**공학설계(캡스톤디자인)**

**프로젝트**

|  |  |
| --- | --- |
| 팀 | *자유주제 9조* |
| 팀 명 | *1learnmask (일런마스크)* |
| 문서 제목 | 결과보고서 |

|  |  |
| --- | --- |
| **팀원** | 오 동근 (팀장) |
| 안 찬혁 (팀원) |
| 이 상혁 (팀원) |
|  |
|  |

**목 차**

[1 개요 4](#_Toc43103653)

[2 개발 내용 및 결과물 5](#_Toc43103654)

[2.1 목표 5](#_Toc43103655)

[2.2 개발 내용 및 결과물 6](#_Toc43103656)

[2.2.1 개발 내용 6](#_Toc43103657)

[2.2.2 시스템 구조 및 설계도 6](#_Toc43103658)

[2.2.3 활용/개발된 기술 6](#_Toc43103659)

[2.2.4 현실적 제한 요소 및 그 해결 방안 6](#_Toc43103660)

[2.2.5 결과물 목록 7](#_Toc43103661)

[3 자기평가 8](#_Toc43103662)

[4 참고 문헌 8](#_Toc43103663)

[5 부록 8](#_Toc43103664)

[5.1 사용자 매뉴얼 8](#_Toc43103665)

[5.2 설치 방법 8](#_Toc43103666)

# 개요

|  |
| --- |
| **한국 사람은 코로나 팬데믹 이전에 종종 마스크를 쓰곤 했다. 황사가 심할 때, 미세 먼지나 초미세먼지 때문에 대기 질이 좋지 않을 때, 감기가 심하게 유행할 때에는 마스크를 썼다. 하지만 코로나 팬데믹 동안 마스크를 항상 쓰고 다녔고, 코로나의 위험도가 내려갔다고 판단하는 지금은 마스크를 거의 쓸 생각을 하지못한다. 엄밀히 말하자면 대기 질이 좋지 않을 때도 마스크를 써야 겠다고 생각하지만 데이터를 한번에 보지 못해서, 시간이 없어서 그냥 집에서 나오곤 한다. 그래서 한번에 순식간에 데이터들을 취합하여 보여주고, 취합한 데이터를 토대로 오늘 마스크를 써야 하는 지를 가르쳐주는 프로그램을 만들었다.** |

# 프로젝트 설명

## 전체 프로젝트 구조

|  |
| --- |
| 프로젝트는 백엔드, 프론트, 하드웨어로 이루어져 있다.   * 백엔드 : 백엔드에선 한 시간에 한 번 데이터들을 가져온다. 데이터를 가져오는 방법은 아래 ‘3.2 데이터 확보 방법’에서 설명하기로 한다. 데이터들을 가져오고 그걸 일정 알고리즘을 통해서 점수를 낸 다음 점수로 마스크를 써야 하는지를 판별한다. * 프론트엔드 : 프론트에선 사용자가 웹 페이지에 접근 할 때마다 벡엔드에서 데이터를 받아와서 그래프와 함께 각 데이터의 심각도를 함께 보여준다. 프론트엔드는 컴퓨터 뿐만 아니라 탭, 핸드폰에서도 지원한다. * 하드웨어 : 하드웨어에선 초음파 센서에 30cm 이내의 거리가 인식 될때마다 3초동안 오늘 마스크를 써야 하는지 여부를 RGB LED 로 보여준다. |

## 회로 구성 사진 또는 실행 화면 사진

|  |
| --- |
| **사진이 없어서 회로 설계도로 대체한다.**  **아두이노에 RGB LED 를 연결하고, 초음파 센서를 연결 하였다.**  **초음파 센서는 HC-SR04 를 이용했으며 회로 설계 프로그램엔 HC-SR06 밖에 없어서 HC-SR06 을 대신 넣었다.** |

## 구현한 기능들에 대한 구체적인 설명

|  |
| --- |
| * 프로그램이 시작할 때와 매 한시간에 한번씩 데이터들을 가져온다. * 가져온 데이터들을 db 에 저장하고 로컬 캐시에 저장을 한다. * 저장된 데이터의 심각도를 계산하고, 마스크를 써야 하는지 여부를 판별한다. * 사용자가 웹에 접근하면, 일정 기간동안의 데이터 들과 마스크를 써야 하는지 여부를 프론트엔드에 보내준다. * 초음파 센서에 30cm 이내의 거리가 인지되면 아두이노에 시리얼 통신으로 마스크를 써야 하는지 여부를 보내준다. * 프론트 엔드는 데이터들을 chart.js 로 보여줌과 동시에 데이터의 심각도, 마스크를 써야 하는지를 보여준다. * Led에서는 마스크를 써야 한다면 빨간색을, 안써도 된다면 파란색을 3초동안 보여준다. |

## 구현한 기능들에 대한 사진

|  |
| --- |
| 마스크를 쓰지 않을 때의 화면 사진. 접근 시각이 18시 이후, 06시 이전일 경우 이런 배경을 보여준다. 다시 확인해보기를 클릭하면 창이 새로고침 된다.    마스크를 써야 할때의 화면 사진    06시 이후 18시 이전일 때 보여주는 배경 사진이다. |
| 모바일 버전으로 만든 화면이다. 모바일에선 데이터의 심각도, 그래프 두 이렇게 두개의 사각형만 보이며 화살표 버튼을 누르거나 밑으로 슬라이드 하면 다음 데이터를 볼 수 있다.    미세먼지 데이터를 보여주는 화면    초미세 먼지 데이터를 보여주는 화면 스크롤을 내리면 다음 데이터가 애니메이션 효과로 보옂고, 이전 데이터는 애니메이션 효과로 사라진다.    코로나 데이터를 보여주는 화면    감기, 황사 데이터를 보여주는 화면  모든 화면에 있는 화살표를 누르면 다음 화면, 맨 아래 있는 화살표를 누르면 맨위로 올라간다.  화면 우측에 있는 버튼들은 리모컨이며 항상 사용자의 화면 오른쪽에 떠 있다. 다섯개의 버튼 중 하나를 누르면 해당 버튼에 적혀 있는 데이터가 있는 화면으로 이동한다. |

# 파일 리스트

## 결과물 목록 & 목록 설명

|  |
| --- |
| 1learnmask-main.zip  1learnmask 의 코드이다. 안에 db 도 포함되어 있다. Db 의 이름은 data.db 이다.  1learnmask.pptx  발표할 때 쓰였던 ppt 이다. 안에 사진 자료들이 많아서 결과물 목록에 추가하였다.  Web\_demo.mp4  프로젝트의 웹 데모 영상이다.  자유주제9조보고서  본 보고서이다. |

## 데이터 확보 방법

|  |
| --- |
| * 모든 데이터는 프로그램이 처음에 실행될 때, 그리고 매 시간 5분에 데이터를 모두 가져오고 저장한다.        * 시간별 황사 데이터 : 기상청의 데이터를 가져오는 네이버에서 셀레니움을 통해 크롤링을 통해 데이터를 가져온다. * 코로나 데이터 : 질병관리청에서 beautiful soup 를 통해서 크롤링을 통해 데이터를 가져온다. * 감기 데이터 : 국민건강보험공단에서 질병예측 정보 csv 를 다운 받았으며, 이를 db 에 넣어놓은다. 데이터는 ‘예측’ 정보이기에 반기 (6개월) 에 한번 업데이트 된다. 그래서 최신 데이터는 없는 대신에 최근 10년 동안의 데이터는 있기에 시계열 예측을 통해 당일의 예상 감기 확진자의 수를 예측한 다음 db 에 저장한다 * 시간별 미세먼지 데이터 : 공공데이터를 api 를 통해 데이터들을 json 으로 가져온다. |

## 테이블 구조

|  |
| --- |
| * air\_quality : 시간별 미세먼지, 초미세먼지의 데이터가 들어있다. * cold : 날짜별 최근 10년치의 감기 확진자 데이터가 들어있다. * cold\_grade : 최근 10년치의 감기 확진자 수를 오름차순으로 정렬 한 다음, 정확히 5분할 하는 기준 데이터 값을 저장한다. 원할한 설명을 위해 db 사진을 업로드 한다.      * covid : 일자별 코로나 확진자 수 데이터가 들어있다. * predicted\_cold : 일자별 예측한 감기 확진자 수 데이터가 들어있다. * sqlite\_sequence : sqlite 에서 auto\_increment 를 위한 데이터를 저장한 것으로, 개발자가 만든 db 는 아니다. |

## 기타 중요 파일에 대한 추가 기술

|  |
| --- |
| * 코드 폴더의 가장 최상위 폴더에 data.db 가 있다. 이 파일이 데이터 폴더이다. * 코드 폴더의 가장 최상위 폴더에 db\_init\_all.txt 가 있는데, 이 파일은 초기 환경 세팅할 때 테이블을 만드는 쿼리들이 적혀 있다. 그대로 복사, 붙여넣기로 실행 시킬 수 있도록 작성해놨다. * 코드 폴더의 /const/config 파일에 아두이노를 연결할 포트를 적는 변수가 있다. 이 프로그램은 아두이노가 연결되어 있지 않으면 실행되지 않기 때문에 각자의 컴퓨터에 맞는 포트 이름을 적어줘야만 한다. |

# 강조하고 싶은 사항

## 내세우고 싶은 강조점, 차별점

|  |  |
| --- | --- |
| 백엔드 | * 최적화가 가장 중요했다. * 쓰레딩을 통해 초음파 센서 값을 읽어오고, led 색깔을 보여주는 프로그램을 데이터를 가져오고 웹에 보여주는 프로그램과 병렬로 돌 수 있게 하였다. 그렇게 해서 초음파 센서, 웹 접근 동시에 작동하고 동시에 작동해도 속도에는 차이가 없게 하였다.      * 쓰레딩을 통해 한시간에 한번씩 데이터를 업데이트 하는 잡이 백그라운드에서 돌아가게 하였다. 백 그라운드에서 돌아가기 때문에 데이터를 업데이트 하는 동안에도 프로그램은 정상적으로 작동한다.      * 가져온 데이터를 db 에 저장하기도 하고, 로컬에 캐싱하기도 하여서 최대한 데이터를 가져오는 시간을 줄였다. * 이를 통해 초음파 센서에 30cm 이하의 거리가 인식되면 0.2초 안에 반응 할 수 있도록 하였다. 초음파 센서 때문에 최적화에 가장 신경을 썼다. * 웹에 접근 했을 때에도 초음파 센서와 같이 데이터를 업데이트 하는 것이 아닌, 저장되어 있는 데이터를 보내주는 것이기 때문에 접근 시간이 매우 빠르다. 맨 처음 접근시에는 이미지등의 렌더링 때문에 시간이 조금 걸리지만, 이후 새로고침을 누르면 데이터 값들을 가져오는 시간은 굉장히 짧아서 0.2 초 이내에 새로고침이 완료 될 수 있도록 하였다. |
| 프론트엔드 | * 웹에 접속하자마자 한눈에 보기 쉽게 결과를 알려주고 아래로 갈수록 근거들을 나열해주는 구조를 통해 편의성과 디테일을 모두 살린 디자인 * 사용자가 밑 / 위 로 스크롤링 하거나 화살표 버튼을 눌렀을 때 다이나믹 함을 보여주기 위해 이전 / 이후 데이터들을 표시하는 부분이 애니메이션으로 생기고 사라지게 하였다. * 우측에 리모컨처럼 작동하는 버튼들을 고정시켜놔서 클릭을 통해 원하는 데이터를 한번에 볼 수 있도록 하였다. 사진은 ‘2.4 구현한 기능들에 대한 사진’에 첨부하였다. * 아침 / 저녁 의 배경을 다르게 하여서 다이나믹한 화면을 보여주었다. 사진은 ‘2.4구현한 기능들에 대한 사진’ 에 첨부하였다. |

## 프로젝트 진행중에 어려웠던 사항들과 그 해결 방법

|  |
| --- |
| 최적화가 가장 문제였다. 초음파 센서도 다른 파이썬 프로그램이 진행되느라 거리 인식이 늦어선 안되었고, 초음파 센서에 데이터를 보내느라 웹이 늦게 떠서도 안되었다. 또한 웹 페이지가 뜰 때에도 속도가 느리면 앞서 말한 것처럼 시간이 없을 때를 위해서 만든 프로그램인데 그 취지에 위배되었다. 이때문에 쓰레딩과 캐싱, db 저장의 방법을 사용하여서 속도를 최적화를 해서 아무리 느려도 0.2 초 이내에 결과를 볼 수 있게 하였다. |

# 소감

|  |  |
| --- | --- |
| 오동근 | 백엔드는 많이 해봤지만 역시 최적화와 속도의 중요성을 다시 한번 느꼈다. 속도가 느리면 사용자는 해당 프로그램을 사용하기 싫어진다. 또한 교수님과도 계속 얘기해 보고 다른 친구들의 프로젝트를 하면서 느꼈는데, 프론트 디자인이 확실히 중요하다. 프론트 디자인이 일단 멋지고 좋아야 사람들이 관심을 갖고 본다. |
| 안찬혁 |  |
| 이상혁 | 웹 개발은 처음이었지만 수업에서 배운 내용을 토대로 프론트와 백의 동작 원리와 구조에 대해서 자세히 알 수 있었고 빠르게 적응하여 개발할 수 있었다. 또한 사용자 관점에선 내용보다 디자인이 더 중요할수도 있다는것에 대해 깨닫는 계기가 되었다. |