* 서버 클래스
* StartServer.java
* 소켓을 생성하고 클라이언트의 접속요청을 받아들인다. 접속된 클라이언트의 요청 처리를 위해 스레드를 생성하고 다시 클라이언트의 접속요청을 기다린다.
* MainServer.java(서버시작클래스)
* Main메소드를 포함. StartServer()객체를 생성한 후 Server.start()함수를 실행하여 서버구동.
* MainThread.java
* 스레드의 Run()메소드 구현 클래스. 접속 요청된 클라이언트의 요청에 따라 각 기능별로 구분(로그인, 회원가입 등), receive\_Image(), send\_Image() 는 이미지를 전송하거나 받는다.
* DBConnection.java
* 오라클 드라이버를 연결하여 Conn 객체를 반환.(오라클 DB만 사용할 것이라 생각하여 따로 Interface를 만들지 않았슴)
* Functions\_Server.java
* 메인스레드에서 객체로 생성되어 함수로 호줄되는 클래스. 메인스레드로부터 conn, DTO객체를 인자값으로 받아, 기능별로 데이터베이스에 접근(데이터 삽입,삭제,수정)
* DTO.java
* 데이터 전송 객체
* DAO.java
* Post\_Dao.java
* PostInfo.java
* User\_Dao.java
* Userinfo
* 클라이언트 클래스
* Main.java
* Main()포함, 클라이언트 시작클래스
* Requestable.java
* 클라이언트에서 필요한 기능을 함수로 정의.
* Functions\_Client.java
* Requestable 인터페이스를 상속받아서 함수들을 재정의 및 구현
* DTO.java
* 데이터 전송객체
* Access.java
* 서버에 접속하는 하여 데이터를 전송하고 수신하는 객체.
* 회원가입
* (서버) MainServer() 에서 StartServer객체를 생성한후 Start()함수를 실행한다.
* (서버) Start()가 실행되면 소켓을 생성한 후 클라이언트로부터 접속요청을 기다린다.
* (클라이언트) Main클래스에서 Functions\_client객체를 생성한 후 Functions\_client의 Register()를 실행한다. Register()의 인자값은 회원가입시 필요한 정보를 포함(아이디, 이름, 비밀번호등). Register()는 DTO객체를 생성한 후 DTO객체에 필요한 정보를 싣고, Access객체를 생성한후 Request()를 실행한다. Request()의 인자로 DTO객체를 받아들인 후 소켓을 생성하고 DTO객체를 전달한다. 만일, 이미지전송이 있을경우 Request()의 인값으로 DTO객체와 이미지파일의 path를 인자 값으로 한다.
* (서버) 접속요청된 클라이언트의 요청처리를 위해 MainThread를 실행시킨다.
* (서버) 연결된 소켓을 통해 DTO객체를 받아들인후 option값을 통해 회원가입요청임을 파악한다.
* (서버) 스레드에서 Functions\_Server 객체를 생성한 후 register()를 실행한다. Register()의 인자값은 DTO객체와 conn. 만일 회원가입시 Image를 전송하고자 할 경우(DTO의 isImage값 true) receive\_Image()를 실행하여 클라이언트로부터 이미지를 받아들인다.
* (서버) register 함수를 통해 데이터베이스에 접근하여 DTO에 담겨있는 회원정보를 데이터베이스에 저장한다. 성공할 경우 true를 반환하고 실패할 경우 false를 반환한다.
* (서버) Send\_DTO객체의 result값에 Register()의 반환값을 저장한 후 소켓을 통해 클라이언트로 전송한다.
* (클라이언트) 연결되어 있는 소켓을 통해 서버로부터 DTO객체를 받아들이고 DTO객체의 isResult()함수를 통해 회원가입의 성공여부를 true와 false로 반환한다.
* DB
* 테이블 생성 쿼리

Create table userInfo(

ID varchar2(30) primary key not null,

name varchar2(20),

password varchar2(30) not null,

age varchar2(10),

sex varchar2(10),

PhoneNumber varchar2(30),

Job varchar2(20));

Create table userimage (

UI\_num number primary key,

imagename varchar2(50),

id varchar2(30));

Create table postimage(

PI\_num number primary key,

imagename varchar2(50),

id varchar2(30),

postnum number);

Create table postinfo(

postnum number primary key,

id varchar2(30),

comments varchar2(100));

* 시퀀스 생성
* PostImage 시퀀스

create sequence PI\_seq

start with 1

increment by 1

maxvalue 100000;

* UserImage 시퀀스

create sequence UI\_seq

start with 1

increment by 1

maxvalue 100000;

* PostInfo 시퀀스

create sequence post\_seq

start with 1

increment by 1

maxvalue 100000;

* 트리거 생성
* UserInfo의 ID값이 변할 때 PostInfo의 Id값도 자동으로 변경

Create or replace trigger trier\_id

after update of id on USERINFO

FOR EACH ROW

BEGIN

UPDATE POSTINFO

SET ID=:NEW.ID

WHERE ID=:OLD.ID;

END;

* UserInfo의 ID값이 변할 때 UserImage의 Id값도 자동으로 변경

Create or replace trigger trier\_id2

after update of id on USERINFO

FOR EACH ROW

BEGIN

UPDATE userimage

SET ID=:NEW.ID

WHERE ID=:OLD.ID;

END;

* 다이어그램



DB password 사용자 : project 패스워드 : kitri