

**Final Project Report (프로젝트 최종 보고서)**

Class Name. Introduction to DB in Spring 2019

Prof. Jongwek Lee

Student No. 2017313233

Student Name. 서창연(Changyeon Seo)

**1. Overview of project.**

-> 이 The subject of this database project is the development of second-hand trading services for college students. At the community "Everytime", which is emerging as a major student community in the university, the trade is frequently occurred between students who have been certified as SKKU students. Expansion of the service to a separate and independent system will not only increase the trust between the seller and the buyer, but also help to improve the economic growth of the members of the school.

-> First, we will restrict users of the above services to Sungkyunkwan University members. If we assume that everyone is a user, we need a separate procedure to certify that a particular user has a student ID to our university. Therefore, regardless of graduation / suspension / enrollment, we will assume that the person who has the student number is assumed to be a user of the service.

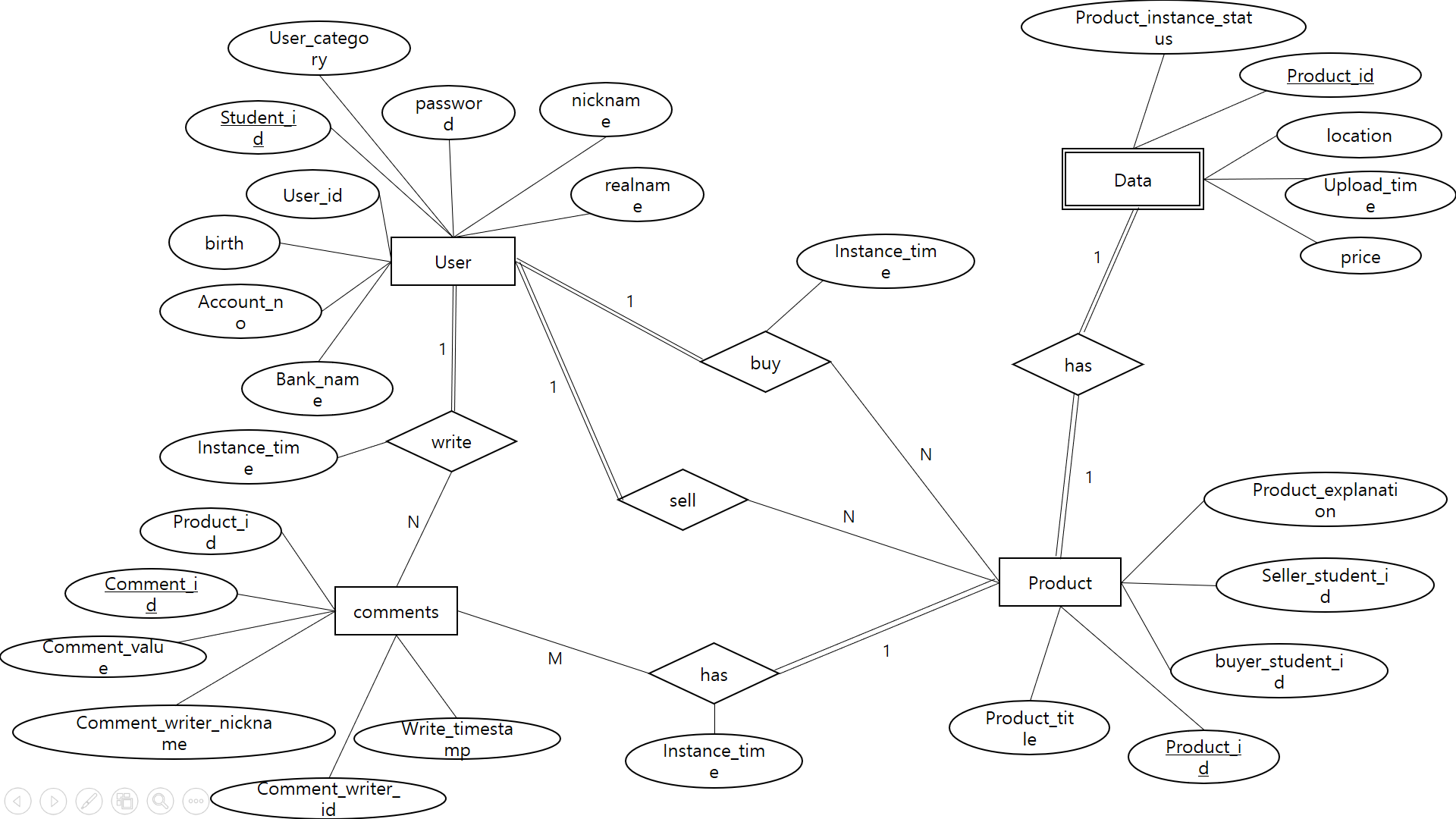
-> The user can join the membership. At this time, the DB receives information such as student number, ID, password, nickname, real name, date of birth, student status, account number, etc. from the user. After the join, user can then sell the item.

-> Sellers can create sales posts based on information about their items. At this time, the user enters information such as the name of the article, the id of the seller, description of the article, place of the transaction, price, etc.

-> Buyers can choose the items they want from the bulletin board and post comments that indicate their purchase intention to the post. The information that corresponds to the comment will be the comment writer, comment, and comment time.

-> The specific may choose one of the users who commented on his / her sales post to show his / her sales intent. After that, the seller and the buyer coordinate the opinions with the message.

**2. ER Model Explanation.**



1) User : An entity that contains user information

- student\_id : A numeric value from 0 to 9 consisting of 10 arrays, the number that all users will be given at admission. The above information can be used to uniquely identify a specific user.

- user\_id / password : A value consisting of a string of 20 or fewer characters, allowing a specific user to log in to the service through the above two values. The ID is disclosed on the web page at the time of input, and the password is not disclosed.

- nickname / realname : The above entity includes a nickname, which is a pseudonym for a particular user with 20 characters or less, and a real name information of up to 20 characters.

- user\_category : Value that identifies the status (undergraduate/graduate school/teacher) of this student.

- account\_no / bank : A string of up to 30 characters each, the value received from the user to provide financial services.

2) Product : Entity that contains product selling post information

- product\_id : A unique number assigned to a sales post for a particular product. It consists of an integer, which must be incremented each time the article is written.

- product\_title / product\_explanation : A value that represents the title of a product sales post that is less than 100 characters long and the content of the post, which is a string of up to 1000 characters.

- seller\_student\_id / buyer\_student\_id : An attribute that contains the information of the seller and buyer of the item. This value should have the same attribute as the student\_id in the user entity.

3) Data : Sub-entity that contains more specific value of product information.

- product\_instance\_status : A value indicating the current status of the used item. It is divided into five categories: Best, Upper, Middle, Lower, and Lowest.

- location : A trading place consisting of a string of up to 100 characters. It is organized in the form of 'campus + building name' considering the specificity of the community in the school.

- price : The price of a commodity that can be expressed in integer units.

- upload\_time : It means the writing time with form (YYYY-MM-DD HH: mm).

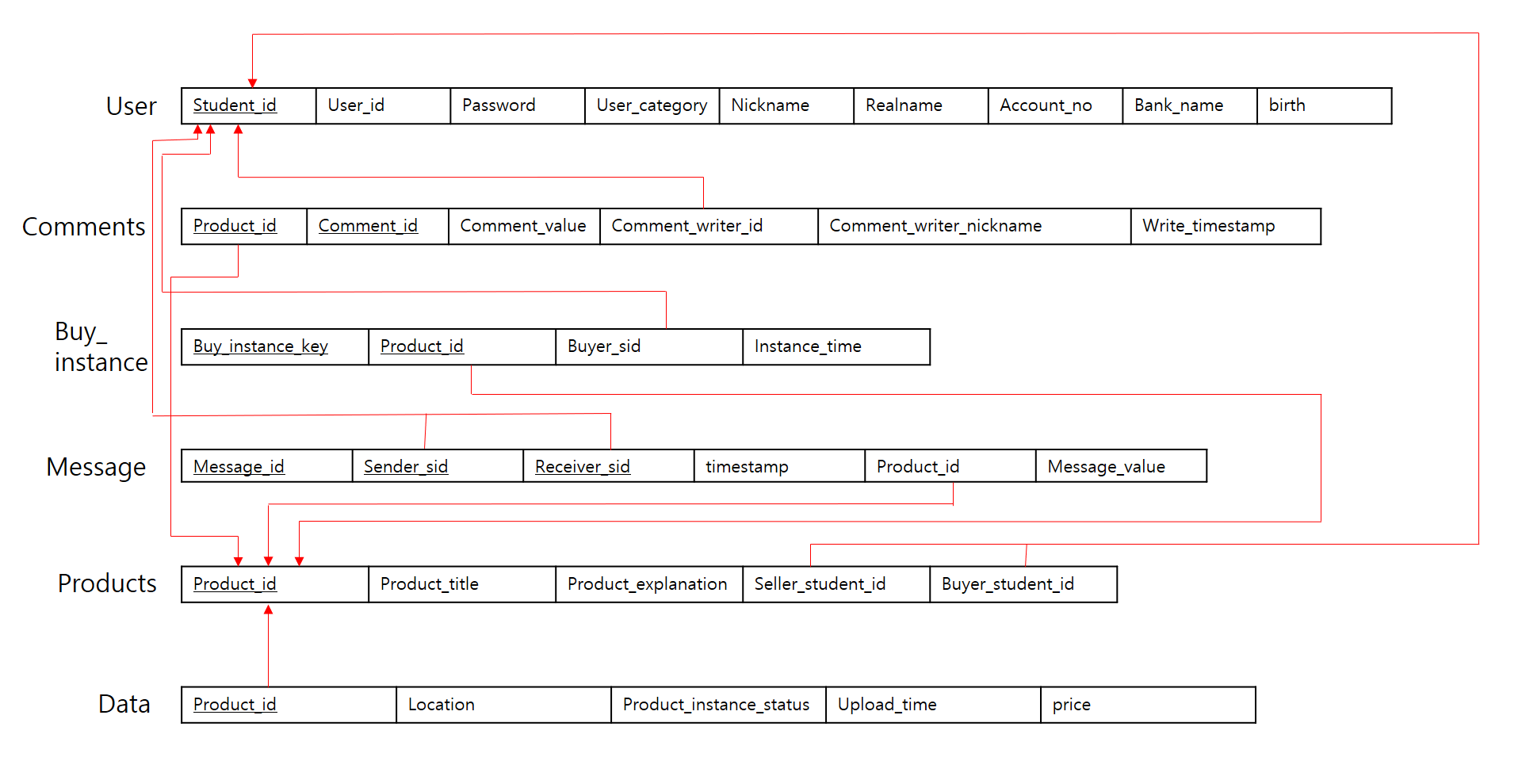
4) Comments : Entity that contains information of message between sellers and buyers.

- comment\_writer\_id / comment\_writer\_nickname : A value with the same attributes as the student\_id and nickname attributes in the user table. It contains information about the student's unique student number and nickname.

- comment\_value : A string of up to 200 characters containing information about the comment created.

- write\_timestamp : It means the writing time with form (YYYY-MM-DD HH: mm).

**3. Relational Data Model.**



1) Primary keys

- student\_id, comment\_id, buy\_instance\_key, message\_id, product\_id

2) Foreign key -> References

- Comments(Comment\_writer\_id) -> User(Student\_id)

- Message(Sender\_sid) -> User(Student\_id)

- Message(Receiver\_sid) -> User(Student\_id)

- Buy\_instance(Buyer\_sid) -> User(Student\_id)

- Buy\_instance(Seller\_sid)-> User(Student\_id)

- Product(Seller\_student\_id) -> User(Student\_id)

- Product(Buyer\_student\_id) -> User(Student\_id)

- Data(Product\_id) -> Product(Product\_id)

- Comments(Product\_id) -> Product(Product\_id)

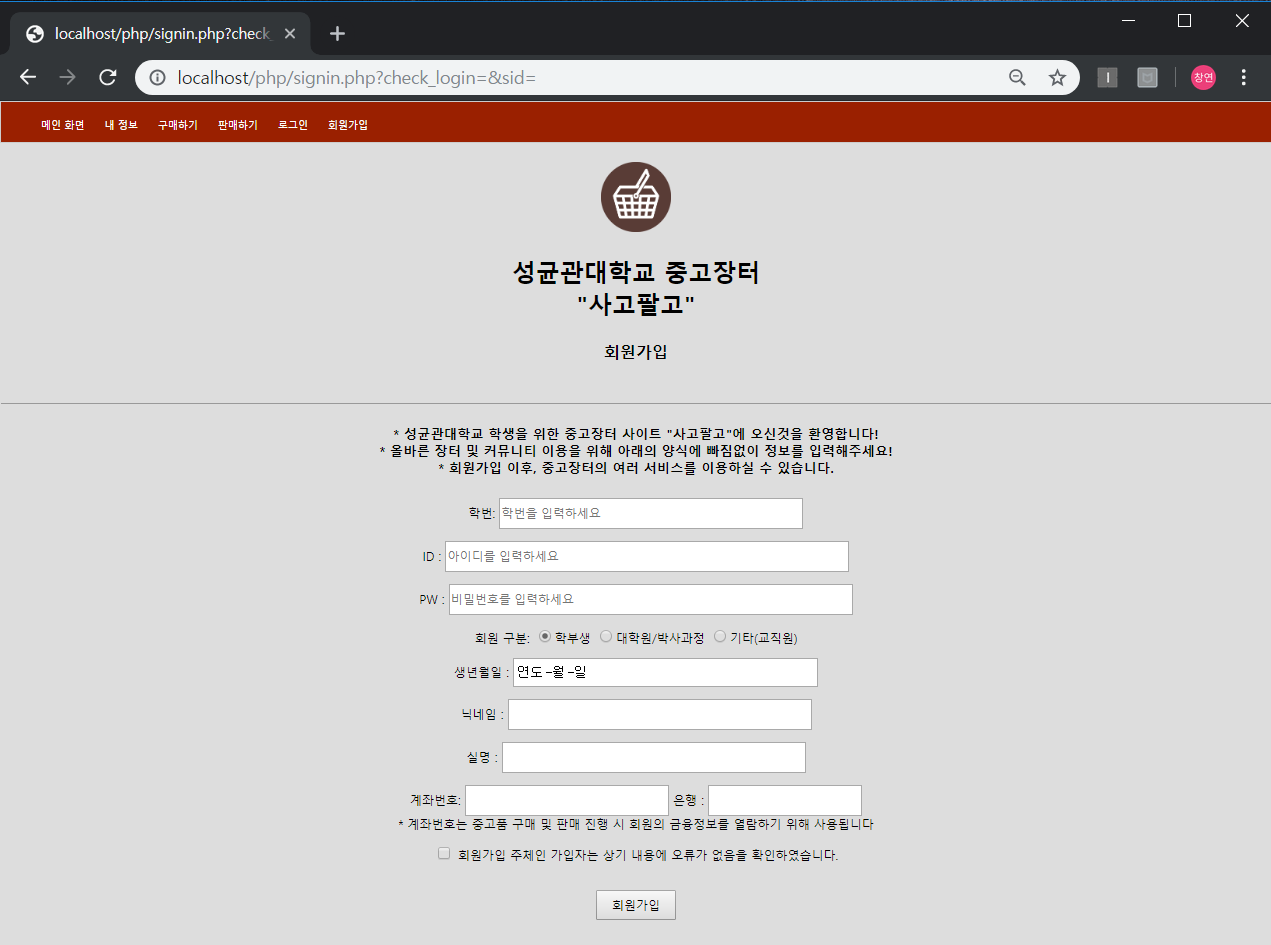
- Buy\_instance(Product\_id) -> Product(Product\_id)

- Message(Product\_id) -> Product(Product\_id)

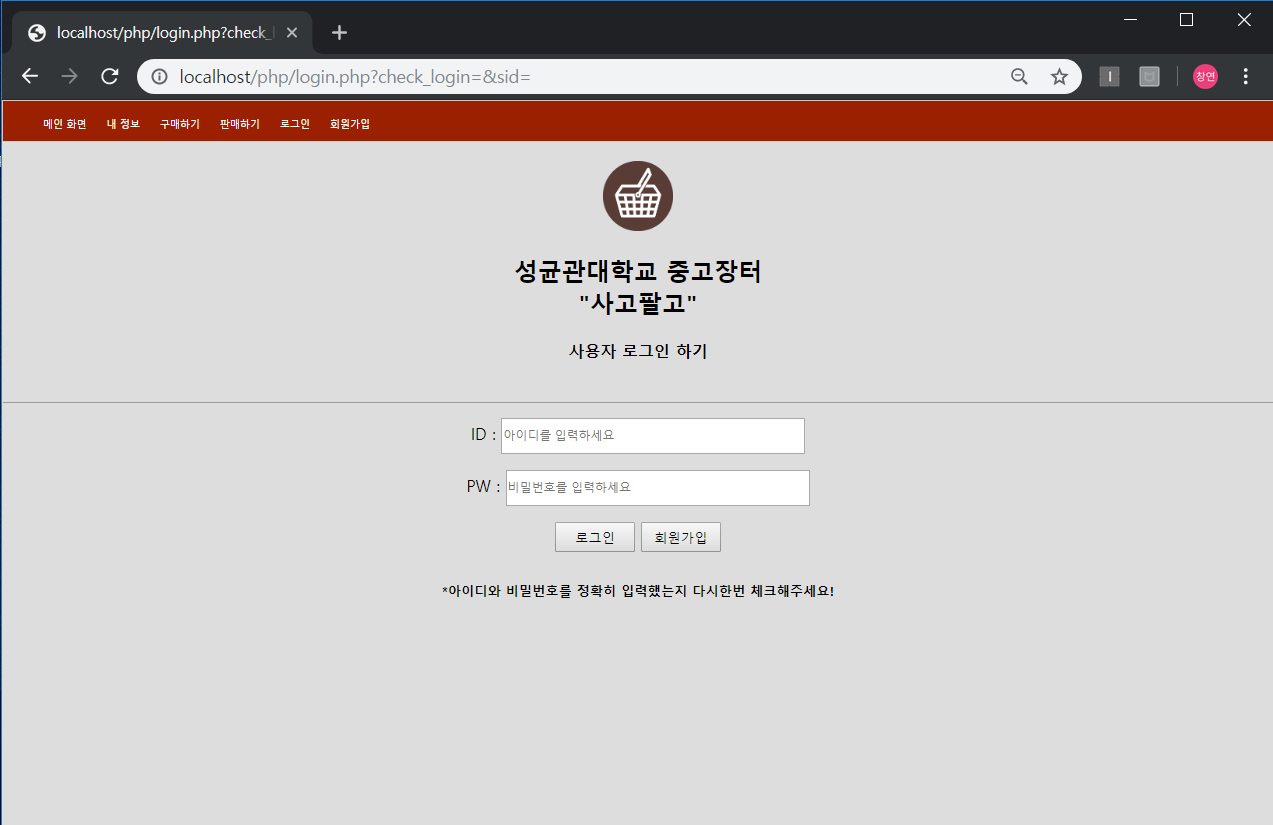
3) Change : Buy relation에 부수적으로 따라오는 attribute인 timestamp를 별도의 entity인 Buy\_instance를 생성해 줌으로서 해결하였다. 또한, ER diagram에서 미처 넣지 못했던 entity였던 Message entity를 만들어, seller와 buyer 간에 메시지를 주고받을 수 있도록 하였다. Data entity를 제외한 모든 entity는 별도의 primary key를 가지고 있다.

**4. Implementation and Results**

1) User enrollment function ( Signin / Signup / Signout )

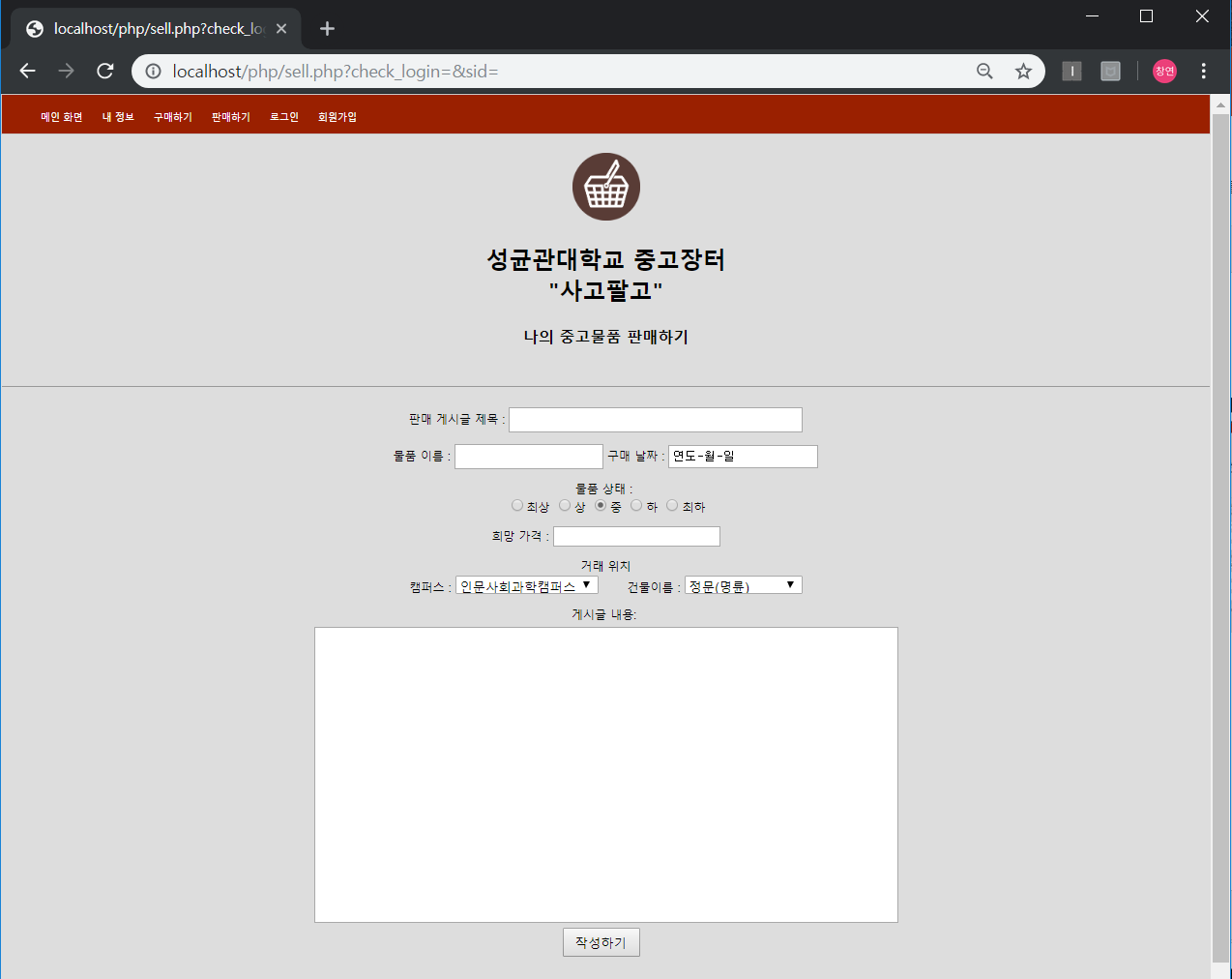


이 서비스는 회원가입과 로그인 기능을 지원한다. 회원가입의 경우 signin.php와 signin\_process.php의 두 가지 php 파일로 구현하였는데, 사용자로부터 회원가입에 필요한 여러가지 정보를 받아 POST 방식으로 signin\_process.php파일로 값을 넘겨준 뒤, "INSERT INTO user (student\_id, user\_id, password, user\_category, nickname, realname, account\_no, bank\_name, birth) VALUES ('$sid', '$uid', '$pwd', '$status', '$nick', '$real', '$acco', '$bank', '$birth');" 쿼리를 통해서 user 데이터베이스에 사용자 정보를 추가해준다.



로그인 기능은 "SELECT \* FROM user WHERE user\_id='$id' AND password='$pw';" 쿼리를 이용하여 구현하였다. User 테이블에서 입력받은 id와 비밀번호 값이 일치하는 컬럼을 얻어 체크한다.

2) Writing Post about product

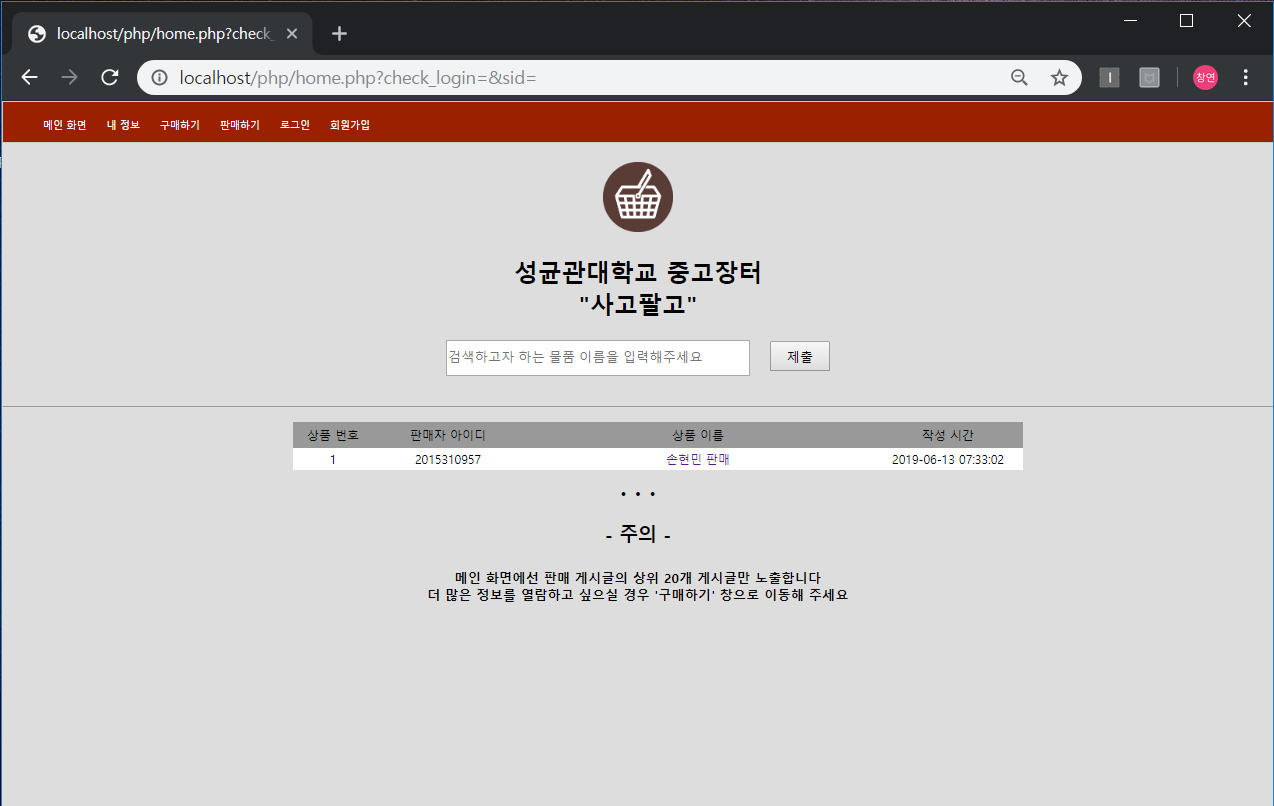


사용자로부터 값을 입력받아, product와 data table에 각각 별도로 값을 저장하는 쿼리를 사용한다.

1. product 삽입 : "INSERT INTO product (product\_id, product\_title, product\_explanation, seller\_student\_id, buyer\_student\_id) VALUES (0, '$title', '$contents', '$seller\_sid', null);"

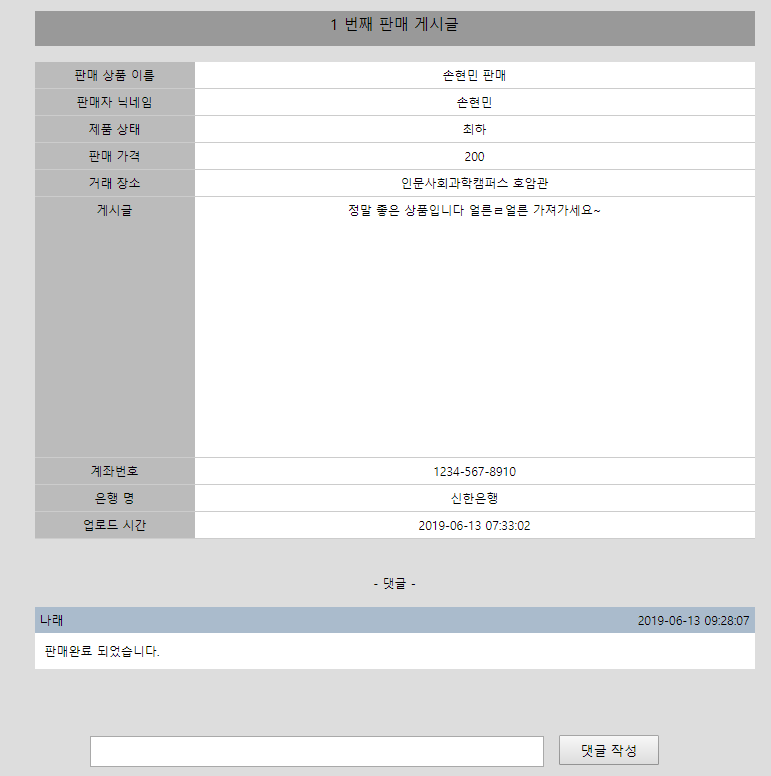
2. data 삽입 : "INSERT INTO data (product\_id, location, product\_instance\_status, upload\_time, price) VALUES (0, '$location', '$stuffstate', '$time', '$price');”

3) Searching Post



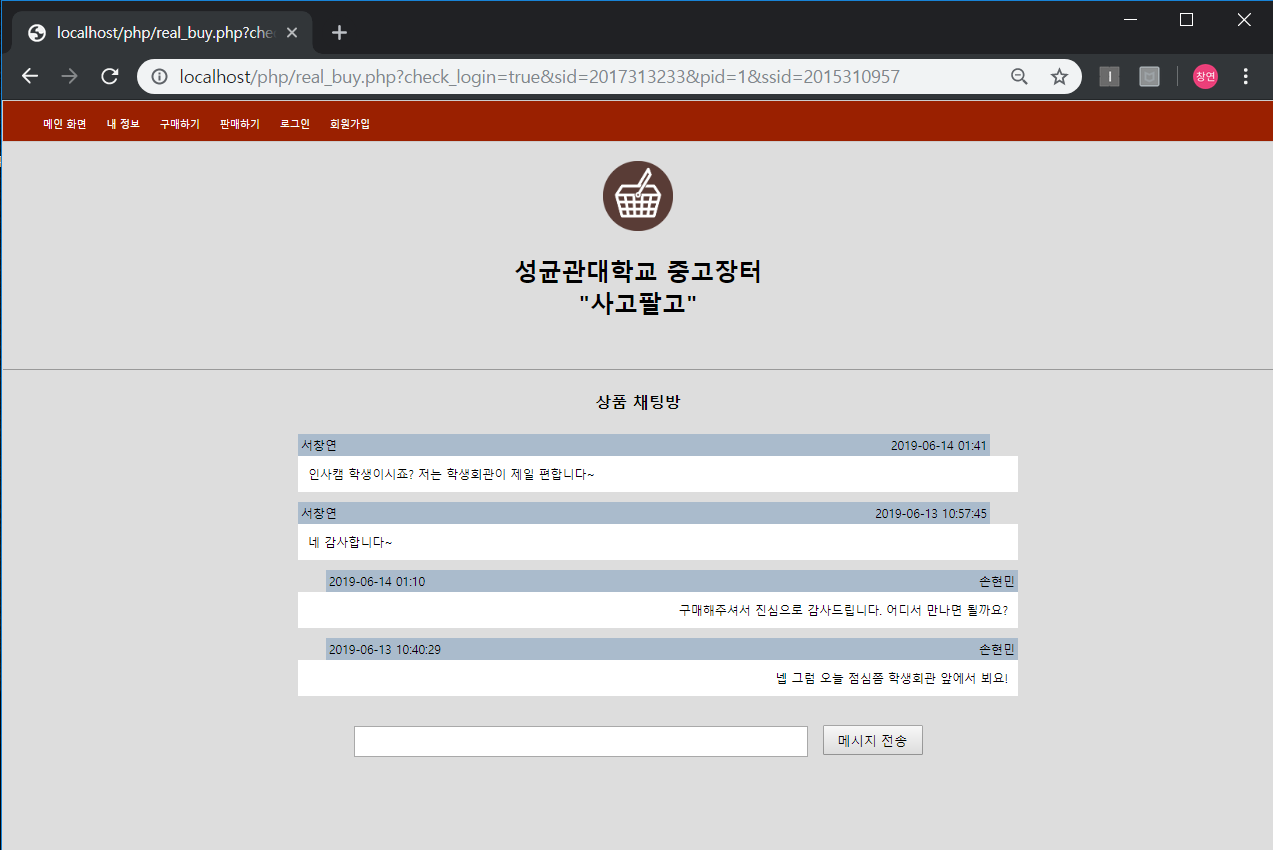
Product table에 있는 정보를 바탕으로 사용자로부터 게시글(상품) 이름 정보를 입력받아, 해당 문자열을 포함하고 있는 컬럼을 구합니다. 이 기능을 구현하기 위해 사용한 쿼리는 다음과 같습니다. "SELECT p.product\_id, p.seller\_student\_id, p.product\_title, d.upload\_time FROM product p, data d WHERE p.product\_id = d.product\_id AND p.product\_title LIKE '%$search\_stuffname%';"

4) Comments



"SELECT \* FROM comments WHERE comments.product\_id=".$row['product\_id'].";" 쿼리를 사용하여, 특정 product\_id를 가지고 있는 모든 comment들을 가져오도록 했습니다. 그리고 가져온 댓글들은 시간 순서대로 위에서부터 정렬하여 표시하였습니다.

5) Message



User\_id와 product\_id를 바탕으로 message table을 별도로 만들어, sender와 receiver로 나누어 메시지 기능을 구현하였습니다. 이 때 사용한 쿼리는 아래와 같습니다.

"INSERT INTO message (message\_id, sender\_sid, receiver\_sid, tstamp, product\_id, message\_value) VALUES (0, '$sender', '$receiver', '$time', '$pid', '$msg');"