**팀 프로젝트**

**모바일 프로그래밍**

**KPU코로나 알리미**

7팀

컴퓨터공학과 2018152010 김지안

컴퓨터공학과 2018152015 박재현

컴퓨터공학과 2018152043 최진혁

컴퓨터공학과 2018152044 한창현

2021.05.27(목)

목차

**1. 목적 및 개요 p.3**

**2. 제약사항 p.3**

**3. 역할 분담 p.3**

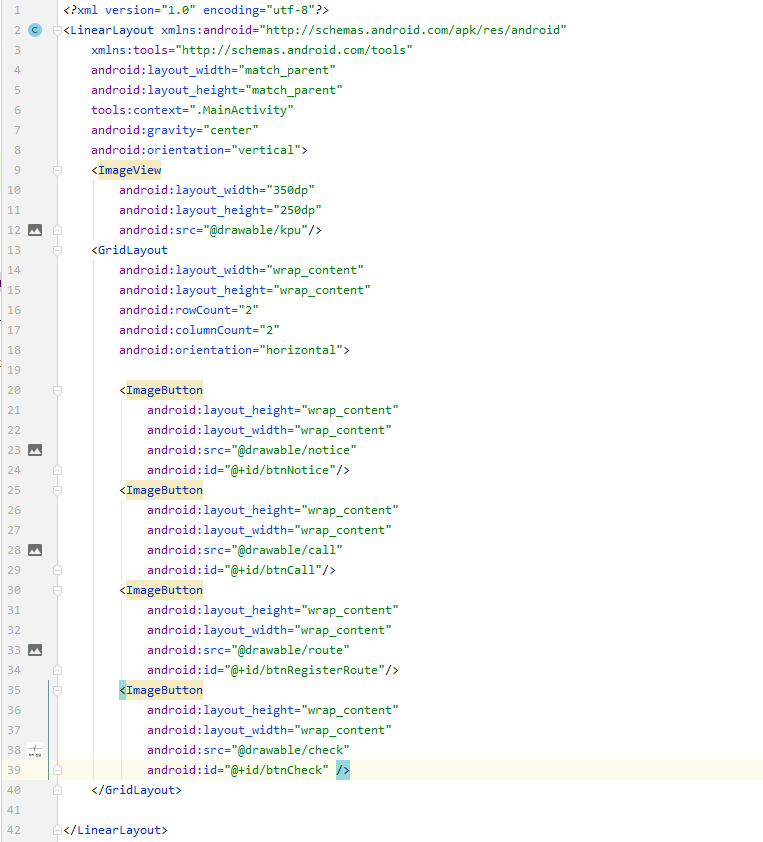
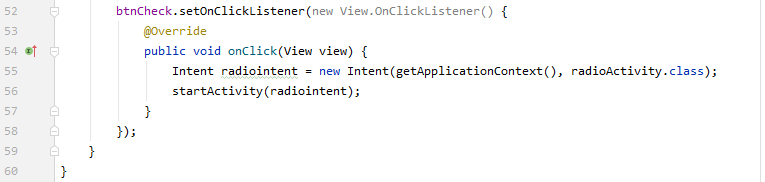
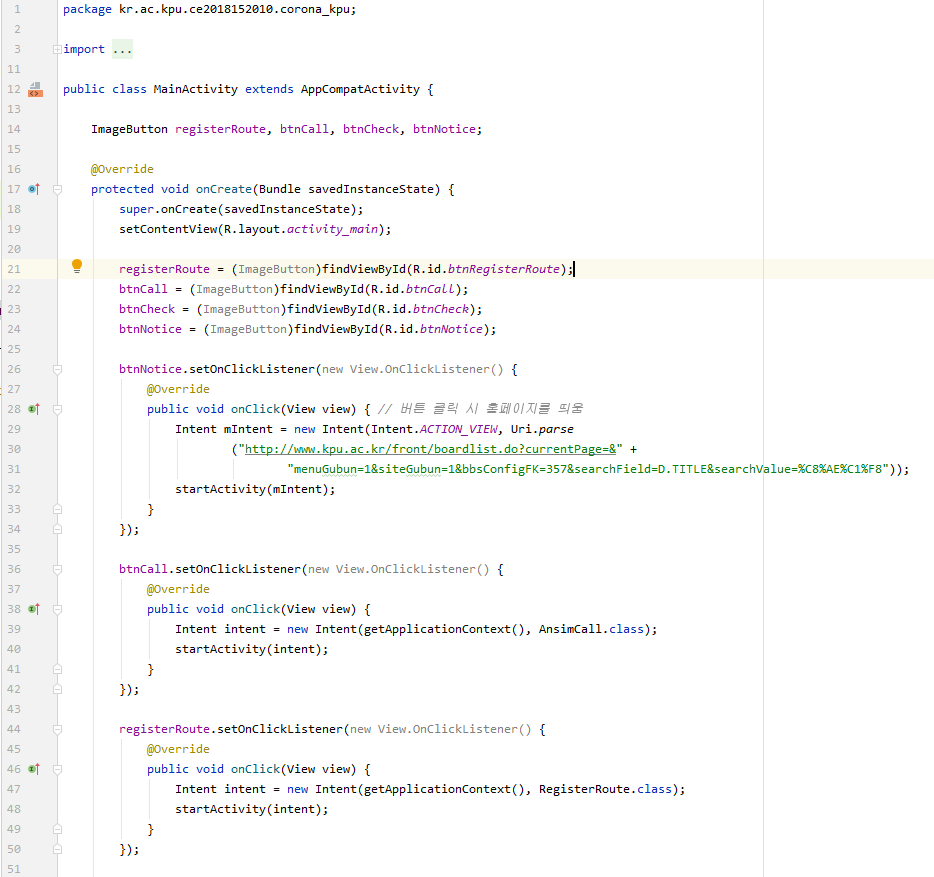
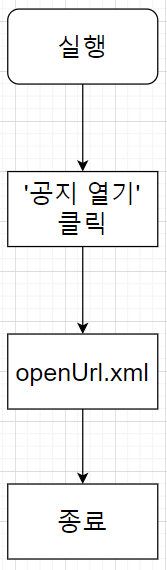
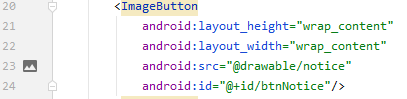
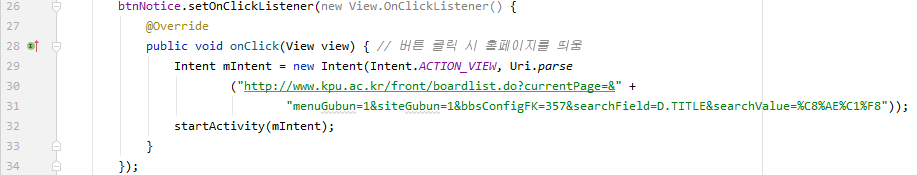
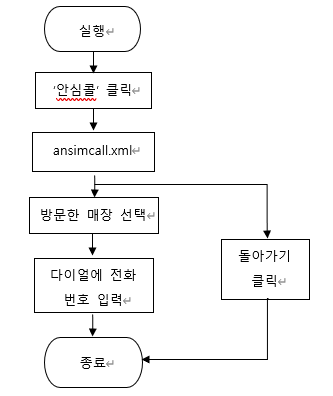
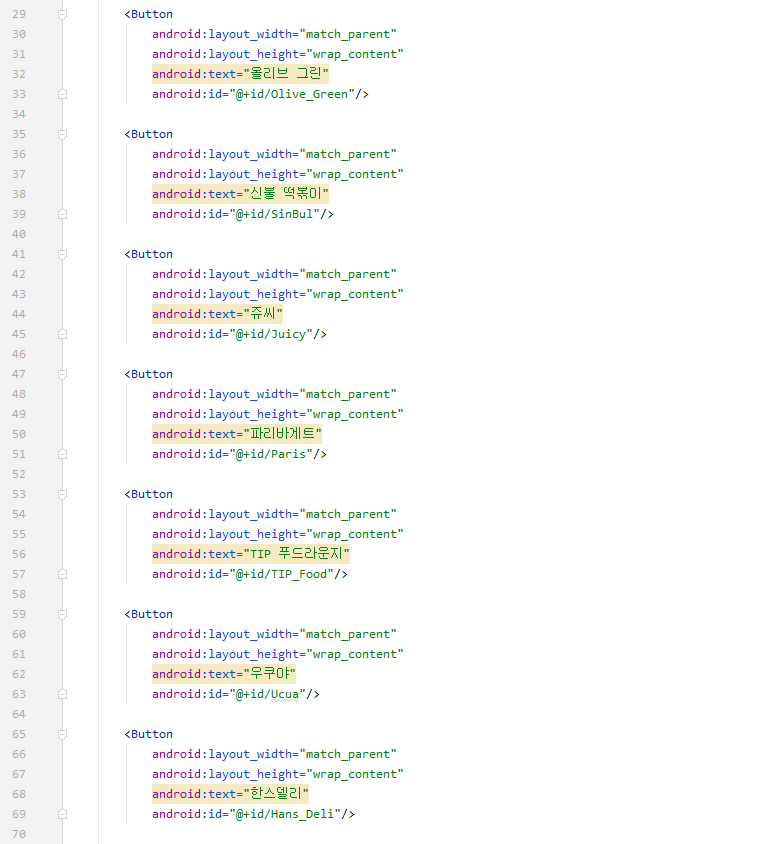
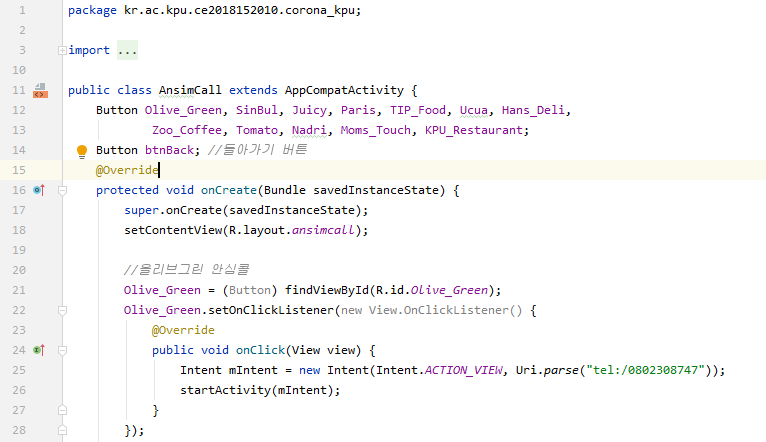
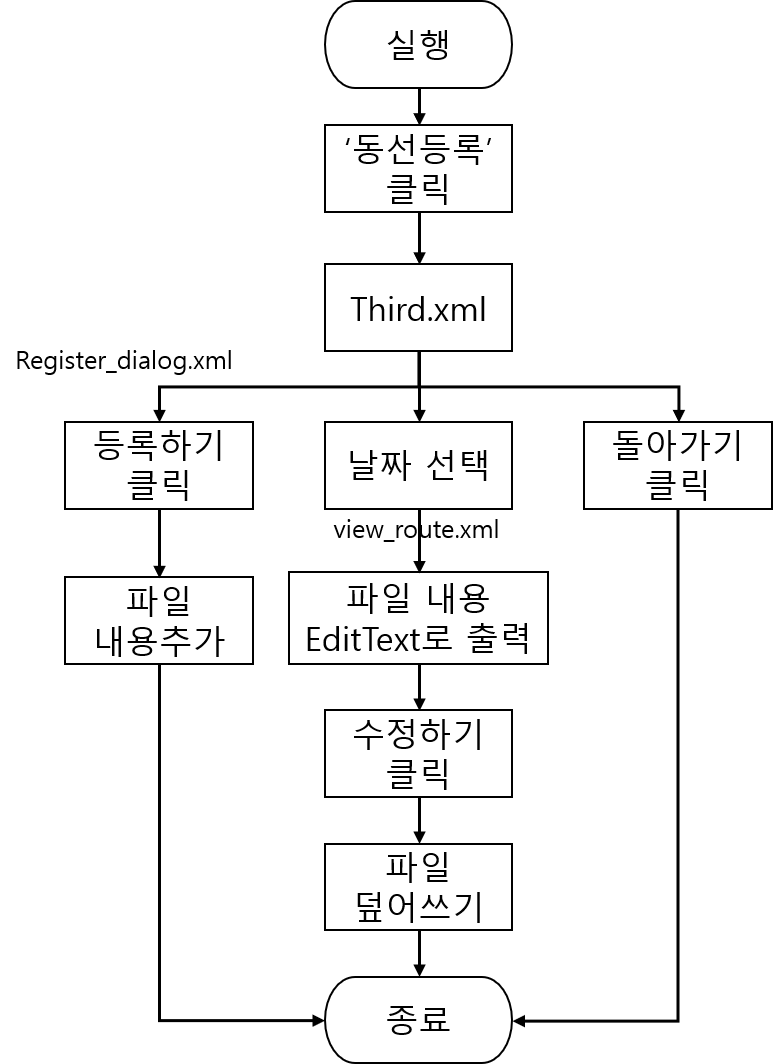
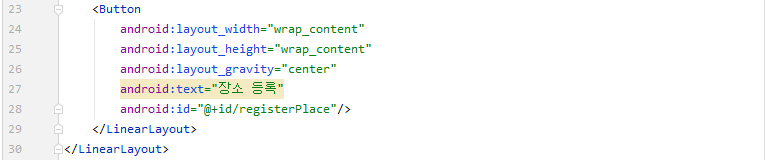
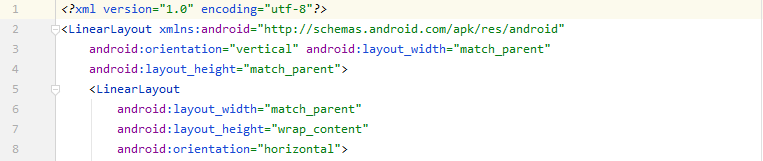
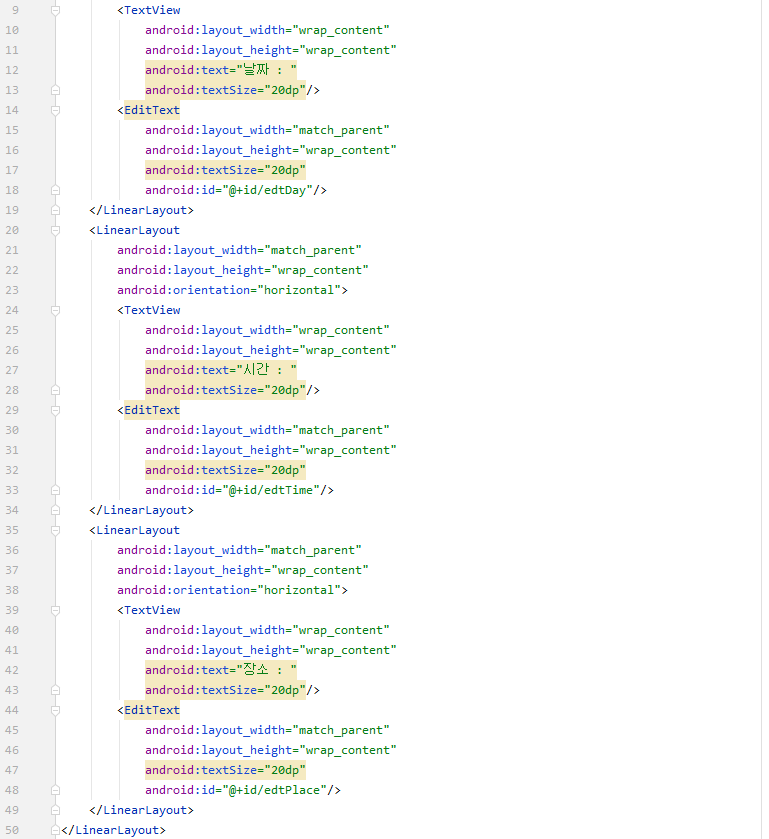
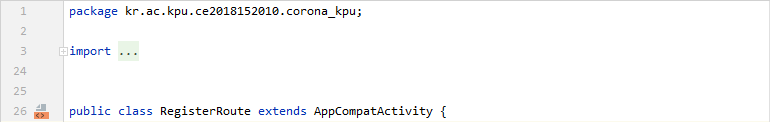
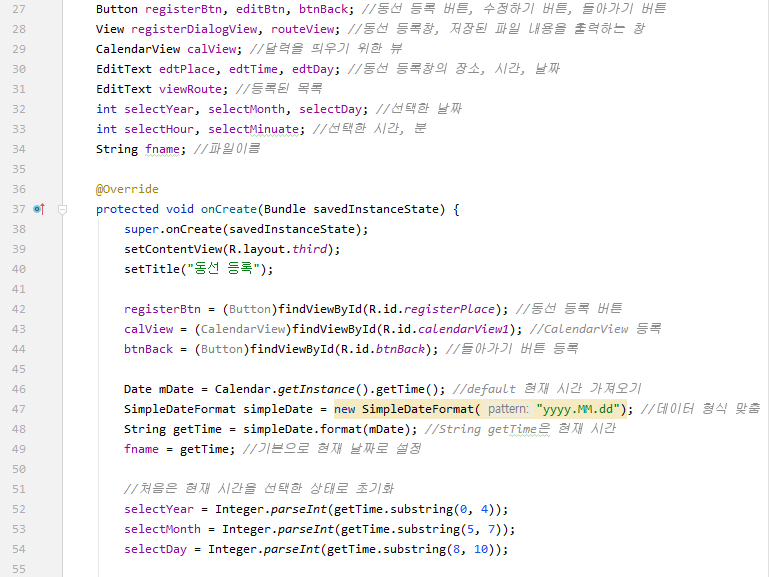
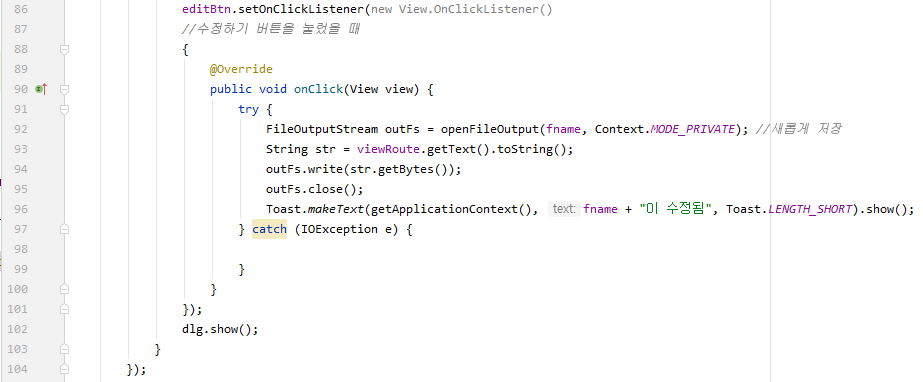
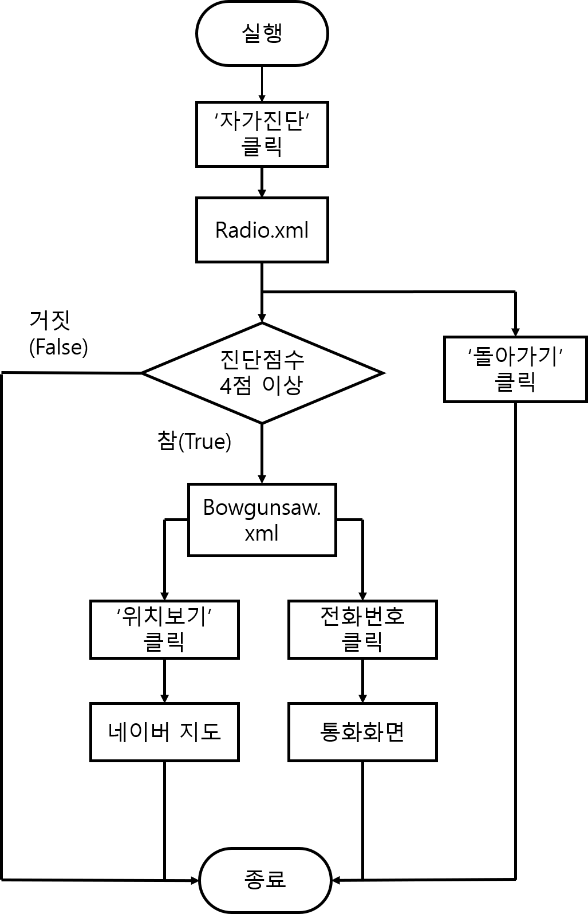
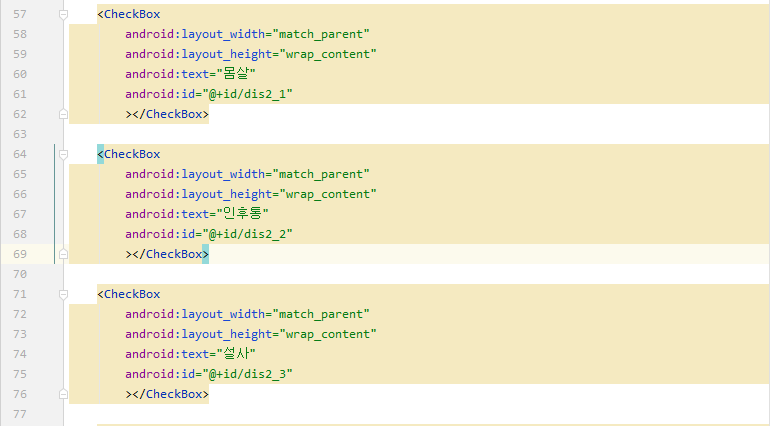
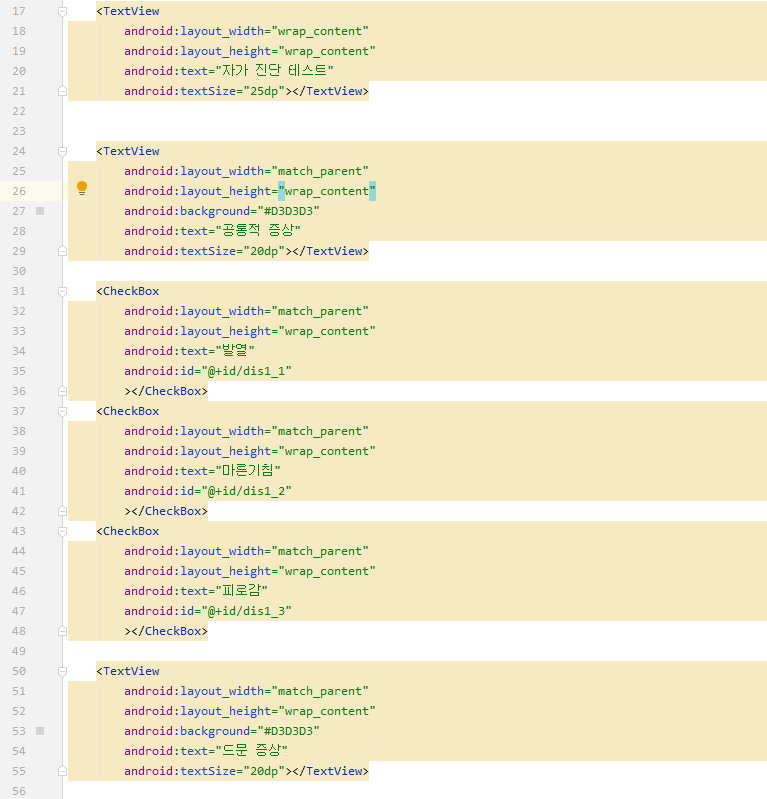
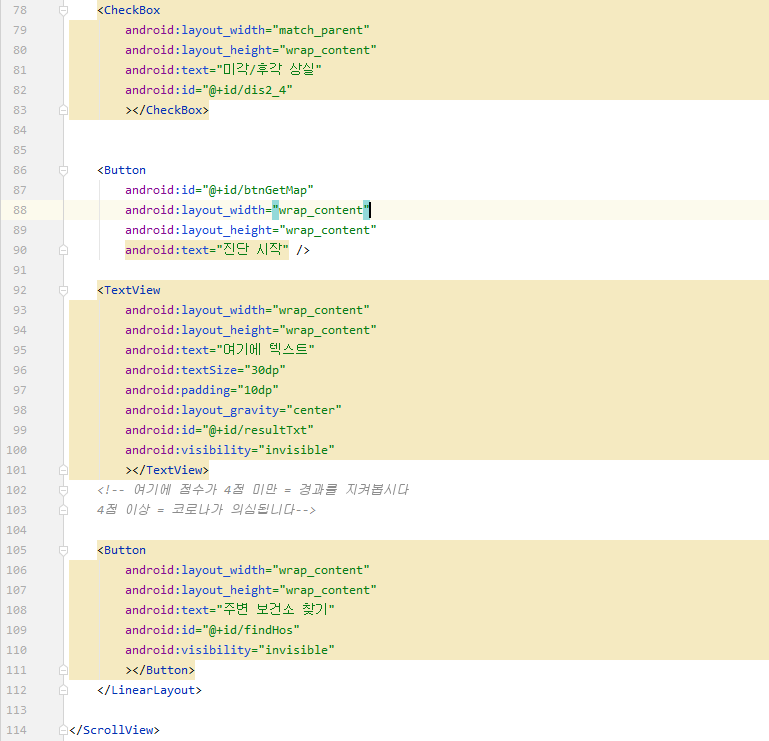
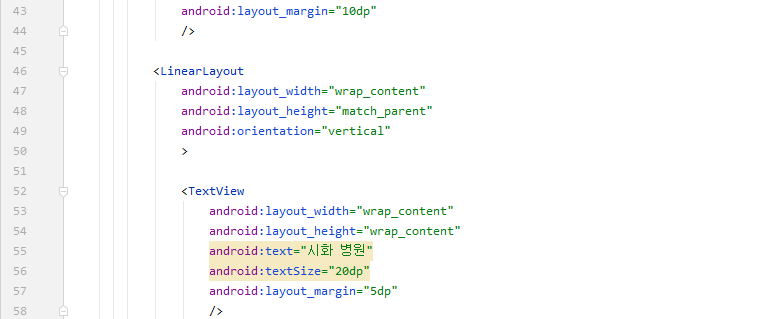
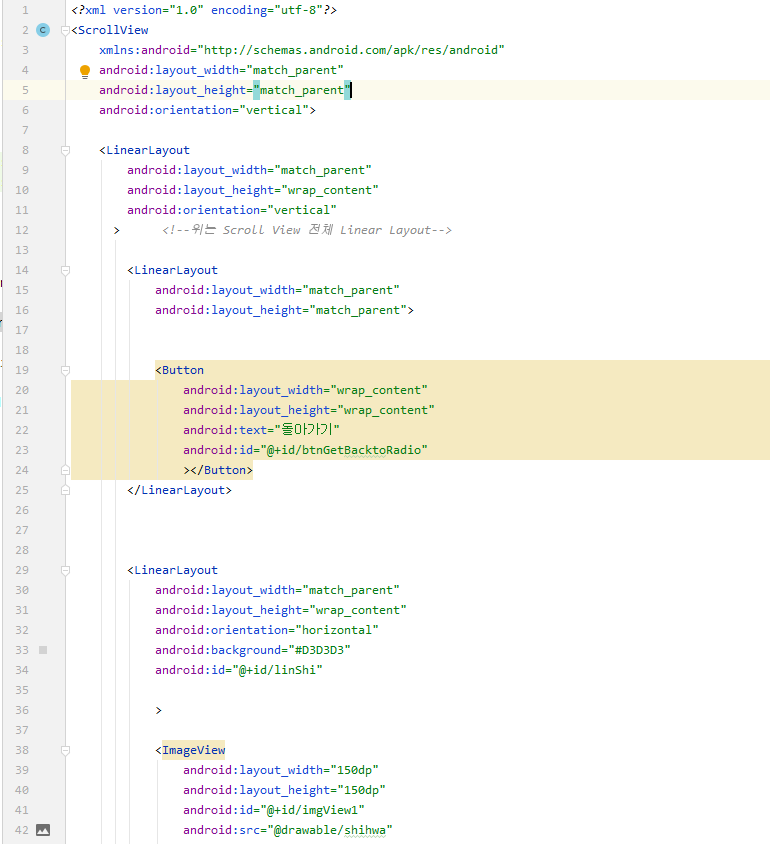
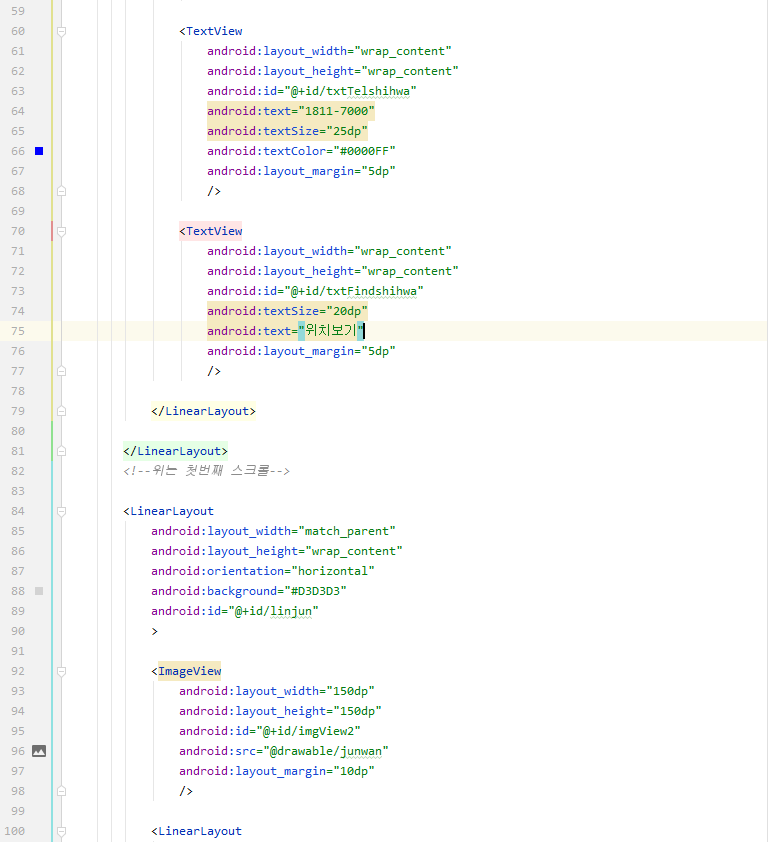
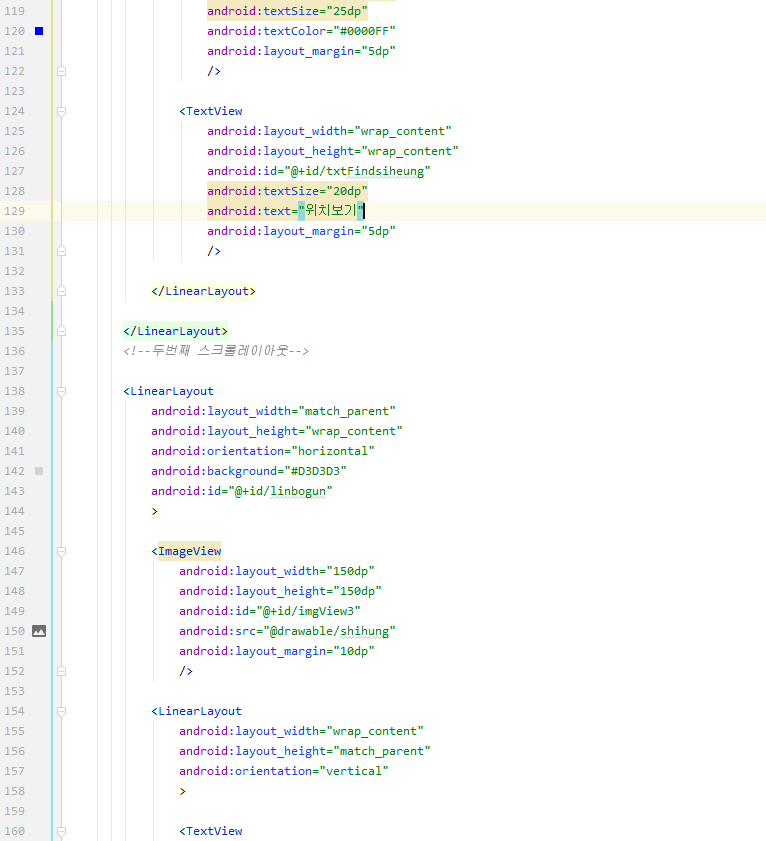
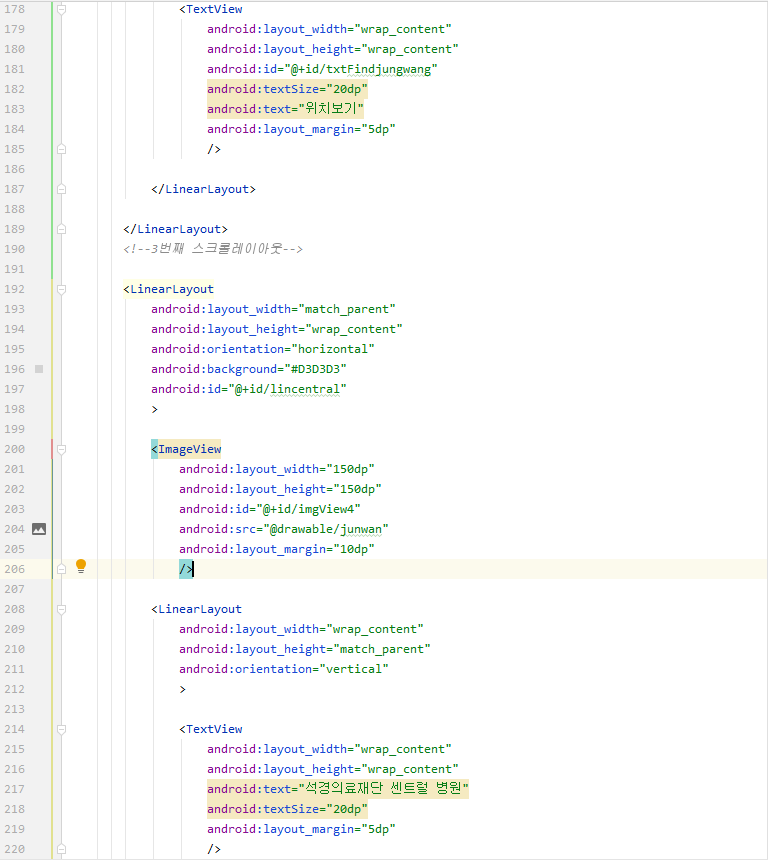
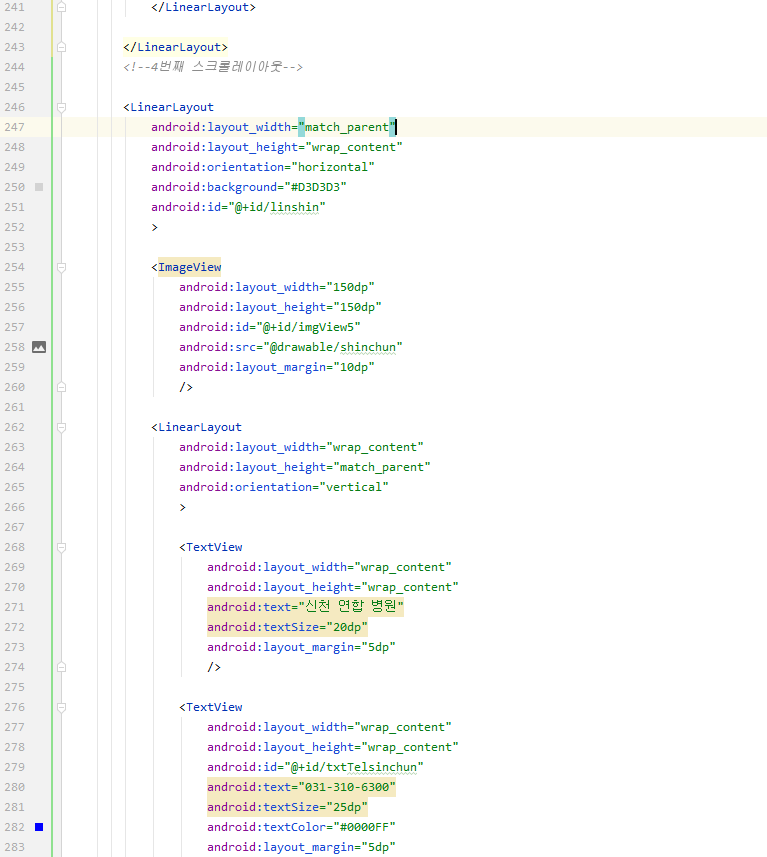
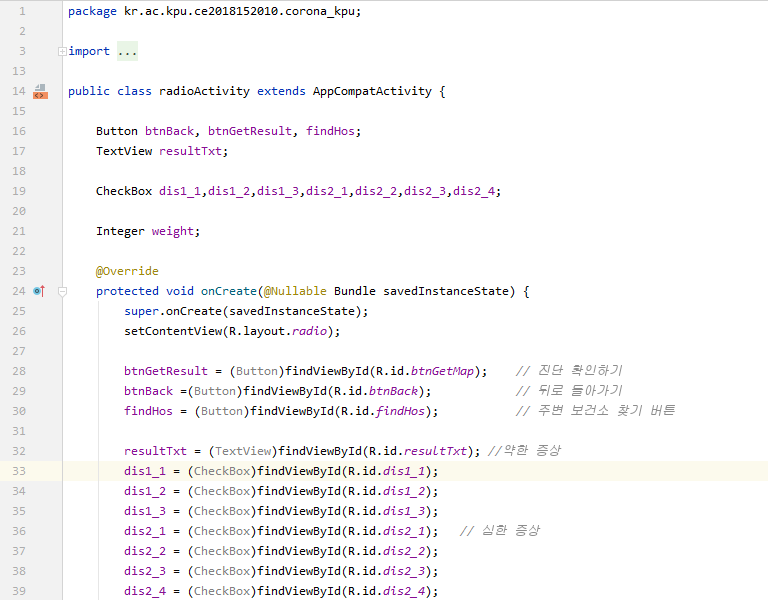
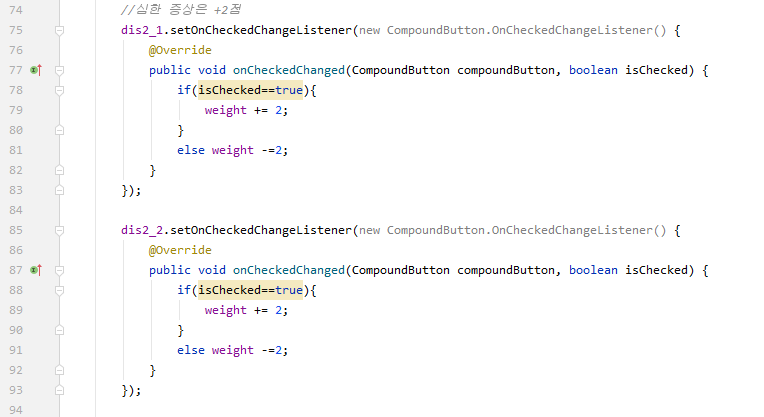
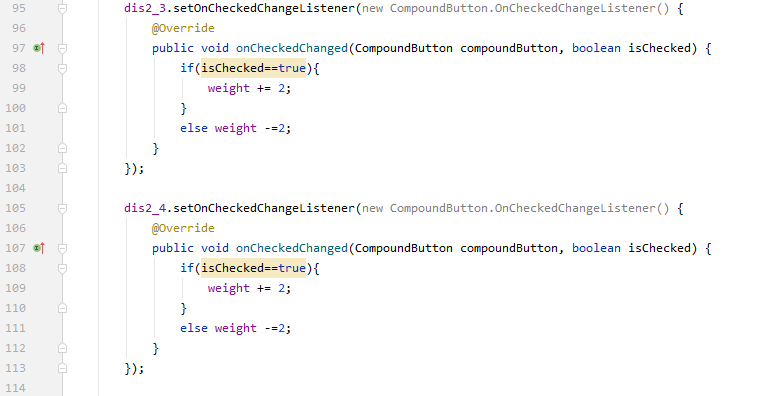
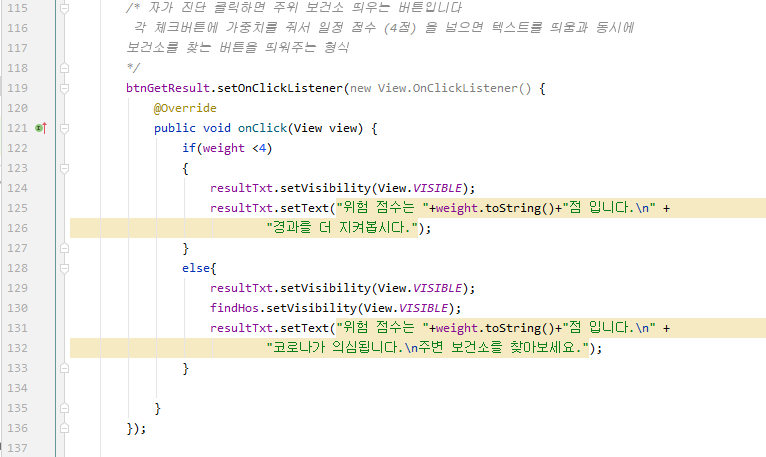
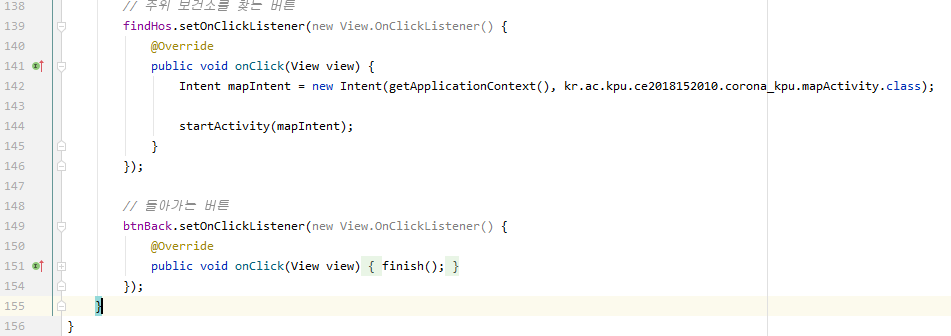
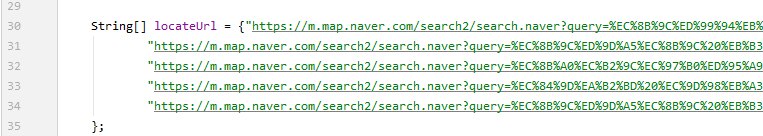
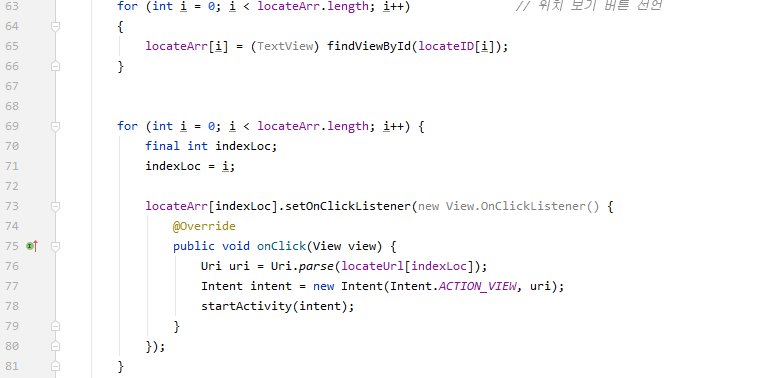
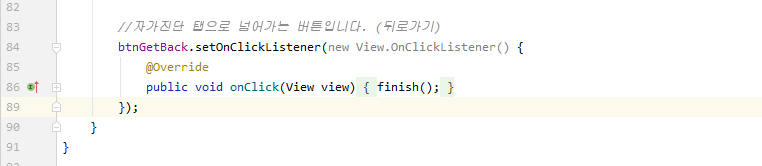
**4. 계획 일정 p.4**

**5. 앱 기능 p.5**

**6. 실행 화면 p.33**

**7. 느낀 점 p.36**

**8. 참고 자료 p.37**

1. **목적 및 개요**
   1. 개발 배경  
      - 코로나 사태에 코로나로 인해 발생하는 위험과 불안함 속에서 학교생활을 해야만 하는 학우들이 존재하고 이러한 요소들로 인해 생겨나는 불편함이 존재한다.
      1. 코로나로 인해 발생하는 불편한 상황들
         1. 교내 확진자 발생 시 세부 내용은 학교 홈페이지에서 직접 확인해야한다.  
            → 앱을 통해 해당 게시글을 바로 확인 가능한 기능을 제공한다.
         2. 자주 가는 교내 식당의 안심콜 번호를 매번 눌러야 하는 불편함이 존재한다.  
            → 버튼 하나로 안심콜이 가능한 기능을 제공한다.
         3. 방문했던 교내 동선을 일일이 기억하는 것의 한계가 존재한다.  
            → 실시간으로 앱을 통해 방문했던 장소와 일시 기록하는 기능을 제공한다.
         4. 단순 감기 증상을 코로나 감염으로 여기는 불안감이 존재한다.  
            → 자가 진단이 가능하고, 코로나가 의심될 경우 주변 보건소나 병원의 위치와 전화번호를 제공하는 기능을 개발한다.
   2. 목적  
      - 코로나 사태에서 대면수업에 참여하고 있는 학생들이 코로나 관련 정보를 빠르게 수집할 수 있도록 하고, 코로나로 인해 발생하는 불편한 상황들을 개선하는 것이다.
2. **제약 사항**- 4개 이상의 액티비티를 구현한다.  
   - 수업시간에 배운 내용을 기반으로 만든다.  
   - 액티비티 별로 별도의 java파일이 존재한다.  
   - 실제로 우리 사회에 도움이 되는 주제를 선정한다.  
   - 동선 데이터를 2주동안 유지한다.
3. **역할 분담**
   1. 코딩  
      - 4개의 액티비티를 팀원이 하나씩 맡아 구현한다.  
      - 김지안 : 동선 등록 구현  
      - 박재현 : 안심콜 구현  
      - 최진혁 : 공지 확인 구현  
      - 한창현 : 자가 진단 구현
   2. 보고서 작성  
      - 박재현 : 최종 보고서 작성 및 제출
   3. 발표  
      - 최진혁 : 발표 및 데모 시 팀원이 각각 구현한 부분에 대해 기술
4. **계획 일정**>> 5/8 ~ 5/16 : 코딩 구현  
   >> 5/17 ~ 5/19 : 코딩 통합  
   >> 5/20 ~ 5/24 : 보고서 작성  
   >> 5/25 ~ 5/26 : 수정  
   >> 5/27 : 보고서 제출 및 발표
5. **앱 기능**
   1. 메인 화면
      1. 코드 분석  
         - 메인 화면은 activity\_main.xml과 MainActivity.java로 구성되어 있다.
         1. activity\_main.xml  
              
            7 줄 – 메인 화면에 위젯들을 중앙에 위치시킨다.  
            8 줄 – 위젯들이 수직으로 배치되도록 ‘orientation = “vertical”’로 설정한다.  
            9 ~ 12 줄 – 학교 이미지를 배치시킨다.  
            13 ~ 18 줄 – 버튼을 2 x 2 형태로 배치시키기 위해 그리드 레이아웃을 사용한다.  
            20 ~ 39 줄 – 공지 확인, 안심콜, 동선 등록, 자가 진단의 이미지 버튼을 생성한다.
         2. MainActivity.java  
            14 줄 – 이미지 버튼 변수 4개를 선언한다.  
            21 ~ 24 줄 – 각 이미지 버튼을 activity\_main.xml에서 찾아 변수에 등록한다.  
            26 ~ 58 줄 – 이미지 버튼이 클릭되었을 때 인텐트를 통해 알맞은 액티비티에 데이터를 넘긴다.
   2. 공지 확인
      1. 기능  
         - 학교 홈페이지에서 코로나와 관련된 게시글을 바로 확인 가능하도록 ‘코로나’를 검색한 페이지의 url 주소를 연결한다.
      2. 순서도  
         
      3. 코드 분석  
         - 첫 번째 액티비티인 ‘공지 확인’ 기능은 메인 화면과 동일하게 activity\_main.xml과 MainActivity.java로 이루어져 있다.
         1. activity\_main.xml  
            - 아래 그림은 activity\_main.xml의 일부이다.  
              
            20 ~ 24 줄 – 화면에 ‘공지 확인’ 버튼을 생성한다.
         2. MainActivity.java  
            - 아래 그림은 MainActivity.java의 일부이다.  
              
            26 ~ 34 줄 – ‘공지 확인’ 버튼이 클릭되었을 때 학교 홈페이지의 기본 주소가 아닌 ‘코로나’를 검색했을 때의 url 주소를 연결한다.
   3. 안심콜
      1. 기능  
         - 교내 안심콜 번호를 앱에 저장하여 버튼 하나로 전화 걸기가 가능하도록 한다.
      2. 순서도  
         
      3. 코드 분석  
         두 번째 액티비티인 안심콜은 ansimcall.xml과 AnsimCall.java로 구성되어 있다.
         1. ansimcall.xml  
              
              
            2 줄 – 전체를 스크롤 뷰로 설정한다.  
            9 줄 – 스크롤 뷰 안의 위젯을 리니어 레이아웃으로 설정한다.  
            12 줄 – 위젯들이 수직으로 배치되도록 orientation = “vertical”로 설정한다.  
            13 ~ 18 줄 – 돌아가기 버튼을 만든다.  
            19 ~ 23 줄 – 돌아가기 버튼 아래의 빈 공간을 만들기 위해 버튼을 하나 만들고 visibility = “invisible”로 설정한다.  
            19 ~ 109 줄 – 건물은 텍스트 뷰로 매장은 버튼으로 등록한다.
         2. AnsimCall.java  
              
              
            12 ~ 13 줄 – 각 매장의 버튼 변수를 선언한다.  
            14 줄 – 돌아가기 버튼 변수를 선언한다.  
            21 줄 – 매장 버튼을 ansimcall.xml에서 찾아 변수에 등록한다.  
            22 ~ 28 줄 – 버튼이 클릭되었을 때 다이얼에 매장의 번호가 입력된다.  
            30 ~ 127 줄 – 21 ~ 28줄의 내용을 반복하되 매장마다 지정되어 있는 번호가 입력되도록 한다.  
            128 줄 – 돌아가기 버튼을 ansimcall.xml에서 찾아 변수에 등록한다.  
            130 ~ 135 – 돌아가기 버튼이 눌렷을 때 이전 화면을 보여주기 위해 현재 화면을 제거하는 finish()함수를 넣는다.
   4. 동선 등록
      1. 기능  
         - 실시간으로 쉽고 빠르게 언제, 어디를 방문했는지 기록이 가능하도록 한다.
      2. 순서도  
         
      3. 코드 분석  
         - 세 번째 액티비티인 ‘동선 등록’은 register\_dialog.xml, third.xml, view\_route.xml, RegisterRoute.java로 구성되어 있다.
         1. third.xml  
              
            2 줄 – 전체를 리니어 레이아웃으로 설정한다.  
            3 줄 – 안의 위젯들이 수직으로 배치되도록 orientation = “vertical”로 설정한다.  
            5 줄 – layout\_weight = “1”로 설정하여 화면이 내용들로 꽉 찰 수 있도록 한다.  
            7 ~ 13 줄 – 이전화면으로 돌아갈 수 있는 돌아가기 버튼을 등록한다. 버튼의 위치를 상단 왼쪽으로 하기 위해서 13줄의 layout\_gravity = “left”로 설정한다.  
            19 ~ 22 줄 – 사용자들이 편하게 이용할 수 있도록 캘린더 뷰를 사용한다.  
            23 ~ 28 줄 – 장소 등록을 편하게 할 수 있도록 가운데에 장소 등록 버튼을 위치시킨다.
         2. view\_route.xml  
              
            2 줄 – 전체를 리니어 레이아웃으로 설정한다.  
            3 줄 – 위젯들이 수직으로 배치되도록 orientation = “vertical”로 설정한다.  
            5 ~ 8 줄 – 에디트 텍스트를 위치시켜서 파일의 내용이 이곳에서 출력 및 수정이 가능하도록 한다.  
            9 ~ 13 줄 – 에디트 텍스트에서 편집한 내용을 다시 파일에 저장하기 위한 수정하기 버튼이다.
         3. register\_dialog.xml  
              
              
            2 줄 – 전체를 리니어 레이아웃으로 설정한다.  
            3 줄 – 위젯들이 수직으로 배치되도록 orientation = “vertical”로 설정한다.  
            5 ~ 8 줄 – 날짜와 입력을 할 수 있는 한 줄을 배치하기 위해 리니어 레이아웃을 중첩시킨다.  
            14 ~ 18 줄 – 사용자에게 날짜, 장소, 시간 입력을 에디트 텍스트로 받는다.
         4. RegisterRoute.java  
              
              
             27 줄 – 동선 등록 버튼, 수정하기 버튼, 돌아가기 버튼이다.  
            28 줄 – 동선을 등록할 수 있는 dialog, 저장된 내용을 출력기 위한 창은 뷰 이다.  
            29 줄 – 달력을 띄우기 위한 뷰이다.  
            30 줄 – 동선 등록창의 장소, 시간, 날짜를 입력하는 에디트 텍스트이다.  
            31 줄 – 등록된 동선이 저장된 파일을 출력하는 에디트 텍스트이다.  
            32 ~ 33 줄 – 각 선택한 년, 월, 일, 시간, 분을 나타내며 자료형은 int이다.  
            34 줄 – 파일을 저장하거나 불러올 때 사용되는 파일 이름을 저장하는 String 타 입 변수이다.  
            39줄 – 화면에 third.xml이 보이도록 한다.  
            40 줄 – 상단바에 ‘동선 등록’ 텍스트가 보이도록 한다.  
            42 ~ 44 줄 – 동선 등록 버튼, 캘린더 뷰, 돌아가기 버튼을 third.xml에서 찾아 각 변수에 등록한다.  
            46 줄 – 날짜를 누르지 않고 등록하기 버튼을 눌렀을 때 날짜를 현재 시간에 맞출 수 있도록 현재 시간을 가져와 mDate 변수에 저장한다.  
            47 ~ 49 줄 – 날짜를 “년도.월.일” 형식으로 하기 위해 SImpleDateFormat을 이용한다. 현재 시간을 형식에 맞추어 파일 이름을 저장하는 fname 변수에 대입한다.  
            51 ~ 54 줄 – 동선을 등록하는 창에 현재 날짜를 맞출 수 있도록 selectYear, selectMonth, selectDay에 getTime을 인덱싱하여 int 형태로 저장한다.  
            57 ~ 62 줄 – 돌아가기 버튼이 눌렷을 때 이전 화면을 보여주기 위해 현재 화면을 제거하는 finish()함수를 넣는다.  
            64 줄 – 캘린더 뷰에 날짜를 클릭하였을 때 나타나는 이벤트 함수를 등록한다.  
            65 줄 – 날짜가 선택될 때 실행되는 함수로 매개변수로는 캘린더 뷰, 선택한 년, 월, 일을 받는다  
            69 ~ 71 줄 – 선택한 날짜로 selectYear, selectMonth, selectDay을 초기화한다.  
            74 줄 – 파일에 저장된 정보와 날짜가 같도록 fname를 재설정한다.  
            76 줄 – 파일 내용을 출력해주는 dialog인 view\_route를 읽어온다.  
            77 ~ 79 줄 - dialog창을 출력한다.  
            80 줄 - routeView에서 에디트 텍스트인 viewRoute를 읽어온다.  
            81 줄 – 수정하기 버튼을 xml에서 가져와 등록한다.  
            83 줄 – readRoute 함수에 매개변수 파일이름을 넘겨주며 호출한다.  
            84 줄 – readRoute 함수 반환 값을 에디트 텍스트를 통해 보여준다.  
            86 줄 – 수정하기 버튼을 눌렀을 때 작동하는 이벤트를 등록한다.  
            90 ~ 96 줄 – 파일이름은 fname이며 기존에 동일한 파일이 있을 경우 지우고 새롭게 저장한다. 에디트 텍스트인 viewRoute에서 텍스트를 가져와 String으로 변환하여 파일을 저장하고 완료후 토스트 메시지로 알려준다.  
            97 줄 – 파일을 여는데 오류가 발생했을 때 90 ~ 96 줄은 수행하지 않는다.  
            102 줄 – routeView dialog 화면을 띄우기 위해서 show() 함수를 호출한다.   
            106 줄 – 등록하기 버튼을 눌렀을 때 동작하는 이벤트를 등록한다.  
            111 ~ 114 줄 – register\_dialog.xml을 dialog창으로 띄우기 위해 뷰를 등록한다.  
            116 ~ 118 줄 – 날짜, 장소, 시간을 입력하기 위한 에디트 텍스트를 등록한다.  
            121 줄 – 선택한 날짜를 자동으로 띄울 수 있도록 edtDay에 현재 선택한 년, 월, 일 을 출력한다.  
            122 줄 – 시간을 입력할 때 edtTime을 클릭하면 동작하는 이벤트를 등록한다.  
            127 ~ 128 줄 – 타임피커 dialog를 화면에 띄울 수 있도록 객체를 생성한다.  
            130 줄 – 시간을 설정하고 타임피커 dialog에서 확인을 누르면 edtTime이 선택한 시간을 출력한다.  
            134 줄 – 타임피커 dialog를 화면에 출력하기 위해 show()함수를 호출한다.  
              
              
              
              
            139 줄 – 동선을 등록하는 창에서 등록 버튼을 누르면 동작할 수 있도록 함수를 사용한다.   
            145 ~ 152 줄 – 입력된 날짜가 fname가 되어 파일을 열어 저장한다. 기존 파일이 있을 경우 해당 파일에 이어서 작성될 수 있도록 한다. 저장하는 내용은 날짜, 시간, 장소 입력정보이다. 파일을 성공적으로 작성 및 저장하면 완료되었다는 토스트 메시지를 출력한다.  
            153 줄 – 파일을 열 때 오류가 발생하면 145 ~ 152 줄은 수행되지 않는다.  
            159 줄 – 취소 버튼을 눌렀을 때 이벤트가 발생하지 않도록 한다.  
            160 줄 – 화면에 registerDialogView가 출력되도록 show(0함수를 호출한다.  
            165 줄 – 리턴 타입은 파일 내용과 매개변수를 String타입으로 받는다.  
            169 ~ 173 줄 – 매개변수로 받은 fname으로 된 파일을 읽어 리턴하는 변수 routeStr에 저장한다. 이 때, 최대로 읽을 수 있는 크기는 500바이트이다.  
            174 줄 – 파일을 읽을 때 오류가 발생하면 169 ~ 173 줄은 수행되지 않는다.  
            177 줄 – routeStr를 반환하며 종료된다.
   5. 자가 진단
      1. 기능  
         - 코로나로 의심되는 증상들을 체크하여 본인의 상태를 파악이 가능하고, 코로나로 의심될 경우 주변 보건소 또는 병원의 위치와 번호를 제공한다.
      2. 순서도
      3. 코드 분석  
         - 네 번째 액티비티인 자가 진단은 radio.xml, bowgunsaw.xml, radioActivity.java, mapActivity.java로 구성되어 있다.
         1. radio.xml  
              
              
              
            2 줄 – 전체를 스크롤 뷰로 설정한다.  
            6 줄 – 스크롤 뷰 안에 위젯을 리니어 레이아웃으로 설정한다.  
            9 줄 – 위젯들이 수직으로 배치되도록 orientation = “vertical”로 설정한다.  
            10 ~ 15 줄 – 돌아가기 버튼을 생성한다.  
            24 ~ 29 줄 – 공통적 증상 텍스트를 생성한다.  
            31 ~ 48 줄 – 공통적인 증상의 체크박스를 생성한다. 각 증상들은 1의 가중치를 가진다.  
            50 ~ 55 줄 – 드문 증상 텍스트를 생성한다.  
            57 ~ 83 줄 – 드문 증상의 체크박스를 생성한다. 각 증상들은 2의 가중치를 가진다.  
            86 ~ 90 줄 – 진단 시작 버튼을 생성한다.  
            92 ~ 111 줄 – 진단 결과를 알려주는 텍스트 뷰와 주건 보건소를 알려주는 버튼을 생성하고 visibility = “invisible”로 설정한다.
         2. bowgunsaw.xml  
              
              
              
              
              
              
            2 줄 – 전체를 스크롤 뷰로 설정한다.  
            8 줄 – 스크롤 뷰 안에 위젯을 리니어 레이아웃으로 설정한다.  
            11 줄 – 위젯들이 수직으로 배치되도록 orientation = “vertical”로 설정한다.  
            19 ~ 24 줄 – 돌아가기 버튼을 생성한다.  
            29 ~ 81 줄 – 하나의 목록은 만드는 리니어 레이아웃이다. 좌측에 이미지 뷰를 배치하고 새로운 리니어 레이아웃을 배치한다.  
            33 줄 – 배경색을 회색으로 설정한다.  
            46 ~ 79 줄 – 리니어 레이아웃에 병원의 이름, 연락처, 위치를 나타낸다.  
            49 줄 - 위젯들이 수직으로 배치되도록 orientation = “vertical”로 설정한다.  
            84 ~ 295 줄 – 29 ~ 81 줄의 내용을 반복하여 다른 병원을 추가한다.
         3. radioactivity.java   
              
            16 ~ 19 줄 – 사용할 버튼, 텍스트 뷰, 체크 박스 변수를 생성한다.  
            21 줄 – 자가 진단 점수를 위해 가중치 값을 integer로 선언한다.  
            28 ~ 39 줄 – 각 버튼, 텍스트 뷰, 체크 박스를 xml에서 찾아 변수에 등록한다.  
            41 줄 – 자가 진단 점수에 사용할 가중치를 0으로 초기화한다.  
              
            44 ~ 72 줄 – 약한 증상인 체크 박스를 선택하면 가중치를 +1 하고, 체크를 풀면 가중치를 -1 하도록 이벤트를 등록한다.  
              
              
            75 ~ 113 줄 – 심한 증상인 체크 박스를 선택하면 가중치를 +2 하고, 체크를 풀면 가중치를 -2 하도록 이벤트를 등록한다.  
              
            119 줄 – 자가 진단 버튼을 눌렀을 때 발생하는 이벤트를 등록한다.  
            122 ~ 127 줄 – 위에서 증상들을 체크했을 때 가중치가 4점 미만이면 위험 점수와 “경과를 더 지켜봅시다.”라는 문장을 텍스트 뷰에 저장하고 텍스트 뷰가 보이도록 setVisibility를 VISIBLE로 설정한다.  
            128 \_ 133 줄 – 가중치가 4점 이상이라면 위험 점수와 “코로나가 의심됩니다. 주변 보건소를 찾아보세요.” 라는 문장을 텍스트 뷰에 저장하고 텍스트 뷰와 주변 보건소 찾기 버튼이 보이도록 setVisibility를 VISIBLE로 설정한다.  
              
            139 ~ 146 줄 – 주변 보건소 찾기 버튼을 누르면 mapActivity에 데이터를 넘긴다.  
            149 ~ 154 줄 – 돌아가기 버튼이 눌렀을 때 이전 화면을 보여주기 위해 현재 화면을 제거하는 finish()함수를 넣는다.
         4. mapActivity.java  
              
            16 줄 – 돌아가기 버튼 변수를 선언한다.  
            19 줄 – 보건소의 전화번호를 띄우기 위해 텍스트 뷰를 선언한다.  
            20 ~ 21 줄 – 병원의 전화번호가 적혀있는 텍스트 뷰들의 아이디를 배열로 저장한다.  
            23 줄 – 병원의 전화번호를 배열로 저장한다.  
            26 줄 – 보건소의 위치를 띄우기 위한 텍스트 뷰를 선언한다.  
            27 ~ 28 줄 – 병원별로 위치보기 텍스트 뷰의 아이디를 배열로 저장한다.  
            30 ~ 35 줄 – 병원의 위치를 네이버 맵 정보에 연결하여 보여주기 위한 url을 배열에 저장한다.  
              
            44 ~ 46 줄 – 전화번호를 telArr 베열에 하나씩 할당한다.  
            48 ~ 60 줄 – 각 병원의 전화번호를 클릭하면 해당 전화번호로 통화하기 위해 53 줄에서 이벤트를 등록한다.  
            63 ~ 66 줄 – 병원의 위치보기 텍스트 뷰를 locateArr 베열에 하나씩 할당한다.  
            69 ~ 81 줄 – 각 병원의 위치보기 텍스트 뷰를 클릭하면 30 ~ 35 줄에서 알맞은 병원의 url주소를 연결해주는 이벤트를 73 줄에 등록한다.  
              
            84 ~ 89 줄 – 자가진단 화면으로 넘어가는 돌아가기 버튼을 만든다.
6. **실행 화면**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 메인 화면 | 공지 확인 버튼 클릭 | 안심콜 버튼 클릭 |
|  |  |  |
| 원하는 매장 클릭 | 동선 등록 버튼 클릭 | 장소 등록 클릭 |
|  |  |  |
| 시간 입력부분 클릭 | 장소를 성공적으로 등록할 때 | 날짜 클릭(파일이 있을 경우) |
|  |  |  |
| 수정하기 버튼 클릭 | 날짜 클릭(파일이 없을 경우) | 자가 진단 버튼 클릭 |
|  |  |  |
| 진단 시작 버튼 클릭(위험 점수가 4점 미만일 때) | 진단 시작 버튼 클릭(위험 점수가 4점 이상일 때) | 주변 보건소 찾기 버튼 클릭 |
|  |  |  |
| 병원의 전화번호 클릭 (시화 병원) | 위치 보기 텍스트 뷰 클릭 (시화 병원) | 보건소 위치찾기 액티비티에서 돌아가기 버튼 클릭 |

1. **개인 별 느낀점**
   1. 김지안  
       주제를 정할 때, 사회적으로도 도움이 되는 앱을 만드는 것이 좋겠다는 의견이 나왔고 그 중 하나가 바로 지금 이 코로나였다. 이 앱의 기능은 실제로 팀원들이 학교에서 코로나 때문에 불편한 점들, 신경 쓰이는 점들을 기능화 한 것이라고 볼 수 있다. 맡은 부분이 동선 등록 부분이었는데, 이것을 리스트 뷰로 구현하여 우리가 일상생활에 쓰이는 앱처럼 좀 더 전문적인 것처럼 보였다면 많은 우리 학생도 사용할 수 있지 않을까 라는 생각이 들었다. 지금 에디트 텍스트로 구현한 장점은 내가 갔던 장소만 저장하는 것이 아닌, 그날 있었던 특별한 사건도 저장할 수 있다는 것이다. 이런 부분에 있어서는 많은 사용자들이 사용을 해보고 어떤 부분이 더 좋은 지 의견을 들어보고 싶다. 그리고 리스트 뷰를 사용하여 똑같은 기능을 구현해보고 싶다는 생각이 들었다.  
       그리고 이번 프로젝트를 진행하면서 지금까지 공부했던 내용들을 실제로 구현함으로써 이론만 공부하는 것보다 훨씬 와 닿았고 실전 경험을 한번 쌓는 것이라고 생각하여 안드로이드에 대한 개발 자신감이 약간은 상승한 것 같다.
   2. 박재현  
       처음 팀프로젝트 주제를 정할 때 ‘내가 잘할 수 있을까, 민폐가 되지는 않을까’라는 생각이 있었는데 걱정했던 것 보다 잘 마무리된 것 같아서 뿌듯하다. 지금까지 경험한 다른 팀 프로젝트와는 다르게 만든 내용을 직접 스마트폰에 다운받아 사용할 수 있다는 것이 신기했다. 다만 아직 경험이 많지 않아서 앱이 완벽하지 못하고 디자인이 너무 기본인 점이 아쉽다. 그래도 능력 좋고 실력 좋은 팀원들과 함께해서 적절히 역할을 분담하여 원활하게 팀 프로젝트를 진행한 것 같다.   
       이번 프로젝트에서 안심콜 번호로 전화를 거는 기능을 맡아서 거의 똑같은 코드만을 만들었지만 안드로이드 스튜디오로 코딩하는 것은 작업이 쉽고 만드는 디자인을 수시로 확인 가능해서 재미있게 진행한 것 같고 좀 더 전문적인 내용을 공부해보고 싶고 더 난이도 높은 앱을 만들어보고 싶다. 여건이 된다면 졸업작품을 앱 형식으로 만들고 싶다.
   3. 최진혁  
       코딩을 잘 못하고 어려워하는 나로서는 팀원들을 상당히 의지할 수 밖에 없었다. 그래서 팀원들에 비해 굉장히 간단한 기능만을 구현하였고, 제안서 작성이나 발표 등 내가 할 수 있는 부분을 좀 더 맡아서 하려 했다. 여러 번의 팀플을 해오면서 항상 열심히 임해야지라는 생각은 갖고 있지만, 코딩에 자신이 없는 나는 매번 코딩이 포함된 팀플을 하는 경우에는 민폐는 되지 말아야지라는 다짐을 하게 되었다. 다행스럽게도 팀원들의 실력이 뛰어났기에 이번 팀플도 무사히 마무리할 수 있던 것 같다.  
       팀플을 마무리 지어 가면서도 여전히 안드로이드 스튜디오는 어려운 과제로 남아있지만, 팀원들이 구현한 액티비티의 알고리즘을 보면서 앱이 진행되는 흐름을 어느 정도 파악할 수는 있었다. 또한 상황에 따라서, 어떠한 위젯을 사용하는 것이 적절한지 판단할 수 있게 되었고, 왜 여러 개의 xml 파일과 java 파일을 두는 것이 효율적이고 필요한 것인지 이해할 수 있었던 것 같다.
   4. 한창현  
       이번 Corona\_KPU 앱을 만들 때 내가 맡은 부분은 자가진단과 시흥의 의료시설 정보를 보여주는 액티비티였다. 액티비티를 구현하는 과정에서 조금 아쉬웠던 점은 실시간으로 현재 위치 정보를 받아서 그 정보를 기반으로 한 주변 의료시설을 지도에 보여주는 기능을 넣고 싶었는데, 내 실력이 부족해서 구현해내지 못한 것이 아쉽다. 만약 다음에 기회가 된다면 다시 도전해 보고싶다.  
       여태가지 모바일 프로그래밍을 배우며 얻은 지식을 활용하여 하나의 프로젝트를 진행하는 것이 배운 점을 복기하는 데에도 도움이 됐고, 여러모로 보람찼다.
2. **참고 자료**- 안드로이드 프로그래밍(6판), 우재남/박길식, 한빛미디어, 2021, p.104,226~228, 255, 324~329, 396, 414  
   - SPRINGNOTE, “[android] Intent로 Web URL 연결”, <https://argument.tistory.com/63>, 2021. 05. 13