

모바일 포렌식 시각화 툴 개발

K-shield 주니어 보안사고 분석대응반

월목반 김석진 | 월목반 문재식

월목반 민서정 | 월목반 박귀수

월목반 송어진 | 월목반 이용하

월목반 이재훈 | 월목반 이지호

월목반 한택승

목차

1. 프로젝트 개요	3
1.1 프로젝트 명 및 기간	3
1.2 프로젝트 목적	3
1.3 기대 효과	3
1.4 프로젝트 배경	3
1.5 프로젝트 목표	4
2. 프로젝트 조직	5
2.1 프로젝트 조직도	5
2.2 프로젝트 책임 및 역할	5
3. 프로젝트 수행 일정	6
3.1 프로젝트 추진 일정	6
3.2 프로젝트 세부 절차	6
3.3 프로젝트 개발 시 사용한 라이브러리 및 공유자원	7
4. 분석용 데이터 생성 및 추출	8
4.1 사용 기기 루팅	8
4.2 추출한 데이터	9
5. GUI 화면 구성	12
6. 시각화	15
7. 추후 연구 예정	26
8. 관련 자료	27

1. 프로젝트 개요

1.1 프로젝트 명 및 기간

프로젝트 명: 모바일 포렌식 시각화 툴 개발

프로젝트 기간: 2021.03. ~ 2021.06. (2.5개월)

프로젝트 수행 팀: 티미룸

1.2 프로젝트 목적

모바일 포렌식을 이용하는 사람들에게 기존에 있는 모바일 포렌식 도구들 보다 더 보기편하고 무료로 사용할 수 있는 툴을 제공하는 것이다. 또한 입문자가 쉽게 모바일 포렌식을 할 수 있도록 도와주는 것이다.

1.3 기대 효과

- 고도의 경험을 가진 숙련자 외에 입문자 또한 쉽게 모바일 포렌식이 가능
- 모바일 포렌식 업을 하는 사람들이 보기 편한 모바일 포렌식 도구를 이용 가능

1.4 프로젝트 배경

현재 스마트폰 시장은 급속도로 팽창하는 중이다. 그 중에서도 안드로이드 스마트폰의 성장은 더욱 더 두드러진다. 모바일 기기의 사용이 증가하면서 모바일 포렌식에 대한 수요도 증가하고 있다. 최근 다양한 뉴스 기사에서 확인할 수 있다.

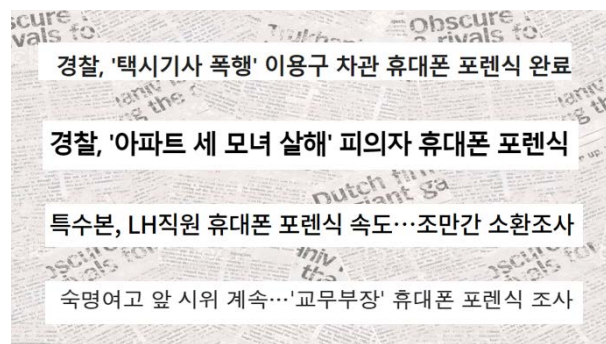


그림 1 모바일 포렌식 관련 기사

우리는 2021 년 기준 4 월 안드로이드 시장 점유율이 72.2%로 가장 높기에 우리는 안드로이드 운영체제를 타깃으로 잡았다. 오픈 소스의 장점을 기반으로 여러 제조사들의

다양한 기기 출시와 꾸준한 안드로이드 OS 업데이트는 사용자들에게 장점으로 부각되지만, 디지털 포렌식 관점에서는 어려움으로 작용하고 있다. 이는 논리적 포렌식은 대부분 루팅을 전제로 하는데, 다양한 기종과 지속적인 업데이트는 루팅을 어렵게 하기 때문이다.



그림 2 2021년 4월 세계 모바일 os 이용 비중



그림 3 2021년 4월 국내 모바일 os 이용 비중

현재 상용화 되어있는 도구들은 비용적인 측면에서 개인이 쉽게 사용이 불가능하며, 기존에 있는 오픈소스에 의존할 수밖에 없다. 하지만 기존에 있는 오픈 소스들은 최신의 기기들에 적용하기 어려운 부분들이 존재하고, 시각화 부분에 아쉬움을 보인다. 우리는 이러한 현실태에서 모바일 논리적 이미징을 통해 사용자 위주의 행위에 대한 데이터를 추출하고 이를 시각화 작업을 통해서 더욱 분석하기 편리한 모바일 포렌식 시각화 도구를 제작할 것이다.

1.5 프로젝트 목표

- 휴대폰에서 추출한 데이터를 이해하기 쉽게 시각화하는 툴 만들기
- 위치정보와 시간정보를 바탕으로 사용자의 행위를 파악하기 쉽게 시각화

2. 프로젝트 조직

2.1 프로젝트 조직도

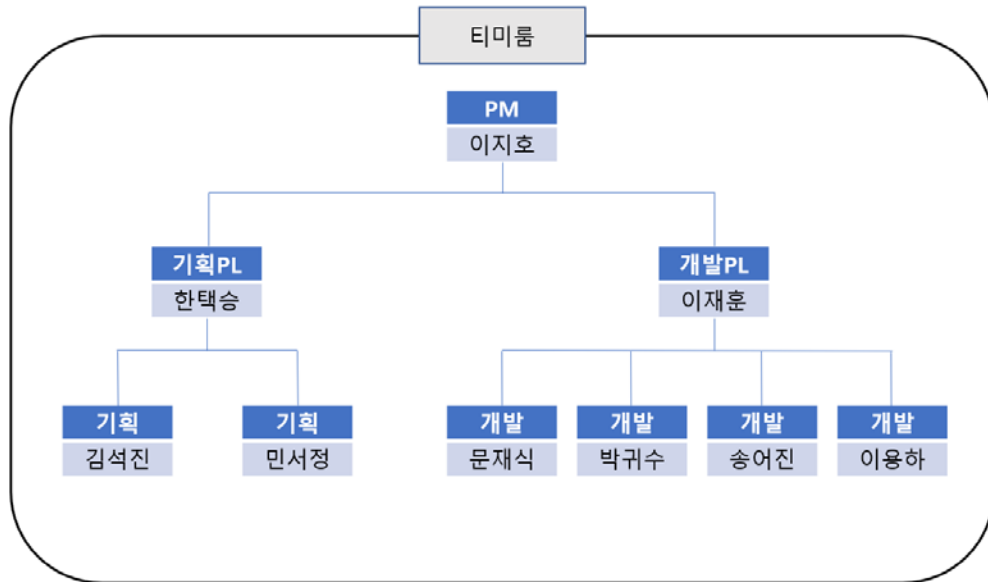


그림 4 프로젝트 팀 도식화

2.2 프로젝트 책임 및 역할

책임	역할	담당자명
PM	프로젝트 총괄 책임자	이지호
개발 PL	개발 책임자	이재훈
기획 PL	기획 및 조사 책임자	한택승
개발	장고를 이용한 대시보드 생성 및 시각화	문재식
개발	GUI 프로그램 개발, 시각화	박귀수
개발	장고를 이용한 대시보드 생성 및 시각화	송어진
개발	장고를 이용한 대시보드 생성 및 시각화	이용하
기획	기획 및 자료 조사, 케이스 생성	김석진
기획	기획 및 자료 조사, 케이스 생성	민서정

표 1 책임 및 역할

3. 프로젝트 수행 일정

3.1 프로젝트 추진 일정

모바일 포렌식 시각화 툴 개발				완료	진행중	예정													
No.	구분	항목	세부사항	3월	4월				5월				6월						
				30~31	1~5	6~12	13~19	19~25	25~30	1~3	4~10	11~17	18~24	25~31	1~7	8~14			
1	사업관리	주제 선정	아이디어 도출 및 정리																
			보완사항 도출 및 의견수렴																
	보고	주간 보고	[Trello] 주간 이슈사항 보고 및 회의																
		최초 보고	사업 계획서 제출																
2	데이터 생성	생성	모바일 데이터 생성																
			전체 백업본 데이터 생성																
			루팅 작업 시험																
			모바일 아티팩트 추출																
	프로그램 개발	파일 분류	tar(or dd) 파일 추출																
			tar(or dd)파일 읽기																
			tar(or dd) 파일 내용 분류																
		GUI 프로그래밍	GUI 프로그래밍																
			웹, DB, 변수설정																
			시각화 1차 수정																
	시각화	인프라 구축	시각화 2차 수정																
			UI 디자인																
			최종 시각화 모델 구축																
			중간보고																
3	테스트	통합테스트	모듈 테스트																
			사용자 테스트																
			최종 테스트																
		사후관리	최종보고서																

그림 5 WBS

3.2 프로젝트 세부 절차

구분	수행업무	설명	담당자
착수	프로젝트 착수	프로젝트 수행 절차 및 일정 계획 수립	모두
준비	기초자료 수집	모바일 포렌식 관련 논문, 문서 등 자료수집	모두
	프로젝트 방향 설정	프로젝트 진행 방향 및 방법 설정	모두
	휴대폰 확보	안드로이드 기기 확보 및 버전 확인	이지호
	데이터 생성	확보된 기기에 데이터 쌓기	김석진
분석	데이터 파악	추출할 수 있는 데이터 분석	모두
개발	GUI프로그램 개발	Qt5를 이용하여 프로그램을 GUI로 사용할 수 있도록 개발	박귀수
	웹 서버 구축	장고를 이용한 웹 서버 구축 및 모델 구축	이재훈
	데이터 추출 및 분류	tar 데이터 추출 및 분류 자동화	이지호
	데이터 시각화	와이파이, 캘린더 데이터 시각화	박귀수
		타임라인 개발	이용하
		전화, 메시지 시각화 및 페이지 이동 개발	송어진
		미디어 시각화, 브라우저 및 메시지 기록 시각화 및 키워드 추천 개발	이재훈
		위치 정보 시각화	문재식, 이재훈

표 2 프로젝트 인원 별 역할

3.3 프로젝트 개발 시 사용한 라이브러리 및 공유자원

구분	TYPE	Spec
개발 언어	Python	3.6.0
웹 프레임워크	Django	3.2.2
오픈 API	Google map API	

4. 분석용 데이터 생성 및 추출

4.1 사용 기기 루팅



그림 6 A8star 이미지

사용 기기 Galaxy A8 Star – 버전 안드로이드 10(최신버전 안드로이드 11)

루팅 과정

Frija 에서 순정 펌웨어를 찾은 후 오딘을 이용하여 순정 펌웨어 설치

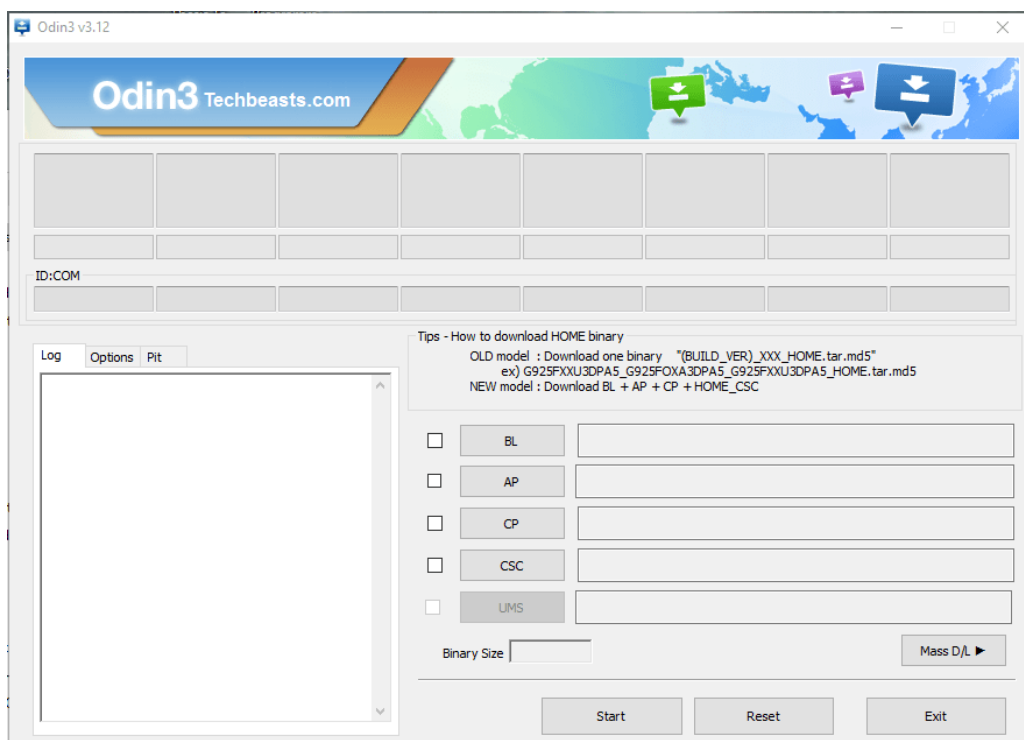


그림 7 odin3 실행창

Magisk 를 이용한 루팅

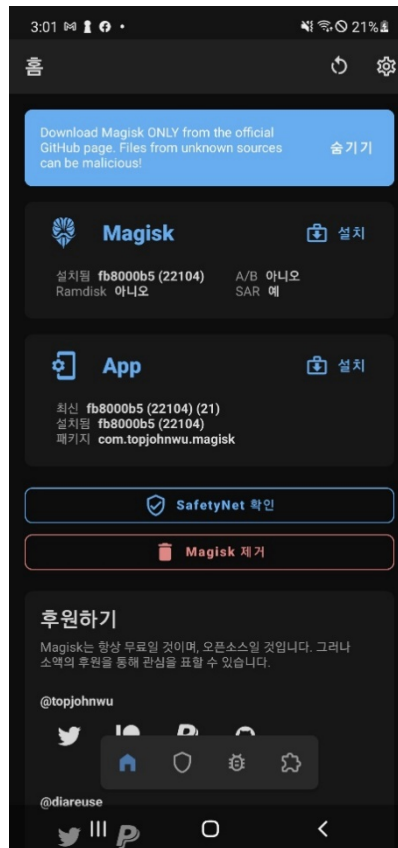


그림 8 Magisk 실행 화면

4.2 추출한 데이터

루팅을 하여 뽑아낸 데이터는 다음과 같다.

←Android 의 /data 안에 있는 폴더

그림 9 /data/

[illegible]

그림 10 /data/data/

↑ 추출한 데이터 안 userdata

userdata 안에는 db 파일, cache 등 관련 데이터가 들어 있다.

경로	설명
/data/data/[패키지 이름]/shared_prefs	어플리케이션의 공유 설정 파일이 저장되는 영역

/data/data/[패키지 이름]/cache	어플리케이션이 필요한 임시파일이 저장되는 영역
/data/data/[패키지 이름]/database	어플리케이션이 필요한 데이터베이스 파일이 저장되는 영역
/data/data/[패키지 이름]/files	데이터베이스와 캐시를 제외한 어플리케이션에서 사용하는 일반 파일이 저장되는 영역
/data/data/[패키지 이름]/lib	어플리케이션이 필요한 라이브러리 파일이 저장되는 영역

표 3 어플리케이션 데이터 설명

5. GUI 화면 구성

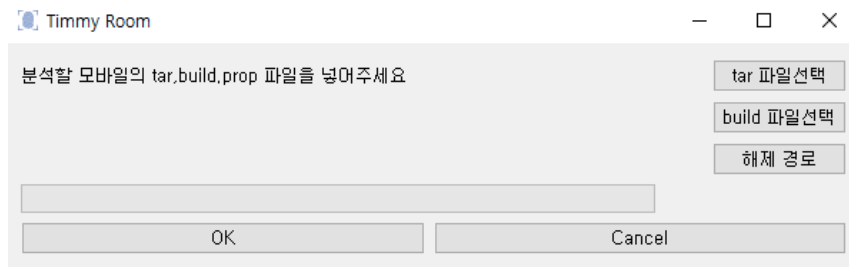


그림 11 실행 창

분석할 모바일의 /data/폴더는 모바일 기기의 userdata 가 들어있는 폴더로 이를 tar 로 압축한 압축파일을 경로에 넣는다. 모바일 기기의 정보가 담겨있는 build.prop 는 /system/의 밑에 있는 파일이다. 위의 두개의 파일은 루팅된 기기에서 추출해야 한다. 이후 파일위치를 선택하여 넣어준다. 이후 tar 파일에서 필요한 파일들을 추출하여 둘 위치경로를 선택한다.

pyQt5 를 이용하여 파일을 선택할 수 있는 코드를 작성하였다.

```
def file_select1(self): #파일 선택
    FileOpen = QFileDialog.getOpenFileName(self, 'Open file', './',"datafile(*.dd *.tar)")#해당 확장자만 파일만 고를수 있도록
    if(FileOpen[0]!=''):
        self.label1.setText(f"tar 경로:{FileOpen[0]}")
        self.tarfilepath=FileOpen[0]
```

그림 12 파일 선택 코드

Tar 로 압축되어 있는 파일들을 압축 푸는 코드를 작성하였다.

```
def decompression(file_path, output_path):
    tar = tarfile.open(file_path)
    for folder in target_folder:
        if folder == './media/':
            for img in image_folder:
                image_name = set(tarinfo for tarinfo in tar.getmembers() if (tarinfo.name.startswith('./media/') and tarinfo.name.endswith(img)))
                tar.extractall(members=image_name, path=f"{image_path}/../was/static/assets/images/")
            continue
        target_name = set(tarinfo for tarinfo in tar.getmembers() if tarinfo.name.startswith(folder)) # target_folder에 있는 폴더의 이름을 가진 모든 tarinfo를 target_name으로 둔다.
        tar.extractall(members=target_name, path=output_path)
    tar.close()
```

그림 13 tar 압축 해제 코드

확인 버튼을 누르고 대시보드를 열어주는 코드를 작성하였다.

```
def ok(self):
    if self.tarfilepath!='' and self.buildfilepath!='' and self.outputpath!='': #파일 경로를 지정하였다면 시스템 명령으로 대시보드를 띄운다.
        try:
            path=os.path.dirname(os.path.abspath(__file__))
            self.progress(self)
            f=open(f"{path}/../경로.txt",'w')
            self.progress(self)
            data = f'{self.tarfilepath}\n{self.outputpath}' #파일 경로를 기록한다. 후후에 was에서 읽음
            f.write(data)
            f.close()
            self.progress(self)
            shutil.copy(f"{self.buildfilepath}", f"{path}/../was/app/")#build.prop 파일 복사
            self.progress(self)
            self.label1.setText("127.0.0.1:8000 으로 접속중입니다...기다려 주세요")
            self.label2.clear()
            self.label3.clear()

            self.label1.repaint() #객체의 label1을 다시 repaint 해준다.
            self.label2.repaint()
            self.label3.repaint()

            read_tar.decompression(self.tarfilepath,self.outputpath)
            self.progress(self)
            subprocess.run(f'python {path}/../was/manage.py runserver',shell=True,timeout=0.5)
        except Exception as e:
            print(e)
            webbrowser.open("http://127.0.0.1:8000",1)#해당 url을 새 창으로 연다.
            sys.exit()
```

그림 14 대시보드 및 웹 시작 코드

이를 통하여 GUI 로 데이터를 선택하고 압축해제한 데이터들을 다음과 같은 과정을 거쳐서 대시보드에 보여준다. (example) contacts2.db

Db 의 각 테이블을 객체화 한다.

```
class contacts_model(models.Model): #어진

    class Meta:
        managed = False
        app_label = "contacts"
        db_table = 'search_index'

    id = models.AutoField(db_column='contact_id',primary_key=True)
    name = models.TextField(db_column='sec_name',blank=True,null=True)
    number = models.TextField(db_column='tokens',blank=True,null=True)
```

그림 15 DB 테이블 객체화 코드

DB 의 경로를 선택한다.

```
DATABASES = {

    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': f'{BASE_DIR}/db.sqlite3',
    },
    'contacts': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME':f'{OUTPATH}/data/com.samsung.android.providers.contacts/databases/contacts2.db',
    }
}
```

그림 16 DB 경로 설정 코드

DB 라우터를 통해 정의한 모델이 각 데이터베이스를 찾아갈 수 있게한다.

```
class MultiDBRouter(object):
    def __init__(self):
        self.model_list =
        ['default','contacts','calllog','message','mms','map','chrome2','SAMINT','WebDowndata','Webext','Appinslog',
        'Media','Calendar','Kakao1','Kakao2']
    def db_for_read(self, model, **hints):
        if model._meta.app_label in self.model_list:
            return model._meta.app_label

        return None
    def db_for_write(self, model, **hints):
        return None
    def allow_relation(self, obj1, obj2, **hints):
        return None
    def allow_migrate(self, db, app_label, model_name=None, **hints):
        return None
```

그림 17 DB 라우팅 코드

객체화한 모델을 통하여 필요한 데이터를 불러와 웹에서 시각화 해준다.

```
if context['url']=="contact.html": ##여진
    page = request.GET.get('page', '1')

    c=contacts_model.objects.values()
    paginator = Paginator(c, 10)
    page_obj = paginator.get_page(page)

    context['callog']=callog_model.objects.raw("SELECT _id,datetime((date / 1000), 'unixepoch','localtime') AS date FROM calls where m_content IS NULL ORDER BY date ASC ") #메시지 아닌것만
    context['contact']=c
    context['page']=page_obj
```

그림 18 DB 데이터

6. 시각화

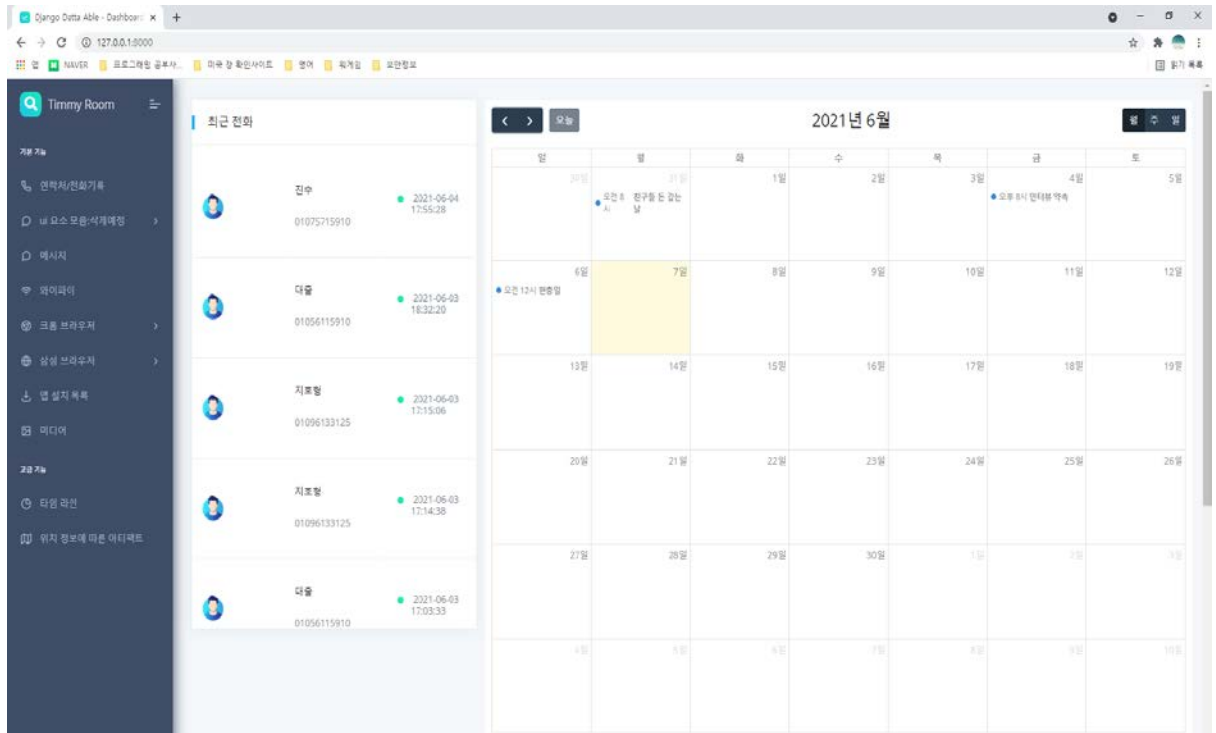


그림 19 메인화면1

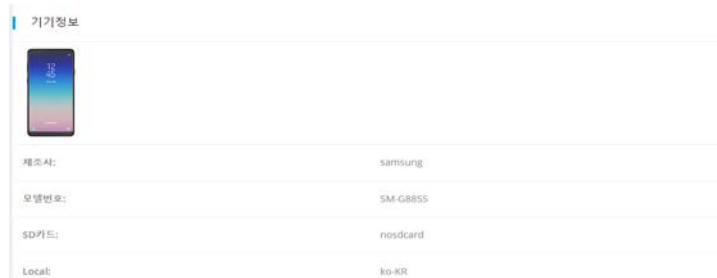


그림 20 메인화면2

탭 안에는 연락처/전화기록, 메시지, 브라우저, 와이파이, 미디어, 앱 설치목록, 타임라인, 위치정보에 따른 아티팩트, 카카오톡 등이 있다. 메인 화면 안에는 캘린더와, 최근 전화, 기기 정보 등이 존재한다. 기기정보는 크롤링을 사용하여 기기의 사진을 불러온다.

크롤링 코드 (현재는 구글에 robots.txt 에서 disallow 되어있는 곳에서 하고 있는 중입니다. 추후 허용된 사이트로 변경 혹은 이미지를 추가하여 진행할 예정입니다.)

```
def image1(model): #크롤링코드

    baseUrl = 'https://www.google.com/search?q=' + 검색
    plusUrl = '&asf=4e3820v1-d8AcF-V5c3hg7Dko4370Wc:1589354087979&source=images&as=KWed-2ahKJWjyo_EGqDpAbhbc-EE8UdrGCIQ0_AhbaVoZC8Qhwhiw9588biw927?'
    url = baseUrl + quote_plus(model) + plusUrl #url도 이동하기 위한 쿼리문자열 만들기
    print("url :",url)

    headers = {'User-Agent' : 'Mozilla/5.0'} #Mozilla요청, 연하면 403으로 크롤링 차단
    req = urllib.request.Request(url, headers=headers)
    html = urllib.request.urlopen(req) #url 열기
    soup = BeautifulSoup(html, 'html') #html.parser ,,,, lxml

    IMG=[]
    result = []

    a= 1

    img = soup.find_all('img')
    for img in soup.find_all("img"):
        IMG = (img['src'])

        if(a==1): #두번째 사진 사용
            break
        a += 1

    img_obj = {
        'title' : model,
        'image' : IMG
    }
    #print(img_obj)
    result.append(img_obj)

    return IMG #올라간 이미지 주소만 반환
```

그림 21 크롤링 코드

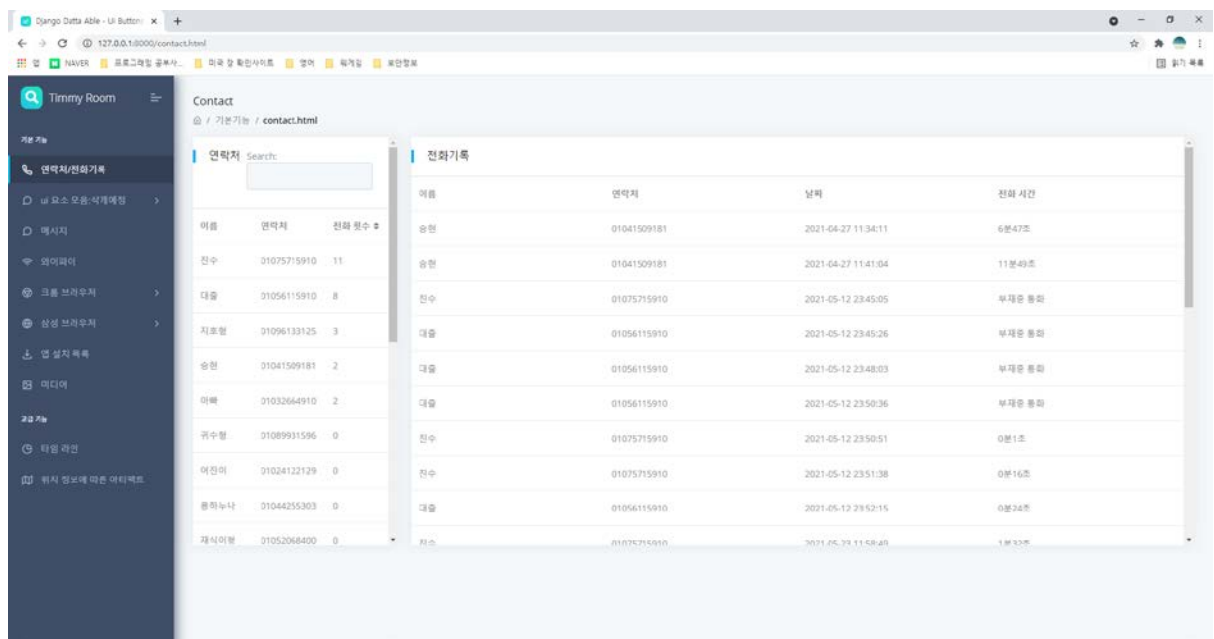


그림 22 연락처/ 전화기록 화면

화면은 연락처와 전화기록으로 되어있으며, 검색으로 검색도 가능하고 연락처의 이름을 클릭하여 전화기록에 그 사람과 연락한 내용만 보게 할 수 있다. 연락처의 원하는 사람의 전화기록을 보는 기능을 활용하면 다음과 같이 된다.

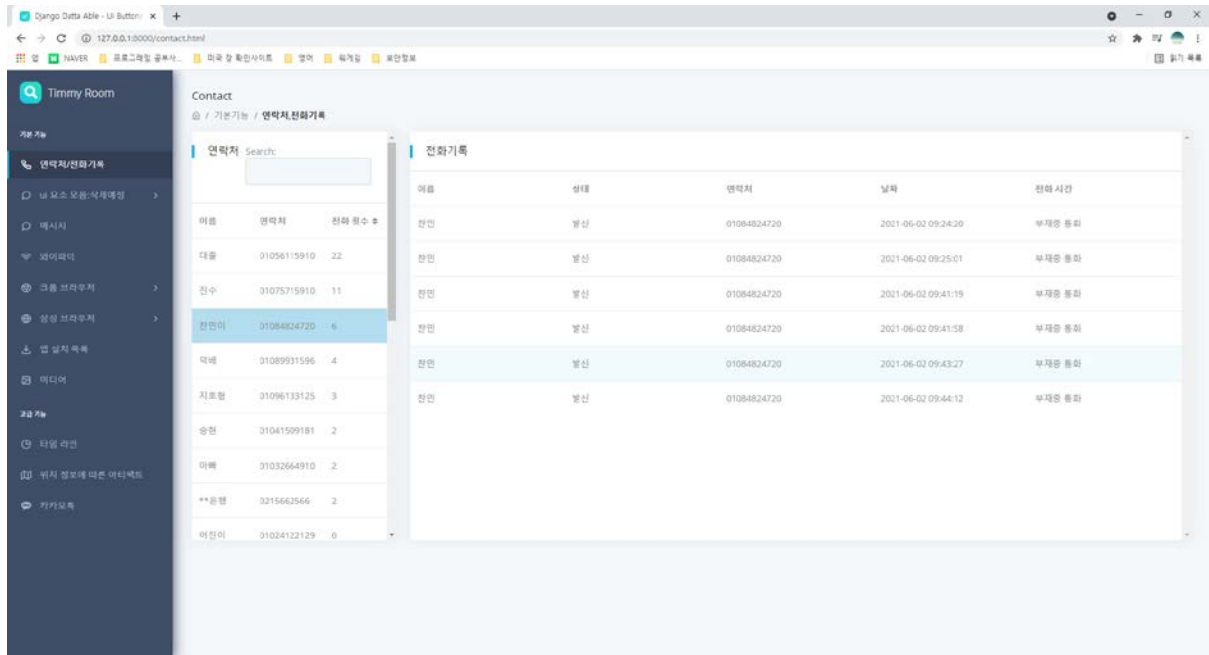


그림 23 원하는 연락처 번호 전화기록 표시

원하는 전화기록만 볼 수 있게 하는 코드를 작성하였다.

```
var cti="{{contactlength}}" //연락처 길이
var cli="{{callloglength}}" //전화기록 길이
$( document ).ready(function() {
    console.log(cti)
    for(var i=1;i<cti;i++){//연락처의 전화횟수 알려내는 소스
        var findnumber=$("#"+i).find("td").eq(1).html()
        var count=$("#calllog").find("tr[number="+findnumber+"]").length
        $("#"+i).find("td").eq(2).text(count)
    }

    $("#contact").tablesorter({//세번째 필립한 정렬 기능 활성화
        headers: {
            0: {
                sorter: false
            },
            1: {
                sorter: false
            },
        },
    });

    $("#tr").click(function(){//연락처에서 선택시 전화기록에 해당 연락처 관련 기록만 보여준다.
        if(this.id!=""){
            //toggle 방식이었는데 직관적인 if문으로 그냥 썼다.
            $(this).attr("flag", "true")

            if($("#"+this.id).attr("style")=="undefined"){//선택한 줄 색깔
                $(this).css("background-color", "#b2ddef")
            }else{
                $(this).css("background-color", "")
                $(this).attr("flag", "false")
            }

            var findnumber=$("#"+this.id).find("td").eq(1).html()

            if($("#"+this.id).attr("flag")=="true"){
                $("#calllog").find("tr[number="+findnumber+"]").attr("flag", "true")
            }else{
                $("#calllog").find("tr[number="+findnumber+"]").attr("flag", "false")
            }

            $("#calllog").find("tr[flag=false]").hide()
            $("#calllog").find("tr[flag=true]").show()

            if($("#contact").find("tr[flag=true]").length==0){//아무것도 선택할 없을때 다시 초기 전화기록
                $("#calllog").find("tr").show()
            }
        }
    });

    $("#tr").trigger("click");//간재로 두번 클릭을 발생시켜서 정렬
    $("#tr").trigger("click")
});
```

그림 24 원하는 전화기록만 선택하는 코드

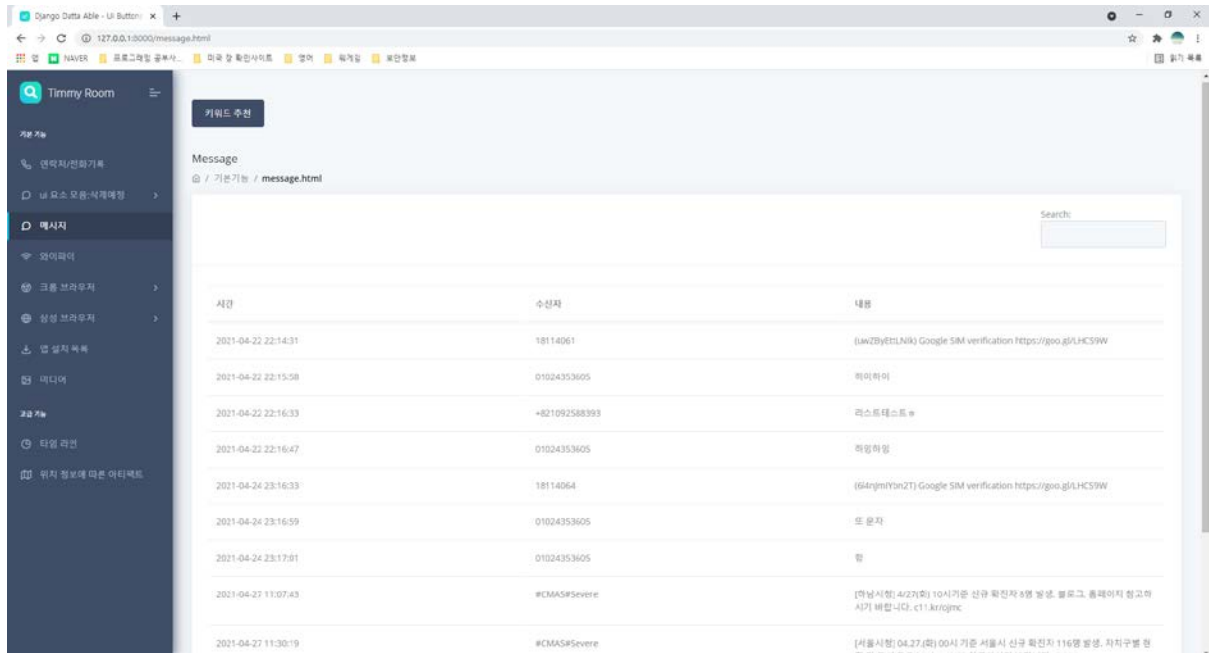


그림 25 메시지 화면

메시지, 브라우저의 검색 기록 등은 키워드 추천기능이 있어서 어떤 키워드를 많이 쳤었는지 보여줄 수 있다. 또한 메시지로 도착한 사진을 화면에 보여줄 수 있다.

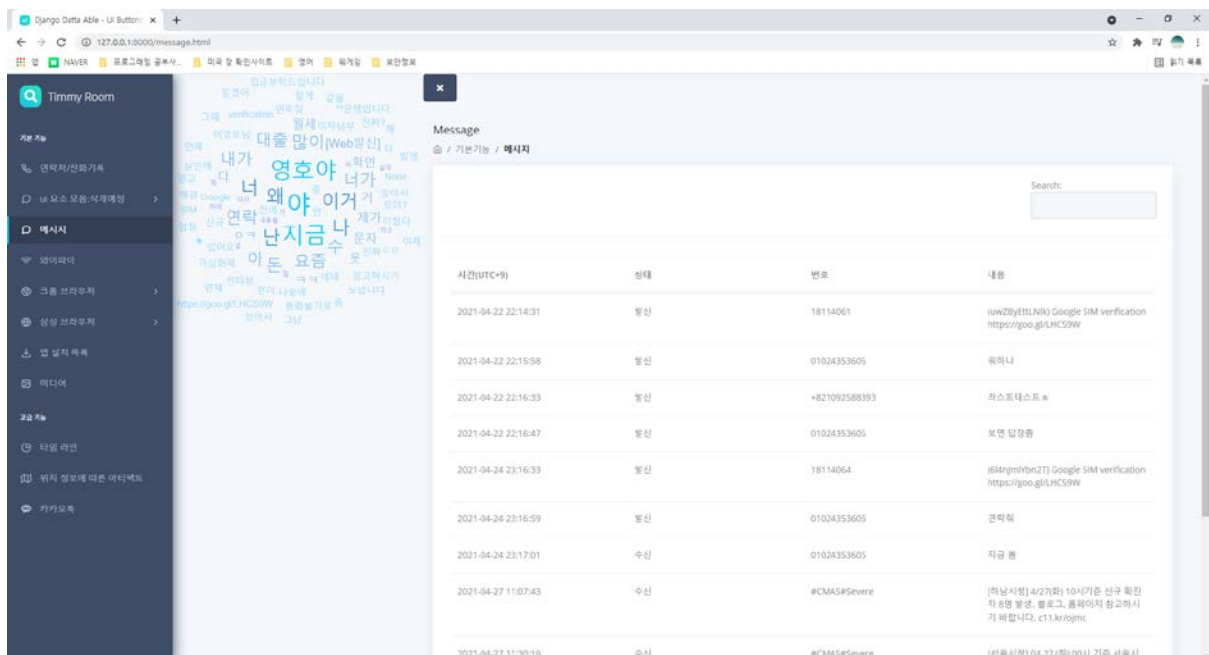


그림 26 키워드 추천 기능 화면

키워드를 추천하는 코드를 다음과 같다.

```
def count(inputli,key):
    #####
    c=list()
    for a in inputli:#쿼리셋->list
        c.append(a[key])
    litost=" ".join(map(str,c)) #list를 전체 문자열로 만든다.

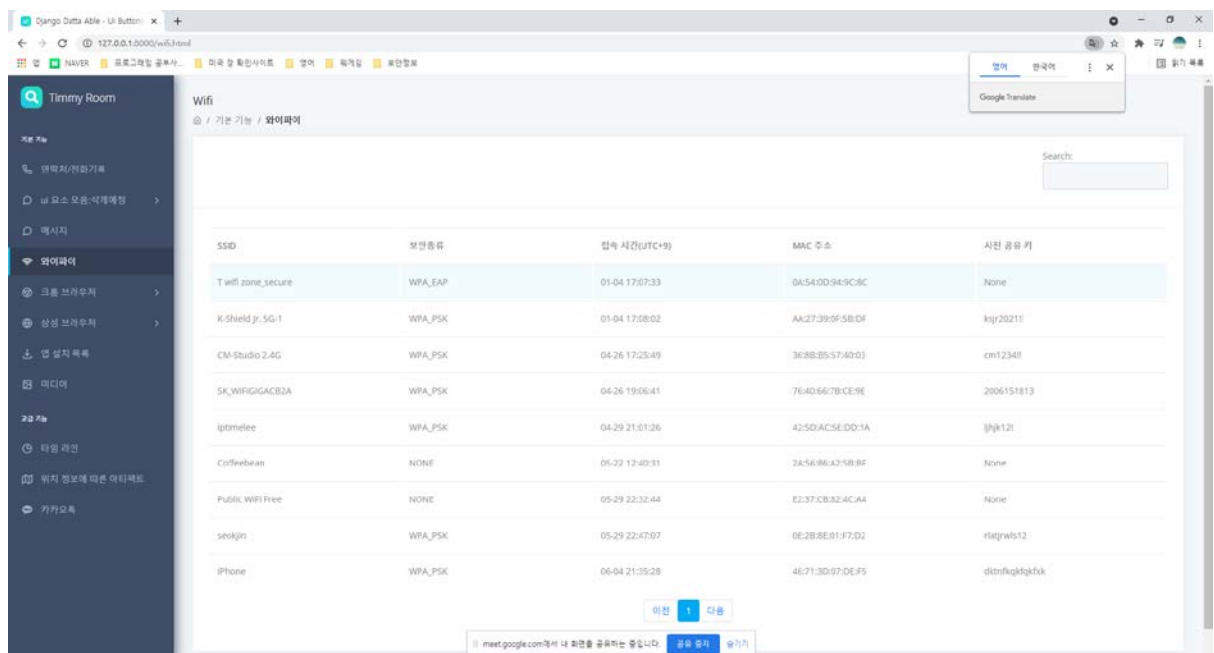
    c=litost.split()#중복이 있는 list
    list1=set(c)
    list1=list(list1)#중복이 제거된 list

    list2=list()
    list3=list()
    for text in c:
        list2.append({"text":text})
    for text in list1:
        list3.append({"text":text})
    a=1

    for a in list2:
        find =a['text']
        for e in list3:
            if find==e['text']:
                try:e['weight']=e['weight']+1
                except:e['weight']=1
    for e in list3:
        e['html']="{\"title\":f\"빈도수:{e['weight']}\"}"

    return list3
```

그림 27 키워드 추천 코드



SSID	보안종류	접속 시간(UTC+9)	MAC 주소	시작 공유 키
T wifi_zone_secure	WPA_EAP	01-04 17:57:33	0A:54:00:94:9C:8C	None
K-Shield Jr.-5G-1	WPA_PSK	01-04 17:58:02	AA:27:39:0F:5B:DF	hjr/2021f
CM-Studio 2.4G	WPA_PSK	04-26 17:25:49	36:8B:B5:57:AD:03	cm1234f
SK_WIFI@GACB2A	WPA_PSK	04-26 19:06:41	76:AD:66:7B:CE:9E	2006151813
iptimelee	WPA_PSK	04-29 21:51:26	42:5D:AC:5E:DD:1A	hjh12f
Coffeebean	NONE	05-22 12:40:31	2A:56:86:A2:5B:8F	None
Public WiFi Free	NONE	05-29 22:32:44	E2:37:CB:82:4C:A4	None
seokjin	WPA_PSK	05-29 22:47:07	0E:2B:8E:01:F7:D2	rlatgrws12
iPhone	WPA_PSK	06-04 21:35:28	46:71:3D:07:DE:F5	distnfkgkdkfuk

그림 28 와이파이 화면

AP name, 최초 접속시간, MAC 주소, 비밀번호 등의 정보들이 나타난다.

와이파이는 .xml 파일에 들어있어 파싱하여 데이터를 보여준다. 코드는 다음과 같다.

```
def xml_parsing(file_path):
    file_path=f'{file_path}/misc/wifi/WifiConfigStore.xml'
    doc = ET.parse(file_path) # xml파일 열기
    root = doc.getroot() # root가 xml문서의 최상위 태그를 가리킴
    networklist = root.find("NetworkList")
    wifi_list = []
    for networks in networklist.findall("Network"):
        i = 0
        for WifiConfiguration in networks.findall("WifiConfiguration"):
            net_info = {}
            SSID_list = WifiConfiguration[0].text.split("\\") # SSID
            SSID = SSID_list[1]
            encryption = SSID_list[2]
            creation_time = WifiConfiguration[30].text # 생성시간
            creation_time = creation_time[5:19]
            MAC = WifiConfiguration[36].text # MAC주소
            pre_shared_key = WifiConfiguration[3].text # 키
            #원하는 값 = WifiConfiguration[각인번호].text를 사용하여 출력
            if pre_shared_key != None:
                pre_shared_key = pre_shared_key.replace("\\", "")
                net_info['SSID']=SSID
                net_info['encryption']=encryption
                net_info['creation_time']= creation_time
                net_info['MAC']=MAC.upper()
                net_info['pre_shared_key']=pre_shared_key
                #net_info.extend([SSID, encryption, creation_time, MAC.upper(), pre_shared_key])
            wifi_list.append(net_info)
    # print(wifi_list)
    """리턴 값 예시) [ [],[],[],[],[] ] """
    return wifi_list # wifi_list 안에 net_info 리스트 반환
```

그림 29 xml파일 시각화 코드

브라우저 탭에는 브라우저 히스토리, 브라우저 다운로드, 브라우저 검색 기록 등이 있으며 각 화면은 다음과 같다.

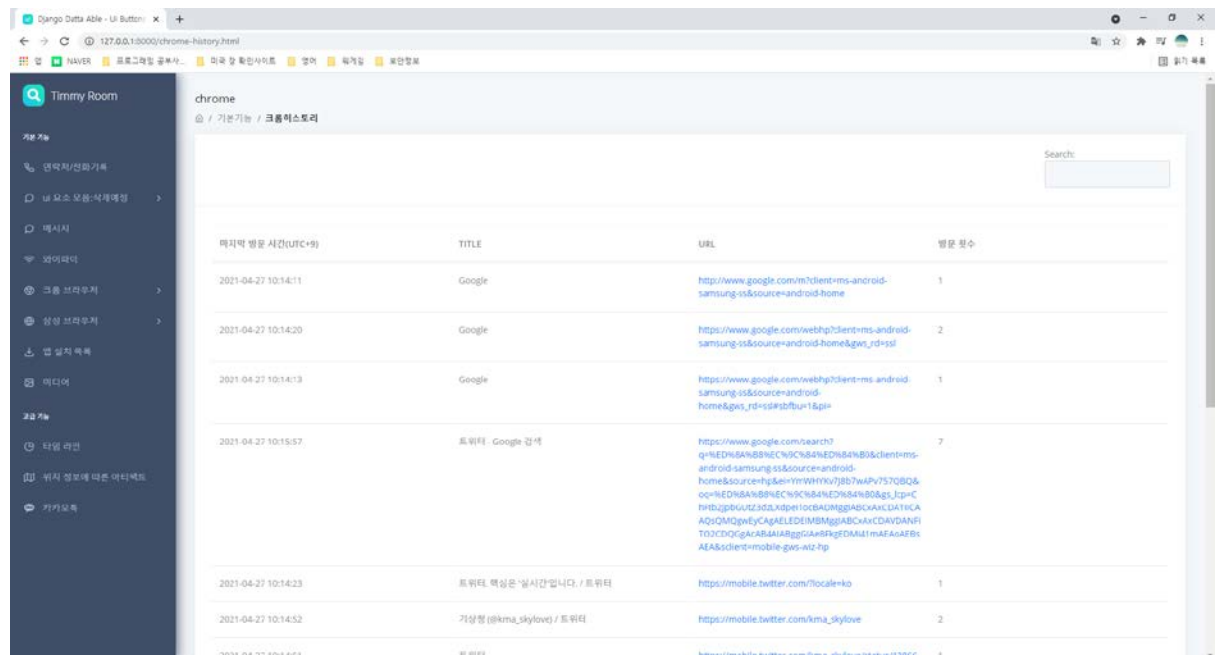
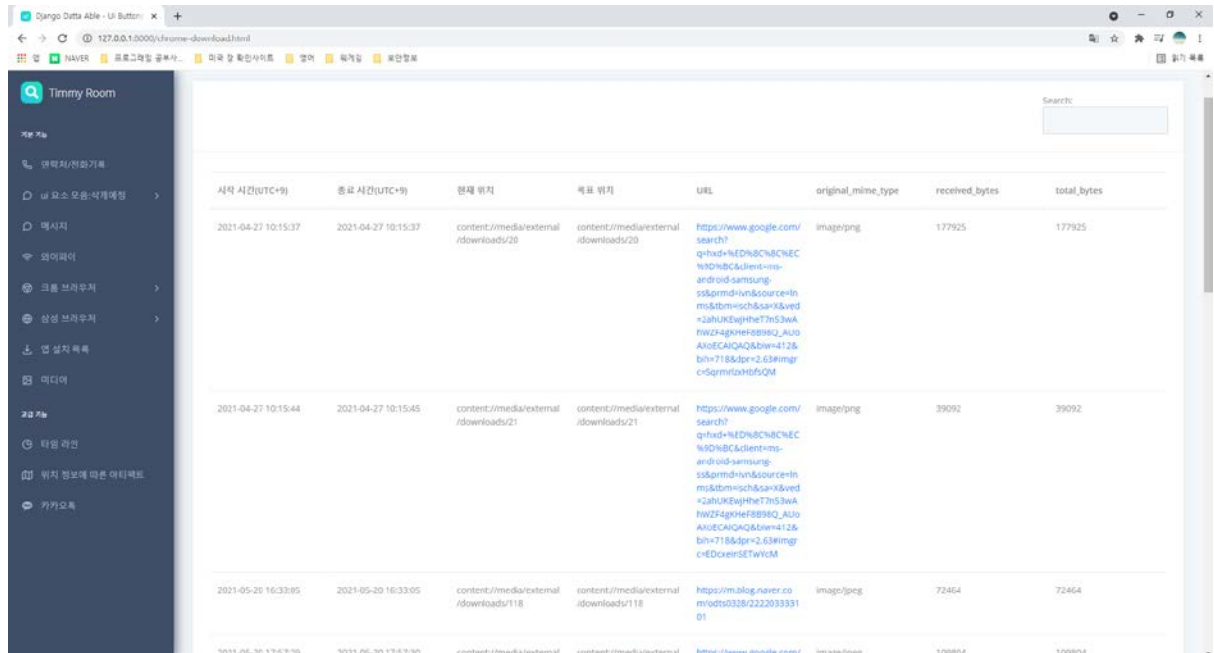


그림 30 크롬 히스토리 화면

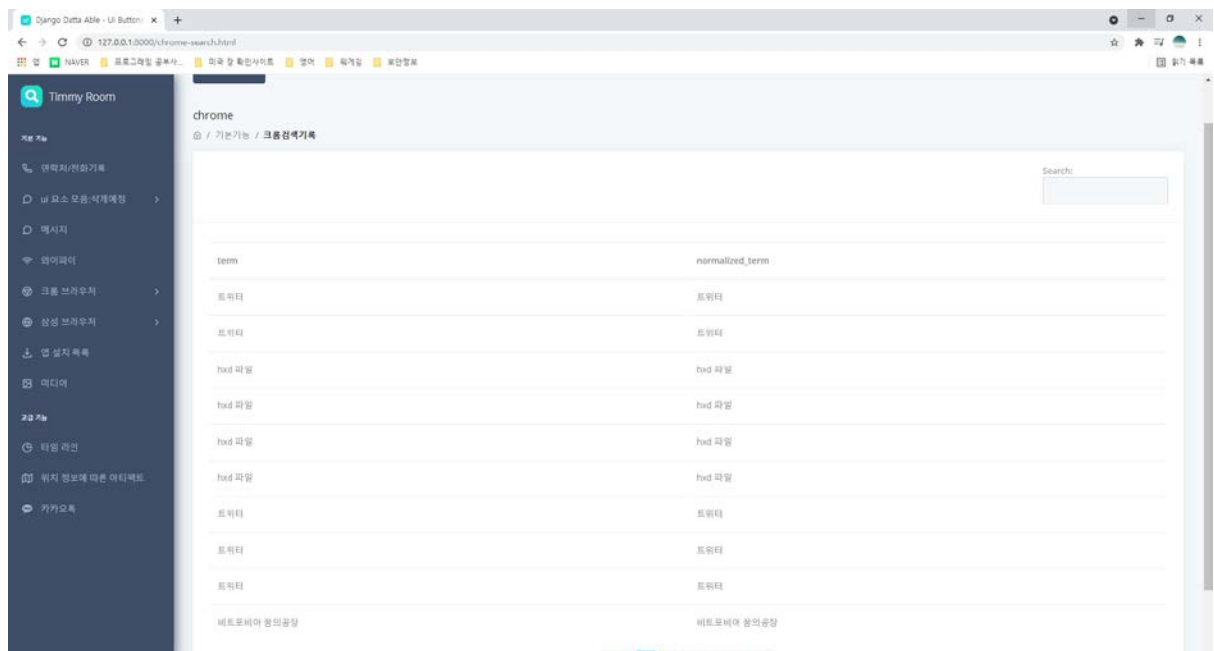
마지막 방문 시간, TITLE, URL, 방문 회수 등이 나오며 URL 을 클릭할 경우 인터넷과 연결하여 페이지를 보여준다.



시작 시간(UTC+9)	종료 시간(UTC+9)	현재 위치	목표 위치	URL	original_mime_type	received_bytes	total_bytes
2021-04-27 10:15:37	2021-04-27 10:15:37	content://media/external/downloads/20	content://media/external/downloads/20	https://www.google.com/search?q=hd+%ED%8C%BC%9D%BC&client=ms-android-samsung-ss&gclid=ea126&bi=718&ip=2.63img/c=5&me=5&f=QM	image/png	177925	177925
2021-04-27 10:15:44	2021-04-27 10:15:45	content://media/external/downloads/21	content://media/external/downloads/21	https://www.google.com/search?q=hd+%ED%8C%BC%9D%BC&client=ms-android-samsung-ss&gclid=ea126&bi=718&ip=2.63img/c=5&me=5&f=QM	image/png	39092	39092
2021-05-20 16:33:05	2021-05-20 16:33:05	content://media/external/downloads/118	content://media/external/downloads/118	https://m.blog.naver.com/odts038/22203333101	image/jpeg	72464	72464
2021-05-20 17:57:26	2021-05-20 17:57:30	content://media/external/downloads/119	content://media/external/downloads/119	https://www.ancle.com/	image/jpeg	109804	109804

그림 31 브라우저 다운로드 화면

시작 시간, 종료 시간, 현재 위치, 목표위치, URL, orinial_mime_type, received_bytes, total_bytes 등의 컬럼이 존재하며 URL을 누르면 받았던 페이지로 이동한다.



term	normalized_term
트위터	트위터
트위터	트위터
하드 파일	하드 파일
하드 파일	하드 파일
하드 파일	하드 파일
하드 파일	하드 파일
하드 파일	하드 파일
트위터	트위터
트위터	트위터
트위터	트위터
비트코인	비트코인

그림 32 브라우저 검색 기록 화면

브라우저 검색기록은 구글에서 검색한 내용 등을 기록하고 있으며 이를 통하여 키워드 추천으로 자주 검색했던 내용을 볼 수 있다.

앱 이름	패키지명	다운로드 완료 시간(UTC+9)	다운로드 요청 시간(UTC+9)	마지막 업데이트 시간(UTC+9)
android.autoinstalls.config.samsung	android.autoinstalls.config.samsung	2021-04-26 14:00:24	2021-04-26 14:00:12	2021-04-26 14:01:33
Termux	com.termux	2021-04-26 14:11:46	2021-04-26 14:11:46	1970-01-01 09:00:00
Bill Letter	com.skt.smartbill	2021-04-26 14:33:40	2021-04-26 14:33:39	2021-04-26 14:33:51
11번가	com.elevenst	2021-04-26 14:33:16	2021-04-26 14:33:16	2021-04-26 14:33:25
File Explorer Root Browser	com.jummy.root.browsefree	2021-04-26 14:11:33	2021-04-26 14:11:32	1970-01-01 09:00:00
멜론	com.lben.melon	2021-04-26 14:34:19	2021-04-26 14:34:18	2021-04-26 14:34:32
Root Checker	com.joeykrim.rootcheck	2021-04-26 14:02:40	2021-04-26 14:02:38	1970-01-01 09:00:00
None	com.socialmobile.dictpass notepad.color note	1970-01-01 09:00:00	1970-01-01 09:00:00	1970-01-01 09:00:00
카카오톡 KakaoTalk	com.kakao.talk	2021-04-26 19:11:17	2021-04-26 19:11:17	1970-01-01 09:00:00
삼성 이메일	com.samsung.android.email.provider	2021-04-26 14:36:24	2021-04-26 14:36:23	2021-04-26 14:36:39

그림 33 앱 설치 목록 화면

앱을 설치한 목록들을 확인할 수 있으며, 앱 이름, 패키지명, 다운로드 완료 시간, 다운로드 요청 시간, 마지막 업데이트 등의 기록을 알 수 있다.

날짜 및 시간 (UTC+9)	생성 도구	Path
2021-04-27 10:15:37	com.android.chrome	
2021-04-27 10:15:44	com.android.chrome	
2021-04-27 10:15:35	None	
2021-04-27 10:25:33	None	/static/assets/images/media/0/DCIM/Camera/20210427_102533.jpg
2021-04-27 10:38:36	None	/static/assets/images/media/0/DCIM/Camera/20210427_103827.mp4

그림 34 미디어 화면

사진들의 시간, 생성 도구 이미지 경로 등이 나타나고 원하는 행 클릭 시 해당 이미지가 보인다.

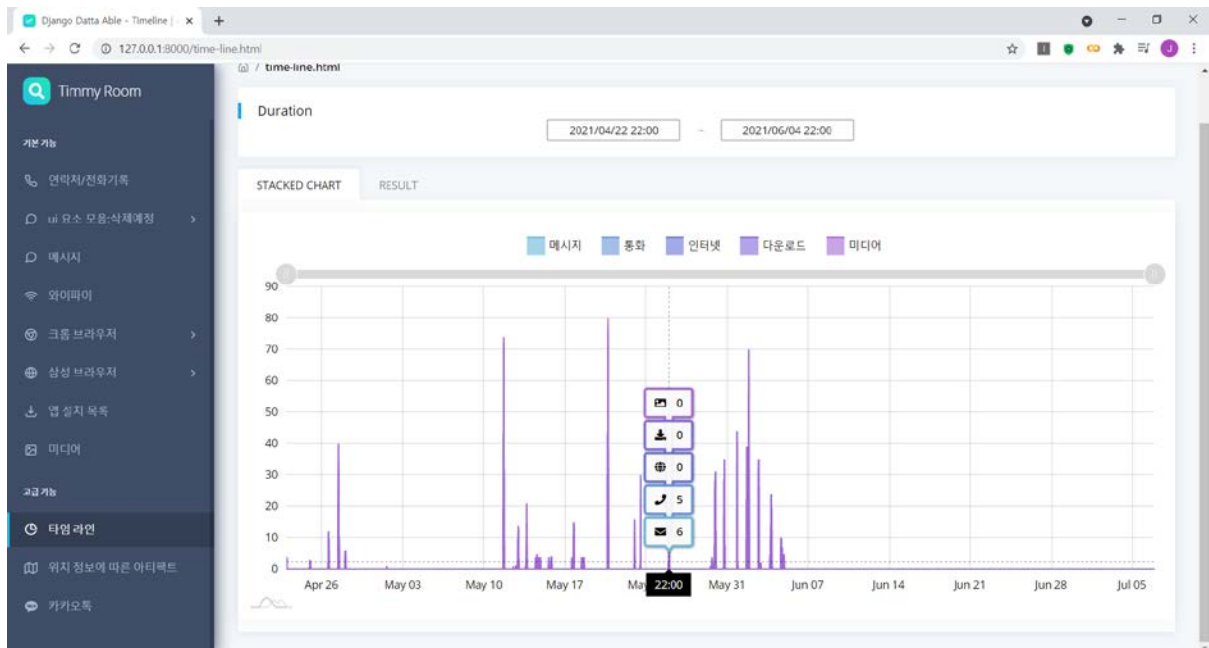


그림 35 타임라인 화면

STACKED CHART 탭에서는 메시지, 통화, 인터넷, 다운로드, 미디어 등의 시간기록을 바탕으로 하여 빈도수를 각각 표시하여 그래프로 보여준다.

The screenshot shows the 'RESULT' tab of the application. It displays a table of activity details. The table has columns for a category icon, timestamp, status, phone number, and a description. The data rows show various activities like '생성 도구' (Creation tool) and '상태' (Status) with specific timestamps and phone numbers. A '전체' (All) dropdown is visible in the top right corner of the table area.

Category	Timestamp	Status	Phone Number	Description
Media	2021/06/05 01:07:11	생성 도구	Camera	저장 위치 /static/assets/images/media/SDCIM/Camera/20210605_010711.jpg
Call	2021/06/05 01:07:41	상태	수신	번호 01075715910 통화 시간 0
Call	2021/06/05 01:14:25	상태	수신	번호 01089931596 통화 시간 0
Call	2021/06/05 01:16:01	상태	수신	번호 01089931596 통화 시간 0
Call	2021/06/05 01:28:01	상태	수신	번호 01089931596 통화 시간 0
Call	2021/06/05 01:28:53	상태	수신	번호 01089931596 통화 시간 0
Message	2021/06/05 01:30:17	상태	수신	번호 01089931596 내용 영우와 여디와 장수가 많이 다져서 우리 병원은 가 문자 확인하면 연락함

그림 36 타임라인 화면2

RESULT 탭에는 문자, 통화, 인터넷, 다운로드, 미디어 등의 것들로 시간안에 있었던 데이터들을 상세하게 볼 수 있다.

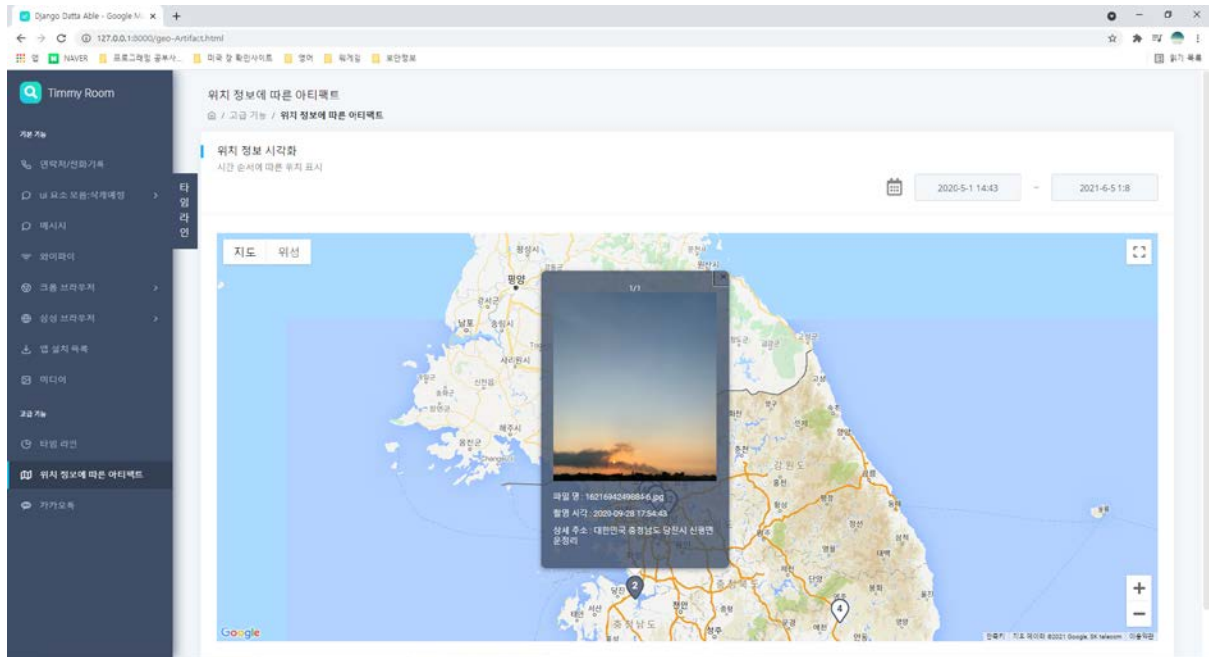


그림 37 위치 정보에 따른 아티팩트 화면

오른쪽에 있는 date picker 의 기간을 조정하여 원하는 시간대의 사진을 화면의 마크로 순서대로 보여준다. 이를 통해 동선을 파악할 수 있다. 마크를 클릭하면 사진을 볼 수 있다.

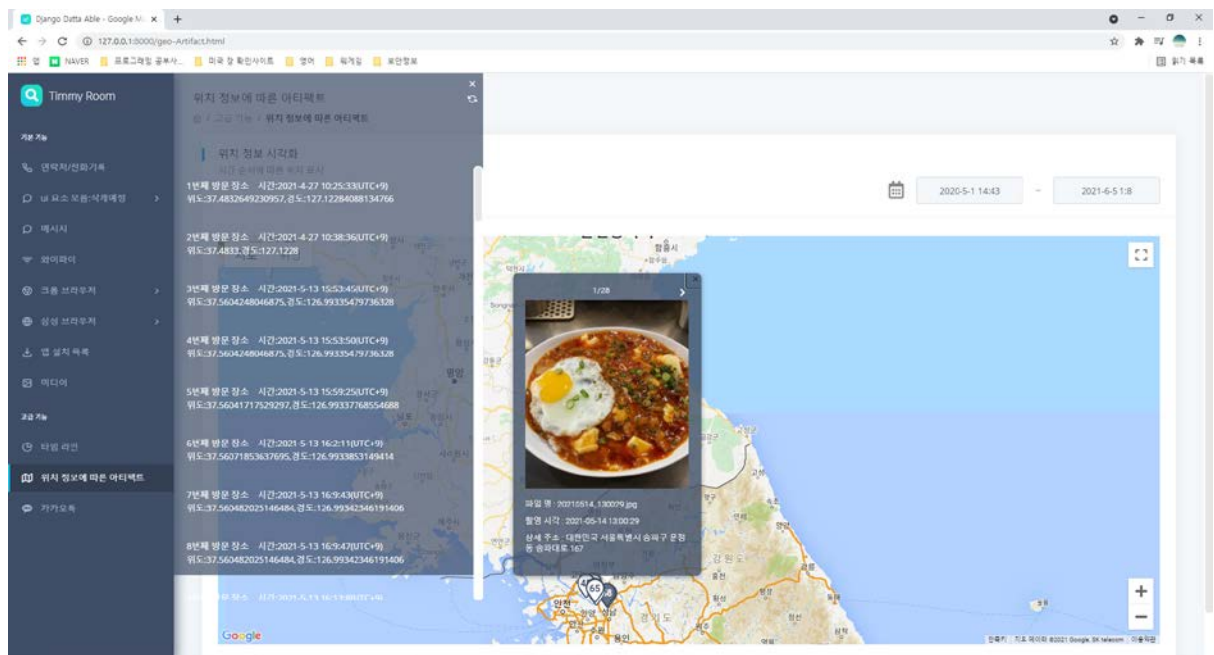


그림 38 위치 정보에 따른 아티팩트 화면2

타임라인 버튼을 이용하여 중복된 사진들을 하나하나 확인할 수 있다. 사진들의 시간을 보여주면서 위치정보가 같아 중복된 마크로 헛갈리게 되는 경우를 줄인다.

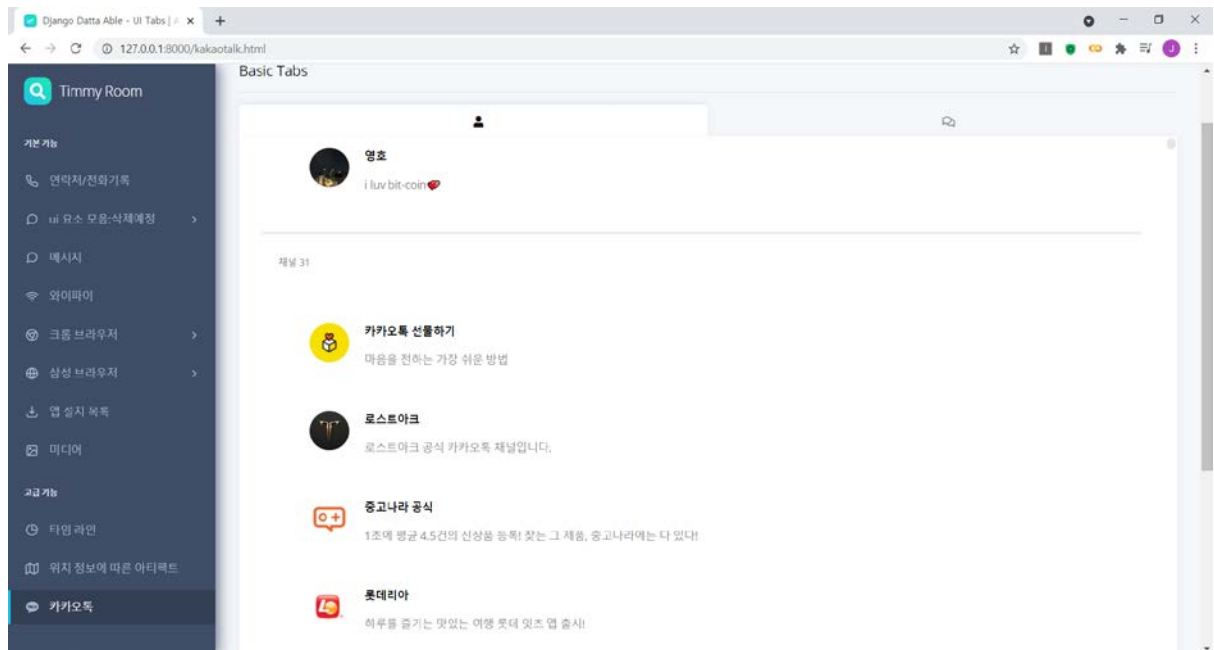


그림 39 카카오톡 화면

- 카카오톡 화면

카카오톡의 친구와 채팅 내역을 볼 수 있다.

카카오톡 복호화 코드가 존재하여 카카오톡 내용이나, 친구들을 볼 수 있으나, 개인정보 문제로 코드는 따로 올리지 않았습니다. (시나리오 데이터 포함)

7. 추후 연구 예정

- 사진데이터를 바탕으로 인물 식별 기능
- 키워드 분석 기능 -> 단순 추천이 아닌 연관할 수 있는 카테고리 분석
- 영상 데이터, 음성 데이터의 소리를 텍스트로 변환하여 보여주는 기능
- 삼성기기가 아닌 다른 기기들도 실행할 수 있도록 개발 예정
- Dd 파일에서 추출하거나, 더 나아가 휴대폰을 연결하면 바로 시각화 가능하도록 진행
- S 노트 시각화 기능
- 원하는 데이터 PDF로 출력 기능

8. 관련 자료

<https://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-202103-202103-bar>

<https://gs.statcounter.com/os-market-share/mobile/south-korea#monthly-202103-202103-bar>

<https://www.w3schools.com/>

<https://developer.mozilla.org/ko/docs/Web/JavaScript>

<https://www.kci.go.kr/kciportal/ci/sereArticleSearch/ciSereArtiView.kci?sereArticleSearchBean.artild=ART002550554>

<https://digitalis.postype.com/post/2290617>

http://forensic.korea.ac.kr/DFWIKI/index.php/%EC%8A%A4%EB%A7%88%ED%8A%B8_%EA%B8%B0%EA%B8%B0_%EC%95%B1

<https://scienceon.kisti.re.kr/srch/selectPORSrchArticle.do?cn=JAKO201633056056055>

<https://www.dbpia.co.kr/Journal/articleDetail?nodeId=NODE07480412>