와따렌트 포팅 매뉴얼

1️⃣ 프로젝트 기술 스택

A. FE

- 기술 스택 (버전)
  - React.js 18
- 사용 툴 (버전)
  - Visual Studio Code 1.74.2

B. EMB

- 기술 스택 (버전)
  - Python3 `3.6.9`
  - Morai Simulator `22.R4.1`
  - ROS `1.14.13`
- 사용 툴 (버전)
  - Oracle VM VirtualBox `7.0.6`
  - Visual Studio Code `1.77.0`

C. 기타

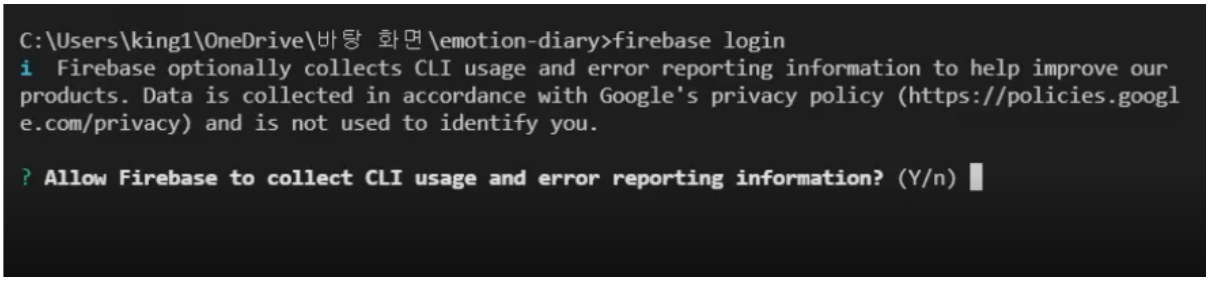
- 형상 관리: GitLab
- 이슈 관리: Jira

- 커뮤니케이션: Notion, Mattermost
- 디자인: Figma, Wondershare Filmora, PowerPoint 2016
- 데이터베이스: Firebase Realtime DB
- 배포: Firebase Hosting

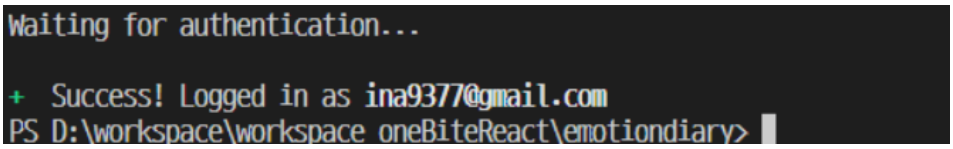
2️⃣ 빌드 방법

A. 프론트엔드 빌드 방법

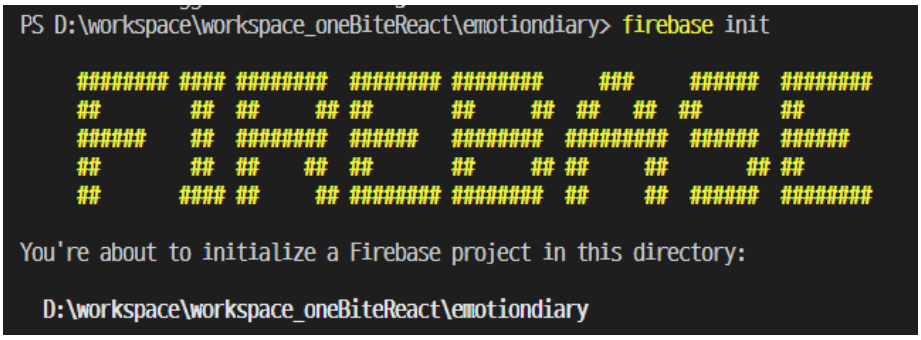
1. `npm install -g firebase=tools`
2. `firebase login`

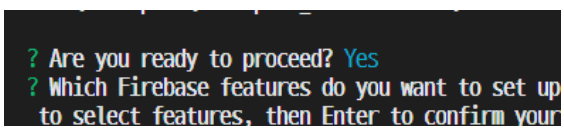


3. y입력 후 엔터
4. 구글 로그인창 오픈 후 해당되는 아이디 클릭후 권한 허용



5. 로그인 완료
6. `firebase init`





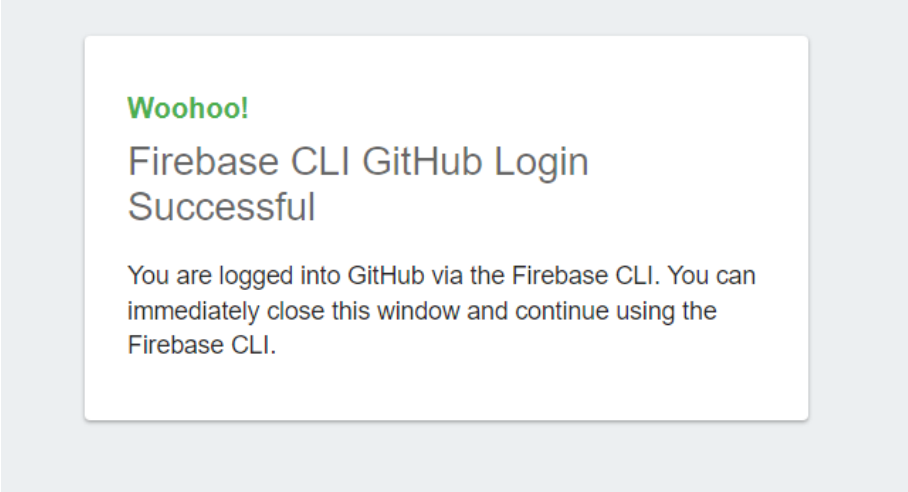
```
 ( ) Firestore: Configure security rules and indexes files for Firestore
 ( ) Functions: Configure a Cloud Functions directory and its files
>(*) Hosting: Configure files for Firebase Hosting and (optionally) set up GitH
ub Action deploys
 ( ) Hosting: Set up GitHub Action deploys
 ( ) Storage: Configure a security rules file for Cloud Storage
```

```
? Please select an option: (Use arrow keys)
> Use an existing project
  Create a new project
  Add Firebase to an existing Google Cloud Platform project
  Don't set up a default project
```

```
? Select a default Firebase project for this directory: (Use arrow keys)
> react-diary-project-fc482 (react-diary-project)
```

!! *중요 !!*

```
? What do you want to use as your public directory? build
```

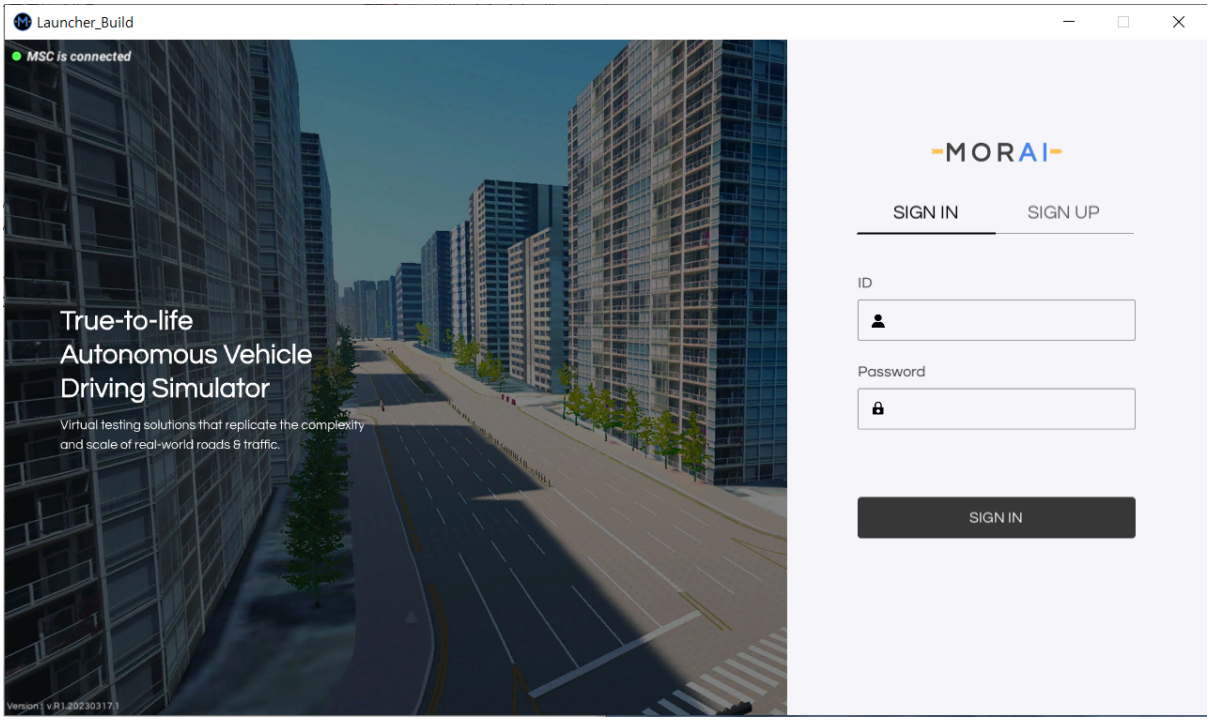


7. `firebase deploy`
8. docker 컨테이너 생성
   `doc`

3️⃣ EMB 매뉴얼

A. Morai Simulator 설치

MoraiLauncher_Win.zip 압축 해제 후 MoraiLancher_Win.exe 파일 실행

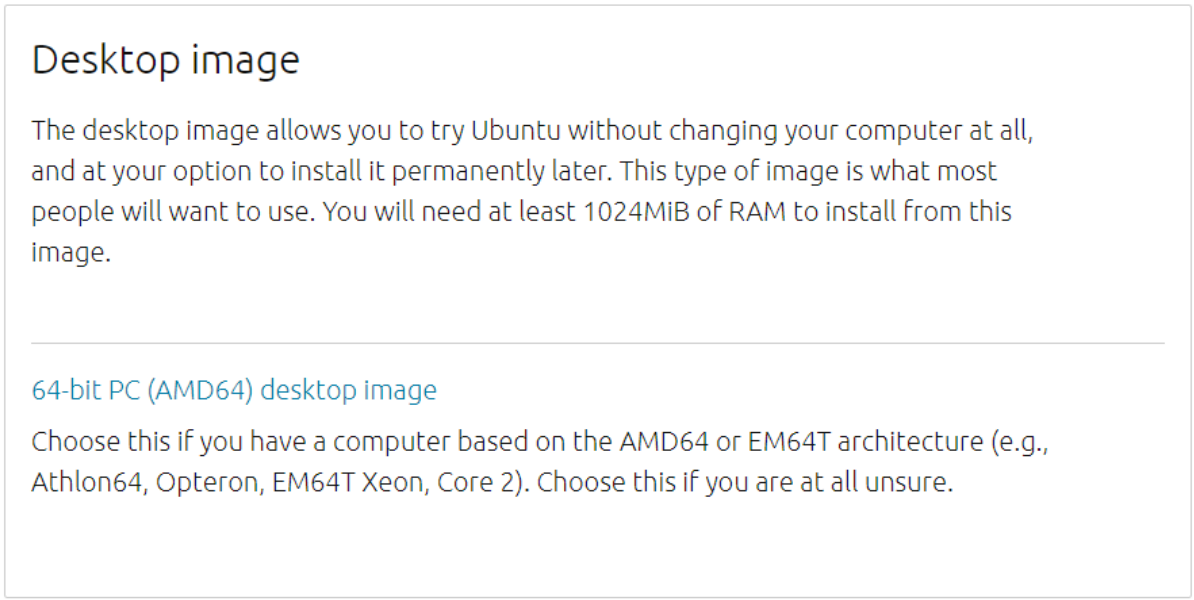


B. VirtualBox 설치

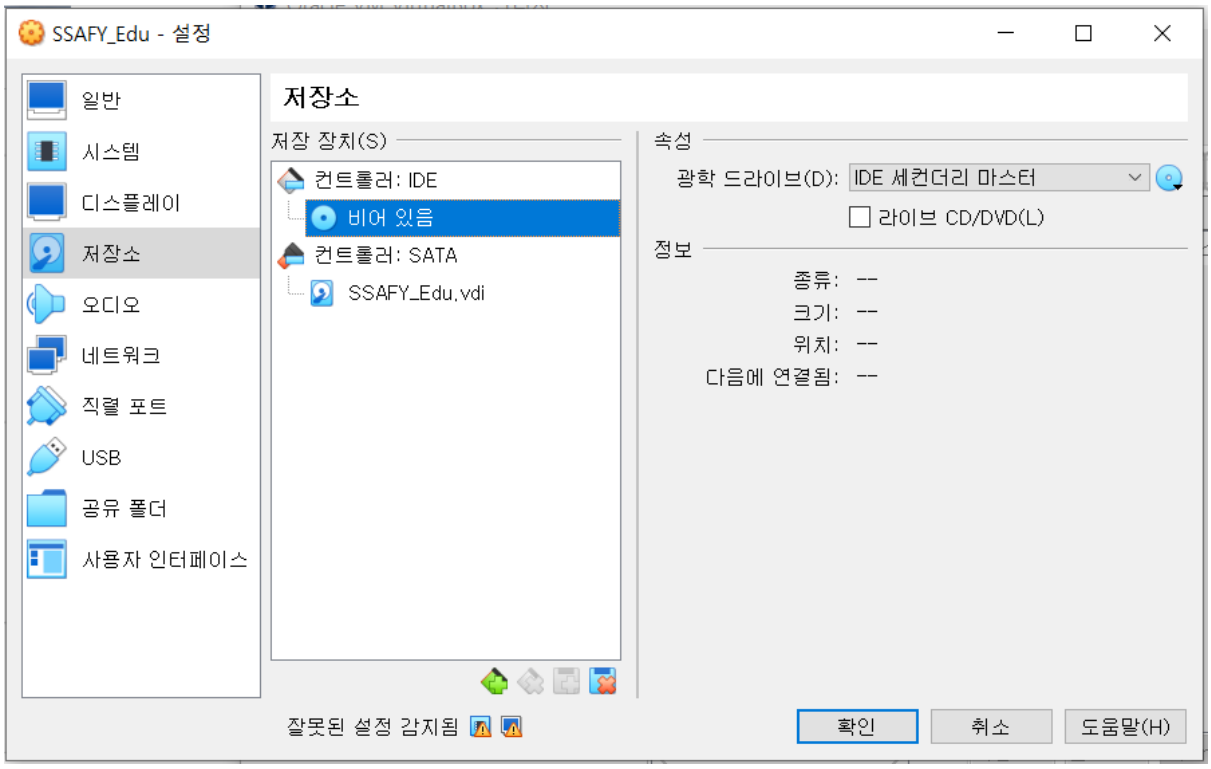
https://www.virtualbox.org/wiki/Downloads 에서 최신 버전 Windows hosts 설치

C. Ubuntu 20.04 설치

1. **https://releases.ubuntu.com/20.04/ 에서 64-bit PC (AMD64) desktop imgae 다운로드**



2. **VMWare > 설정 > 저장소 > 컨트롤러 IDE > 광학 드라이브에 Ubuntu 20.04 iso 설치**



D. ROS 설치

1-1. ROS 기본 설정

```
$ sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
$ sudo apt install curl
$ curl -s https://raw.githubusercontent.com/ros/rosdistro/master/ros.asc | sudo apt-key add -
$ sudo apt update
$ sudo apt install ros-melodic-desktop-full
$ sudo apt-get install python-rosdep
$ sudo rosdep init
$ rosdep update
$ echo "source /opt/ros/melodic/setup.bash" >> ~/.bashrc
```

```
$ source ~/.bashrc
$ sudo apt install python-rosinstall python-rosinstall-generator pythonwstool build-essential
```

1-2. ROS 설치 확인

```
$ roscore
```

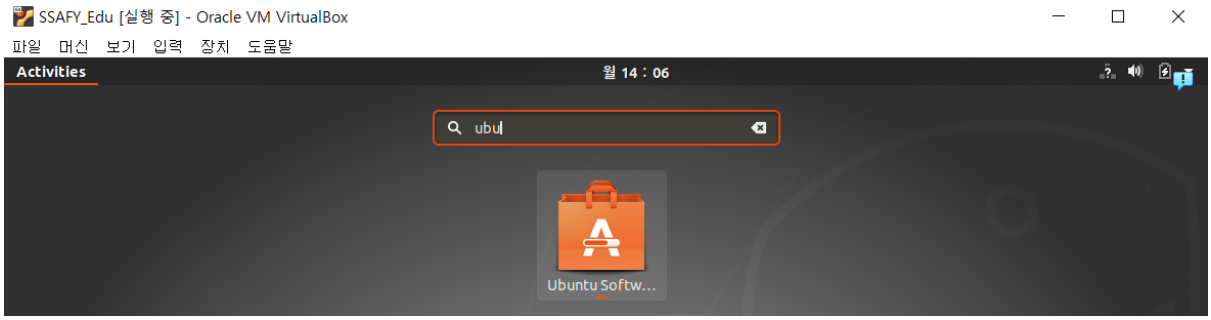
1-3. Morai msg 파일 다운

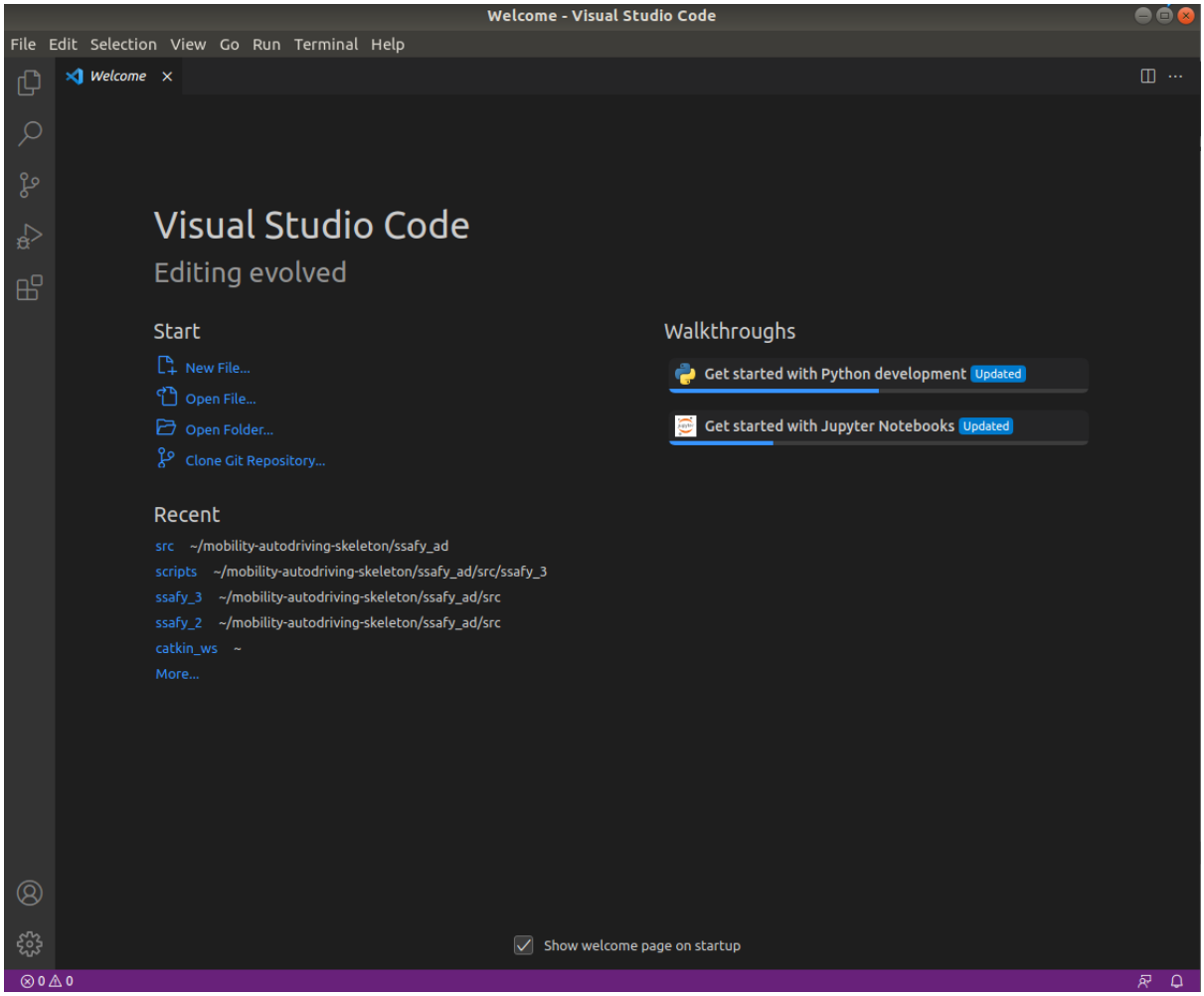
```
$ cd ~/catkin_ws
$ catkin_make
$ cd src
$ catkin_create_pkg ssafy_ad rospy std_msgs
$ cd ~/catkin_ws
$ catkin_make
$ source ~/catkin_ws/devel/setup.bash
$ rospack profile

$ sudo apt-get install git
$ cd ~/catkin_ws/src
$ git clone https://github.com/morai-developergroup/morai_msgs.git
$ cd ~/catkin_ws
$ catkin_make
```

1-4. 종속패키지 및 라이브러리 설치

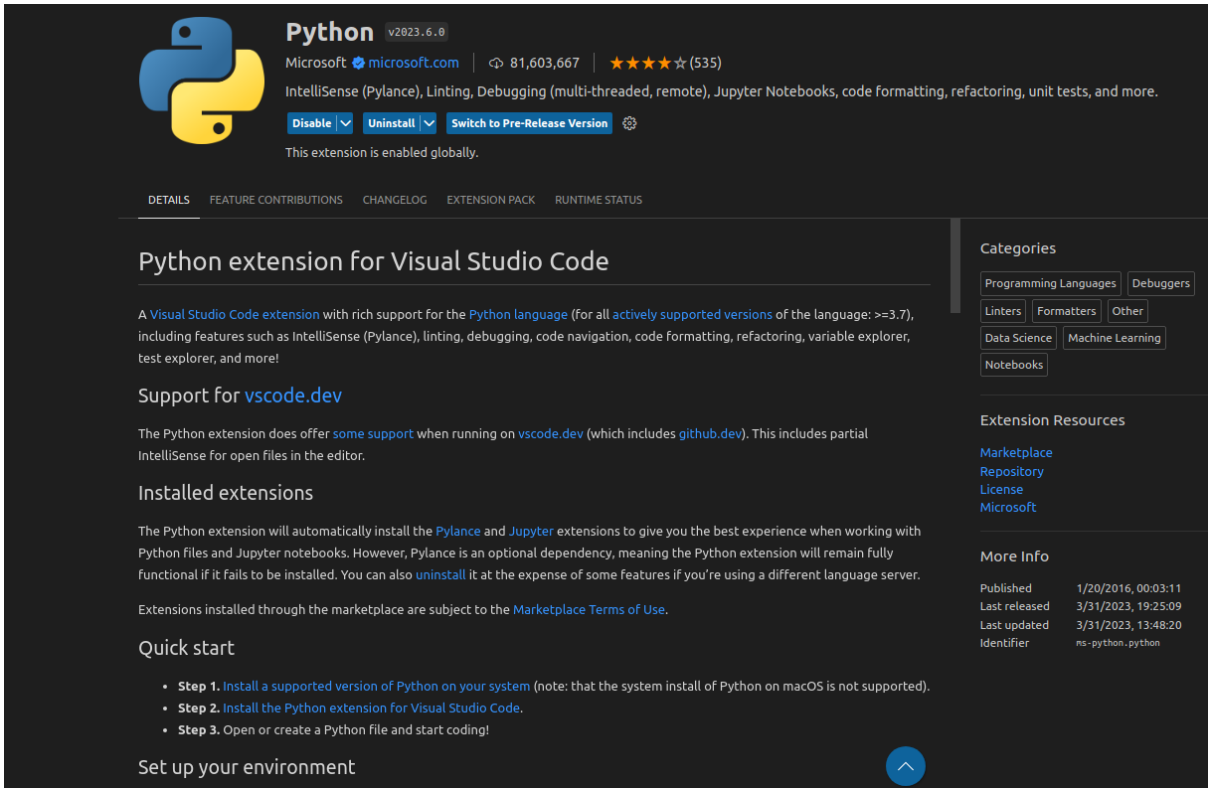
```
$ sudo apt-get install python-pip
$ sudo apt-get install net-tools
$ sudo apt-get install ros-melodic-rosbridge-server
$ sudo apt-get install ros-melodic-velodyne
$ sudo apt install terminator
$ sudo apt install libvulkan1
$ pip install pyproj
$ pip install scikit-learn
$ cd ~/catkin_ws
$ catkin_make
```

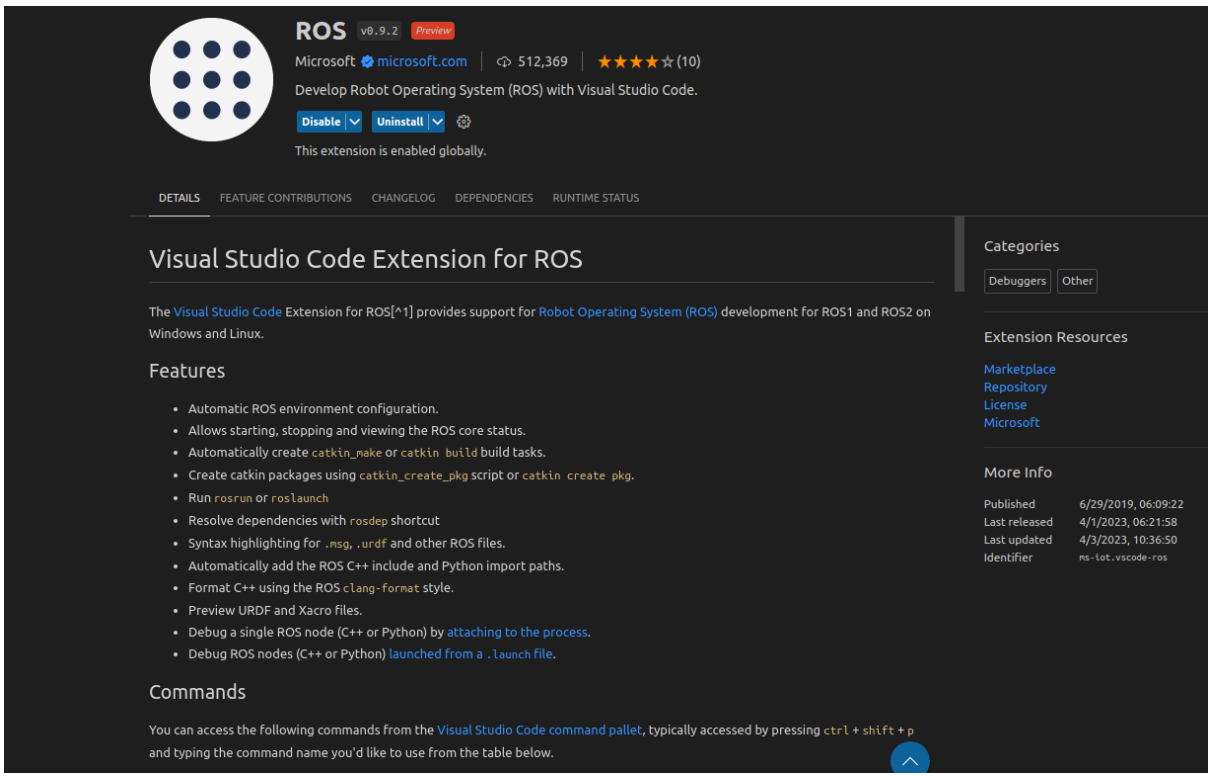
2-1. Visual Studio Code 설치





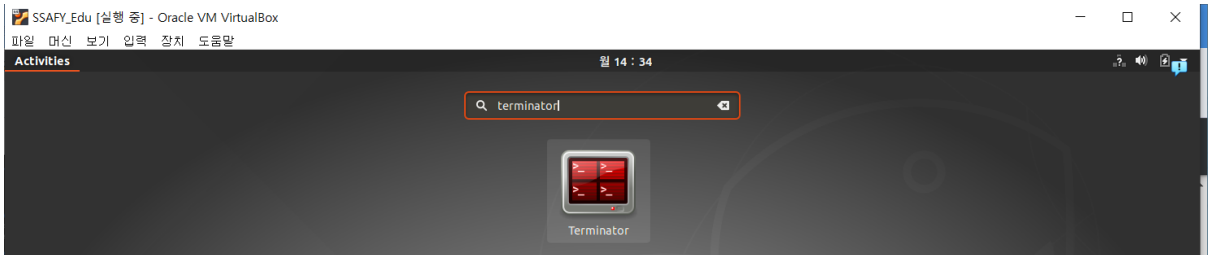
2-2. Visual Studio Code Plugin 설치





3-1. Terminator 설치

```
$ sudo apt install terminator
```



E. 실행

```
$ cd
$ python main.py

# /SelectCar의 선택하기 버튼(DB : Reservation->choice_btn->choice_btn) 누를 시 실행되는 명령어
$ roslaunch watta_dir watta.launch

# /SelectCar의 예약확인 버튼(DB : Reservation->reservation_btn->reservation_btn) 누를 시 실행되는 명령어
$ rosrun watta_dir autodriving.py

# /CompleteCar의 운행종료 모달의 확인 버튼(DB : Reservation->return_btn->return_btn) 누를 시 실행되는 명령어
$ roslaunch watta_dir watta.launch
$ rosrun watta_dir autodriving.py
```

4️⃣ 데이터 통신

A. FE to DB

1. 사용자 현재 좌표
2. 목적지 좌표
3. 차량 call 기능 (flag 값)
4. 차량 도착 기능 (flag 값)

B. EMB to DB

1. GPS기반 Ego차량 위치 좌표
2. 목적지 도착 알림 기능 (flag 값)
3. 목적지와의 거리값

C. DB to FE

1. 2-1값을 받아와 web 상에서 차량이 이동하는 경로를 map에 표시
2. 2-2값을 통해 도착 완료 표기

D. DB to EMB

1. 1-3값을 통해 차량 출발
   case a) 해당 값이 true, 목적지 좌표가 가능할 때
   case b) 해당 값이 true, 목적지 좌표가 불가능할 때
   case c) 해당 값이 false일 때

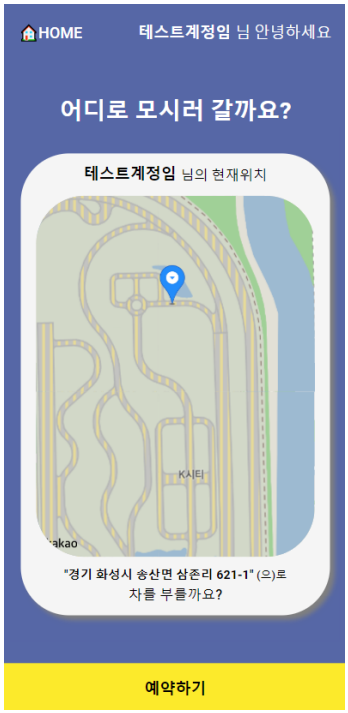
2. 1-1, 1-2값을 바탕으로 최단경로로 이동(Dijkstra)
3. 목적지 부근 도착시 도착 알림 flag 전송
4. 사용자가 운행종료 버튼 누르면 1-4값 확인해서 차고지(start point 고정값)로 다시 이동

5️⃣시나리오

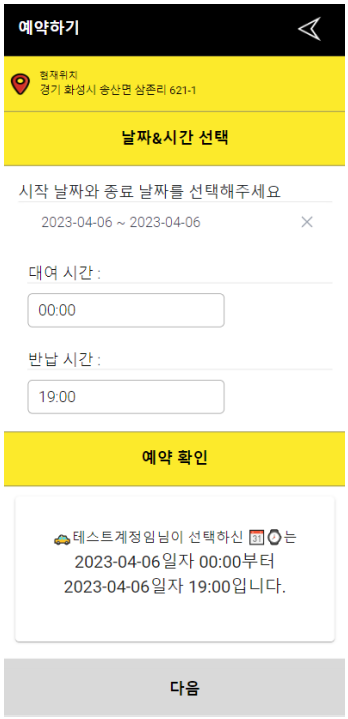
A. 홈 화면

- 시작화면이다.
- 화면을 누르면 로그인 화면으로 넘어간다.

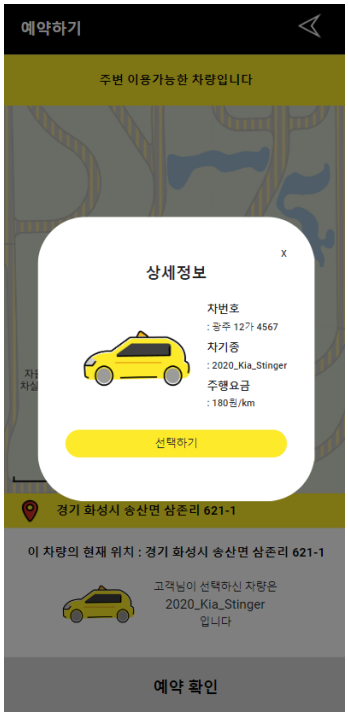
B. 메인 화면



- 차량 픽업받을 장소를 설정하는 페이지이다.



- 예약 날짜 시간을 설정하는 페이지이다.

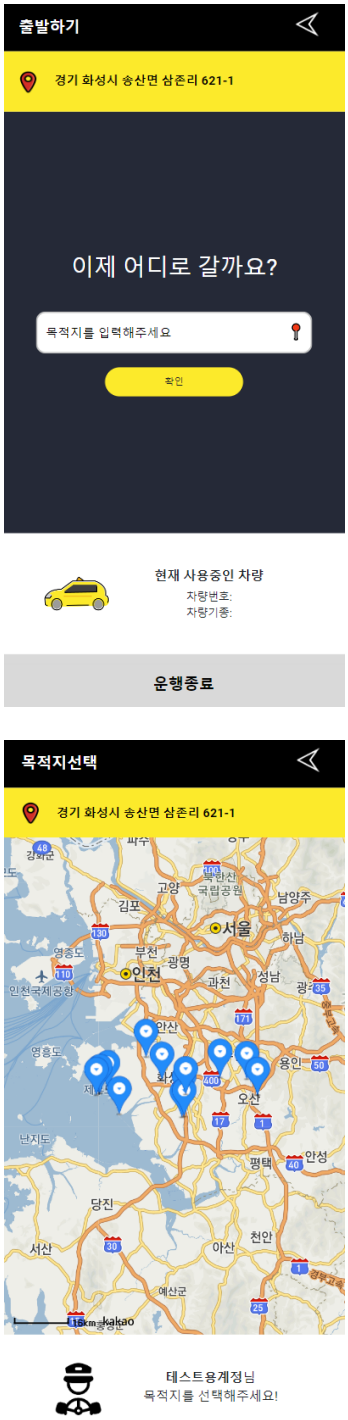


- 차량 선택 화면이다

C. 차량 픽업 화면

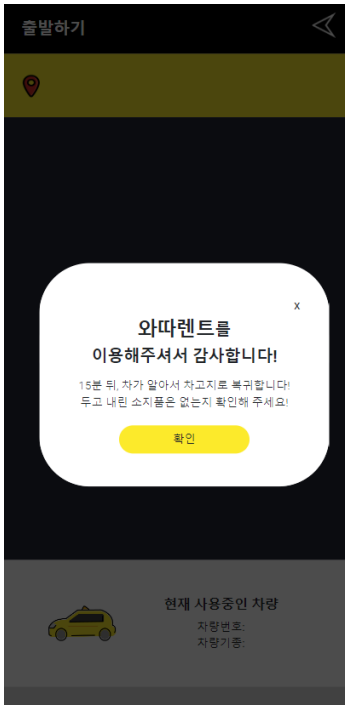
- 차량 픽업 화면이다.

D. 목적지 설정 화면



- 주소를 검색하고 마커를 선택해 목적지를 설정한다.

- 목적지로 이동하는 차량의 모습이다



- 반납 버튼을 누른 모습이다.