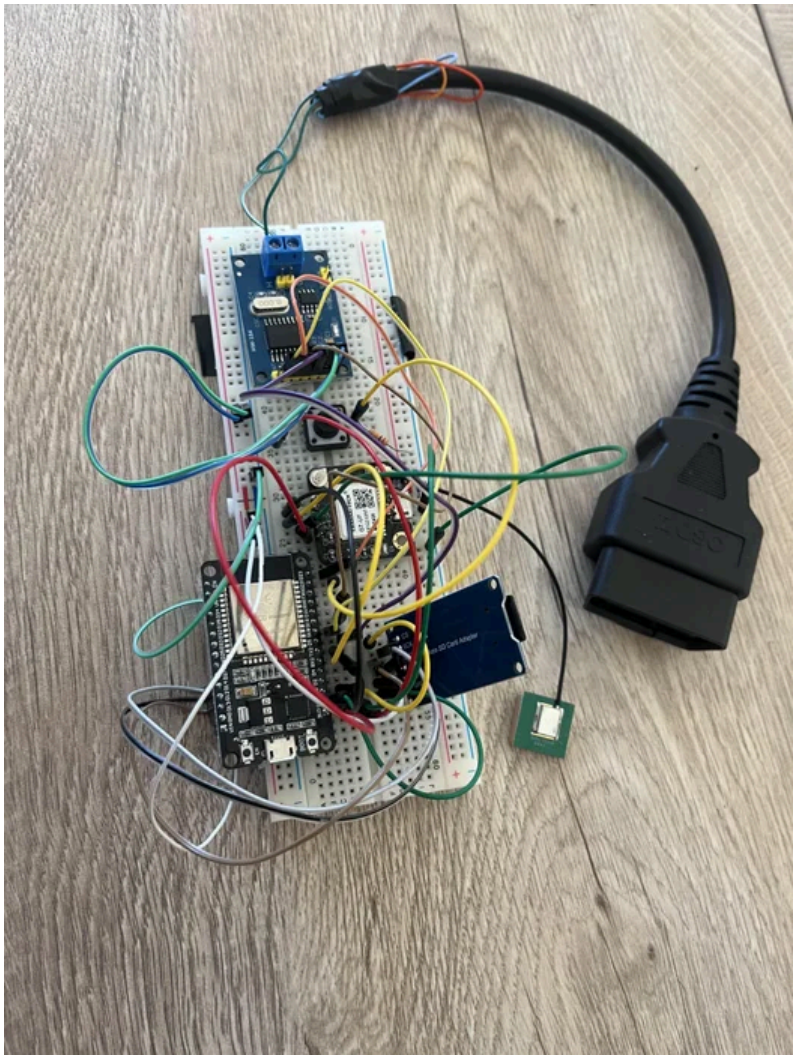
AUTODESK Instructables

ESP32 차량 텔레매틱스 - OBD2(CAN) 및 GPS 보고 시스템

bkennar2 의 회로 마이크로 컨트롤러

소개: ESP32 차량 텔레매틱스 - OBD2(CAN) 및 GPS 보고 시스템



이 프로젝트의 코드는 https://github.com/bkennar2/vehicle-telematics에서 찾을 수 있습니다.

이 프로젝트의 목적은 차량을 추적하고 OBD2 데이터를 읽을 수 있는 장치를 만드는 것입니다. Samsara와 Verizon과 같은 회사는 이와 같은 장치와 모니터링 서비스를 차량에 판매합니다. 모든 신형 차량(일부 EV 제외)에는 OBD2 포트가 있습니다. OBD2 포트를 사용하면 CAN을 사용하여 차량에서 직접 데이터를 가져올 수 있습니다. OBD2에 대해 자세히 알아보려면 여기를 읽어보세요:

데이터 처리가 Python Flask로 구축되고 라즈베리 파이에 호스팅된 API에 의해 수행됩니다. 그런 다음 해당 데이터는 동일한 API에 연결하여 여행 데이터를 가져오는 웹앱에서 검색할 수 있습니다. Google Maps API를 사용하여 여행을 보여주는 시각화를 만듭니다.

작동 원리:

ESP32는 OBD2 브레이크아웃 케이블을 통해 차량에 연결됩니다. 현재 USB를 사용하여 설정에 전원을 공급합니다. 매초 시스템은 GPS 데이터를 읽고, CAN 메시지를 차량으로 전송한 다음 응답을 읽습니다. 그런 다음 해당 데이터는 carData.txt라는 파일에 저장됩니다. 사용자가 브레드보드의 버튼을 누르거나 코드가 현재 GPS 좌표가 사전 프로그래밍된 홈 좌표에서 30m 이내에 있음을 감지하면 홈 WiFi 네트워크에 연결하여 데이터를 Raspberry Pi 서버에 업로드하려고 시도합니다. 참고: 홈에서 시작하면 30m 밖으로 나갈 때까지 업로드를 시도하지 않습니다.

시스템이 켜지면 새 세션이 carData.txt 파일에 "&"로 기록됩니다. ESP32가 API에 업로드할 데이터를 준비할 때 이를 통해 세션을 구분할 수 있습니다. 세션의 고유 식별자는 VIN과 세션의 첫 번째 타임스탬프입니다. 데이터는 carData/[VIN]/[start_date] 위치에 있는 라즈베리파이의 csv 파일에 저장됩니다. ESP32의 메모리 양은 제한적입니다. 따라서 업로드를 위해 세션을 청크로 나눕니다. 청크가 세션의 첫 번째인 경우 VIN과 시작 날짜만 있는 행을 PostgreSQL 데이터베이스에 추가하고 csv 파일을 만듭니다. 첫 번째 청크가 아닌 경우 이전에 만든 csv에 추가합니다.

우리는 하루에 한 번 실행되는 파이썬 스크립트를 가지고 있는데, 이 스크립트는 종료 날짜가 없는 데이터베이스의 모든 행을 찾습니다. 그런 다음 csv 파일을 처리하여 DB의 메타데이터의 나머지 열을 채웁니다. 이 프로그램은 세션의 종료 날짜, 유휴 시간(0km/h에서 소요된 초), 최대 속도, 여행의 평균 연비를 가져옵니다. 여기서 처리하고 추적할 통찰력을 추가할 수 있습니다.

API를 사용하면 데이터를 검색할 수도 있습니다. 저는 이 데이터를 가져와 Google Maps API를 사용하여 지도에 여행을 표시하는 매우 간단한 Flask 앱을 만들었습니다.

프로젝트 개선 사항:

1) 회로는 차량의 OBD2 포트에서 12V로 구동될 수 있습니다. 12V를 5V로 낮추려면 DC-DC 컨버터가 필요합니다. 그러나 많은 차량은 항상 OBD2에서 12V를 켭니다. 차량의 배터리를 소모하지 않으려면 ESP32를 절전 모드로 전환할 방법이 필요합니다. 일부 OBD2 포트에는 "점화" 핀이 있습니다. 12V까지 올라가면 절전 모드에서 나올 수 있습니다.

2) 라즈베리파이를 사용하는 대신 API와 스토리지를 클라우드에서 실행할 수 있습니다. API에 보안을 추가해야 합니다.

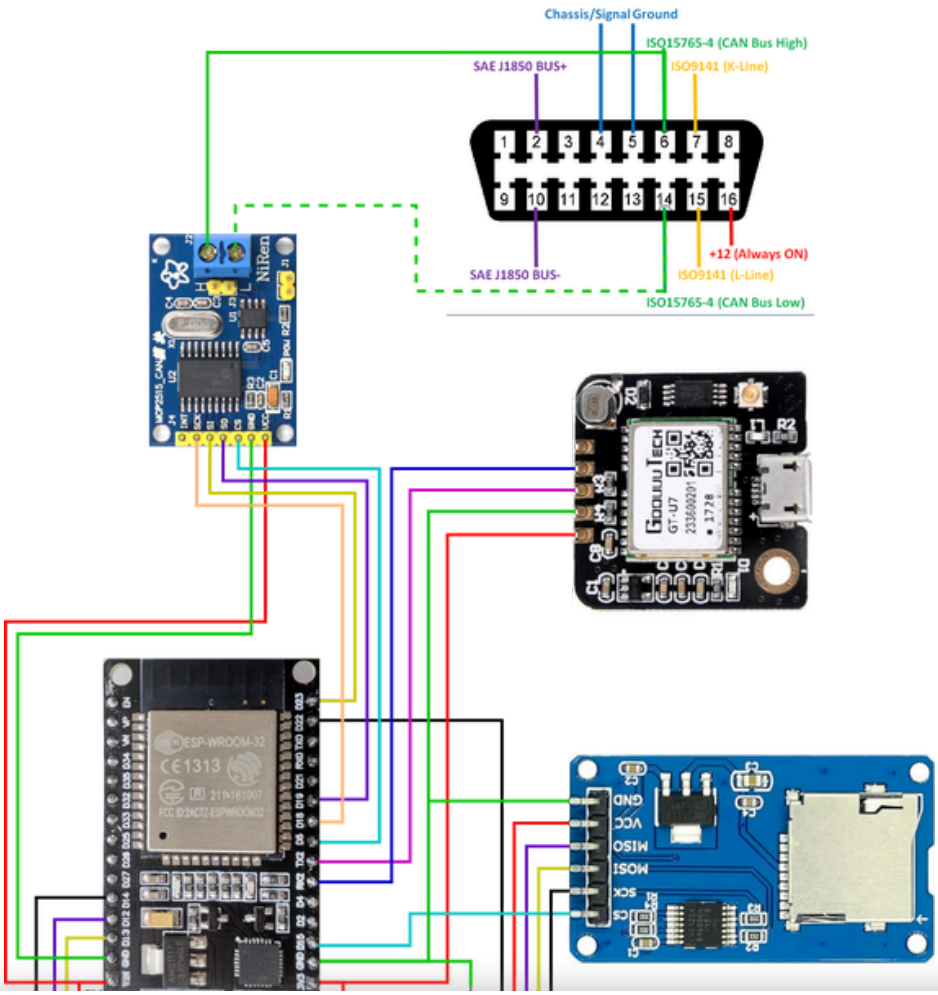
3) 각 모듈의 부품을 OBD2 포트에 직접 꽂는 단일 보드로 통합할 수 있습니다. 이렇게 하면 전체 프로젝트가 훨씬 더 깔끔해질 것입니다.

4) ESP32 코드에 설정에서 차량의 VIN을 검색하는 섹션을 추가할 수 있습니다.

용품

1. ESP32 DevKitV1 보드: https://www.amazon.com/Development-Microcontroller-Integrated-Antenna-Amplifiers/dp/B09GK74F7N
2. CAN 모듈: https://www.amazon.com/HiLetgo-MCP2515-TJA1050-Receiver-Arduino/dp/B01D0WSEWU
3. GPS 모듈: https://www.amazon.com/Navigation-Positioning-Microcontroller-Compatible-Sensitivity/dp/B084MK8BS2
4. SD 모듈: https://www.amazon.com/HiLetgo-Adater-Interface-Conversion-Arduino/dp/B07BJ2P6X6
5. 마이크로 SD 카드
6. OBD2 브레이크아웃 케이블: 여기
7. 20k 옴 저항기
8. 단추
9. 브레드보드
10. 라즈베리파이

1단계: ESP32 및 모듈 배선



2단계: ESP32 코드 설정.

1. 아두이노 IDE를 설치하다
2. 보드 관리자 아래에 Esspressif Systems의 ESP32를 설치합니다.
3. 라이브러리 관리자에서 설치
4. Ress의 TinyGPSPlusPlus
5. autowp-mcp2515 by autowp
6. 내 github repo의 ESP32 폴더에서 .ino 파일을 열고 저장합니다.
7. 값 변경
8. WiFi 크레딧에 대한 SSID 및 비밀번호
9. VIN 또는 12345로 남겨주세요
10. 서버 IP 주소. raspberrypi.local 또는 pi의 정적 IP 주소를 사용할 수 있습니다.
11. 27번과 28번 줄에는 집이나 차량을 주차할 위치의 좌표를 추가합니다.
12. ESP32가 이 지점에서 30m 이내에 있음을 감지하면 WiFi에 연결을 시도합니다.
13. CAN 모듈에 16Mhz 크리스털이 있는 경우 90번 라인 8을 16으로 변경해야 합니다.

3단계: Raspberry Pi 설정

라즈베리파이를 설정하세요. 헤드리스 설정을 보여주는 좋은 영상이 있습니다 . 즉, 모니터, 키보드 또는 마우스가 필요하지 않습니다.

4단계: PostgreSQL 데이터베이스 설정

여기의 지침 에 따라 다음 조정을 수행할 수 있습니다.

"Raspberry Pi에서 첫 번째 PostgreSQL 데이터베이스 만들기" 섹션 제목에서

"cardata"라는 데이터베이스를 만들고 tripData라는 테이블에 다음 열이 있습니다. 추적하려는 데이터에 따라 열을 추가하거나 제거할 수 있습니다. 이것은 여행에 대한 메타데이터일 뿐입니다. 해당 여행에 대한 요약이나 데이터를 제공합니다.

1. cardata 데이터베이스 생성;
2. tripData 테이블 생성(VIN varchar(17), startTime 타임스탬프, endTime 타임스탬프, idleTime 정수, maxSpeed smallint, fuelEff float(1));

5단계: Pi에 API 설정

1. 홈 방향으로 가세요 "cd ~"
2. "mkdir Projects"를 입력하여 프로젝트 폴더를 만듭니다.
3. 디렉토리를 프로젝트 폴더로 변경하려면 "cd Projects"를 입력하세요.
4. 내 github에서 carDataAPI 폴더를 이 위치로 복사합니다.
5. 프로젝트에 대한 파이썬 가상 환경을 생성하세요
6. 파이썬 -m env /home/pi/Projects/carDataAPI
7. carDataAPI 폴더 "cd carDataAPI"로 이동합니다.
8. 가상 환경 "source env/bin/activate"를 활성화합니다.
9. 명령줄 옆에 (env)가 표시되어야 합니다.
10. 요구 사항 파일을 사용하여 필요한 패키지를 설치합니다.
11. "pip install -r requirements.txt"
12. 14, 15번째 줄의 사용자 이름과 비밀번호를 변경하세요.
13. python app.py를 사용하여 프로그램이 실행되는지 확인하세요.
14. "chmod 777 start_flask.sh app.py"를 실행하세요
15. 이렇게 하면 파일의 권한이 변경되어 crontab에서 실행할 수 있습니다.
16. "./start_flask.sh"로 프로그램을 실행하여 제대로 작동하는지 확인하세요.

6단계: Pi 자동 설정 API 시작

Pi가 켜질 때마다 API가 시작되기를 원합니다. 따라서 crontab에 줄을 추가해야 합니다. crontab을 사용하면 프로그램을 자동으로 실행할 수 있습니다. 아래 단계에 따라 crontab에 API를 설정하세요.

1. crontab을 편집하려면 "crontab -e"를 실행하세요.
2. 아래 줄을 crontab에 추가합니다. 이 줄은 재부팅 시 start_flask.sh를 실행하고 출력을 Flask.log로 전송한다는 것을 의미합니다.
3. @reboot /home/pi/Projects/carDataAPI/start_flask.sh >> /home/pi/Projects/carDataAPI/flask.log 2>&1
4. pi를 재부팅하고 Flask 앱이 시작되는지 확인하세요. (sudo reboot now)
5. 어떤 파이썬 파일이 실행 중인지 보려면 "sudo ps -aux | grep python"을 실행하면 됩니다.

7단계: 설정 테스트

1. USB를 사용하여 ESP32를 노트북에 연결하세요
2. Arduino IDE에서는 직렬 모니터에 GPS 데이터가 표시되어야 합니다.
3. 30초 동안 실행한 다음 USB 연결을 끊습니다.
4. 모듈에서 SD 카드를 꺼내 노트북에 넣으세요.
5. 데이터가 저장되고 있는지 확인하세요.
6. 날짜, 시간, 경도, 위도, 고도만 표시됩니다.
7. 나머지는 차량에 연결되어 있지 않기 때문에 쉼표로만 표시됩니다.
8. SD 카드를 모듈에 다시 넣으세요
9. ESP32를 노트북에 연결하세요
10. 프로그램을 30초 더 실행하세요
11. 브레드보드의 버튼을 누르세요
12. ESP32는 API를 통해 라즈베리파이 서버에 데이터를 업로드하려고 시도해야 합니다.
13. 직렬 모니터를 사용하는 경우 "성공"이 표시됩니다.
14. 데이터가 라즈베리파이에 있는지 확인하세요
15. 라즈베리파이에 ssh로 접속
16. carDataAPI 디렉토리로 이동
17. "python readData.py"를 실행하여 메타데이터가 있는지 확인하세요.
18. VIN이 적힌 폴더도 있을 겁니다.
19. 해당 폴더로 이동하여 전체 데이터가 있는지 확인하세요.

8단계: 프로세스 데이터 파일 테스트

1. python processData.py를 실행하여 processData.py 파일을 테스트합니다. 이 파일도 carDataAPI 폴더에 있습니다.
2. 성공하면 readData.py 파일을 실행하면 metaData에 endDate가 표시됩니다.
3. crontab에 작업을 추가하여 processData.py 파일을 시작하는 run_process.sh 파일을 시작합니다.
4. crontab이 파일을 실행할 수 있도록 권한을 변경하려면 "chmod 777 processData.py run_process.sh"를 실행하세요.
5. crontab 업데이트 "crontab -e"
6. 추가: 40 23 * * * /home/pi/Projects/carDataAPI/run_process.sh >> /home/pi/Projects/carDataAPI/process.log 2>&1
7. 이 파일은 매일 오후 11시 40분에 실행되고 출력을 process.log로 보냅니다.

9단계: 시각화 설정

시각화를 실행하기 위해 저는 powershell(Windows) 또는 terminal(mac)을 사용하여 컴퓨터에 Flask 앱을 설정했습니다. 단계는 라즈베리파이에서의 설정과 매우 유사합니다. 이는 컴퓨터에 이미 파이썬이 있다고 가정합니다.

1. "mkdir Projects"를 입력하여 프로젝트 폴더를 만듭니다.
2. 디렉토리를 프로젝트 폴더로 변경하려면 "cd Projects"를 입력하세요.
3. 내 github에서 mapApp 폴더를 노트북의 Projects 폴더로 복사하세요
4. 프로젝트에 대한 파이썬 가상 환경을 생성하세요
5. 파이썬 -m env /mapApp
6. mapApp 폴더 "cd mapApp"으로 이동하세요.
7. 가상 환경 "source env/bin/activate"를 활성화합니다.
8. 명령줄 옆에 (env)가 표시되어야 합니다.
9. 필요한 패키지를 설치하세요
10. pip 설치 -r 요구 사항.txt
11. https://developers.google.com/maps로 이동하여 API 키와 맵 ID를 받으세요.
12. 이 단계를 완료하기 위한 좋은 튜토리얼이 유튜브에 많이 있습니다.
13. index.html 파일의 6번째 줄과 104번째 줄에 각각 API 키와 맵 ID를 추가합니다.
14. 또한 102번째 줄에서 지도의 시작 중심점을 변경할 수도 있습니다.
15. "python app.py"로 앱을 실행합니다.
16. 브라우저에서 localhost:5000으로 이동하여 앱에 액세스하세요.