

객체지향프로그래밍II 결과 보고서

제목: 부동산 매매 정보제공 시스템

팀장 : 장윤재

조원 : 김주의, 김준근, 김상환

목 차

1. 프로젝트 개요	3
1.1. 문제 기술	3
1.2. 비전	3
1.3. 기능목록	3
2. 시스템	4
2.1. 소프트웨어 구조	4
2.1.1. 패키지 설계 구조	4
2.1.2. 클래스 설명	4
2.2. 시스템 인터페이스	8
2.3. 사용자 인터페이스	8
2.4. 통신 인터페이스	9
3. 특징 구현	10
3.1. 부동산 정보 검색	10
3.1.1. 분석	10
3.1.2. 설계	10
3.1.3. 구현과 테스트	11
3.2. 엑셀 파일 파싱	19
3.2.1. 분석	19
3.2.2. 설계	19
3.2.3. 구현과 테스트	20
3.3. 로그인 기능	23
3.3.1. 분석	23
3.3.2. 설계	23
3.3.3. 구현과 테스트	25
3.4. 아이디/비밀번호 찾기 기능	29
3.4.1. 분석	29
3.4.2. 설계	29
3.4.3. 구현과 테스트	30
4. 프로젝트 평가	38
4.1. 프로젝트 완성도	38
4.2. 일정 계획 평가	38
4.3. 역할 수행 평가	40
4.4. 설계 구성요소	40
4.5. 현실적 제한조건	41
5. 소감	43

1. 프로젝트 개요

1.1. 문제 기술

집을 구하기 위해 여러 부동산에 들러 매물로 나와 있는 집을 발로 뛰어다니며 직접 보러 다니며 내가 원하는 부동산정보를 얻으려고 하면 시간과 힘이 많이 든다. 헛걸음하여 시간을 낭비할 수 있는 문제점도 있다. 집을 구하는 사람들에게 직접 찾으러 다녀야 하는 불편함을 줄여주고, 쉽고 간단하고 명확하게 부동산 매물에 관한 정보를 제공하려 한다.

1.2. 비전

요즘 세대에는 무언가를 얻으려고 할 때 발로 뛰어다니며 직접 다니는 것보다 집에서 휴대폰이나 PC를 사용하여 재화나 정보를 편하게 얻는 것을 선호한다. 그러므로 우리는 집을 구하는 사람들에게 직접 다녀야 하는 불편함을 줄여주고 간단하고 명확하게 부동산 매물에 관한 정보를 제공하려 한다. 그것이 우리가 개발한 부동산 매물 정보제공 시스템이다.

부동산 매물 정보제공 시스템은 집을 구할 때 특유의 막막하고, 어렵고, 몸이 힘들고, 무엇을 해야 할지 모를 때 이 프로그램을 통해 부동산 매물에 관한 기본적인 정보(매물 이름, 위치, 가격 등)를 제공하고, 그 정보들을 쉽게 비교할 수 있도록 이해 당사자들의 요구사항을 반영하여, 사용자들에게 편리함을 제공하도록 했다.

사용자에는 학생, 부동산 업자, 일반 가정이 있고 그 개개인의 ID를 가지고 로그인하기 때문에 자신의 정보를 제외한 다른 사람의 정보는 볼 수 없다. 또한, 별도의 보안 장치를 통해 다른 사람이 자신의 ID로 로그인하는 경우를 막았다.

현재는 매물 이름, 위치, 가격 등의 정보를 주고 비교할 수 있는 기능을 제공하며 자신이 원하는 옵션(아파트, 연립다세대, 단독다가구)을 선택하여 필터링하여 검색 기능을 제공한다.

1.3. 기능목록

연번	우선순위	기능
F-01	1	공공 API를 이용하여 매물 정보(이름, 가격, 위치, 설명 등) 제공
F-02	2	원하는 매물 위치 검색
F-03	3	사용자 로그인/회원가입
F-04	4	아이디/비밀번호 찾기

2. 시스템

2.1. 소프트웨어 구조

2.1.1. 패키지 설계 구조

패키지	클래스
client	Client TcpClientTest TcpEchoClient
login	AptPrice AptResultFrame FindIDPasswordFrame FirstFrame LoginFrame OftResultFrame RegisterFrame RhtRegisterFrame ShtRegisterFrame
parsing	CategoryParsing ExcelPs ParsingApt ParsingBase ParsingOft ParsingRht ParsingSht TestDrive
server	Server TcpEchoServer TcpServerTest

2.1.2. 클래스 설명

연번	클래스 이름	책임
		public method 및 생성자
1	Client	사용자가 프로그램을 사용할 수 있도록 서버와 연결하는 메소드
		▪ public void startClient()
		사용자가 프로그램을 종료하면 서버와 연결을 끊는 메소드
		▪ public void stopClient()
		클라이언트에서 서버로 보낸 값의 결과를 클라이언트로 받아 사용자에게 결과를 보여주는 메소드

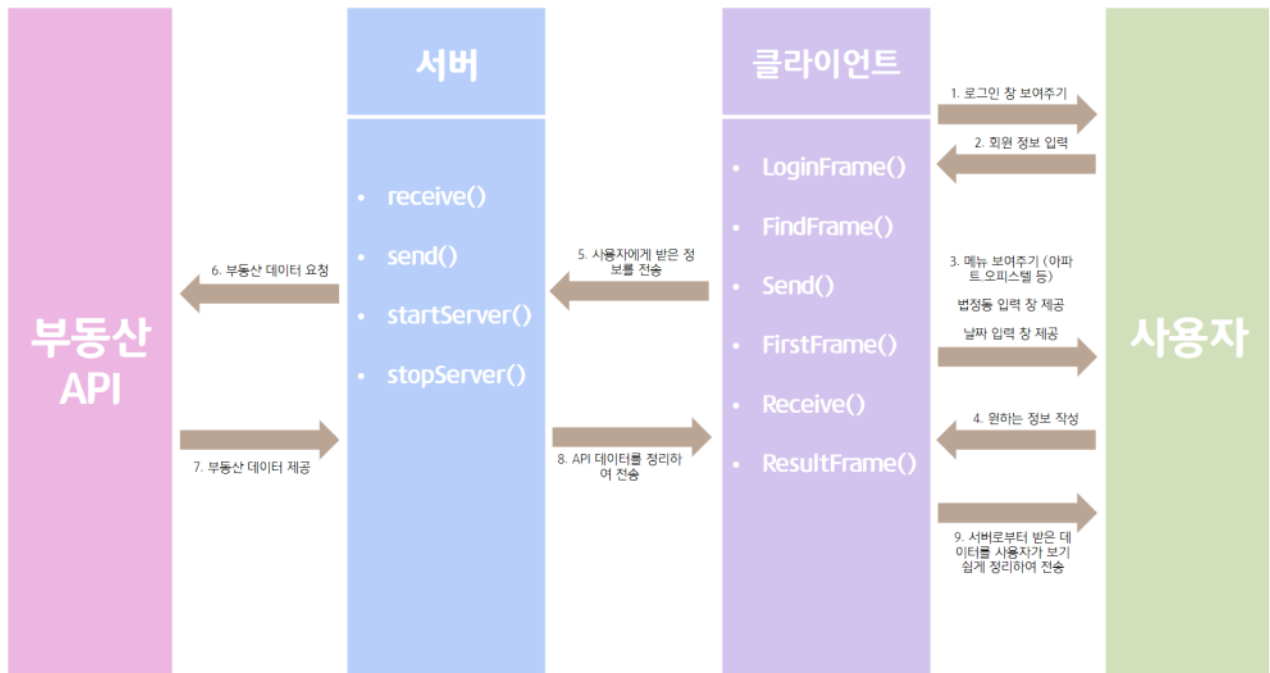
		<ul style="list-style-type: none"> ▪ public void receive() <p>사용자가 입력한 값(아이디, 비밀번호, 이메일 등)을 클라이언트가 서버에 전달하는 메소드</p> <ul style="list-style-type: none"> ▪ public void send(String data)
2	TCPClientTest	<p>Client 클래스를 테스트하는 메소드</p> <ul style="list-style-type: none"> ▪ public static void main(String[] args)
3	TcpEchoClient	<p>소켓을 생성하여 서버로 word 전송하고 받아보는 메소드</p> <ul style="list-style-type: none"> ▪ public void go()
4	AptPrice	<p>매물 위치 검색화면 GUI 중 하나로 '시도를 선택하세요!'와 필터가 뜨는 창을 구현한 생성자</p> <ul style="list-style-type: none"> ▪ public AptPrice(Client client) <p>AptPrice(Client client) 안의 메소드로 시/도를 선택하는 창을 닫는 메소드</p> <ul style="list-style-type: none"> ▪ public windowClosing(WindowEvent e)
5	AftResultFrame	<p>아파트 매물 정보 결과를 화면에 띄우는 생성자</p> <ul style="list-style-type: none"> ▪ public OftResultFrame(String allReuslt)
6	FindIDPasswordFrame	<p>아이디/비밀번호 찾기 GUI로 아이디/비밀번호 찾기에 필요한 것들이 뜨는 창을 구현한 생성자</p> <ul style="list-style-type: none"> ▪ public FindIDPasswordFrame(Client client) <p>FindIDPasswordFrame(Client client) 안의 메소드로 아이디/패스워드 찾기 창을 닫는 메소드</p> <ul style="list-style-type: none"> ▪ public windowClosing(WindowEvent e) <p>FindIDPasswordFrame창 실행하는 메소드</p> <ul style="list-style-type: none"> ▪ public void run() <p>아이디/비밀번호 찾기 창을 배치하는 메소드</p> <ul style="list-style-type: none"> ▪ public void disposefindIDPasswordFrame() <p>'응답' 메시지가 뜨는 메소드</p> <ul style="list-style-type: none"> ▪ public void showMessage()
7	FirstFrame	<p>매물 위치 검색화면 GUI 중 하나로 '방 찾기' 버튼이 뜨는 창을 구현한 생성자</p> <ul style="list-style-type: none"> ▪ public FirstFrame(Client client) <p>FirstFrame(Client client) 안의 메소드로 방 찾기 창을 닫는 메소드</p> <ul style="list-style-type: none"> ▪ public windowClosing(WindowEvent e)
8	LoginFrame	<p>로그인 GUI로 아이디, 비밀번호 적는 칸, 회원가입, 아이디/비밀번호 찾기 등이 뜨는 창을 구현한 생성자</p> <ul style="list-style-type: none"> ▪ public LoginFrame(Client client)

		안에 들어가는 요소들의 위치를 배열 해주고 폰트, 폰트 사이즈 등 세부적인 것들을 설정해주는 메소드
		▪ public void runLoginFrame()
		LoginFrame(Client client) 안의 메소드로 로그인 창을 닫는 메소드
		▪ public windowClosing(WindowEvent e)
		로그인에 성공했을 때 '회원님 환영합니다~ 로그인 성공' 메시지를 뜨게 하는 메소드
		▪ public void disposeLoginFrame()
		로그인에 실패했을 때 '로그인에 실패하였습니다' 메시지를 뜨게 하는 메소드
		▪ public void loginFailed()
		아이디/비밀번호 찾기 창을 배치하는 메소드
		▪ public void findIDframedispose()
		아이디/비밀번호 찾기를 했을 때 아이디, 비밀번호를 출력해주는 메소드
		▪ public void findIDframeShowMessage(String message)
9	OftResultFrame	오피스텔 매물 정보 결과를 화면에 띄우는 생성자
		▪ public OftResultFrame(String allReuslt)
10	RegisterFrame	회원가입 GUI로 아이디와 비밀번호를 등록하는 생성자
		▪ public RegisterFrame(Client client)
		RegisterFrame(Client client) 안의 메소드로 회원가입 창을 닫는 메소드
		▪ public windowClosing(WindowEvent e)
		RegisterFrame 창 실행하는 메소드
		▪ public void run()
11	RhtResultFrame	연립다세대 매물 정보 결과를 화면에 띄우는 생성자
		▪ public RftResultFrame(String allReuslt)
12	ShtResultFrame	단독다가구 매물 정보 결과를 화면에 띄우는 생성자
		▪ public SftResultFrame(String allReuslt)
13	CategoryParsing	Api에서 받아온 지역 코드와 요청 월, 항목별로 변수에 담아주는 생성자
		▪ public CategoryParsing(int reignCode, int dealMonth, String cateGory)
14	ExcelPs	지역 코드가 포함된 Excel 파일을 파싱 하는 메인 메소드
		▪ public static void main(String[] args)
15	ParsingApt	아파트 매매정보를 위하여 지역 코드, 요청 일을 저장하는 생성자

		<ul style="list-style-type: none"> ▪ public ParsingApt(String reignCode, String dealMonth)
		항목들을 NodeList로 저장하는 메소드
16	ParsingBase	<ul style="list-style-type: none"> ▪ public void contents(NodeList nodeList)
		URL을 만드는 메소드
		<ul style="list-style-type: none"> ▪ public void setUrl()
		URL의 정보를 전부 받아서 공백을 없애고 결과를 저장하는 메소드
		<ul style="list-style-type: none"> ▪ public void setReadResult()
		URL을 만드는 함수를 호출하고 URL의 정보를 전부 받아서 공백을 없애고 결과를 저장하여 예외처리한 부분을 실행시키는 메소드
		<ul style="list-style-type: none"> ▪ public void run()
		result(결과)를 리턴 하는 메소드
17	ParsingOft	<ul style="list-style-type: none"> ▪ public String getResult()
		오피스텔 매매정보를 위하여 지역 코드, 요청 일을 저장하는 생성자
		<ul style="list-style-type: none"> ▪ public ParsingOft(String reignCode, String dealMonth)
		NodeList를 받아와서 오피스텔 정보를 파싱 할 때 사용하는 메소드
18	ParsingRht	<ul style="list-style-type: none"> ▪ public void contents(NodeList nodeList)
		연립다세대 매매정보를 위하여 지역 코드, 요청 일을 저장하는 생성자
		<ul style="list-style-type: none"> ▪ public ParsingRht(String reignCode, String dealMonth)
		NodeList를 받아와서 연립다세대 정보를 파싱 할 때 사용하는 메소드
19	ParsingSht	<ul style="list-style-type: none"> ▪ public void contents(NodeList nodeList)
		단독주택 매매정보를 위하여 지역 코드, 요청 일을 저장하는 생성자
		<ul style="list-style-type: none"> ▪ public ParsingSht(String reignCode, String dealMonth)
		NodeList를 받아와서 단독주택 정보를 파싱 할 때 사용하는 메소드
20	Server	<ul style="list-style-type: none"> ▪ public void contents(NodeList nodeList)
		서버를 시작하는 메소드
		<ul style="list-style-type: none"> ▪ public void startServer()
		서버를 종료하는 메소드
21	TcpEchoServer	<ul style="list-style-type: none"> ▪ public void stopServer()
		클라이언트가 보내온 단어를 수신하고 다시 보내주는 메소드

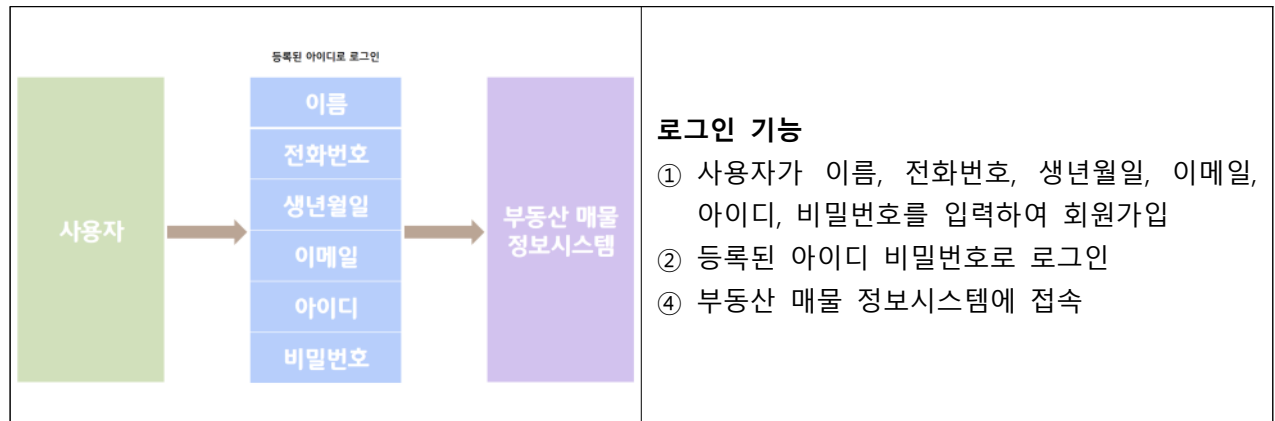
		<ul style="list-style-type: none"> ▪ public void go()
22	TcpServerTest	수신한 입력 스트림을 그대로 출력 스트림을 통하여 보내주는 메소드 <ul style="list-style-type: none"> ▪ public static void main(String[] args)

2.2. 시스템 인터페이스

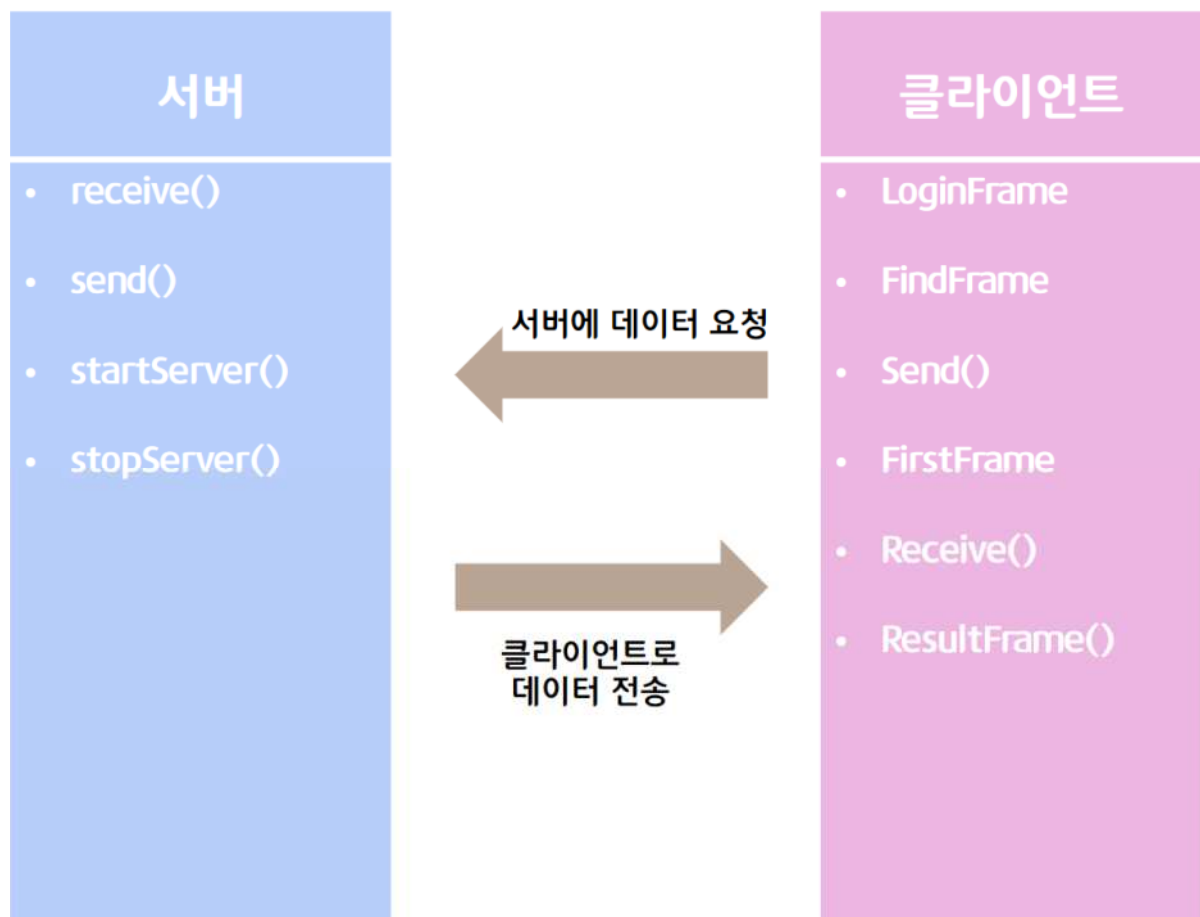


2.3. 사용자 인터페이스

화면 흐름	설명
<p>The flowchart shows the process for the 'Real Estate Search Function'. It starts with the 사용자 (User) selecting 지역 (Area), 등록 날짜 (Registration Date), and 건물 종류 (Building Type). These selections are then processed by the 부동산 매물 정보시스템 (Real Estate Listing Information System) to provide results.</p>	매물 검색 기능 ① 사용자가 지역, 등록날짜, 건물 종류를 선택하여 입력 ② 입력받은 데이터를 파싱 하고 파싱 된 데이터를 한 번 더 엑셀 파싱 하여 서버에 저장 ③ 부동산 매물 정보시스템이 그에 맞는 정보를 출력해 결과화면을 보여줌



2.4. 통신 인터페이스



3. 특징 구현

3.1. 부동산 정보 검색

✓ 기술

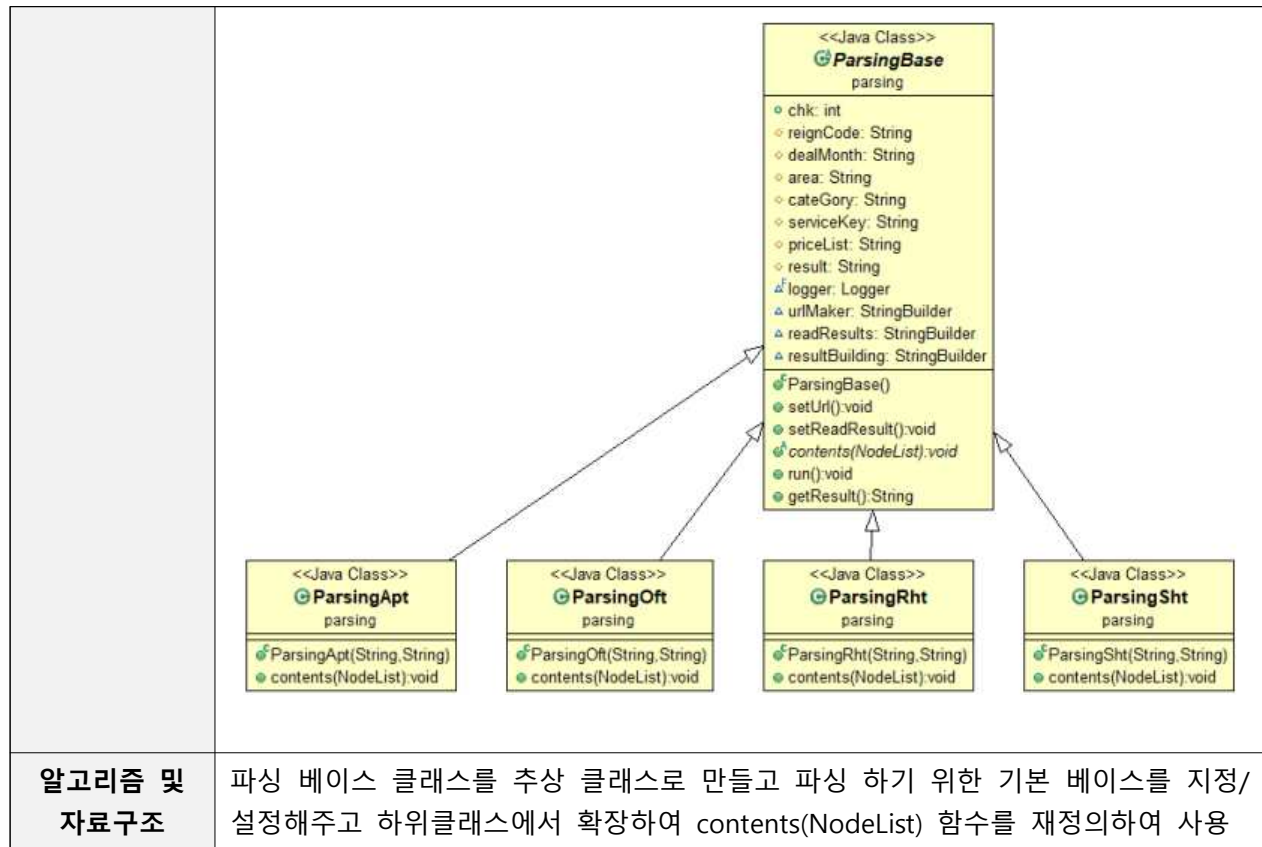
전국에 있는 아파트, 단독다가구, 연립다세대, 오피스텔에 대한 매매정보(가격, 건축년도, 평수, 지역구, 주택 유형)등을 검색을 할 수 있게 하여 정보를 제공해주는 기능

3.1.1. 분석

		내용
요구 사항		공공 API를 파싱을 통해 아파트, 단독다가구, 연립다세대, 오피스텔 부동산정보를 요구함
비기능 요구 사항	성능 효율성	방 찾기 버튼을 클릭했을 때 10초 안에 정보를 받아와야 함
	공간 효율성	검색 기능을 수행할 때 메모리를 최대한 적게 차지해야 함
	사용성	사용자 인터페이스는 사용자가 회원가입하고 로그인하여 자신이 원하는 동 이름을 검색하고 그 동의 매물 정보를 제공한다. 도중에 오류가 발생한다면 이를 로거로 콘솔 창에 표시해줌
	신뢰성	특정한 기능을 실행할 때 실패할 가능성이 5%보다 낮아야 함
특징 구현 방안		각 파싱 클래스를 분할 하고, GUI 창에서 주택 유형을 선택해 원하는 지역, 동을 사용자 입력으로 검색하면 GUI 창으로 부동산 정보가 나오도록 구현함

3.1.2. 설계

		내용
특징 설계		각 파싱 클래스를 분할 하여 각자 다른 공공 API를 파싱 하도록 하며, ParsingBase로부터 기본 구조를 받아오며, 아파트, 단독다가구, 연립다세대, 오피스텔 부동산정보를 받아서 서버로 보내 클라이언트와 연결 하여 부동산정보를 보여주도록 설계
클래스 다이어그램		<pre> classDiagram class Client { <<Java Class>> +socketChannel: SocketChannel +Client(SocketChannel) +receive():void +send(String):void } class Server { <<Java Class>> +executorService: ExecutorService +serverSocketChannel: ServerSocketChannel +databaseScanner: Scanner +databaseWriter: FileWriter +databasePath: String +database: File +Server() +startServer():void +stopServer():void } Client --> "0..*" Server : -connections </pre>



3.1.3. 구현과 테스트

파싱 한 클래스의 기본적인 구조는 ParsingBase 클래스를 추상 클래스로 만들어 하위클래스에서 추상 메소드를 재정의하여 소스코드의 경제성을 높였다. 공공 API에서 받은 정보를 nodeList에 저장하고 resultBuilding에서 append() 함수를 사용하여 파싱 하고 콘솔 창에서 결과 확인 후 GUI 창에 파싱 한 결과를 제공하도록 구현하였다.

파싱 베이스 (추상 클래스)	
클래스	public abstract class ParsingBase
변수 및 생성자	<pre> public int chk = 0; protected String reignCode; //지역 코드 protected String dealMonth; //계약 월 protected String area; //법정동 protected String cateGory = ""; //종류 //서비스 키 protected String serviceKey = "qraPJG00hKQsPxMhFB1dnPyXND1kHr51C1WU2HY0R1Ww5CcLG4YBPq%2BeBcYUmcYEJr%2BvXZM3LKFDZmRQ73u3A%3D%3D"; //URL을 잘라서 붙일 때 필요 protected String priceList = "http://openapi.molit.go.kr:8081/OpenAPI_ToolInstallPackage/service/rest/RTMSOBJSvc/getRTMSDataSvc"; protected String result = ""; //결과를 저장할 변수 final Logger logger = Logger.getLogger(ParsingBase.class.getName()); //logger를 사용하기 위해서 선언 //URL 만들어주는 객체 StringBuilder urlMaker = new StringBuilder(""); //""에 스트링형 객체 넣어줘야 함 StringBuilder readResults = new StringBuilder(""); //""결과를 읽는 스트링 빌더 StringBuilder resultBuilding = new StringBuilder("\n"); //결과를 저장하는 스트링 빌더 StringBuilder lastBuilding = new StringBuilder("\n"); </pre>

URL을 만드는 코드

```
public void setUrl() {
    urlMaker.append(priceList); //시작 URL
    urlMaker.append(cateGory); //아파트, 단독/다가구, 연립다세대의 종류를 선택
    urlMaker.append("?LAWD_CD=");
    urlMaker.append(reignCode); //int형으로 선언해준 지역 코드를 string형으로 형 변환
    urlMaker.append("&DEAL_YMD=");
    urlMaker.append(dealMonth);
    urlMaker.append("&ServiceKey=");
    urlMaker.append(serviceKey);
    System.out.println(urlMaker);
}
```

파싱을 하기 위해서는 URL 주소가 필요. 주소를 만들기 위해서는 서비스키가 필요. 그것을 사람 손으로 직접 다 만들 수 없기에 setUrl 메소드를 통해 URL을 만들어 줌.

URL의 정보를 전부 받아 공백을 없애고 저장하는 코드

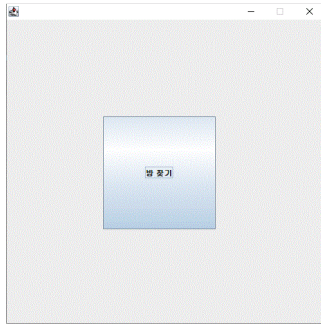
```
public void setReadResult() {
    try {
        URL url = new URL(urlMaker.toString());
        HttpURLConnection urlconnection = (HttpURLConnection) url.openConnection();
        BufferedReader br = new BufferedReader(new InputStreamReader(urlconnection.getInputStream(), "UTF-8"));
        //기존 string 배열로 반복문을 사용하여 받지 않고 stringBuilder 객체를 사용하여 바로 추가
        readResults.append(br.readLine());
    } catch (MalformedURLException e) { //URL이 잘못되었을 경우 예외처리
        logger.warning("MalformedURLException");
    } catch (IOException e) { //어떤 오류가 발생했을 경우 예외처리
        // TODO Auto-generated catch block
        logger.warning("오류 발생");
    }
    //System.out.println(readResults);
}
}
```

URL을 읽고 만든 URL을 String형으로 변환. 그리고 한 줄로 전부 받아 기존 String 배열로 반복문을 사용하여 받지 않고 StringBulider 객체를 사용하여 바로 추가

아파트 매매 거래 정보 분류(파싱 베이스를 상속받는 클래스) (오피스텔, 연립다세대, 단독다가구 정보를 분류하는 클래스도 아래와 비슷한 원리로 동작)	
클래스	<code>public class ParsingApt extends ParsingBase</code>
변수 및 생성자	<pre> public ParsingApt(String reignCode, String dealMonth, String area) { cateGory = "AptTrade"; //유형 : 아파트 매매정보 this.reignCode = reignCode; //reignCode와 dealMonth를 int에서 String으로 변경 this.dealMonth = dealMonth; //등록년월 this.area = area; //지역 run(); } </pre>

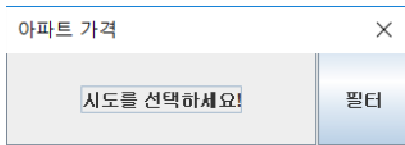
파싱 한 아파트 매물 정보를 분류하는 코드	
	<pre> public void contents(NodeList nodeList) { for (int i = 0; i < nodeList.getLength(); i++) { NodeList child = nodeList.item(i).getChildNodes(); resultBuilding.setLength(0); //StringBuilder 를 초기화 for (int j = 0; j < child.getLength(); j++) { //items 으로 둘러싸인 부분을 반복 Node node = child.item(j); if (node.getNodeName().contains("거래금액")) { // item 으로 둘러싸인 부분을 반복 // contains 와 equal 사용 고려 String temp = node.getTextContent(); temp = temp.trim(); //문자열의 앞 뒤 공백 제거 함수 resultBuilding.append("거래금액 : "); resultBuilding.append(temp); resultBuilding.append("만원"); resultBuilding.append("\n");// 줄바꿈 } else if (node.getNodeName().contains("건축년도")) { //item 으로 둘러싸인 부분을 반복 //contains 와 equal 사용 고려 String temp = node.getTextContent(); temp = temp.trim(); //문자열의 앞 뒤 공백 제거 함수 resultBuilding.append("건축년도 : "); resultBuilding.append(temp); resultBuilding.append("년"); resultBuilding.append("\n");// 줄바꿈 } else if (node.getNodeName().contains("법정동")) { //item 으로 둘러싸인 부분을 반복 String temp = node.getTextContent(); temp = temp.trim(); //문자열의 앞 뒤 공백 제거 함수 if(temp.equals(area)) { chk=1; } resultBuilding.append("지역구 : "); resultBuilding.append(temp); resultBuilding.append("\n");// 줄바꿈 } else if (node.getNodeName().contains("아파트")) { //item 으로 둘러싸인 부분을 반복 . . . } } } } </pre>

방 찾기 GUI



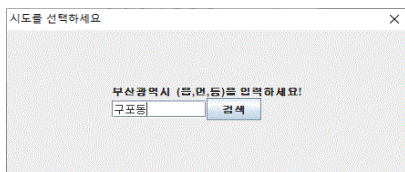
```
public class FirstFrame {
    private JFrame mainFrame;
    private JButton searchButton;
    public FirstFrame(Client client){
        mainFrame = new JFrame(""); //메인프레임
        mainFrame.setLayout(null); //메인프레임 레이아웃설정
        searchButton = new JButton("방 찾기"); //발찾기 버튼
        mainFrame.setSize(new Dimension(500,500)); //메인프레임 크기설정
        searchButton.addActionListener(e -> { //버튼 이벤트 처리기
            AptPrice aptPrice = new AptPrice(client); //아파트 매매정보를 가져옴
        });
        mainFrame.add(searchButton); //메인프레임에 검색버튼 추가
        searchButton.setBounds(150,150,175,175);
        mainFrame.setResizable(false);
        mainFrame.setVisible(true);
        mainFrame.addWindowListener(new WindowAdapter() { // 창 오른쪽 상단 X버튼 눌렀을 경우 처리
            public void windowClosing(WindowEvent e) {
                System.exit(0); //종료
            }
        });
    }
}
```

'시도를 선택하세요' GUI



```
public sidoDongSearch() {
    check = false;
    sidoFrame = new JFrame();
    sidoDlg = new JDialog(sidoFrame, "시도를 선택하세요", Dialog.ModalityType.APPLICATION_MODAL);
    sidoDlg.setLayout(new GridBagLayout());
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 0;
    gridBagConstraints.gridwidth = 4;
    gridBagConstraints.anchor = GridBagConstraints.WEST;
    JLabel text = new JLabel("시도를 선택하세요!");
    sidoPanel = new JPanel();
    sidoPanel.setLayout(new GridBagLayout());
    sidoPanel.setBorder(BorderFactory.createTitledBorder("시/도"));
    sidoDlg.add(text, gridBagConstraints);
    gridBagConstraints.gridwidth = 1;
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 0;
}
```

'- (읍, 면, 동)을 입력하세요!' GUI



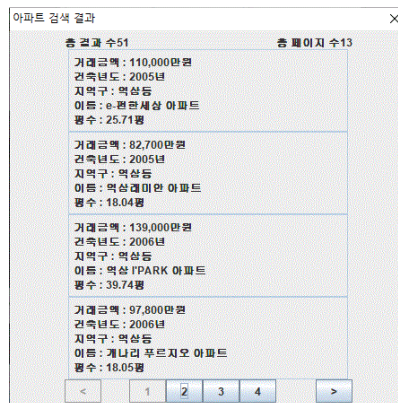
```
for (int i = 0; i < sidoList.length; i++) {
    if (i % 3 == 0) {
        gridBagConstraints.gridx = 0;
        gridBagConstraints.gridy++;
    }
    JButton sido = new JButton(sidoList[i]);
    int finalI = i;
    sido.addActionListener(e -> {
        curSelectedItem = sidoList[finalI];
        sidoName = curSelectedItem;
        text.setText(curSelectedItem + " (읍,면,동)을 입력하세요!");
        sidoPanel.removeAll();
        sidoDlg.remove(sidoPanel);
        sidoDlg.getContentPane().revalidate();
        sidoDlg.getContentPane().repaint();
        JTextField dongInput = new JTextField();
        dongInput.setColumns(10);
        GridBagConstraints gridBagConstraints = new GridBagConstraints();
        gridBagConstraints.gridx = 0;
        gridBagConstraints.gridy = 1;
        sidoDlg.getContentPane().add(dongInput, gridBagConstraints);
        JButton search = new JButton();
        search.setText("검색");
        gridBagConstraints.gridx = 1;
        sidoDlg.getContentPane().add(search, gridBagConstraints);
    });
}
```


필터 클릭 시 '기능 선택' GUI



```
private SelectPage() {
    selectFrame = new JFrame("");
    selectDlg = new JDialog(selectFrame, "기능을 선택하세요", Dialog.ModalityType.APPLICATION_MODAL);
    selectDlg.getContentPane().setLayout(new GridBagLayout());
    gridBagConstraints = new GridBagConstraints();
    parsingAptButton = new JButton("아파트");
    parsingAptButton.setPreferredSize(new Dimension(100, 100));
    parsingOfftButton = new JButton("오피스텔");
    parsingOfftButton.setPreferredSize(new Dimension(100, 100));
    parsingRhtButton = new JButton("연립 다세대");
    parsingRhtButton.setPreferredSize(new Dimension(100, 100));
    parsingShtButton = new JButton("단독/다가구");
    parsingShtButton.setPreferredSize(new Dimension(100, 100));
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 0;
    selectDlg.getContentPane().add(parsingAptButton, gridBagConstraints);
    gridBagConstraints.gridx = 1;
    selectDlg.getContentPane().add(parsingShtButton, gridBagConstraints);
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 1;
    selectDlg.getContentPane().add(parsingRhtButton, gridBagConstraints);
    gridBagConstraints.gridx = 1;
    selectDlg.getContentPane().add(parsingOfftButton, gridBagConstraints);
}
```

검색 결과 GUI (코드는 4가지 비슷하여 아파트 검색결과만)



검색결과

```
public AptResultFrame(String allResult) {
    aptframe = new JFrame();
    aptdlg = new JDialog(aptframe, "아파트 검색 결과", Dialog.ModalityType.APPLICATION_MODAL);
    aptdlg.getContentPane().setLayout(new GridBagLayout());
    gridBagConstraints = new GridBagConstraints();

    curPage = 1;
    String[] temp = allResult.split("\\n+");
    resultList = new StringBuilder[temp.length / 7];
    totalCount = resultList.length;
    if (resultList.length % 4 == 0) {
        totalPage = resultList.length / 4;
    } else {
        totalPage = resultList.length / 4 + 1;
    }
    for (int i = 1; i < resultList.length; i++) {
        resultList[i - 1] = new StringBuilder("<html>");
        for (int j = 0; j < 7; j++) {
            resultList[i - 1].append(temp[i * 7 + j]);
            resultList[i - 1].append("<br />");
        }
        resultList[i - 1].append("</html>");
    }
    prevButton = new JButton("<");
    nextButton = new JButton(">");
    prevButton.addActionListener(e -> {
        newPage(curPage - 1);
    });
    nextButton.addActionListener(e -> {
        newPage(curPage + 1);
    });
    newPage(1);
    aptdlg.setSize(450, 450);
    aptdlg.setResizable(false);
    aptdlg.setVisible(true);
}
```

페이지

```
private void newPage(int pageNum) {
    curPage = pageNum;
    aptdlg.getContentPane().removeAll();

    JLabel totalCounting = new JLabel("총 결과 수" + String.valueOf(totalCount));
    JLabel totalPaging = new JLabel("총 페이지 수" + String.valueOf(totalPage));
    gridBagConstraints.anchor = GridBagConstraints.CENTER;
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 0;
    aptdlg.getContentPane().add(totalCounting, gridBagConstraints);
    gridBagConstraints.gridx = 5;
    aptdlg.getContentPane().add(totalPaging, gridBagConstraints);
    JButton[] pages = new JButton[0];
    int counter = (pageNum - 1) / 4;
    if (curPage == 1)
        prevButton.setEnabled(false);
    else
        prevButton.setEnabled(true);
    if (curPage == totalPage)
        nextButton.setEnabled(false);
    else
        nextButton.setEnabled(true);

    if ((totalPage - 1) / 4 > (pageNum - 1) / 4) {
        pages = new JButton[4];
        for (int i = 0; i < pages.length; i++) {
            pages[i] = new JButton(String.valueOf(1 + i + counter * 4));
            if (curPage == 1 + i + counter * 4)
                pages[i].setEnabled(false);
            else
                pages[i].setEnabled(true);
            int finalI = i;
            pages[i].addActionListener(e -> newPage(1 + finalI + counter * 4));
        }
    } else if ((totalPage - 1) / 4 == (pageNum - 1) / 4) {
        pages = new JButton[totalPage % 4];
        for (int i = 0; i < pages.length; i++) {
            pages[i] = new JButton(String.valueOf(1 + i + counter * 4));
            if (curPage == 1 + i + counter * 4)
                pages[i].setEnabled(false);
            else
                pages[i].setEnabled(true);
            int finalI = i;
            pages[i].addActionListener(e -> newPage(1 + finalI + counter * 4));
        }
    }
    switch (pages.length) {
        case 0:
            //결과가 전혀 없을 경우 처리하고 싶으면 여기를 건드리면 됨
            break;
        case 1:
            gridBagConstraints.gridy = 9;
            gridBagConstraints.gridx = 1;
            aptdlg.getContentPane().add(pages[0], gridBagConstraints);
            break;
        case 2:
            gridBagConstraints.gridy = 9;
            gridBagConstraints.gridx = 1;
            aptdlg.getContentPane().add(pages[0], gridBagConstraints);
            gridBagConstraints.gridx = 2;
            aptdlg.getContentPane().add(pages[1], gridBagConstraints);
            break;
        case 3:
            gridBagConstraints.gridy = 9;
            gridBagConstraints.gridx = 1;
            aptdlg.getContentPane().add(pages[0], gridBagConstraints);
            gridBagConstraints.gridx = 2;
            aptdlg.getContentPane().add(pages[1], gridBagConstraints);
            gridBagConstraints.gridx = 3;
            aptdlg.getContentPane().add(pages[2], gridBagConstraints);
            break;
        case 4:
            gridBagConstraints.gridy = 9;
            gridBagConstraints.gridx = 1;
            aptdlg.getContentPane().add(pages[0], gridBagConstraints);
            gridBagConstraints.gridx = 2;
            aptdlg.getContentPane().add(pages[1], gridBagConstraints);
            gridBagConstraints.gridx = 3;
            aptdlg.getContentPane().add(pages[2], gridBagConstraints);
            gridBagConstraints.gridx = 4;
            aptdlg.getContentPane().add(pages[3], gridBagConstraints);
            break;
        default:
            //심각한 버그가 발생해서 페이지가 -이거나 50이상일 경우
            break;
    }
}
```



```

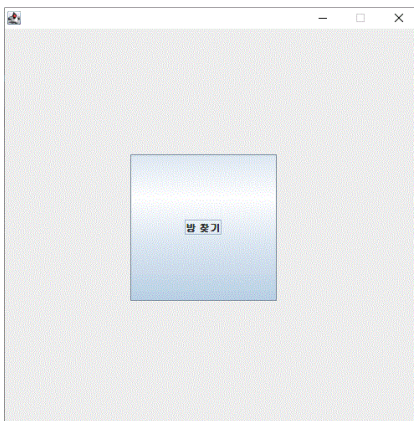
gridBagConstraints.gridx = 0;
gridBagConstraints.gridwidth = 7;
gridBagConstraints.gridy = 1;
JLabel[] result = new JLabel[4];
JPanel[] groupBox = new JPanel[4];
for (int i = 0; i < result.length; i++) {
    if (totalCount <= i + (curPage - 1) * 4) {
        break;
    }
    if (resultList[i + (curPage - 1) * 4] != null) {
        groupBox[i] = new JPanel();
        groupBox[i].setBorder(BorderFactory.createTitledBorder(""));
        result[i] = new JLabel(resultList[i + (curPage - 1) * 4].toString());
        result[i].setPreferredSize(new Dimension(300, 80));
        groupBox[i].add(result[i]);
    }
}
aptdlg.getContentPane().add(groupBox[0], gridBagConstraints);
gridBagConstraints.gridy = 2;
if (result[1] != null)
    aptdlg.getContentPane().add(groupBox[1], gridBagConstraints);

gridBagConstraints.gridy = 3;
if (result[2] != null)
    aptdlg.getContentPane().add(groupBox[2], gridBagConstraints);
gridBagConstraints.gridy = 4;
if (result[3] != null)
    aptdlg.getContentPane().add(groupBox[3], gridBagConstraints);
gridBagConstraints.gridwidth = 1;
gridBagConstraints.gridy = 9;
gridBagConstraints.gridx = 0;
gridBagConstraints.anchor = GridBagConstraints.WEST;
aptdlg.getContentPane().add(prevButton, gridBagConstraints);
gridBagConstraints.gridx = 5;
gridBagConstraints.anchor = GridBagConstraints.EAST;
aptdlg.getContentPane().add(nextButton, gridBagConstraints);
aptdlg.getContentPane().revalidate();
aptdlg.getContentPane().repaint();
}

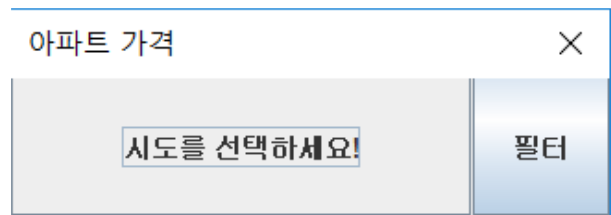
```

테스트 결과

① 초기실행 창



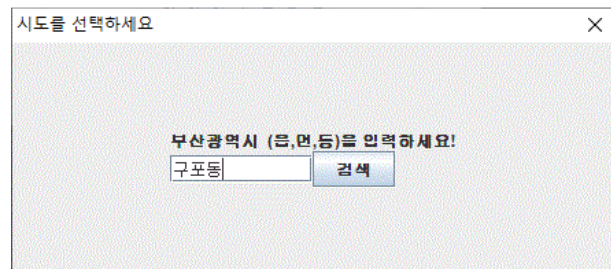
② '방 찾기' 선택 시 실행 창



③ '시도를 선택하세요!' 선택 시 실행 창



④ 지역(시/도) 선택 시 실행 창



⑤ 지역(읍, 면, 동) 검색 시 실행 창

아파트 가격

부산광역시 구포동

필터

11월 2015

일	월	화	수	목	금	토
45	1	2	3	4	5	6
46	8	9	10	11	12	13
47	15	16	17	18	19	20
48	22	23	24	25	26	27
49	29	30				

⑥ 필터 선택 시 실행 창

기능을 선택하세요

아파트	단독/다가...
연립 다세...	오피스텔

⑦ 최종 결과

아파트 검색 결과

총 검색 수 93

총 페이지 수 24

거래금액 : 12,700만원
건축년도 : 1992년
지역구 : 구포동
이름 : 백성그린힐드 아파트
층수 : 23.3층
거래금액 : 21,500만원
건축년도 : 1994년
지역구 : 구포동
이름 : 구포현대 아파트
층수 : 25.69층
거래금액 : 20,750만원
건축년도 : 1999년
지역구 : 구포동
이름 : 삼성그린코아 아파트
층수 : 18.68층
거래금액 : 25,850만원
건축년도 : 1997년
지역구 : 구포동
이름 : 한진재팬 아파트
층수 : 25.64층

< 1 2 3 4 >

3.2. 엑셀 파일 파싱

✓ 기술

공공 API 파싱을 하기 위해서 지역 코드가 필요한데 그 지역 코드가 전국기준 38000개인데 그것을 하나하나 써서 사용할 수 없다고 판단하여 지역 코드가 들어있는 엑셀 파일을 파싱 하여 지역별 코드 정보를 받아오기 위한 기능

3.2.1. 분석

		내용
요구 사항		공공 API를 파싱을 위해 지역 코드가 필요한데 그 지역 코드의 수가 너무 많아 엑셀 파일에서 읽어와야 함
비기능 요구 사항	성능 효율성	엑셀 파싱을 했을 때 10초 안에 코드를 받아와야 함
	기능 효율성	검색 기능을 수행할 때 메모리를 최대한 적게 차지해야 함
	사용성	엑셀 파일을 읽어와 콘솔 창에 띄어줌
	신뢰성	엑셀 파싱을 실행할 때 실패할 가능성이 5%보다 낮아야 함
특징 구현 방안		엑셀 파일을 읽어오기 위해 파일 직렬화나 입출력을 사용하여 구현하려 했으나 POI 라이브러리가 엑셀 파일을 다루는 데 좋다 하여 POI 라이브러리 사용

3.2.2. 설계

		내용
특징 설계		전국의 코드, 시/도명, 구, 동, 년/월 코드가 작성된 엑셀 파일 중 법정동 코드와 동 코드를 나누고 잘 분류해서 저장. 파일 내부에 빈 셀들이 존재하여 그 셀들 때문에 오류가 나서 빈 셀을 제거하여 결괏값이 엑셀 파일의 값과 비교 후 콘솔 창에 출력해가면서 설계함
클래스 다이어그램		<pre> classDiagram class Client { <<Java Class>> +SocketChannel socketChannel +Client(SocketChannel) +receive() void +send(String) void } class Server { <<Java Class>> +ExecutorService executorService +ServerSocketChannel serverSocketChannel +Scanner databaseScanner +FileWriter databaseWriter +String databasePath +File database +Server() +startServer() void +stopServer() void } Client --> "0..*" Server : -connections </pre>

	<div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 5px; background-color: #ffffcc;"> <p align="center"><<Java Class>></p> <p align="center">CategoryParsing</p> <p align="center">parsing</p> <hr/> <p> ■ reignCode: int ■ dealMonth: int ■ cateGory: String ▲F logger: Logger ▲ DataList: ArrayList<StringBuilder> </p> <hr/> <p>●C CategoryParsing(int,int,String)</p> </div> <div style="border: 1px solid black; padding: 5px; background-color: #ffffcc;"> <p align="center"><<Java Class>></p> <p align="center">ExcelPs</p> <p align="center">parsing</p> <hr/> <p> ●C ExcelPs() ●S main(String[]):void </p> </div> </div>
사용 라이브러리	<p><POI 4.0></p> <p>Grid 형태의 데이터를 핸들링하는 화면에는 보통 '엑셀 다운로드', '엑셀 업로드' 버튼이 존재한다. 이를 구현하기 위해서 Apache에서 제공하는 POI 라이브러리를 이용함. Java로 작성된 Client 어플리케이션의 경우 엑셀 업로드는 Client 어플리케이션이 실행되는 로컬에서 파일에 접근하여 내용을 read 하여 서버로 전송할 데이터로 변경하면 되기 때문에 엑셀 파일에 접근하고 내용을 읽기 위해 사용함.</p>

3.2.3. 구현과 테스트

전국의 코드, 시/도명, 구, 동, 년/월 코드가 작성된 엑셀 파일을 문자, 숫자, 빈 공간 등을 잘 분류해서 저장하여 받아오는 API에 지역 코드를 넘겨주도록 구현

엑셀 파일 파싱	
클래스	<code>public class ExclePs</code>
변수 및 생성자	<pre> public String areaCodes = ""; public long areaCode; public String areaName= ""; public String sidoName= ""; XSSFRow row; XSSFCell cell; XSSFCell sido; </pre>

지역 코드가 포함된 엑셀 파일을 파싱 하는 코드													
엑셀 파일							콘솔 창						
18919	4831033024	경상남도	거제시	남부면	다대리	19950101	4831033024	경상남도	거제시	남부면	다대리	19950101	[null]
18920	4831033025	경상남도	거제시	남부면	갈곶리	19950101	4831033025	경상남도	거제시	남부면	갈곶리	19950101	[null]
18921	4831034000	경상남도	거제시	거제면		19950101	4831034000	경상남도	거제시	거제면	[null]	19950101	[null]
18922	4831034021	경상남도	거제시	거제면	법동리	19950101	4831034021	경상남도	거제시	거제면	법동리	19950101	[null]
18923	4831034023	경상남도	거제시	거제면	내간리	19950101	4831034023	경상남도	거제시	거제면	내간리	19950101	[null]
18924	4831034024	경상남도	거제시	거제면	외간리	19950101	4831034024	경상남도	거제시	거제면	외간리	19950101	[null]
18925	4831034025	경상남도	거제시	거제면	옥산리	19950101	4831034025	경상남도	거제시	거제면	옥산리	19950101	[null]
18926	4831034026	경상남도	거제시	거제면	서정리	19950101	4831034026	경상남도	거제시	거제면	서정리	19950101	[null]
18927	4831034027	경상남도	거제시	거제면	서상리	19950101	4831034027	경상남도	거제시	거제면	서상리	19950101	[null]
18928	4831034028	경상남도	거제시	거제면	남동리	19950101	4831034028	경상남도	거제시	거제면	남동리	19950101	[null]


```

public ExcelPs(String areaName, String sidoName) {
    this.areaName = areaName; //지역이름
    this.sidoName = sidoName; //시도이름
    System.out.println(sidoName);
    System.out.println(areaName);
    try {
        //엑셀파일의 경로
        FileInputStream inputStream = new FileInputStream("C:\\Users\\dbswo\\OneDrive\\바탕 화면\\code.xlsx");
        XSSFWorkbook workbook = new XSSFWorkbook(inputStream);
        // sheet 수 취득
        int sheetCn = workbook.getNumberOfSheets();
        System.out.println("sheet 수 : " + sheetCn);
        for (int cn = 0; cn < sheetCn; cn++) {
            System.out.println("취득하는 sheet 이름 : " + workbook.getSheetName(cn));
            System.out.println(workbook.getSheetName(cn) + " sheet 데이터 취득 시작");
            // 0번째 sheet 정보 취득
            XSSFSheet sheet = workbook.getSheetAt(cn);
            // 취득된 sheet에서 rows 수 취득
            int rows = sheet.getPhysicalNumberOfRows();
            System.out.println(workbook.getSheetName(cn) + " sheet의 row 수 : " + rows);
            // 취득된 row에서 취득대상 cell 수 취득
            int cells = sheet.getRow(cn).getPhysicalNumberOfCells(); //
            System.out.println(workbook.getSheetName(cn) + " sheet의 row명 취득대상 cell 수 : " + cells);
            for (int r = 0; r < rows; r++) {
                row = sheet.getRow(r); // row 가져오기
                if (row != null) {
                    for (int c = 0; c < cells; c++) {
                        cell = row.getCell(c);
                        sido = row.getCell(1);
                        if (cell != null) {
                            String value = null;
                            switch (cell.getCellType()) { //셀의 타입이
                                case FORMULA: //수식일 경우
                                    value = cell.getCellFormula();
                                    break;
                                case NUMERIC: //숫자일 경우
                                    value = "" + String.valueOf(new Double(cell.getNumericCellValue()).longValue());
                                    if (c == 0)
                                        areaCodes = value;
                                    break;
                                case STRING: //문자일 경우
                                    value = "" + cell.getStringCellValue();
                                    if (sido.getStringCellValue().contains(sidoName)) { // 시 도
                                        if (c == 3 && value.contains(areaName)) { //법정동
                                            areaCode = Long.parseLong(areaCodes)/100000; // 지역코드
                                        }
                                    }
                                    break;
                                case BLANK: //빈공간일 경우
                                    value = "[null 아닌 공백]";
                                    break;
                                case ERROR:
                                    value = "" + cell.getErrorCellValue();
                                    break;
                                default:
                                    break;
                            }
                            // System.out.print(value + "\t");
                        } else {
                            // System.out.print("[null]\t");
                        }
                    } // for(c)
                } // System.out.print("\n");
            } // for(r)
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

테스트 결과

4831033024	경상남도	거제시	남부면	다대리	19950101	[null]
4831033025	경상남도	거제시	남부면	갈곶리	19950101	[null]
4831034000	경상남도	거제시	거제면	[null]	19950101	[null]
4831034021	경상남도	거제시	거제면	법동리	19950101	[null]
4831034023	경상남도	거제시	거제면	내간리	19950101	[null]
4831034024	경상남도	거제시	거제면	외간리	19950101	[null]
4831034025	경상남도	거제시	거제면	옥산리	19950101	[null]
4831034026	경상남도	거제시	거제면	서정리	19950101	[null]
4831034027	경상남도	거제시	거제면	서상리	19950101	[null]
4831034028	경상남도	거제시	거제면	남동리	19950101	[null]
4831034029	경상남도	거제시	거제면	통상리	19950101	[null]
4831034030	경상남도	거제시	거제면	오수리	19950101	[null]
4831034031	경상남도	거제시	거제면	명진리	19950101	[null]
4831034032	경상남도	거제시	거제면	소량리	19950101	[null]
4831035000	경상남도	거제시	둔덕면	[null]	19950101	[null]
4831035021	경상남도	거제시	둔덕면	상둔리	19950101	[null]
4831035022	경상남도	거제시	둔덕면	시목리	19950101	[null]
4831035023	경상남도	거제시	둔덕면	거림리	19950101	[null]
4831035024	경상남도	거제시	둔덕면	산방리	19950101	[null]
4831035025	경상남도	거제시	둔덕면	방하리	19950101	[null]
4831035026	경상남도	거제시	둔덕면	하둔리	19950101	[null]
4831035027	경상남도	거제시	둔덕면	어구리	19950101	[null]
4831035028	경상남도	거제시	둔덕면	습역리	19950101	[null]
4831035029	경상남도	거제시	둔덕면	학산리	19950101	[null]
4831036000	경상남도	거제시	사등면	[null]	19950101	[null]
4831036021	경상남도	거제시	사등면	사곡리	19950101	[null]
4831036022	경상남도	거제시	사등면	사등리	19950101	[null]
4831036023	경상남도	거제시	사등면	성포리	19950101	[null]
4831036024	경상남도	거제시	사등면	지석리	19950101	[null]
4831036025	경상남도	거제시	사등면	청곡리	19950101	[null]
4831036026	경상남도	거제시	사등면	오량리	19950101	[null]
4831036027	경상남도	거제시	사등면	덕호리	19950101	[null]
4831036028	경상남도	거제시	사등면	창호리	19950101	[null]
4831037000	경상남도	거제시	연초면	[null]	19950101	[null]
4831037021	경상남도	거제시	연초면	한내리	19950101	[null]
4831037022	경상남도	거제시	연초면	오비리	19950101	[null]
4831037023	경상남도	거제시	연초면	연사리	19950101	[null]
4831037024	경상남도	거제시	연초면	죽토리	19950101	[null]
4831037025	경상남도	거제시	연초면	다공리	19950101	[null]

3.3. 로그인

✓ 기술

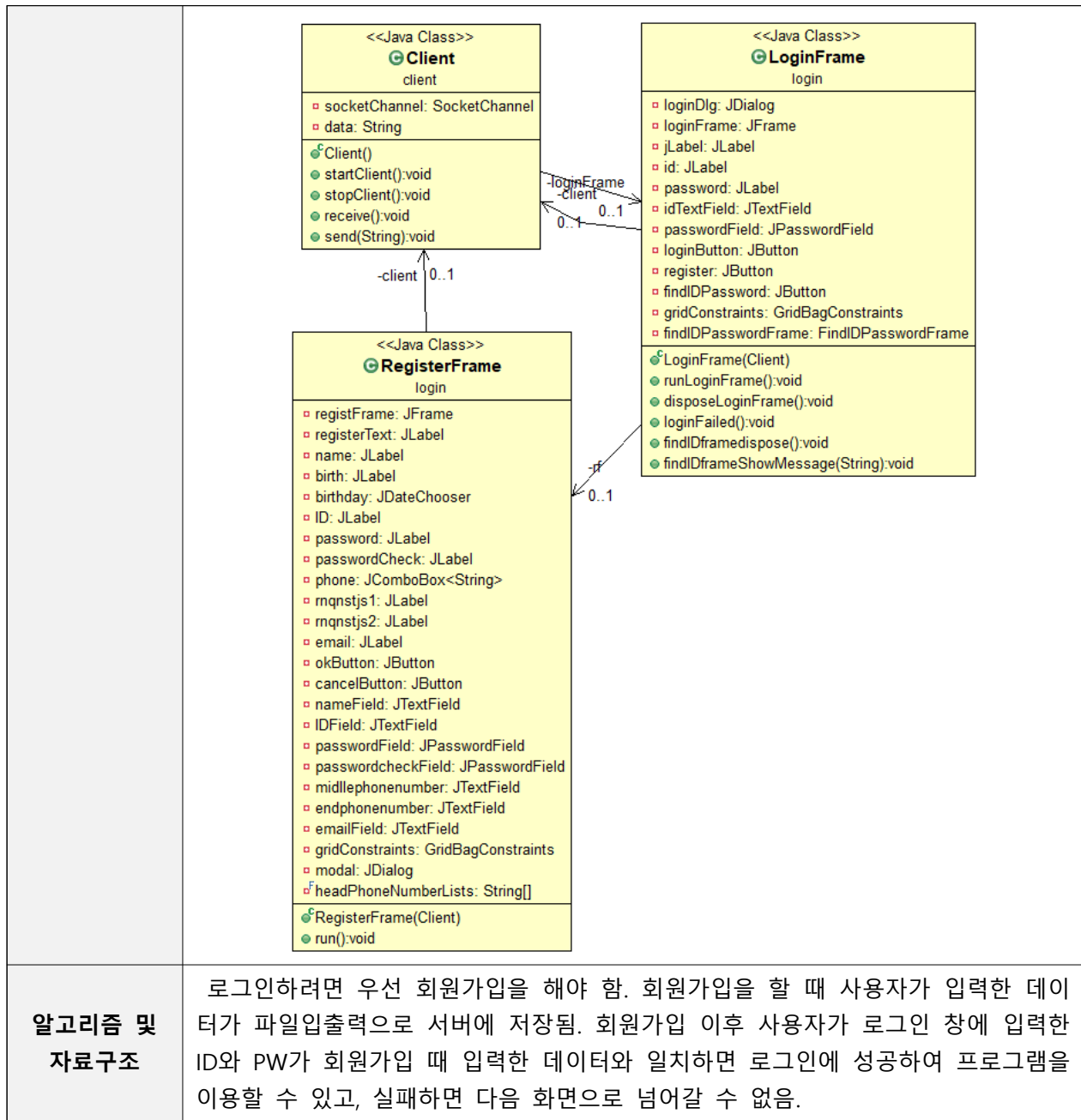
사용자가 부동산 매물 정보제공 시스템을 사용할 수 있도록 사용자의 정보를 받아 회원가입 후 프로그램을 사용할 수 있도록 한다. 부적절하거나 이상한 경로로 접근하는 사용자의 사용을 제한하기 위한 기능이다.

3.3.1. 분석

		내용
요구 사항		아이디와 비밀번호를 만들기 위해 개인의 정보를 필요하고 파일입출력이 요구됨
비기능 요구 사항	성능 효율성	로그인하였을 경우 5초 안에 로그인되어 기능을 수행할 수 있도록 해야 함
	공간 효율성	로그인에 관련된 정보를 저장하는 파일이 너무 크면 안 됨
	사용성	프로그램사용을 허가해줌
	신뢰성	로그인을 실행할 때 실패할 가능성이 5%보다 낮아야 함
특징 구현 방안		텍스트 파일입출력을 통해 작성한 양식을 텍스트 파일에 저장하여 서버로 보내고 그 사용자가 로그인할 때 작성한 데이터와 비교하여 로그인을 허가해주는 방안

3.3.2. 설계

		내용
특징 설계		프로그램을 사용하려면 회원가입을 하게 한다. 회원가입은 이름, 생년월일, 아이디, 비밀번호, 비밀번호 확인 양식에 맞게 쓰면 파일입출력으로 서버에 저장함
클래스 다이어그램		<pre> classDiagram class Client { <<Java Class>> socketChannel: SocketChannel Client(SocketChannel) receive():void send(String):void } class Server { <<Java Class>> executorService: ExecutorService serverSocketChannel: ServerSocketChannel databaseScanner: Scanner databaseWriter: FileWriter databasePath: String database: File Server() startServer():void stopServer():void } Client --> "0..*" Server : -connections </pre>



3.3.3. 구현과 테스트

클래스	<code>public class LoginFrame</code>
변수 및 생성자	<pre> private JDialog loginDlg; private JFrame loginFrame; //화면을 만든다 private JLabel jLabel; //로그인 텍스트 private JLabel id; //ID 텍스트 private JLabel password; //PW 텍스트 private JTextField idTextField; //id 입력 칸 private JPasswordField passwordField; //password 입력 칸 private JButton loginButton; //로그인버튼 private JButton register; //회원가입 버튼 private JButton findIDPassword; //ID/PW찾기 버튼 private GridBagConstraints gridConstraints; //grid 배치 private Client client; private RegisterFrame rf; private FindIDPasswordFrame findIDPasswordFrame; </pre>

로그인 창 X 버튼을 눌렀을 때 이벤트 처리기

```

loginFrame.addWindowListener(new WindowAdapter() {
    @Override
    public void windowClosing(WindowEvent e) {
        System.exit(0); // 창 오른쪽 상단 X버튼 눌렀을 경우 처리
    }
});

```

창 오른쪽 상단 X 버튼 눌렀을 경우 종료

'아이디/비밀번호 찾기'를 눌렀을 때 이벤트 처리기

```

findIDPassword.addActionListener(e -> {
    findIDPasswordFrame = new FindIDPasswordFrame(client);
    findIDPasswordFrame.run();
});

```

아이디/비밀번호 찾기를 눌렀을 때 아이디 패스워드 프레임을 만들. 클라이언트는 이 프레임이 서버에 데이터를 보내기 위해 생성자에 넣어줌. 아이디 패스워드 프레임을 실행

'회원가입'을 눌렀을 때 이벤트 처리기

```

register.addActionListener(e -> {
    rf = new RegisterFrame(client);
    rf.run();
});

```

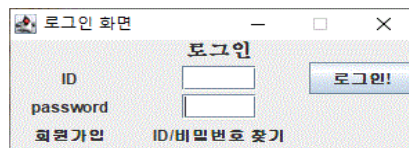
회원가입을 눌렀을 때 회원가입 프레임을 만들. 클라이언트는 이 프레임이 서버에 회원가입 정보를 보내기 위해 생성자에 넣어줌. 회원가입 프레임을 실행

'로그인' 버튼 눌렀을 때 이벤트 처리기

```
loginButton.addActionListener(e -> {
    String ID = idTextField.getText();
    //getPassword()메소드는 char[] 즉 캐릭터 배열 형태
    String password = String.valueOf(passwordField.getPassword());
    StringBuilder stringBuilder = new StringBuilder("로그인");
    stringBuilder.append("\n");
    stringBuilder.append(ID).append("\n");
    stringBuilder.append(password);
    client.send(stringBuilder.toString());
    //보내는 형식 - 로그인\n아이디\n비밀번호
    //반환할 때 하기 위해서 String으로 바꿈
    //ID와 PW에 idTextField, passwordField에 있는 문자열을 받아서 넣음
});
```

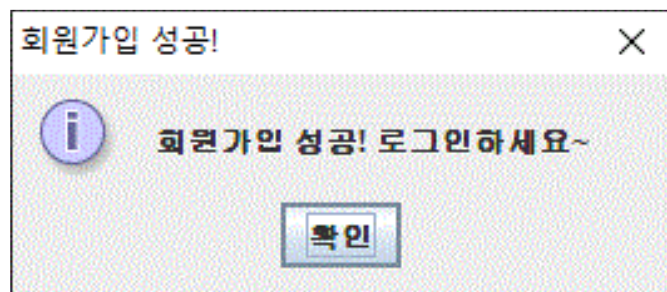
ID 창에 입력된 텍스트를 읽어서 ID에 저장. PW 창에 입력된 비밀번호를 PW에 저장. ID와 PW에 idTextField, passwordField에 있는 문자열을 받아서 넣음.

로그인 창 GUI



```
public LoginFrame(Client client) { //초기화
    this.client = client;
    loginFrame = new JFrame("로그인 화면");
    loginDlg = new JDialog(loginFrame, "로그인 화면", Dialog.ModalityType.APPLICATION_MODAL);
    JLabel = new JLabel();
    id = new JLabel();
    password = new JLabel();
    idTextField = new JTextField();
    passwordField = new JPasswordField();
    loginButton = new JButton();
    register = new JButton();
    findIDPassword = new JButton();
    gridConstraints = new GridBagConstraints();
```

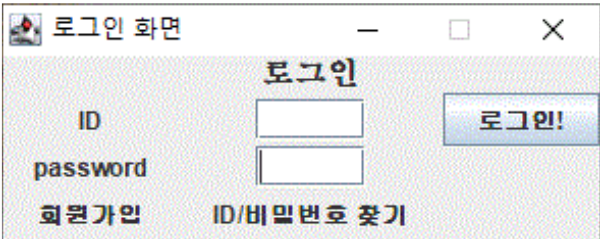

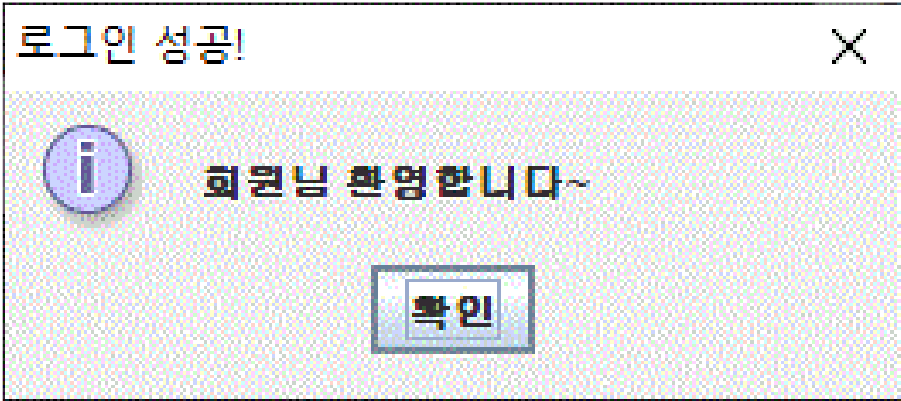
회원가입 GUI

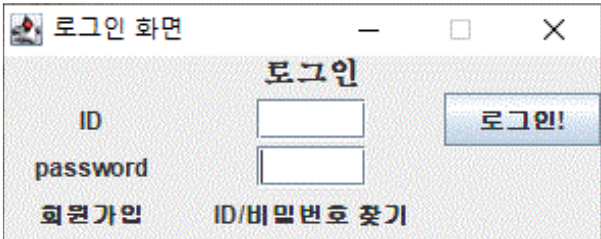
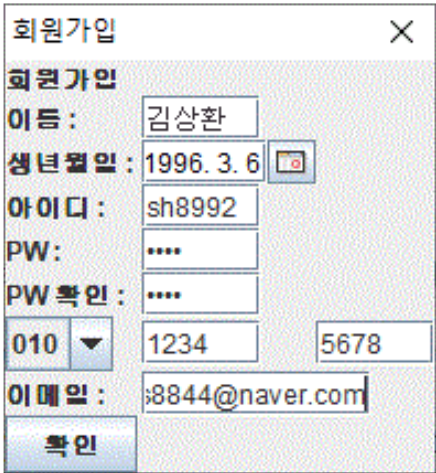
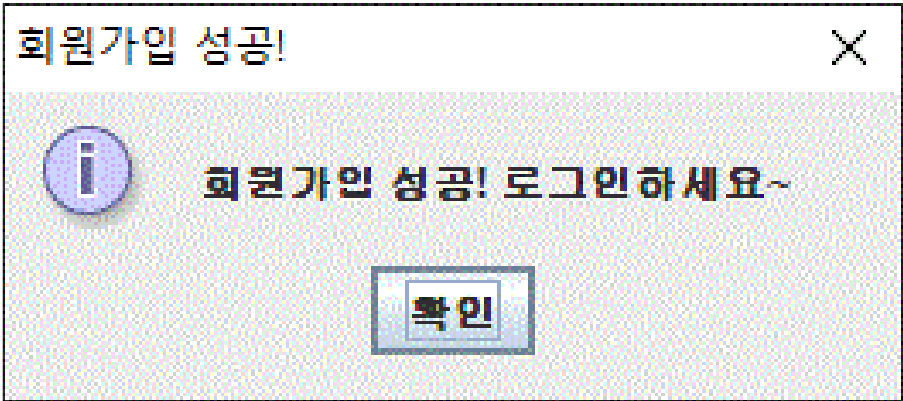


```

okButton.addActionListener(e -> //OK 버튼을 눌렀을 때 이벤트 처리기
    if (!middlephonenumber.getText().matches("[0-9]{4}$")) {
        JOptionPane.showMessageDialog(null, "폰번호 중간번호 오류 다시 입력하세요", "오류", JOptionPane.INFORMATION_MESSAGE);
    }
    else if (!endphonenumber.getText().matches("[0-9]{4}$")) {
        JOptionPane.showMessageDialog(null, "폰번호 끝번호 오류 다시 입력하세요", "오류", JOptionPane.INFORMATION_MESSAGE);
    }
    else if (!emailField.getText().matches("[a-z0-9-]+(.[a-z0-9-]+)*@[?:\\w+\\.]+\\w+$")) {
        JOptionPane.showMessageDialog(null, "이메일 형식 오류 다시 입력하세요", "오류", JOptionPane.INFORMATION_MESSAGE);
    }
    else if (String.valueOf(passwordField.getPassword()).equals(String.valueOf(passwordcheckField.getPassword()))) {
        //확인 버튼을 눌렀을 때 처리할 구문
        StringBuilder stringBuilder = new StringBuilder("회원가입");
        stringBuilder.append("\n");
        stringBuilder.append("이름 : ").append(nameField.getText()).append("\n");
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy년MM월dd일");
        String dateString = simpleDateFormat.format(birthday.getDate());
        stringBuilder.append("생년월일 : ").append(dateString).append("\n");
        stringBuilder.append("id : ").append(IDField.getText()).append("\n");
        stringBuilder.append("pw : ").append(String.valueOf(passwordField.getPassword())).append("\n");
        stringBuilder.append("휴대폰 번호").append(phone.getSelectedIndex());
        stringBuilder.append("-").append(middlephonenumber.getText());
        stringBuilder.append("-").append(endphonenumber.getText()).append("\n");
        stringBuilder.append("email : ").append(emailField.getText()).append("\n");
        client.send(stringBuilder.toString());
        JOptionPane.showMessageDialog(null, "회원가입 성공! 로그인하세요~", "회원가입 성공!", JOptionPane.INFORMATION_MESSAGE);
        registFrame.dispose();
        registFrame.setVisible(false);
    }
    else {
        JOptionPane.showMessageDialog(null, "비밀번호를 다시 확인하세요", "비밀번호 오류", JOptionPane.INFORMATION_MESSAGE);
    }
});

```

로그인 테스트 결과	
① 초기 실행 창	② 아이디, 비밀번호 입력
 <p>로그인 화면</p> <p>로그인</p> <p>ID <input type="text"/></p> <p>password <input type="password"/></p> <p>로그인!</p> <p>회원가입 ID/비밀번호 찾기</p>	 <p>로그인 화면</p> <p>로그인</p> <p>ID <input type="text" value="sh8992"/></p> <p>password <input type="password" value="...."/></p> <p>로그인!</p> <p>회원가입 ID/비밀번호 찾기</p>
③ 최종 결과 창	
 <p>로그인 성공!</p> <p>회원님 환영합니다~</p> <p>확인</p>	

회원가입 테스트 결과	
① 초기 실행 창	② 회원가입 클릭 시 실행 창(입력)
 <p>The initial login screen is titled '로그인 화면' (Login Screen). It features a '로그인' (Login) button and a '회원가입' (Sign Up) link. Below the login button, there is a label 'ID/비밀번호 찾기' (Find ID/Password).</p>	 <p>The sign-up form is titled '회원가입' (Sign Up). It contains the following fields: <ul style="list-style-type: none"> 이름 (Name): 김상환 생년월일 (Date of Birth): 1996. 3. 6 아이디 (ID): sh8992 PW: PW 확인 (Confirm Password): 010 (Phone Area Code): 010 1234 (Phone Number): 1234 5678 (Phone Number): 5678 이메일 (Email): j8844@naver.com There is a '확인' (Confirm) button at the bottom. </p>
③ 최종 결과 창	
 <p>The final result window is titled '회원가입 성공!' (Sign Up Success!). It contains an information icon and the message '회원가입 성공! 로그인하세요~' (Sign Up Success! Please log in~). There is a '확인' (Confirm) button at the bottom.</p>	

3.4. 아이디/비밀번호 찾기

✓ 기술

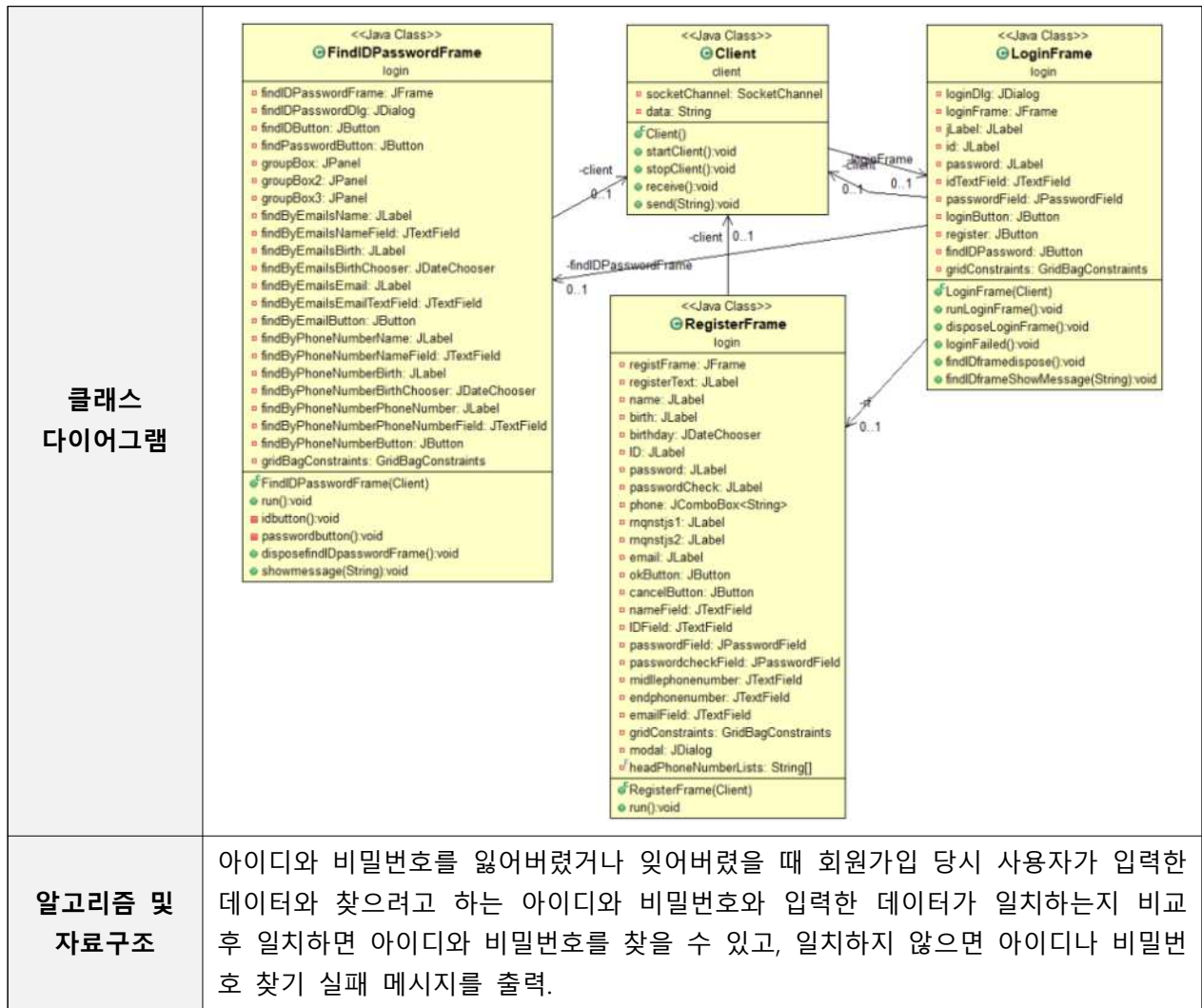
사용자가 부동산 매물 정보제공 시스템을 사용할 수 있도록 사용자의 정보를 받아 회원가입 후 프로그램을 사용할 수 있도록 한다. 그러나 사용자가 아이디와 비밀번호를 잊어버렸을 때 자신이 가입할 때 입력했던 정보와 현재 입력하는 정보를 비교하여 아이디와 비밀번호를 찾아주는 기능이다.

3.4.1. 분석

		내용
요구 사항		아이디 비밀번호를 찾기 위해 입력한 데이터(이름, 생년월일, 전화번호 등)를 서버에 저장해야 하며 파일입출력이 요구된다.
비기능 요구 사항	성능 효율성	아이디 비밀번호 찾기 버튼을 눌렀을 때 아이디와 비밀번호를 바로 알려줄 수 있어야 함
	사용성	로그인에 관련된 정보(아이디, 비밀번호)를 잊어버렸을 사용할 수 있는 기능이다.
	신뢰성	사용자가 입력한 데이터를 다른 사람이 알지 못하면 해당 사용자의 아이디와 비밀번호를 알 수 없다.
특징 구현 방안		파일입출력을 통해 작성한 양식의 데이터를 텍스트 파일에 저장하여 서버로 보낸 데이터와 사용자가 아이디 비밀번호를 찾을 때 작성한 데이터와 비교하여 로그인에 관련된 정보를 알려주는 방법

3.4.2. 설계

		내용
특징 설계		아이디 비밀번호를 자주 까먹는 사람들을 위해 자신이 가입할 때 입력한 데이터와 찾으려고 하는 아이디나 비밀번호를 입력하고 전화번호나 이메일 이름 생년월일 중 하나를 입력받아 서버에 저장되어있는 데이터와 비교하여 그 결과가 일치하면 아이디와 비밀번호를 가르쳐 주도록 설계함.



아이디 찾는 코드 및 GUI

이메일로 아이디 찾기

```
private void idbutton() {
    groupBox3.removeAll();
    findIDPasswordDlg.remove(groupBox3);
    //이메일을 입력하면 아이디를 찾아준다.
    groupBox.setBorder(BorderFactory.createTitledBorder("이메일로 찾기"));
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 1;
    gridBagConstraints.anchor = GridBagConstraints.WEST;
    findIDPasswordDlg.getContentPane().add(groupBox, gridBagConstraints);
    groupBox.setLayout(new GridBagLayout());
    //휴대폰번호를 입력하면 아이디를 찾아준다
    groupBox2.setBorder(BorderFactory.createTitledBorder("휴대폰번호로 찾기"));
    groupBox2.setLayout(new GridBagLayout());

    gridBagConstraints.gridx = 1;
    gridBagConstraints.gridy = 1;
    findIDPasswordDlg.getContentPane().add(groupBox2, gridBagConstraints);
    findByEmailsName.setText("이름 : ");
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 0;
    groupBox.add(findByEmailsName, gridBagConstraints);
    gridBagConstraints.gridx = 1;
    findByEmailsNameField.setColumns(10);
    groupBox.add(findByEmailsNameField, gridBagConstraints);
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 1;
    findByEmailsBirth.setText("생년월일 : ");
    groupBox.add(findByEmailsBirth, gridBagConstraints);
    Calendar cal = Calendar.getInstance();
    cal.add(Calendar.YEAR, -24);
    findByEmailsBirthChooser.setDate(cal.getTime());
    findByEmailsBirthChooser.setMaxSelectableDate(new Date());
    gridBagConstraints.gridx = 1;
    groupBox.add(findByEmailsBirthChooser, gridBagConstraints);
    findByEmailsEmail.setText("이메일 : ");
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 2;
    groupBox.add(findByEmailsEmail, gridBagConstraints);
    gridBagConstraints.gridx = 1;
    findByEmailsEmailTextField.setColumns(10);
    groupBox.add(findByEmailsEmailTextField, gridBagConstraints);
    findByEmailButton.setText("찾기!");
    findByEmailButton.addActionListener(e -> {
        StringBuilder stringBuilder = new StringBuilder("이메일로 아이디 찾기");
        stringBuilder.append("\n");
        stringBuilder.append("이름 : ").append(findByEmailsNameField.getText()).append("\n");
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy년MM월dd일");
        String birthString = simpleDateFormat.format(findByEmailsBirthChooser.getDate());
        stringBuilder.append("생년월일 : ").append(birthString).append("\n");
        stringBuilder.append("이메일 : ").append(findByEmailsEmailTextField.getText()).append("\n");
        client.send(stringBuilder.toString());
    });
}
```

휴대폰 번호로 아이디 찾기

```

gridBagConstraints.gridwidth = 2;
gridBagConstraints.gridy = 3;
gridBagConstraints.gridx = 0;
groupBox.add(findByEmailButton, gridBagConstraints);
gridBagConstraints.gridwidth = 1;
findByPhoneNumberName.setText("이름 : ");
gridBagConstraints.gridx = 0;
gridBagConstraints.gridy = 0;
groupBox2.add(findByPhoneNumberName, gridBagConstraints);
gridBagConstraints.gridx = 1;
findByPhoneNumberNameField.setColumns(10);
groupBox2.add(findByPhoneNumberNameField, gridBagConstraints);
gridBagConstraints.gridx = 0;
gridBagConstraints.gridy = 1;
findByPhoneNumberBirth.setText("생년월일 : ");
groupBox2.add(findByPhoneNumberBirth, gridBagConstraints);
gridBagConstraints.gridx = 1;
groupBox2.add(findByPhoneNumberBirthChooser, gridBagConstraints);
findByPhoneNumberBirthChooser.setDate(cal.getTime());
findByPhoneNumberBirthChooser.setMaxSelectableDate(new Date());
gridBagConstraints.gridy = 2;
gridBagConstraints.gridx = 0;
findByPhoneNumberPhoneNumber.setText("휴대폰 번호 : ");
groupBox2.add(findByPhoneNumberPhoneNumber, gridBagConstraints);
gridBagConstraints.gridx = 1;
findByPhoneNumberPhoneNumberField.setColumns(10);
groupBox2.add(findByPhoneNumberPhoneNumberField, gridBagConstraints);
gridBagConstraints.gridwidth = 2;
gridBagConstraints.gridx = 0;
gridBagConstraints.gridy = 3;
findByPhoneNumberButton.setText("찾기");
findByPhoneNumberButton.addActionListener(e -> {
    StringBuilder stringBuilder = new StringBuilder("휴대폰 번호로 아이디 찾기");
    stringBuilder.append("\n");
    stringBuilder.append("이름 : ").append(findByPhoneNumberNameField.getText()).append("\n");
    SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy년MM월dd일");
    String birthString = simpleDateFormat.format(findByPhoneNumberBirthChooser.getDate());
    stringBuilder.append("생년월일 : ").append(birthString).append("\n");
    stringBuilder.append("휴대폰 번호 : ").append(findByPhoneNumberPhoneNumberField.getText()).append("\n");
    client.send(stringBuilder.toString());
});
groupBox2.add(findByPhoneNumberButton, gridBagConstraints);
findIDPasswordDlg.getContentPane().revalidate();
findIDPasswordDlg.getContentPane().repaint();
}

```


3.4.3. 구현과 테스트

아이디/비밀번호 찾기 GUI

아이디 비밀번호 찾기!!
✕

아이디 찾기

이메일로 찾기

이름 :

생년월일 : 12. 13.

이메일 :

찾기!

비밀번호 찾기

휴대폰번호로 찾기

이름 :

생년월일 : 12. 13.

휴대폰 번호 :

찾기

```

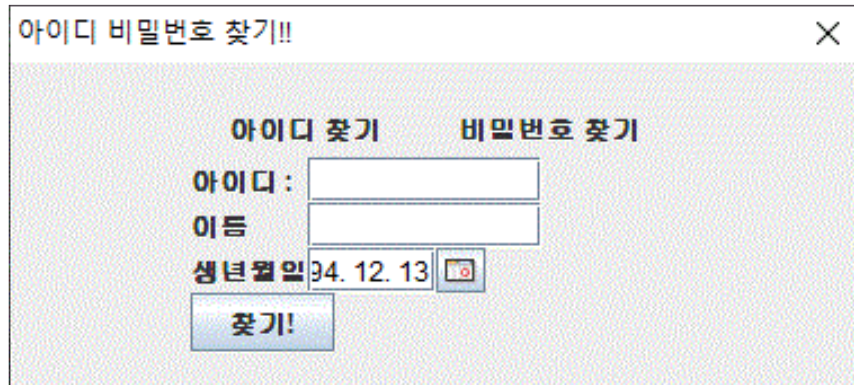
public void run() {

    findIDButton.setText("아이디 찾기");
    findPasswordButton.setText("비밀번호 찾기");
    findIDButton.setBorderPainted(false); // 버튼을
    findIDButton.setFocusPainted(false); // label처럼
    findIDButton.setContentAreaFilled(false); // 보이게
    findIDButton.setOpaque(false); // 하기 위해 하는 작업
    findPasswordButton.setBorderPainted(false); // 버튼을
    findPasswordButton.setFocusPainted(false); // label처럼
    findPasswordButton.setContentAreaFilled(false); // 보이게
    findPasswordButton.setOpaque(false); // 하기 위해 하는 작업
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 0;
    findIDPasswordDlg.getContentPane().add(findIDButton, gridBagConstraints);
    gridBagConstraints.gridx = 1;
    gridBagConstraints.gridy = 0;
    findIDPasswordDlg.getContentPane().add(findPasswordButton, gridBagConstraints);
    idbutton();
    findIDButton.addActionListener(e -> { //아이디 찾기 버튼 눌렀을때 이벤트 처리기

        idbutton();
    });
    findPasswordButton.addActionListener(e -> { //비밀번호 찾기 눌렀을때 이벤트 처리기

        passwordbutton();
    });
    findIDPasswordDlg.pack();
    findIDPasswordDlg.setResizable(false);
    findIDPasswordDlg.setVisible(true);
}
    
```

비밀번호 찾는 코드 및 GUI



```
private void passwordbutton() {
    groupBox.removeAll();
    groupBox2.removeAll();
    findIDPasswordDlg.remove(groupBox);
    findIDPasswordDlg.remove(groupBox2);
    findIDPasswordDlg.getContentPane().revalidate();
    findIDPasswordDlg.getContentPane().repaint();
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 1;
    gridBagConstraints.gridwidth = 2;
    findIDPasswordDlg.getContentPane().add(groupBox3, gridBagConstraints);
    gridBagConstraints.gridx = 0;
    gridBagConstraints.gridy = 0;
    gridBagConstraints.gridwidth = 1;
    JLabel id = new JLabel("아이디 : ");
    groupBox3.add(id, gridBagConstraints);
    gridBagConstraints.gridx = 1;
    JTextField idText = new JTextField();
    idText.setColumns(10);
    groupBox3.add(idText, gridBagConstraints);
    JLabel name = new JLabel("이름");
    gridBagConstraints.gridy = 1;
    gridBagConstraints.gridx = 0;
    groupBox3.add(name, gridBagConstraints);
    gridBagConstraints.gridx = 1;
    JTextField nameText = new JTextField();
    nameText.setColumns(10);
    groupBox3.add(nameText, gridBagConstraints);
    JLabel birth = new JLabel("생년월일");
    gridBagConstraints.gridy = 2;
    gridBagConstraints.gridx = 0;
    groupBox3.add(birth, gridBagConstraints);
    JDateChooser birthChooser = new JDateChooser();
    Calendar cal = Calendar.getInstance();
    cal.add(Calendar.YEAR, -24);
    birthChooser.setDate(cal.getTime());
    birthChooser.setMaxSelectableDate(new Date());
    gridBagConstraints.gridx = 1;
    groupBox3.add(birthChooser, gridBagConstraints);
    gridBagConstraints.gridy=3;
    gridBagConstraints.gridx=0;
    gridBagConstraints.gridwidth=2;
    JButton findButton = new JButton("찾기!");
}
```

```

groupBox3.add(findButton,gridBagConstraints);
findButton.addActionListener(e -> {
    StringBuilder stringBuilder = new StringBuilder("비밀번호 찾기").append("\n");
    stringBuilder.append("아이디 : ").append(idText.getText()).append("\n");
    stringBuilder.append("이름 : ").append(nameText.getText()).append("\n");
    SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy년MM월dd일");
    String birthString = simpleDateFormat.format(birthChooser.getDate());
    stringBuilder.append("생년월일 : ").append(birthString).append("\n");
    client.send(stringBuilder.toString());
});
findIDPasswordDlg.getContentPane().revalidate();
findIDPasswordDlg.getContentPane().repaint();
}

public void disposefindIDpasswordFrame(){
    findIDPasswordDlg.dispose();
    findIDPasswordDlg.setVisible(false);
}
public void showmessage(String message){
    JOptionPane.showMessageDialog(null, message, "응답", JOptionPane.INFORMATION_MESSAGE);
}
}

```

아이디 이메일로 찾기 테스트 결과	
① 초기 실행 창	② 이메일 찾기(입력)
③ 최종 결과 창	

아이디 휴대폰 번호로 찾기 테스트 결과	
① 초기 실행 창	② 이메일 찾기(입력)
<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> <div style="display: flex; justify-content: space-between;"> 아이디 비밀번호 찾기!! × </div> <div style="display: flex; justify-content: space-around;"> <div style="width: 45%;"> <p style="text-align: center;">아이디 찾기</p> <p>이메일로 찾기</p> <p>이름 : <input type="text"/></p> <p>생년월일 : 12. 13. <input type="text"/></p> <p>이메일 : <input type="text"/></p> <p style="text-align: center;">찾기!</p> </div> <div style="width: 45%;"> <p style="text-align: center;">비밀번호 찾기</p> <p>휴대폰번호로 찾기</p> <p>이름 : <input type="text"/></p> <p>생년월일 : 12. 13. <input type="text"/></p> <p>휴대폰 번호 : <input type="text"/></p> <p style="text-align: center;">찾기</p> </div> </div> </div>	<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> <div style="display: flex; justify-content: space-between;"> 아이디 비밀번호 찾기!! × </div> <div style="display: flex; justify-content: space-around;"> <div style="width: 45%;"> <p style="text-align: center;">아이디 찾기</p> <p>이메일로 찾기</p> <p>이름 : <input type="text"/></p> <p>생년월일 : 12. 13. <input type="text"/></p> <p>이메일 : <input type="text"/></p> <p style="text-align: center;">찾기!</p> </div> <div style="width: 45%;"> <p style="text-align: center;">비밀번호 찾기</p> <p>생년월일로 찾기</p> <p>이름 : <input type="text" value="김상환"/></p> <p>생일 : 6. 3. 6. <input type="text"/></p> <p>휴대폰 번호 : 010-1234-5678</p> <p style="text-align: center;">찾기</p> </div> </div> </div>
③ 최종 결과 창	
<div style="border: 1px solid #ccc; padding: 10px; margin: 0 auto; width: 80%;"> <div style="display: flex; justify-content: space-between;"> 응답 × </div> <div style="text-align: center; padding: 20px;"> <p style="font-size: 1.2em; font-weight: bold; color: #000080;">아이디는 sh8992입니다.</p> <div style="border: 1px solid #000080; padding: 5px; display: inline-block; margin-top: 10px;"> <p style="margin: 0;">확인</p> </div> </div> </div>	

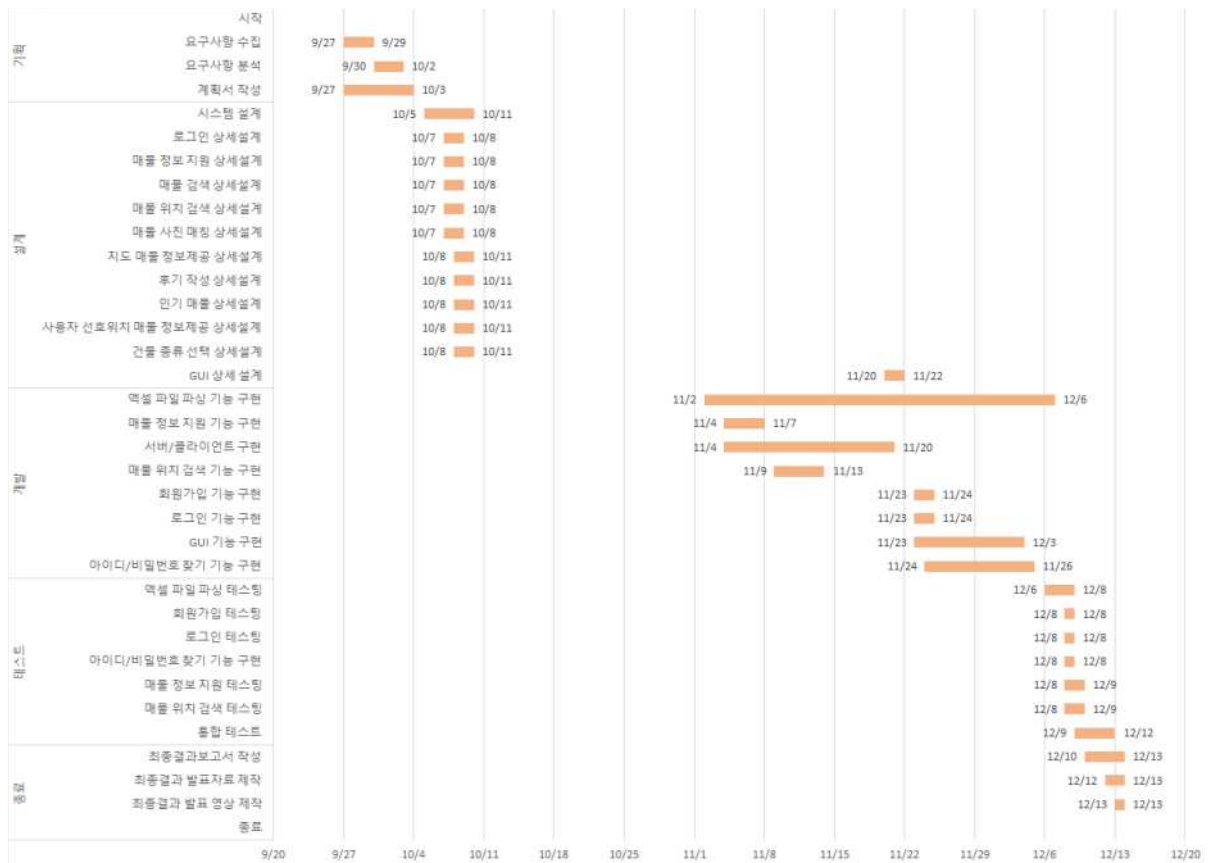
비밀번호 찾기 테스트 결과	
① 초기 실행 창	② 이메일 찾기(입력)
<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> <div style="display: flex; justify-content: space-between;"> 아이디 비밀번호 찾기!! × </div> <div style="display: flex; justify-content: space-around;"> <div style="width: 45%;"> <p style="text-align: center;">아이디 찾기</p> <p>아이디 : <input type="text"/></p> <p>이름 : <input type="text"/></p> <p>생년월일 : 94. 12. 13. <input type="text"/></p> <p style="text-align: center;">찾기!</p> </div> <div style="width: 45%;"> <p style="text-align: center;">비밀번호 찾기</p> </div> </div> </div>	<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> <div style="display: flex; justify-content: space-between;"> 아이디 비밀번호 찾기!! × </div> <div style="display: flex; justify-content: space-around;"> <div style="width: 45%;"> <p style="text-align: center;">아이디 찾기</p> <p>아이디 : <input type="text" value="sh8992"/></p> <p>이름 : <input type="text" value="김상환"/></p> <p>생년월일 : 1996. 3. 6. <input type="text"/></p> <p style="text-align: center;">찾기!</p> </div> <div style="width: 45%;"> <p style="text-align: center;">비밀번호 찾기</p> </div> </div> </div>
③ 최종 결과 창	
<div style="border: 1px solid #ccc; padding: 10px; margin: 0 auto; width: 80%;"> <div style="display: flex; justify-content: space-between;"> 응답 × </div> <div style="text-align: center; padding: 20px;"> <p style="font-size: 1.2em; font-weight: bold; color: #000080;">비밀번호는 1234입니다.</p> <div style="border: 1px solid #000080; padding: 5px; display: inline-block; margin-top: 10px;"> <p style="margin: 0;">확인</p> </div> </div> </div>	

4.1. 프로젝트 완성도

완성도 70%

4.2. 일정 계획 평가





계획단계에서 작성한 일정표를 토대로 프로젝트를 진행하려 했으나 구현하고자 하는 초기기능이 너무 단순하다고 생각하여 추가한 기능과 현실적으로 구현할 수 없는 기능들 그리고 수업시간에 새롭게 배운 내용을 추가하려다 보니 계획과는 조금 다르게 프로젝트가 진행되었습니다. 그 주범으로 엑셀 파싱 부분에서 POI 라이브러리 4.0 버전에서 사라진 메소드를 자꾸 사용하려 하다 보니 생각보다 시간이 많이 지체되었고 그 과정에서 프로젝트 진행이 지체되었고 그래도 결국 오류와 문제점을 찾아내서 그 뒤로는 앞서 계획해 높은 일정과는 조금 다르지만 꾸준하고 정기적으로 프로젝트를 진행하여 초기에 계획했던 기능들은 대부분 구현하였습니다.

4.3. 역할 수행 평가

작업	장윤재	김준근	김주의	김상환
아이디어 제공		√		
주제 선정	√	√	√	√
자료 조사			√	√
계획서 작성	√	√	√	√
회의록 작성	√		√	
엑셀 파일 작업	√	√		√
공공 API 분석	√	√	√	√
파싱	√	√		
코딩	√	√		
GUI 설계			√	√
GUI 구현	√	√		
테스팅	√		√	
형상 관리	√	√		
영상편집				√
최종 보고서	√	√	√	√
PPT			√	

4.4. 설계 구성요소

항목	내 용
분석	문제 정의와 작업 분해를 통하여 프로젝트 주제에 대한 목적 이해 및 해야 할 일을 계획할 수 있다.
	<ul style="list-style-type: none"> ▪ 방을 찾는 동의대학교 학생들, 살 집을 알아보는 사람들과 자신의 부모님께서 부동산 쪽 일을 하시는 분들의 이야기를 들어보았음. ▪ 여기서 직접 매물을 찾아 나서는 게 힘들고 불편하고 가끔은 허탕을 칠때도 있다는 문제점을 발견했음. ▪ 공공데이터 포털에 들어가 부동산정보에 관한 API가 있는지 찾아봄. ▪ 사용할 수 있는 데이터 중 아파트 매매정보와 단독다가구 정보, 연립다세대 정보, 오피스텔 정보에 흥미를 느낌 ▪ 방을 찾는 프로그램을 만들 결정하고 공공데이터를 파싱 하고 결과화면을 콘솔창이 아닌 GUI에 출력하기로 하였음.
시스템 설계	SW의 유연성, 유지보수성과 재사용성을 높이기 위하여 SW의 전체 구조를 클라이

	<p>엔트-서버 구조로 설계할 수 있다.</p> <ul style="list-style-type: none"> ▪ 최초 서버를 동작시키면 서버는 클라이언트가 접속할 때까지 대기하다가 클라이언트가 접속하면 로그인을 할 수 있도록 로그인 창을 실행하여 아이디와 비밀번호를 입력받게 함 ▪ 아이디와 비밀번호가 없으면 회원가입을 통해 회원등록을 하고 로그인을 하면 클라이언트가 정보를 요청할 수 있음. ▪ 클라이언트가 서버에서 파싱 된 부동산 매매정보를 요구했을 때, 서버는 만들어진 부동산정보 파싱 클래스에 연결되어 결과를 서버에 저장하고 클라이언트가 검색하는 지역의 정보를 다시 엑셀 파싱 하여 지역 코드를 보내주며 그 지역에 맞는 부동산 매물 정보를 GUI 창에 출력 ▪ 서버와 클라이언트는 멀티스레드로 구현하여 사용자의 다중 접속에도 대응할 수 있도록 설계
	<p>구현</p> <p>분석과 설계 산출물을 사용하여 정해진 일정에 맞게 소프트웨어를 구현할 수 있다.</p> <ul style="list-style-type: none"> ▪ 분석 단계에서 생각했던 파싱 된 공공 API 정보들을 서버에서 가지고 있도록 하여 서버 설계 ▪ 파싱 된 API 정보들에서 필요하지 않은 부분은 삭제하고 정말 필요한 부분만 가져옴 ▪ 서버와 클라이언트를 통하여 사용자에게 보여줄 GUI를 구현하여 사용자가 콘솔 창을 확인하지 않더라도 화면에 표시된 프레임과 버튼을 통하여 자신이 원하는 정보(부동산 가격, 건축년도, 지역, 건물 이름, 평수 등)를 한눈에 알기 쉽게 표현함 ▪ 초기에 계획했던 기능(지도에 가격정보를 출력하는 기능)은 지도를 받아오고 그 지도가 사용자가 검색한 위치로 이동하는 부분에서 어려움을 겪어 구현에서 제외

4.5. 현실적 제한조건

항목	내 용
생산성	<p>개발 과정에 생산성을 높일 수 있는 도구를 적용하였는지, 그리고 산출물을 활용하여 기존 시스템에 비해 생산성을 높일 수 있는지를 기술한다. 웹 응용 프로그램을 작성하기 위하여 모든 구성요소를 직접 프로그램하지 않고 기존의 잘 알려진 시스템 또는 라이브러리를 활용하여 코딩의 효율성 및 테스트가 보다 간편해짐을 이해하고 활용할 수 있다.</p>
	<ul style="list-style-type: none"> • 사용자가 집을 보러 직접 다니지 않아도 되며 직접 보러 다니는 것보다 작은 힘과 적은 시간으로 많은 매물을 알아보고 검색해볼 수 있으므로 생산성이 생긴다. • 매물의 등록날짜별로 원하는 지역의 부동산 매물 정보를 얻을 수 있으므로 생

	<p>산성이 생김</p> <ul style="list-style-type: none"> 만들어진 프로그램은 기존의 프로그램들과는 달리, 특정 지역에 등록된 모든 부동산정보를 가지고 있는 프로그램으로서, 기존의 다른 프로그램들에 비하여 더 많은 부동산 매물과 더많은 정보로 사용자에게 편의성을 제공
산업표준	<p>개발 프로세스나 개발 산출물에 아래와 같이 산업표준/국가표준/국제표준을 고려하였는지, 고려하였다면 어떻게 했는지를 기술한다.</p> <ol style="list-style-type: none"> 1. 개발 환경은 Eclipse/NetBeans IDE를 사용하여야 한다. 2. 형상 관리, 테스트, 문서화를 적용하여야 한다. 3. 클래스 다이어그램을 사용하여 프로그램의 아키텍처를 표현해야 한다. 4. 서버 프로그램과 클라이언트 프로그램은 TCP 또는 UDP 소켓 프로그래밍을 사용하여 구현한다. 5. 서버/클라이언트 프로그램은 강의 시간에 배운 Java SE를 활용한다.
	<ul style="list-style-type: none"> 개발 환경은 Eclipse Java Photon을 사용하였다. UML은 ObjectAid Class Diagram을 설치하여 만든 클래스의 관계를 그림으로 표현 UML로 만든 클래스의 관계를 그린 그림으로 클래스 구조를 디자인 패턴에 맞게 잘 표현 형상 관리 프로그램으로 git을 사용하였고 git을 편하게 사용하기 위해 스마트 깃을 사용하였다. 형상 관리를 통하여 소스코드를 더욱 효율적으로 관리하고 개발할 수 있었으며 협업의 재미를 느끼게 해준 부분이 여기 있다고 생각 테스팅은 SonarQube를 사용하여 버그와 코드스멜, 취약점을 수정하였으며, 소스코드의 문서화로 설계내용, 프로젝트 진행, 관리, 프로그램 유지 및 보수의 중요성을 알게 되었고 관련 내용을 공부하고 회의록에 표기하였다. 서버 프로그램과 클라이언트 프로그램은 TCP 소켓 프로그래밍을 사용하여 구현

5. 소감

장윤재

“이번 학기 초부터 내주셨던 자바 팀 프로젝트는 내 손으로 시작되었다. 아무도 나서지 않아 내가 직접 나가 교수님께서 만드신 무작위 팀 만들기 프로그램을 행운의 숫자 7을 생각하여 7번을 돌려나온 조가 지금의 자바 1분반 7조이다. 조가 만들어졌을 때 다른 조와는 다르게 조원이 한 명 작은 인원이라는 점에서 눈앞이 깜깜했다. ‘아 조원이 작아서 손해 보면 어쩌나, 열심히 하고자 하는 친구들이 한 명 줄어들 수도 있지 않을까?’ 하며 걱정을 했다. 하지만 그것은 진짜 걱정에 불과했다. 우리 조원들은 나에게 팀장이 되어 달라고 해서 의도하지 않게 팀장이 되어 조원들에게 이것저것을 요구하고 부탁하는 저의 기대에 응해주며 우리 팀의 조원들은 한 명 적은 인원수에도 굴하지 않고 프로젝트를 같이 헤쳐나갔다. 먼저, 훌륭한 자바 실력을 겸한 준근이의 창의적인 코딩 머리와 뛰어난 임기응변 능력으로 우리 조의 부동산 프로그램을 한층 더 멋진 프로그램으로 완성 시켜 주었고, 뛰어난 문서작성능력을 가진 주위는 여러 형식의 회의록과 보고서, 중간보고서, 소나큐브 보고서를 완벽하게 작성하고 미적인 부분까지 겸하여 정말 진짜 정말 문서를 잘 작성하였다. 상환이는 GUI를 그림으로 그려서 설계하고 프로그램이 완성되면 넣을 디자인 아이콘제작, 그리고 PPT를 보고 영상을 편집하는 능력을 보여주었다. 이렇게 모두가 갈등 없이 사고 무탈하게 웃으며 서로 칭찬하며 진행해보는 프로젝트는 처음이었고 정말로 이번 자바프로젝트가 힘들고 어려웠지만 이런 역경을 헤쳐나가는 재미가 있었다.”

김상환

“한 학기 동안 진행하였던 자바 팀 프로젝트는 다른 수업 팀 프로젝트와는 비교적 훨씬 길고 어려웠다. 처음 교수님이 팀을 무작위로 정해 주셨을 때 친하지 않은 조원 그리고 다른 조와는 다르게 조원이 한 명 적은 4명으로 구성된 팀이라 앞길이 막막했다. 하지만 팀원들과 프로젝트를 계획하고 수업시간에 배운 것을 토대로 역할을 분담하여 하나씩 건드려보니 기능이 구현되고 그때의 성취감은 말로 표현할 수가 없었다. 그렇게 팀원들과의 합도 점점 맞추고 프로젝트의 기능들을 하나씩 구현해 낼 때마다 자바라는 언어에 대해 이해할 수 있게 되었고 어느 순간 성장한 팀원들의 모습을 볼 수 있었다. 아쉬운 점은 다른 강의 과제와 시험 기간 때문에 시간이 부족하여 처음 계획했던 프로젝트를 100% 완성시키지 못한 점이다. 그래도 군에서 전역하고 처음 하는 팀 프로젝트였는데 나름 성공적으로 끝내서 뿌듯하고 부족한 프로그래밍 실력이라도 나를 잘 이끌어준 팀원들에게 감사를 전하고 싶다.”

김준근

“자바 팀 프로젝트는 굉장히 길고 어려운 프로젝트였다. 다른 수업과는 다르게 랜덤으로 팀원이 정해져 각기 다른 성격과 프로그래밍 실력을 가지고 있는 사람들과 약 3개월 동안 프로젝트를 진행 한다는 게 쉽지 않겠다고 생각이 들었다. 하지만 팀원 전체가 참여하려는 의지가 강하고 성격도 잘 맞아 순조롭게 프로젝트를 진행하였고 간간이 팀원들 간의 의견충돌도 있었지만 서로 배려하고 이해하며 프로젝트를 진행하였다. 처음 계획만큼 완벽하게 프로젝트를 완성하진 못했지만 버려지는 팀원 없이 역할분담을 잘 하여 끝까지 프로젝트를 완성했다는 점에서 팀원들에게 크게 감동했다. 프로젝트가 끝날 때쯤에는 자바라는 언어에 대해 좀 더 다가가 있는 나 자신을 보았고 이번 프로젝트가 팀 프로젝트, 자바를 이해할 수 있는 뜻깊은 시간이 되어 기쁘다.”

김주의

“힘들었다. 5명이 한 팀인데 우리 조는 4명이 팀이었고 다른 강의 과제들이 너무 많아서 힘들었다. 과제 때문에 하루종일 컴퓨터 보는 날이 많아져서 눈도 피로해지고 몸도 많이 안 좋아졌다. 그래도 상황이 오빠 빼고 다 아는 오빠들이어서 편했다. 원래 친하지 않은 사람에게는 말도 잘못하는데 아는 사람이니 불만이 생기면 바로 말할 수 있어서 좋았던 것 같다. 그리고 이 프로젝트로 몰랐던 상황이 오빠랑도 친해지고 다른 오빠들도 더 친해져서 좋았다. 그리고 제대로 된 프로젝트를 만들고 보고서를 쓰는게 진짜로 내가 프로그램을 만든 것처럼 느껴져서 좋았고 의미 있었다. 아쉬운 것은 이 프로젝트에서 나는 회의록, 중간/결과 보고서, 소나큐브 보고서 등 보고서들을 쓰는 게 중점이어서 코딩은 오빠들이 거의 다 하고 나는 옆에서 피드백을 조금 해줬다. 개인 컴퓨터도 없어서 커밋도 못해보는 게 아쉬웠다. 그리고 다들 다른 강의 과제들도 하다 보니 시간이 부족해서 프로젝트를 미완성한 것이 아쉬웠다. 힘들었던 점은 회의록을 쓰는 게 처음이라 어떤 식으로 써야 할지 몰라서 막막했고 보고서 쓰는 것도 개념들을 많이 알아야 하는데 나는 모르는 게 너무 많아서 답답했다. 그래도 오빠들이 많이 도와주었고 역할분담을 해서 거기에 맞춰서 하니 나는 글 쓰는 것에 집중할 수 있어서 보고서와 글들을 쓸 수 있었던 것 같다. 프로젝트를 하면서 도움을 많이 받았고 다들 말하면 많이 도와줘서 고마웠다.