Feel Easy

20152761 김소현 20157495 김민지

20161091 김예진 20160940 배새연

목차

1. 작품 소개

2. 작품 구성

2-1. DB 구성
2-2. 아두이노
2-3. 어플리케이션
2-4. 핵심 코드 구성

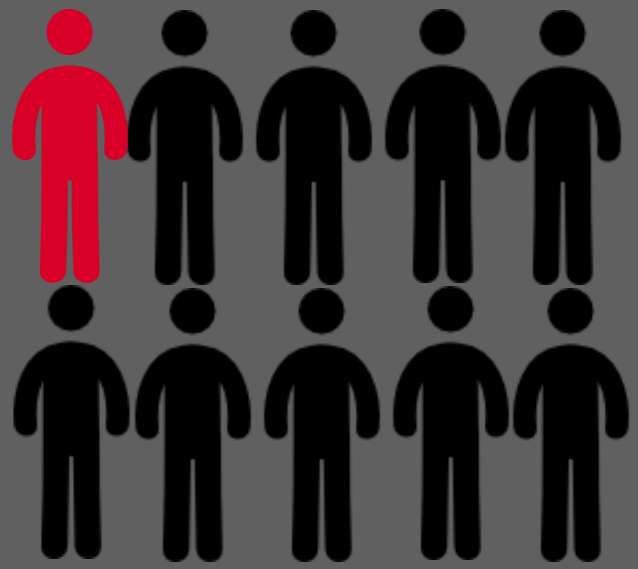
3. 기대효과

1. 작품 소개

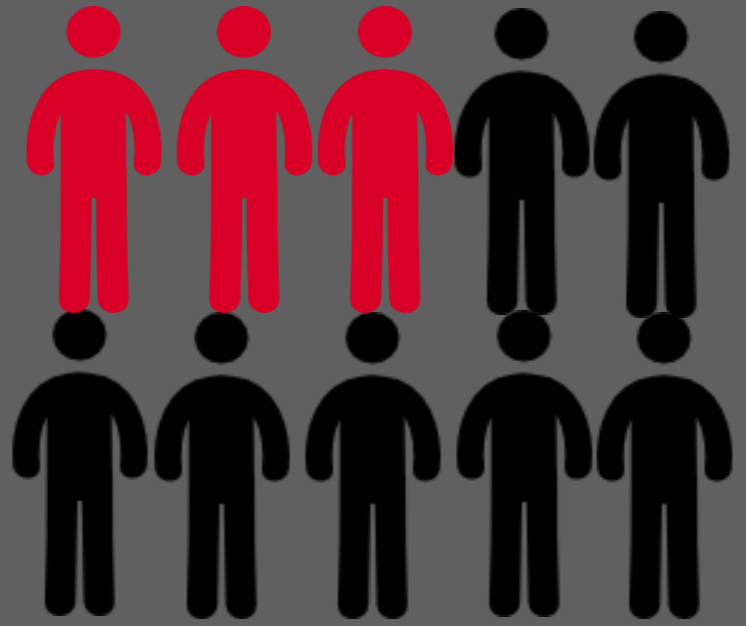
'1인가구 전성시대'

1인가구 비중 27.9%
1인가구 수 539만 7,615가구

2018년



2045년



1인가구 급증

소통의 부재

고독사

범죄의 표적

1인 가구 맞춤 안전 서비스

1인 가구를 위한

안전성 보장

독립성 보장

실시간 감지

생활 패턴 모니터링
위험 알림 서비스

Feel Easy

2. 작품 구성

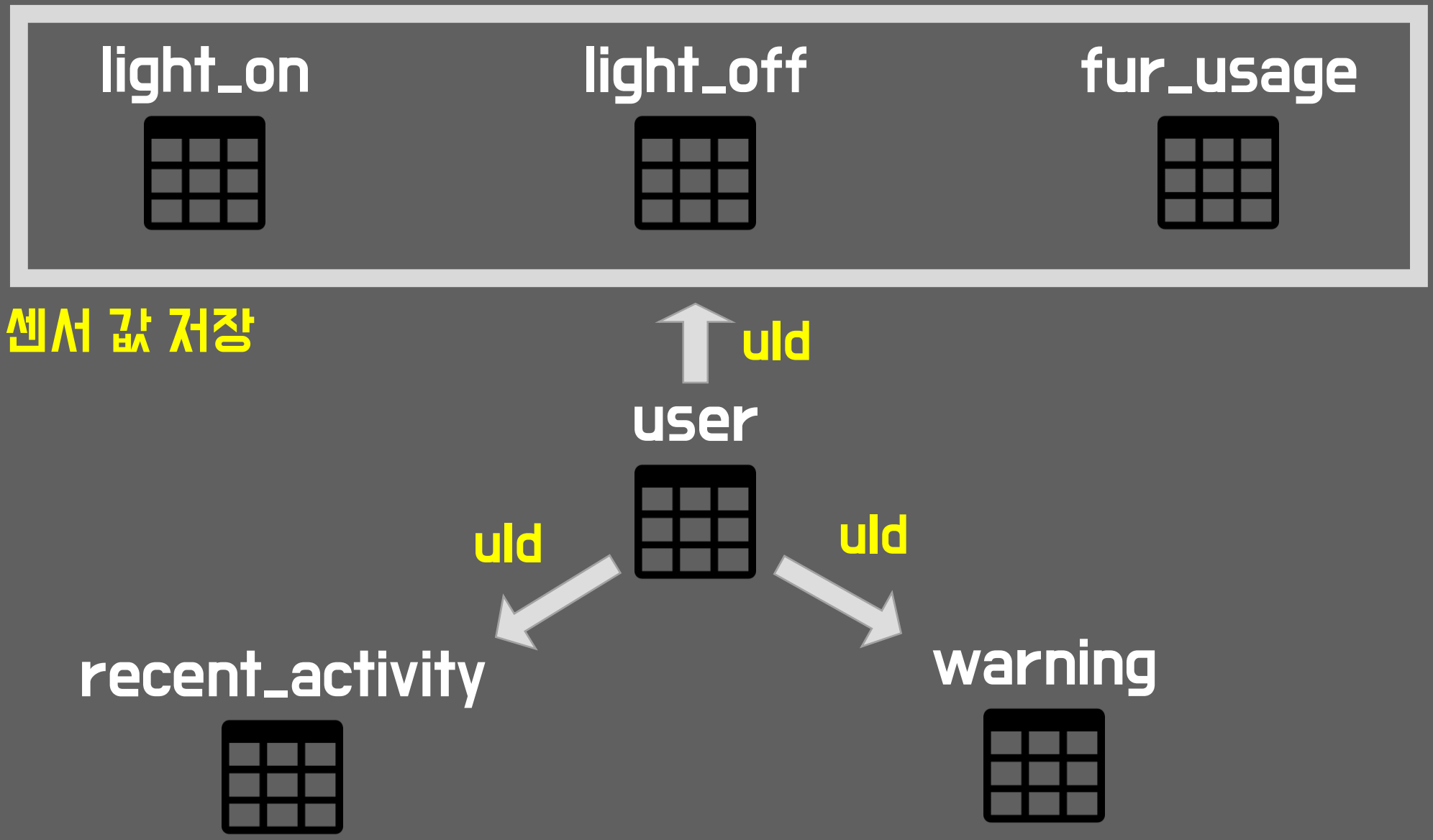
아두이노

어플리케이션

데이터베이스

생활 패턴 분석

위험 상황 알림


light_on
light_off
fur_usage
센서 값 저장
uId
user
uId
uId
recent_activity
warning

user : 앱을 사용하는 거주자, 보호자에 대한 정보를 저장하는 테이블

#	이름	종류	NULL
1	uId	int(11)	
2	type	char(1)	
3	name	varchar(10)	
4	tel	varchar(10)	
5	uFur	varchar(10)	
6	agree	int(11)	
7	related	int(11)	
8	date	datetime	허용

light_on, light_off, fur_usage

: 각각 전등이 켜지는 횟수, 전등이 꺼지는 횟수, 가구 사용 횟수를 요일별로 측정하여 저장하는 테이블

#	이름	종류	NULL	기본값
1	uid	int(11)		
2	sun	int(11)		0
3	mon	int(11)		0
4	tue	int(11)		0
5	wed	int(11)		0
6	thur	int(11)		0
7	fri	int(11)		0
8	sat	int(11)		0
9	date	datetime	허용	

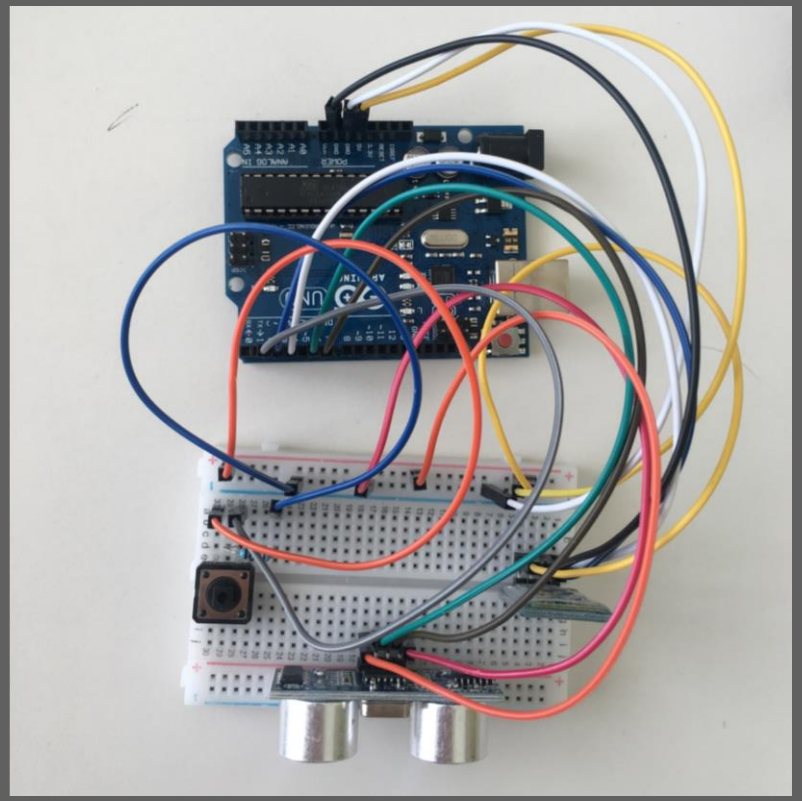
recent_activity
: 거주자의 최근 전등/가구 사용 기록을 저장하는 테이블

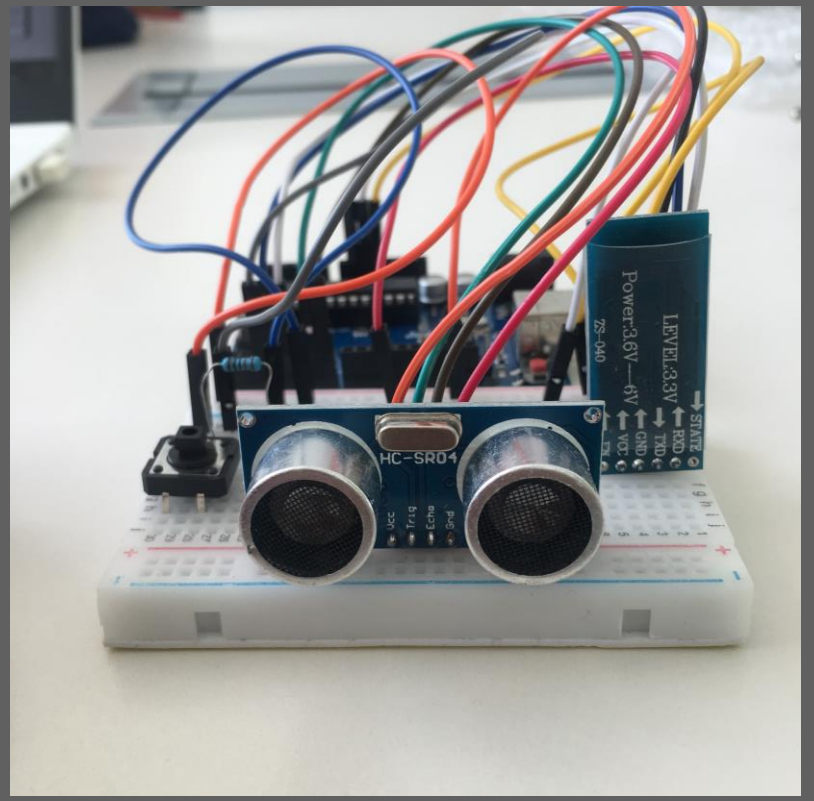
#	이름	종류	NULL	기본값
1	uId	int(11)		
2	act1	varchar(30)	허용	
3	act2	varchar(30)	허용	
4	act3	varchar(30)	허용	
5	act4	varchar(30)	허용	
6	act5	varchar(30)	허용	
7	act6	varchar(30)	허용	
8	position	int(11)	허용	-1

warning
: 거주자, 보호자에게 알리는 경고 알림 기록을 저장하는 테이블

#	이름	종류	NULL	기본값
1	uid	int(11)		
2	warn1	varchar(30)	허용	
3	warn2	varchar(30)	허용	
4	warn3	varchar(30)	허용	
5	warn4	varchar(30)	허용	
6	warn5	varchar(30)	허용	
7	position	int(11)	허용	-1

푸시 버튼 + 초음파 거리 센서 + 블루투스 모듈





기능 1. 실시간 모니터링

거실 전등과 등록된 가구 사용 현황을 실시간으로 조회

기능 2. 사용 패턴 조회

최근 일주일간의 전등, 가구 사용 기록을 요일별 그래프로 조회

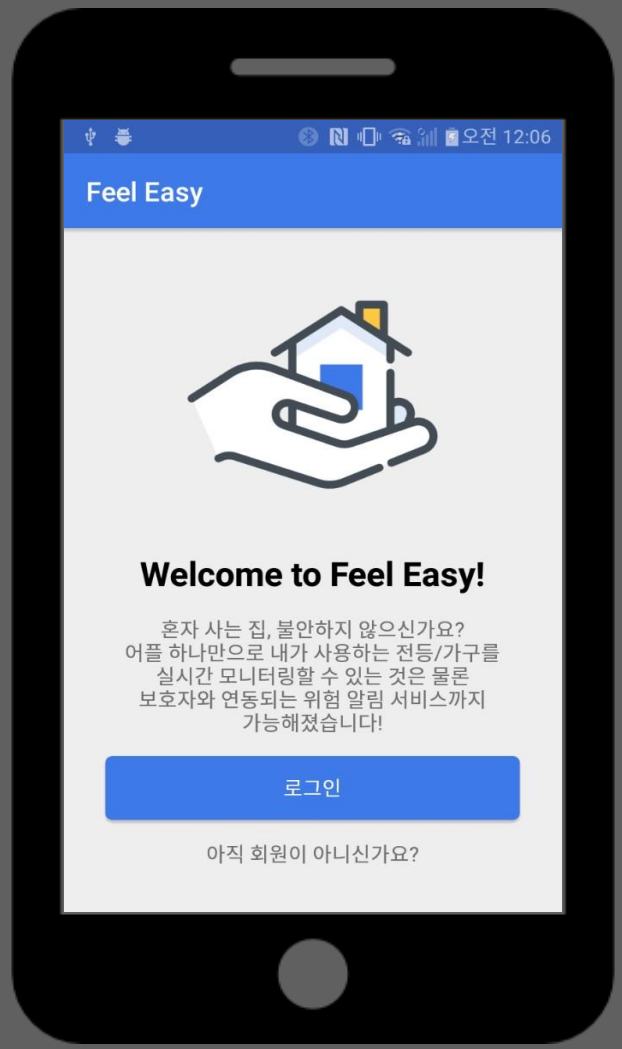
기능 3. 경고 알림 서비스

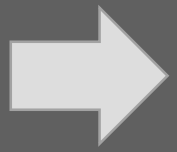
외출 모드 시 주기적 확인 알림, 평상시 생활 패턴과 다른 양상이 보일 시 경고 알림

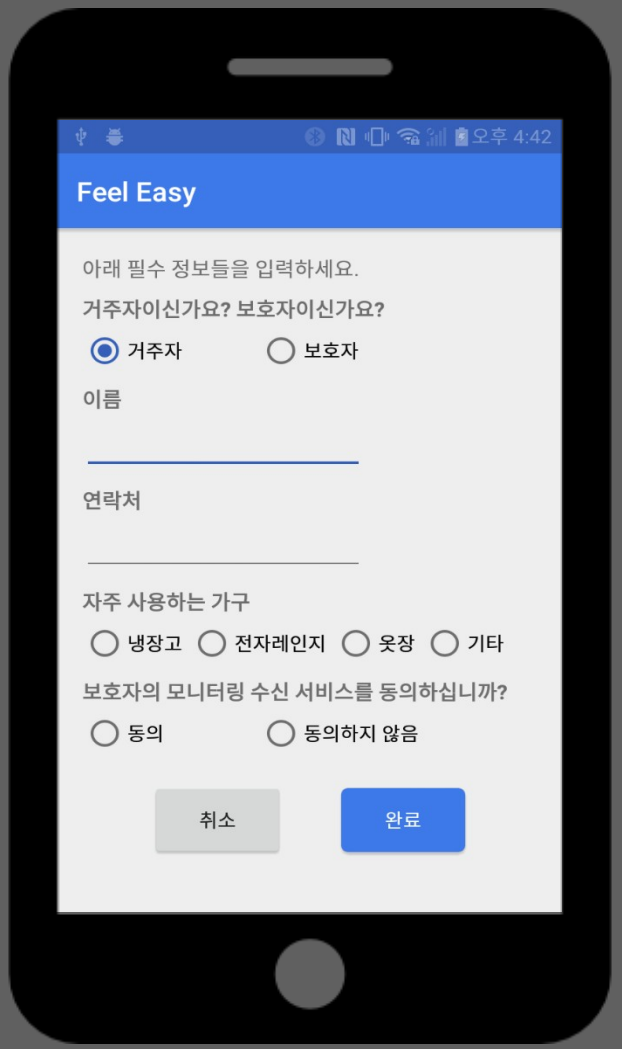
기능 4. 보호자와 공유

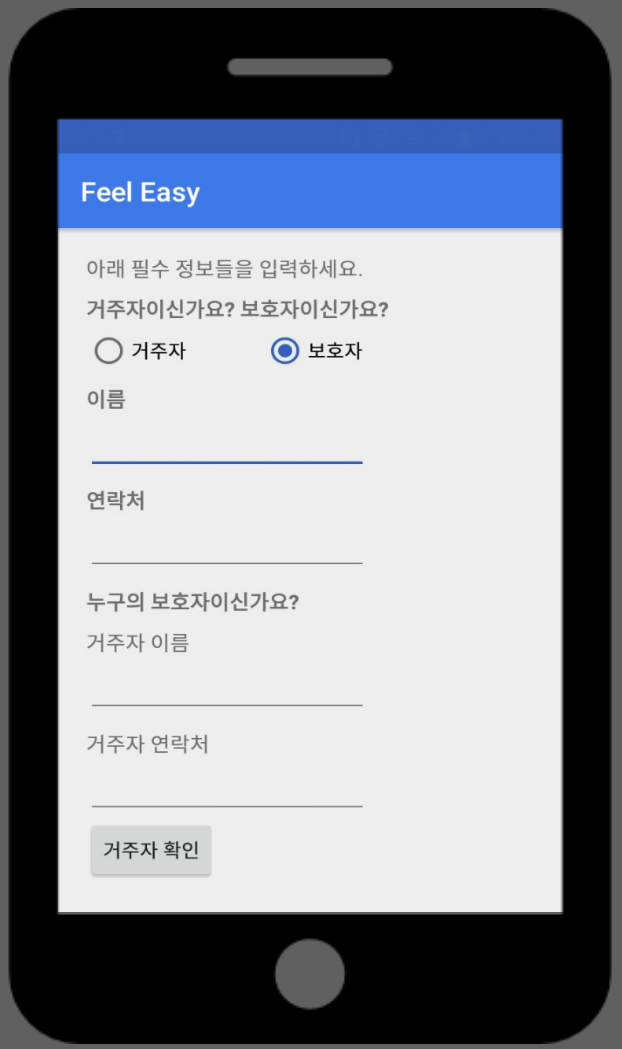
거주자가 동의할 시 모든 상황을 보호자와 함께 공유

거주자 또는 보호자로 회원가입

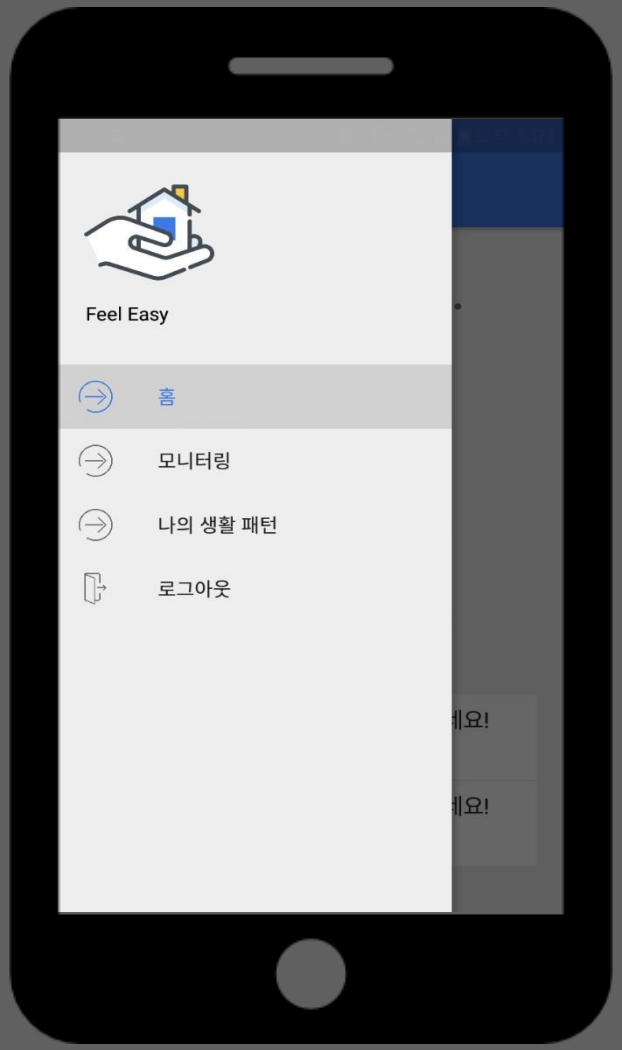




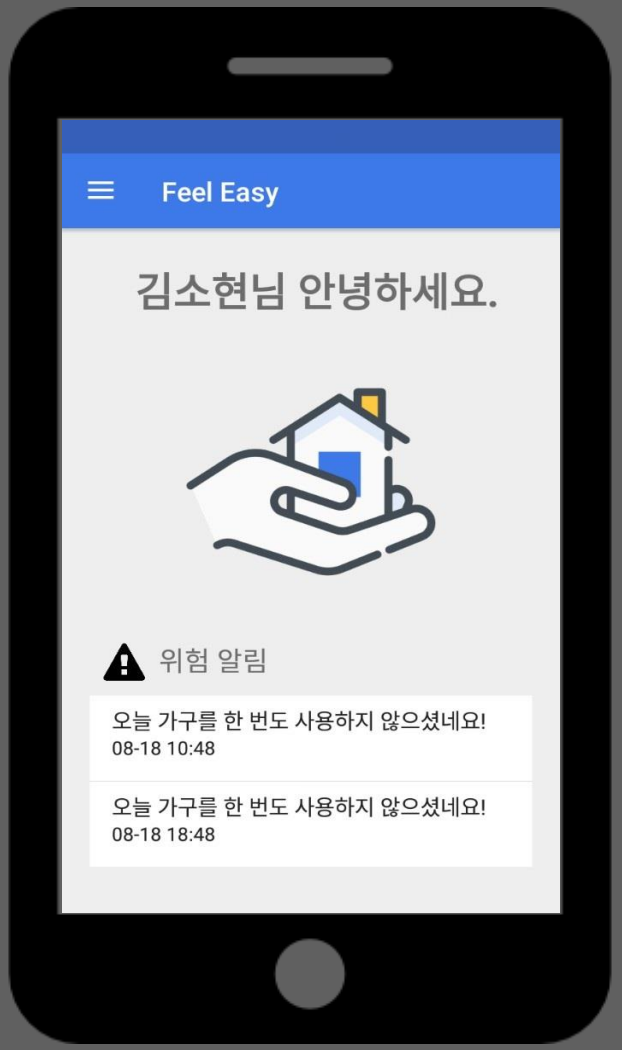


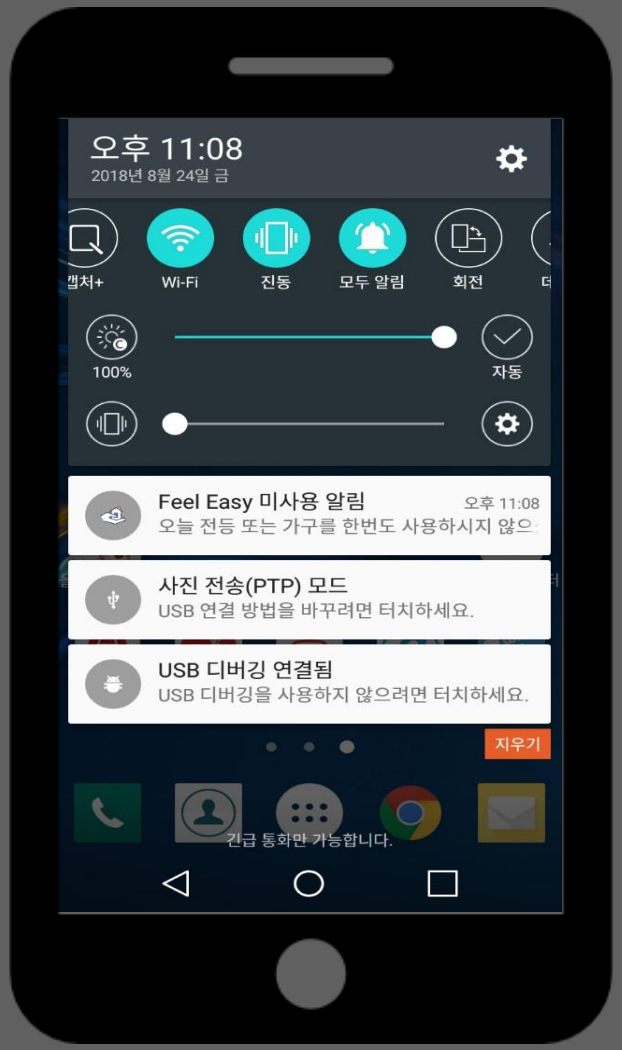


로그인 후

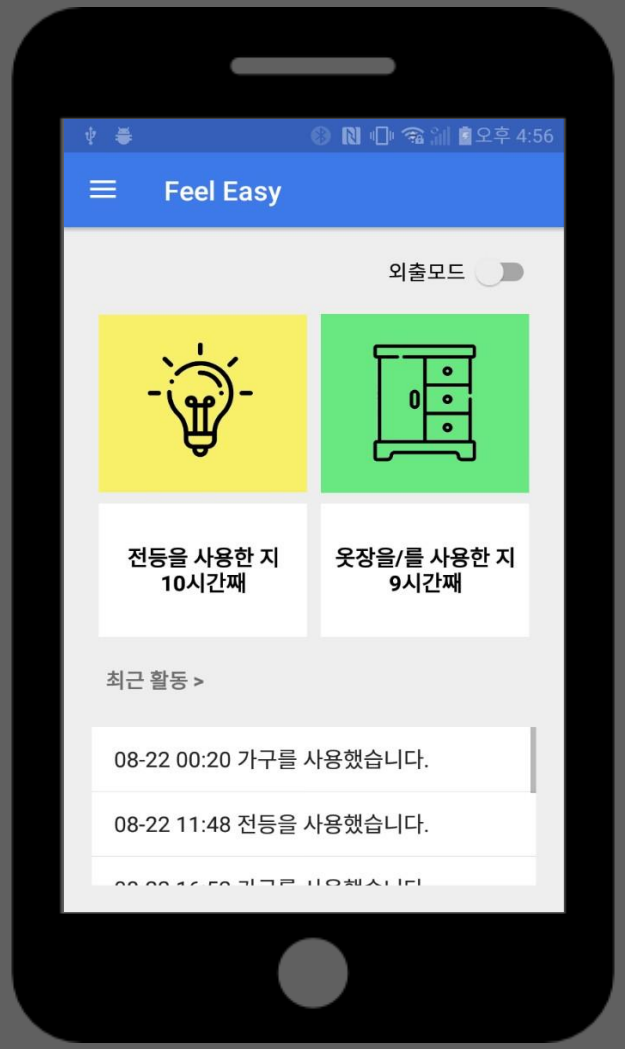

Feel Easy
홈
모니터링
나의 생활 패턴
로그아웃

홈 화면

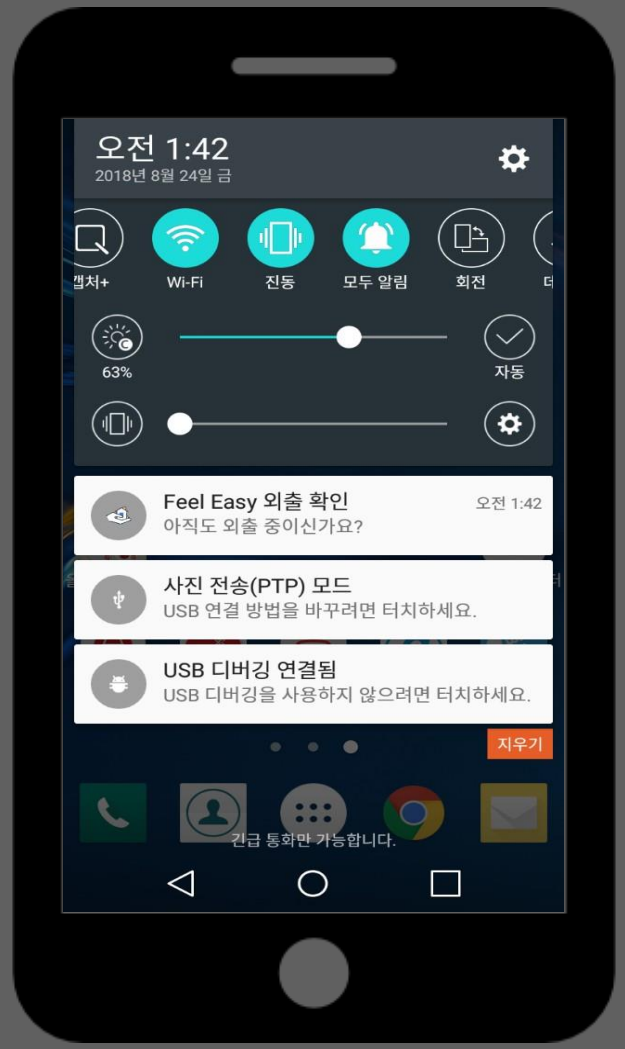

Feel Easy
김소현님 안녕하세요.
위험 알림
오늘 가구를 한 번도 사용하지 않으셨네요!
08-18 10:48
오늘 가구를 한 번도 사용하지 않으셨네요!
08-18 18:48


오후 11:08
2018년 8월 24일 금
Wi-Fi
진동
모두 알림
회전
100%
자동
Feel Easy 미사용 알림
오후 11:08
오늘 전등 또는 가구를 한번도 사용하시지 않으
사진 전송(PTP) 모드
USB 연결 방법을 바꾸려면 터치하세요.
USB 디버깅 연결됨
USB 디버깅을 사용하지 않으려면 터치하세요.
지우기
긴급 통화만 가능합니다.

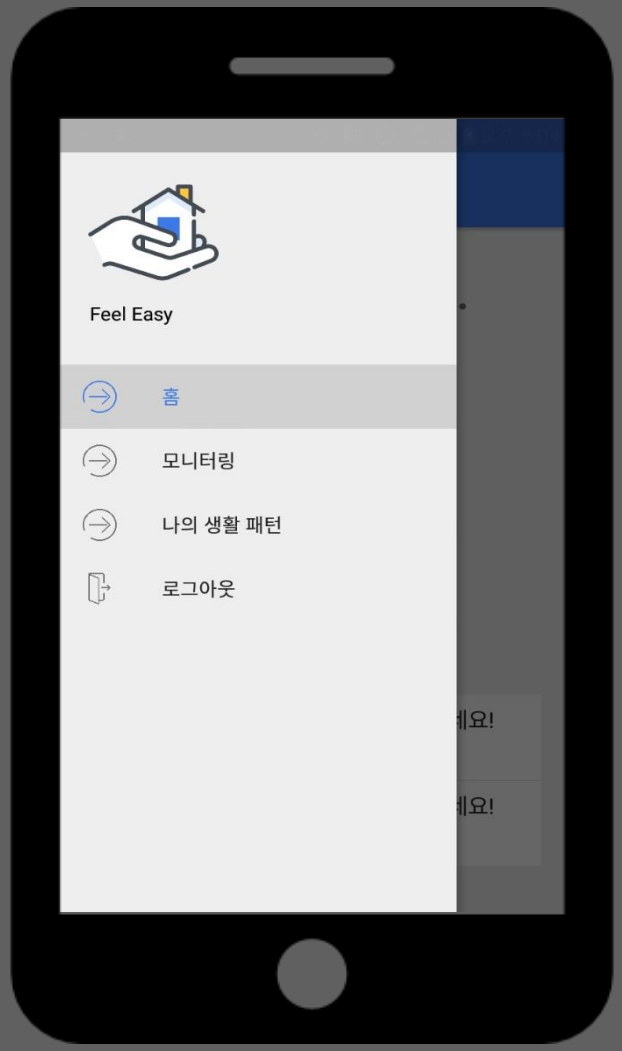
모니터링 화면



외출 모드 시 푸시 알림



로그인 후


Feel Easy
홈
모니터링
나의 생활 패턴
로그아웃

생활 패턴 화면


오후 5:01
Feel Easy
최근 일주일간 전등 켜진 횟수
Description Label
08/17
08/18
08/19
08/20
08/21
08/22
08/23
전등 켜진 횟수
최근 일주일간 가구 사용 횟수
Description Label
가구 사용 횟수

〈아두이노 코드〉

전등의 on/off 상태 판단과 눌린 횟수 계산

```
btnState = digitalRead(BTN);
if (btnState != lastBtnState) {
  if (btnState == HIGH) {
    btnPushCount++;
    if (btnPushCount % 2 != 0) {
      BTSerial.println("on");
    }
    else if (btnPushCount % 2 == 0) {
      BTSerial.println("off");
    }
    Serial.println(btnPushCount);
  }
  delay(50);
}
lastBtnState = btnState;
```

거리 계산을 통한 가구 사용 판단

```
digitalWrite(TRIG, LOW);
delayMicroseconds(2);
digitalWrite(TRIG, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG, LOW);

long distance = pulseIn(ECHO, HIGH)/58.2;
if (distance < 10) {
  if (sw == 1) {
    BTSerial.println("fur");
    sw = 0;
  }
} else {
  sw = 1;
}
```

2.4 핵심 코드 구성

〈어플리케이션 코드〉

블루투스를 통해 아두이노에서 센서 값 수신

```
void beginListenForData() {
    final Handler handler = new Handler();

    readBuffer = new byte[1024];  //수신 버퍼
    readBufferPosition = 0;  //버퍼 내 수신 문자 저장 위치

    //문자열 수신 쓰레드
    thread = new Thread((Runnable) () -> {
            while (!Thread.currentThread().isInterrupted()) {
                try {
                    int bytesAvailable = iStream.available();
                    if (bytesAvailable > 0) {  //데이터가 수신된 경우
                        byte[] packetBytes = new byte[bytesAvailable];
                        iStream.read(packetBytes);
                        for (int i=0; i<bytesAvailable; i++) {
                            byte b = packetBytes[i];
                            if (b == charDelimiter) {
                                byte[] encodedBytes = new byte[readBufferPosition];
                                System.arraycopy(readBuffer,  i: 0, encodedBytes,  i1: 0, encodedBytes.length);
                                final String data = new String(encodedBytes,  charsetName: "US-ASCII");
                                readBufferPosition = 0;

                                handler.post(() -> {
                                        update(data);  //수신된 데이터 앱에 반영
                                });
                            } else {
                                readBuffer[readBufferPosition++] = b;
                            }
                        }
```

서버를 통해 DB에 전등/가구 사용 데이터 업데이트

```
public class UpdateDatas extends AsyncTask<String,Void,String> {
    Context context;

    public UpdateDatas(Context context) { this.context = context; }

    @Override
    protected void onPostExecute(String s) {
        Toast.makeText(context,  text: "데이터가 수집되었습니다!", Toast.LENGTH_LONG).show();
    }

    @Override
    protected String doInBackground(String... arg0) {
        String Table =  arg0[0];
        String Uid =  arg0[1];
        String Content =  arg0[2];

        try {
            String link = "http://192.168.35.159/FeelEasy/putDatas.php?table=" + Table
                    + "&uid=" + Uid + "&content=" + Content;
            URL url = new URL(link);
            HttpURLConnection httpURLConnection = (HttpURLConnection) url.openConnection();
            httpURLConnection.setReadTimeout(5000);
            httpURLConnection.setConnectTimeout(5000);
            httpURLConnection.setRequestMethod("GET");
            httpURLConnection.setDoInput(true);
            httpURLConnection.connect();

            int responseStatusCode = httpURLConnection.getResponseCode();
            InputStream inputStream;
            if(responseStatusCode == HttpURLConnection.HTTP_OK) {
                inputStream = httpURLConnection.getInputStream();
            }
            else {
                inputStream = httpURLConnection.getErrorStream();
            }
```

〈어플리케이션 코드〉

서버를 통해 DB에 접근하여 전등/가구 사용 횟수를 조회

```
//전등, 가구 사용 횟수를 조회하는 클래스
public class SelectCounts extends AsyncTask<String,Void,String> {

    @Override
    protected void onPostExecute(String result) {
        try {
            String[] datas = result.split( regex: " ");
            String[] lights = datas[0].split( regex: ",");  //요일별 전등 켜진 횟수
            String[] furs = datas[1].split( regex: ",");  //요일별 가구 사용 횟수

            int day = (Calendar.getInstance()).get(Calendar.DAY_OF_WEEK);  //1~7 사이 값 반환
            int index = day - 1;
            for (int i=0; i<7; i++) {
                entriesLight.add(new Entry(i, Integer.parseInt(lights[index])));
                if (index == (lights.length-1))
                    index = -1;
                index++;
            }
            //조회한 데이터를 바탕으로 그래프 좌표 설정
            setValues(chartLight, entriesLight,  label: "전등 켜진 횟수");

            for (int i=0; i<7; i++) {
                entriesFur.add(new Entry(i, Integer.parseInt(furs[index])));
                if (index == (lights.length-1))
                    index = -1;
                index++;
            }
            setValues(chartFur, entriesFur,  label: "가구 사용 횟수");
        } catch (Exception e) {
```

조회한 데이터를 바탕으로 라인 그래프 그리기

```
public void setValues(LineChart chart, ArrayList<Entry> entries, String label) {
    LineDataSet dataSet = new LineDataSet(entries, label);
    dataSet.setColor(ContextCompat.getColor(getContext(), R.color.colorPrimary));
    dataSet.setValueTextColor(ContextCompat.getColor(getContext(), R.color.colorPrimaryDark));

    XAxis xAxis = chart.getXAxis();
    xAxis.setPosition(XAxis.XAxisPosition.BOTTOM);
    //그래프 x축에 라벨 달기
    setLabels();

    IAxisValueFormatter formatter = new IAxisValueFormatter() {
        @Override
        public String getFormattedValue(float value, AxisBase axis) {
            return labels.get((int) value);
        }
    };
    xAxis.setGranularity(1f);
    xAxis.setValueFormatter(formatter);

    YAxis yAxisRight = chart.getAxisRight();
    yAxisRight.setEnabled(false);

    YAxis yAxisLeft = chart.getAxisLeft();
    yAxisLeft.setGranularity(1f);

    LineData data = new LineData(dataSet);
    chart.setData(data);
    chart.animateX( durationMillis: 2500);
    chart.invalidate();
}
```

3. 기대효과

낮은 진입 장벽

발전된
맞춤형 서비스

사용 범위 확대

감사합니다.