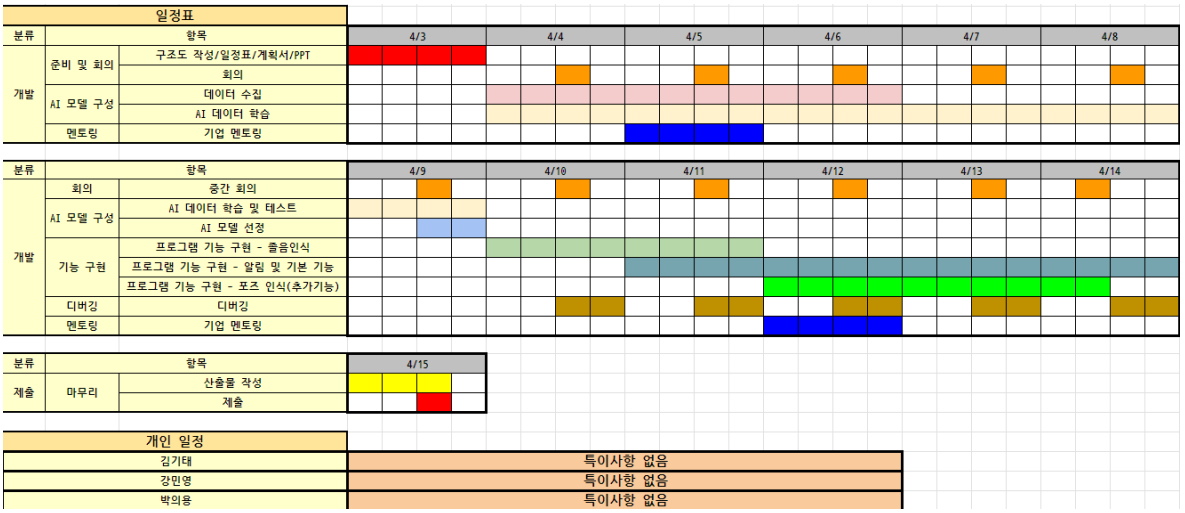
개발 완료 보고서

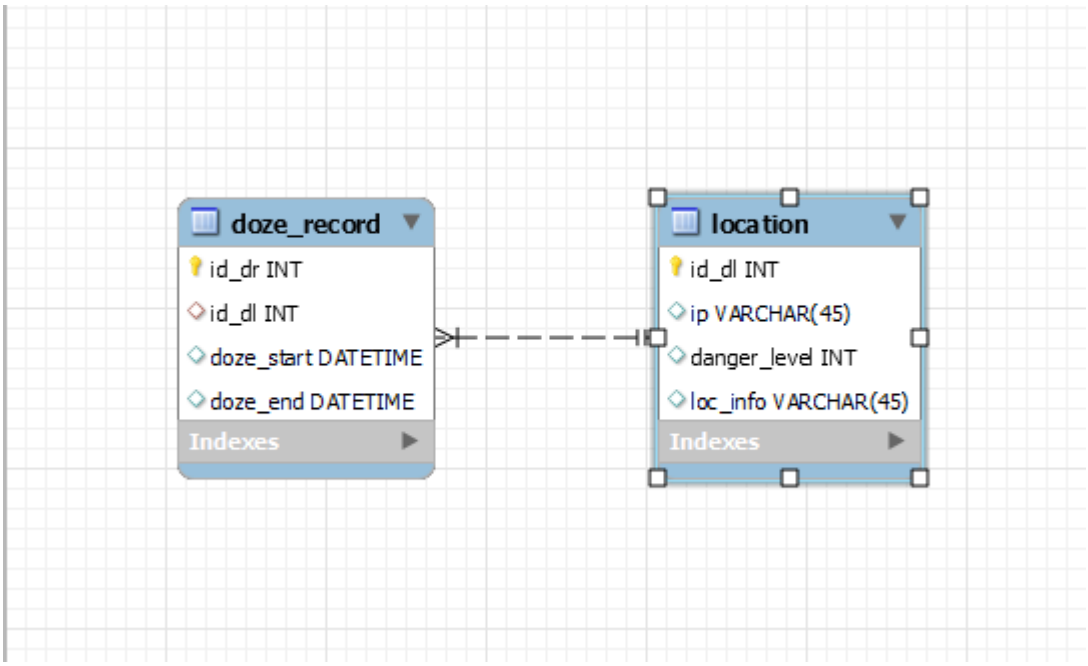
제출일 : 2023년 04월 15일

<table>
<tr><td>팀명</td><td colspan="3">5팀 탕탕특공대</td></tr>
<tr><td>참여인원</td><td colspan="3">김기태, 강민영, 박의용</td></tr>
<tr><td colspan="4">프로젝트 소개</td></tr>
<tr><td>프로젝트 명</td><td colspan="3">CNN을 활용한 졸음 방지 솔루션 개발 프로젝트</td></tr>
<tr><td>활동일시</td><td>2023.04.03(월) ~ 2023.04.15(토)</td><td>장소</td><td>광주인력개발원
공학 1관 드론융합실</td></tr>
<tr><td>주요 주제</td><td colspan="3">얼굴 인식을 통한 노동자의 졸음 방지</td></tr>
<tr><td>개발목적</td><td colspan="3">- 위험 구역 혹은 공장에서의 반복 작업시에 불량률을 낮출뿐만 아니라 근로자의 안전환경을 구현하기 위하여 근로자의 졸음을 인식하여 졸음을 깰수 있게 하고 일정 횟수 이상 알림시 관리자에게 알림을 보냅니다.</td></tr>
<tr><td rowspan="3">개인별
임무분담</td><td>김기태</td><td colspan="2">- 데이터 수집, CNN을 이용한 딥러닝, GUI 구현, 알림 기능 구현 , TCP/IP 구현</td></tr>
<tr><td>강민영</td><td colspan="2">- 데이터 수집, CNN을 이용한 딥러닝, GUI 구현, 얼굴 인식 관리자 클라이언트의 기록 조회 및 시각화 구현</td></tr>
<tr><td>박의용</td><td colspan="2">- 데이터 수집, CNN을 이용한 딥러닝, GUI 구현, 클라이언트 GUI 구현</td></tr>
<tr><td>개발환경</td><td colspan="3">IDE : Pycharm, VS 2022 community, winform C#, Jupyter Notebook
DBMS : MySQL / 공유환경 : GitHub / AI Tool : Keras</td></tr>
<tr><td>구현기능</td><td colspan="3">- 파이썬과 CNN을 활용한 졸음 인식 인공지능 구현
- 눈꺼풀 감기는 정도에 따른 공장 단순 작업자의 졸음 인식 구현
- 사용자의 얼굴 각도에 따른 단순 작업자의 졸음 인식 구현
- 사용자의 행동에 따른 알림 Off 기능 구현</td></tr>
</table>

일정표



ERD



요구사항 분석서

5팀 탕탕특공대 요구사항 분석서

번호	유형	요약	요구사항	요구분석 내용	정	부	비고
1	공통	공통사항	OpenCV 사용	OpenCV 4.5.5.64 을 사용한다.	김기태 강민영 박의용	-	
			서버 네트워크 구조	TCP/IP 서버 클라이언트 구조를 이용함.			
			GUI	C# winform으로 구현한다			
			졸음 인식 AI 딥러닝	Jupyter notebook , Keras, OpenCV를 이용한다.			
			얼굴 인식 모델	haar cascade를 사용합니다.			
			IDE	Visual Studio Community 2022			
			DB	1.DBMS 는 MySQL을 사용한다.			
				2.스키마의 이름은 FDsolution 으로 한다.			
				3. 구역별 위험도를 표시한 dangerLevel 테이블 (ID,IP,위험도)			
				4. 구역별 졸음 감지 기록을 담은 dozeRecord 테이블 (ID,LocationID,졸음시간,알림해제까지 걸린 시간)			
2	졸음인식 프로그램 (근로자)	근로자의 얼굴 혹은 자세를 분석하여 졸음을 인식한다.	졸음 인식 및 알림 기능	1. 얼굴의 각도를 인식하거나 눈꺼풀을 인식한다.	강민영 박의용	김기태	
				2. 카메라에서의 얼굴 인식시 제대로 검출되지 않는 포인트가 생기는게 반복되면 알림을 준다.			
				3. 카메라에서의 눈 인식시 눈꺼풀이 3초 이상 감겨 있을시 알림을 줌.			
				4. 카메라에서의 눈 인식시 눈꺼풀이 2초이상 감겨 있는 시간이 3번 이상 반복되면 알림을 준다.			
				5. (추가기능) 특정 동작 인식시에 알림이 꺼진다.			
				6. 버튼을 누를시 알림이 꺼진다.			
				7. 알림이 몇초 이상 꺼지지 않는다면 서버에 신호를 송신하여 관리자 측으로 전달한다.			
3	관리자 프로그램 및 서버	특정 구역 알람 발생시 구역을 확인할 수 있다.	졸음 기록 조회 및 서버 기능	1. 알람 발생시 구역을 확인할 수 있다.	김기태	강민영 박의용	
				2. 알람 발생 기록을 확인할 수 있다.			
				3. 사용자측 인식 프로그램에서 정보를 전달 받아 DB에 기록한다.			

코드 설명	서버 및 관리자	
	Record.cs - DB의 기록을 조회하고 구역별 그리고 요일별 졸음 횟수를 시각화 해준다.	
	private void Record_Load(object sender, EventArgs e)	두번째 폼을 로드하는 동시에 DB에 연결하여 doze_start 테이블에 있는 번호, 구역, 졸음 시작/종료 시각 정보를 표에 시각화한다.
	private void btn_search_Click(object sender, EventArgs e)	- 조회버튼을 클릭하여, 콤보박스에서 선택한 년, 월, 일에 해당하는 졸음 기록을 불러온다.
	private void btn_graph_Click(object sender, EventArgs e)	그래프 버튼을 클릭하여 막대 그래프 두개를 표시한다. - char 1: 공장내 구역 별 졸음 빈도수 시각화 - chart 2: 요일 별 졸음 빈도수 시각화
	fd_solution.cs - 구역 경보를 나타내주는 윈폼	
	private void fd_solution_FormClosing (object sender, FormClosingEventArgs e)	가장 주된 기능이라고 생각되어서 닫기 버튼을 누르더라고 꺼지지 않게 했다.
	private void fd_solution_Load(object sender, EventArgs e)	타이머 로드
	private void timer1_Tick(object sender, EventArgs e)	타이머간격 마다 라벨의 시간을 바꿔줌.
	Main.cs - MDI 부모폼 - 서버를 만들어서 비동기로 각 클라이언트별 스레드를 돌린다.	
	private async void Main_Load(object sender, EventArgs e)	메인폼이 로드될때 TCPListener를 열고 자식폼을 열어주고 각 클라이언트들이 연결될때 비동기로 Task를 실행한다.

코드 설명	private async void AsyncTcpProcess(object o)	서버에 클라이언트가 연결되었을때 각 신호를 받는 수신 스레드이다. 특정 신호가 오면 자식폼에 invoke하여 GUI에 시각화하여 표시해준다.
	private async Task InsertSleepStart(string area)	졸음감지시 DB dozerecord 테이블에 졸음 시작 기록을 한다.
	private async Task UpdateSleepEnd(string area)	졸음 종료시 해당 구역의 마지막 기록에 기록을 저장한다.
	private string CheckIPArea(string ip)	클라이언트 접속시 DB에 접근하여 IP를 가지고 해당구역이 어디인지 확인한다.
	private void PlaySound()	'구역알림' 신호를 수신받을때 특정 알람이 실행된다.
코드 설명	클라이언트	
	class ThreadClass(QThread) def eye_cropper(self, frame)	Main Ui와 별도로 큐 쓰레드를 형성하여 웹캠에 모델을 인식하고, 송출하기 위해 별도의 쓰레드를 형성하여 pyqtsignal을 이용하여 시그널을 전달한다. 이후 프레임을 매개변수로 받아 눈 주위를 80x80 사이즈로 자른 이후 이미지를 return 하여 최종 영상 속 모델 인식을 구현한다.
	class boardInfo(QThread): cpu, ram	Main Ui와 별도로 큐 쓰레드를 형성하여 컴퓨터 cpu와 ram을 시그널로 전달하여, 컴퓨터 현재 상태를 확인할 수 있다.
	class server_connect(QThread)	Main Ui와 별도로 큐 쓰레드를 형성하여 서버 연결 시 프로그램 동작이 멈추지 않도록 한다.
	class MainWindow(QMainWindow)	Main Ui 주기능 실행 클래스. ip와 port를 설정하고, 여러 시그널 및 메서드를 포함한다.
	def __del__(self) def connectconn(self, check)	서버와 연결하기. 소켓 소멸 , 위 쓰레드에서 시그널로 받아, 접속 확인 후 완료 시 서버 메시지 전송,

코드 설명	def request(self) def closeconn(self)	버튼 클릭시 connectconn 함수 연결, 연결 종료 서버 메시지 전송 및 소켓을 닫고 여러 버튼 및 라벨 메서드 변화 담당.
	def mousePressEvent(self, event) def mouseMoveEvent(self, QMouseEvent) def mouseReleaseEvent(self, QMouseEvent)	클라이언트 ui가 마우스로 드래그하여 움직일 수 있게 하였다.
	def Client_lamp(self) def clock(self) def getCPU_usage(self, cpu) def getRAM_usage(self, ram) def Close_software(self)	서버와 연결 시 server 연결 상태 확인 라벨에 변화를 준다. 시간 변화를 timer를 이용하여 표시한다. cpu와 ram을 위 쓰레드에서 시그널로 받아 표시한다. (이 부분은, 웹캠 사용 및 모델 인식 그에 따른 컴퓨터 과부하를 알고 싶어서 넣었습니다.) x버튼을 눌러 종료할 때 실행되는 모든 쓰레드 종료
	def StartWebCam(self) def StopWebCam(self) def opencv_emit(self, Image) def cvt_opencv_qt(self, Image)	웹캠을 실행하고 종료하기 위한 함수 웹켐 실행시 모델 인식 쓰레드가 별도로 실행이 된다. 그에 따른 frame 에 대한 상태를 시그널로 받아온 이후 opencv_emit, cvt_opencv_qt 두 함수의 기능으로 pixmap을 이용하여 웹캠 영상을 Main Ui에 송출한다.
	def get_drowsiness(self, counter) def alarm_stop(self)	졸음 방지 함수. counter을 위 쓰레드에서 받아온다. 눈을 뜨면 0이다. 눈을 감으면 프레임 별로 카운트가 1씩 증가한다. if (조건문을 이용하여 1에서 3이면 눈을 감았다는 카운트 시그널을 쓰레드에서 보낸다) 카운트가 4이상이면 졸음 감지로 판단한 이후, winsound를 이용하여 알람이 울린다. 이후 서버에 졸음감지 시그널 전송, 알람 중지 버튼을 클릭하면 서버에 졸음에 깨기 위해 버튼을 눌러 알람을 중지한 걸로 판단하여 졸음종료로 서버에 시그널 전송.
	ImageUpdate =pyqtSiganl(np.ndarray) Eye Crimination =pyqtSignal(int) cpu = pyqtSignal(float) ram = pyqtSignal(tuple) check conn=pyqtSignal(int)	사용된 pyqt 시그널은 총 5개이다. ImageUpdate는 모델 예측을 하고, 그 이미지를 MainUi로 보내기 위한 시그널이다. Eye Crimination은 모델 예측을 통해 프레임별로 영상 속에서 식별하여 눈을 뜨고 감고 졸음을 보내기 위한 시그널이다. cpu, ram 확인을 위한 시그널이다. 서버 연결이 완료인지? 기다리고 있는지? 에러가 있는지? 확인하기 위한 시그널이다. 시그널에 따라 방출되는(emit) 또한 총 5개로 구성되어 있다.

<table>
<tr><td rowspan="3">개인후기</td><td>강민영</td><td>CNN 딥러닝에 대해 처음에는 잘 몰라서 Teachable Model을 통해서 눈 뜨고 있는 얼굴 사진과 눈 감은 사진을 두 그룹으로 나눠서 학습을 시키고 정확도가 어느정도는 있을 거라고 예상했지만 전혀 아니었다. 팀원들의 수고로 얻은 사진으로 학습을 시켜 정확도가 높은 모델을 가져올 수가 있었고, dlib을 직접 사용해보면서 이 라이브러리만으로도 눈꺼풀 사이의 거리를 측정하여 졸음 감지가 가능하다는 것을 알게 되었다.
이 후 서버에서 저장해 놓은 DB정보를 C# UI 에 띄우는 역할을 맡아, 완성 후에는 그래프도 만들어보았다.
이번에는 C#에서 DB를 연결하는 것이 두번째 이기 때문에 처음 했을 때 보다는 능숙하게 처리하였고, 전반적으로 비슷한 기능을 구현하는 일은 반복을 통해 익숙해진 느낌이 든다.

CNN 은 양질의 데이터를 1만개 이상 모아서 모델에게 학습을 여러번 시켜 정확도를 높이는 것이 관건인데 이 부분을 팀원들이 매우 적극적으로 참여해 준 덕분에 프로젝트가 빠르게 진행될 수 있었다.</td></tr>
<tr><td>김기태</td><td>쥬피터 노트북으로 모델을 학습시키고 다시 C#에서 불러오는 과정에서 제대로 불러오지 못해
아쉽게도 C# 관리자클라이언트및 서버 그리고 파이썬 클라이언트구조를 사용하게 되었다.
팀언이 파이썬클라이언트 틀을 QT로 사용할수 있는 사용법들을 많이 사용했길래 보면서 많이 배웠습니다.</td></tr>
<tr><td>박의용</td><td>CNN 딥러닝 모델 구현을 처음부터 끝까지 직접 해 보면서, 예상치 못한 문제를 계속 직면했다. 정확히 어떤 부분에서 문제가 일어났는지, 그 문제를 해결하기 위해 어떻게 해야 하는지에 대한 문제해결 능력을 이번 과제를 통해 피하지 않고 직면했다.
자료들을 참고할 때는, 영어로 된 원서들을 직접 해석하고, 코드 한줄 한줄 어떻게 실행이 되고 구성이 되는지 이해하려고 노력하였다. 또한 관련 자료들을 찾을 때는 논문 등을 참고하여, 어떻게 모델을 프로그램에 내장 시키는지에 대한 방법들을 배울 수 있었다. 정답은 정해져 있지 않지만, 정답을 찾아 가는 과정 속에서 많은 성장이 있었다고 생각한다. 팀장인 기태형이 C# 서버에 구현을 맡아줘서, 모델 구현 및 어떻게 프로그램을 동작시킬 수 있을지 더욱 집중하여 할 수 있었다. 혼자였다면 프로젝트를 완료할 수 없었을 것이다. 민영 누나 또한 피드백 및 여러 부족한 점들에 대한 이야기들을 같이 나누는 과정에서, 요구사항에 맞지 않는 부분들을 수정 해 나가며, 프로젝트를 완수 할 수 있었다고 생각한다. 코드적인 부분은 아직도 부족한 점이 많지만, 이번 과제를 통해서 많은 성장이 있었다고 생각하고 감사하다.</td></tr>
<tr><td>비고</td><td colspan="2">수고 많으셨습니다</td></tr>
</table>