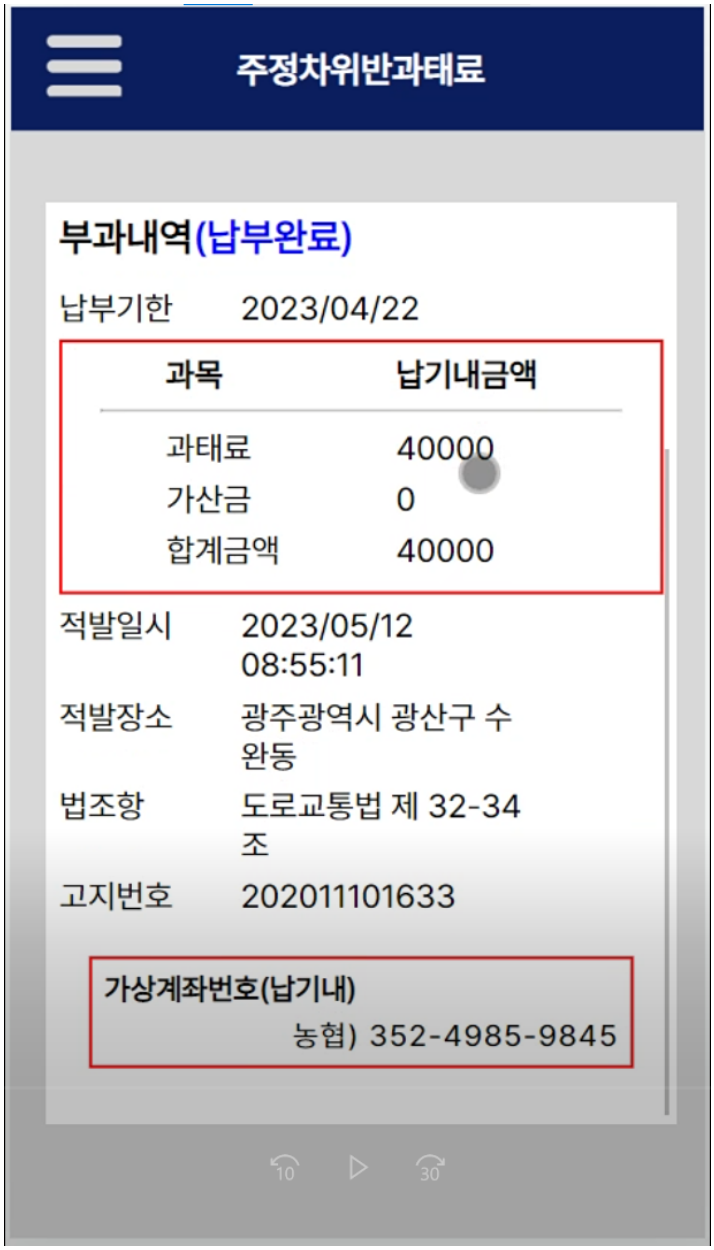
C. 차량 등록



수정/삭제 및 알람메세지 전송


주정차위반과태료
331우7799
차량소유자: 이영도
주정차위반과태료
납부이력
고지서 확인완료
납세자 정보
납세자 이영도
전화번호 010-4047-3617
주소 광주광역시 광산구 하남동
번호판
331우7799
부과내역 미납
납부기한 2023/04/26
과목 납기내금액
과태료 40000
가산금 0
합계금액 40000
적발일시 2023/05/16 14:32:52
적발장소 광주광역시 광산구 우산동
법조항 도로교통법 제32-34조
고지번호 202011101633
가상계좌번호(납기내) 농협) 352-4985-9845

고지서 미리보기


주정차위반과태료
부과내역(납부완료)
납부기한 2023/04/22
과목 납기내금액
과태료 40000
가산금 0
합계금액 40000
적발일시 2023/05/12 08:55:11
적발장소 광주광역시 광산구 수완동
법조항 도로교통법 제 32-34조
고지번호 202011101633
가상계좌번호(납기내)
농협) 352-4985-9845

문자메세지 내용