DC 스마트그리드 홈 네트워크

팀 : 지하보다 지상

16011009 컴퓨터공학과 서지상, 16011040 컴퓨터공학과 장현희

16011034 컴퓨터공학과 박찬영, 16010995 컴퓨터공학과 조성우

목차

**1. 시스템 개요1**

**2. Use Case :: P2P 전력거래2**

1) Actor 목록3

2) P2P 전력거래3

2-1) 개요4

2-2) 사건 흐름4

**3. Data Map6**

**4. DB ER Diagram8**

**5. Sequence Diagram7**

**6. Class Diagram8**

**7. 필요 부품 목록10**

**시스템 개요**

DC 스마트그리드는 현재 사용되고 있는 시스템의 AC-DC로 변환하는 과정 없이 DC 전력만을 사용하여 전력 손실을 줄이고자 개발되었다. 본 시스템은 각 가정에서 신재생 에너지를 생산하게 하여 공적 전기 수요량을 줄여 공적 전기를 생산할 때 발생되는 환경 오염을 줄이고자 하였다. DC 스마트그리드 방식의 전력망은 양방향 통신이 가능하므로 신재생 에너지를 생산하는 가정 간의 p2p 거래로 잉여 전력의 처리와 공적 전기 수요량 감소뿐만 아니라 경제적 이득을 보게 했다.

본 시스템은 외부의 중앙 웹 서버와 각 가정 별 컨트롤 모듈들로 구성되며, 해당 웹 서버에 DB를 통해 데이터들을 저장 및 관리한다. 사용자는 모두 회원가입이 필수이며 각각 가정의 컨트롤 모듈을 독립적으로 가진다. 또한 사용자는 사용자가 설정한 기기들의 전원을 웹 서버를 통하여 원격제어가 가능하며 신재생 에너지의 전력 충전량과 총 사용중인 전력량, 배터리 잔량을 확인할 수 있다.

**Use Case :: P2P 전력거래**

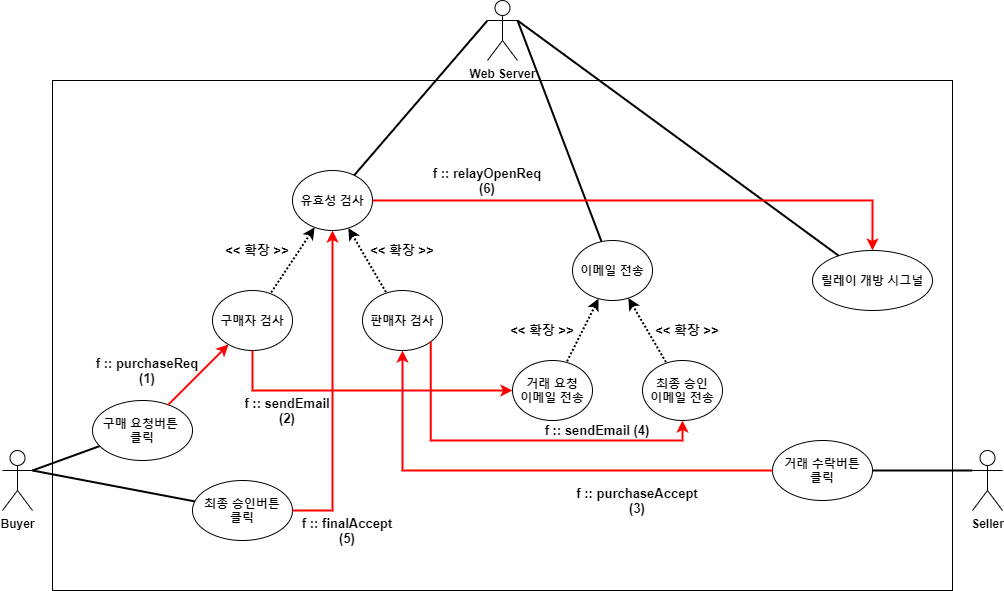


그림 1. 전력 거래 Usecase

**1) Actor 목록**

|  |  |  |
| --- | --- | --- |
| **Actor** | **구분** | **설명** |
| Seller | 시스템 사용자 | 전력 판매자, 거래 글 게시자 |
| Buyer | 시스템 사용자 | 전력 구매자, 거래 신청자 |
| Web Server | 제어 및 관리 | 전력 거래 중개 및 거래 과정 통괄 |

**2) P2P 전력거래**

**1. 개요**

등록된 판매 글에 구매자가 거래를 요청할 시, 구매자에 대한 유효성 검사가 시행되고, 통과 시, 판매자에게 거래 요청에 대한 이메일이 전송된다. 판매자가 거래 수락을 하면, 판매자에 대한 유효성 검사가 시행된다. 통과 시, 웹 서버에서 구매자 측에 최종 승인에 관한 메일을 전송한다. 제한시간 내에 구매자가 최종 승인을 하면, 구매자와 판매자 측에 대한 유효성 검사를 시행한다. 이후, 웹 서버에서 릴레이 컨트롤 모듈로 송전 채널을 열기 위한 신호를 보낸다. 이를 수용한 컨트롤러는 채널을 열어주고 송전을 진행한 후 전력 거래가 완료되면 채널을 닫고 웹 서버에 거래 완료 신호를 보낸다.

**2. 사건 흐름**

**\* 기본 흐름 (T)**

|  |  |  |
| --- | --- | --- |
| **T** | **Action** | **System** |
| T-1 | 구매자(Buyer)가 웹페이지에서 구매하고자 하는 전력량을 정한 후 ‘구매 신청’ 버튼을 누른다. | **purchaseReq(buyer, transaction, reqAmount)**을 호출하여 다음을 진행한다.  1. **buyerValidationTest(buyer, reqAmount)** 호출한다.  **True면** 다음 과정 시행, **False**면 대안상황으로 이동한다.  **(user\_battery\_max – user\_elec\_charge > reqAmount)**  2. **transaction\_status = 1**로 변경하여. ‘구매 신청’ 상태로 만든다.  3. **transaction\_buyer\_id**에 구매자의 **user\_id** 를 입력한다.  4. **buyer\_id**, 현재시간, **reqAmount** 등을 이용하여 해시값을 생성하고 **transaction\_hash**에 저장한다.  5. **sendEmail(seller, transaction)**호출하여 T-2를 이행한다. |
| T-2 | 웹 서버(Web Server)에서 판매자(Seller)에게 구매 요청 이메일을 전송한다. | **sendEmail(user, transaction)**을 호출하여 다음 중 하나를 진행한다.  1. user가 판매자(**seller**)일 경우,  **transaction**의 **transaction\_seller\_id == seller**의 **user\_id**  가 **True**이면 다음 이메일을 판매자에게 전송한다.  [“ {**TransAction.description**}에 구매 요청 되었습니다. 아래. 해당 거래를 아래 버튼을 통해 승인 또는 거절을 해 주시기 바랍니다.”]  2. user가 구매자(**buyer**)일 경우,  **transaction**의 **transaction\_seller\_id == seller**의 **user\_id**  가 **True**이면 다음 이메일을 판매자에게 전송한다.  [“ {**transaction.description**}의 거래가 수락되었습니다. 해당 거래를 아래 버튼을 통해 최종 승인 또는 거절을 해 주시기 바랍니다.”] |
| T-3 | 이메일을 받은 판매자(Seller)가 웹페이지에서 ‘거래 수락’ 버튼을 누른다. | **purchaseAccept(transaction\_id, hash)**을 호출하여 다음을 실행한다.  1. **transaction\_hash**에 저장된 값과 **hash**를 비교하여 올바른 요청인지 확인한다.  2. **sellerVallidationTest(seller, transaction)**을 호출 하여,  1). 판매자의 잔여 배터리 용량이 판매 전력량보다 큰지 확인한다.  (**user\_elec\_charge > reqAmount)**  2). 판매자의 초당 생산량이 초당 소모량보다 큰지 확인한다.  (**user\_elec\_supply > user\_elec\_usage)**  3. 1)과 2) 모두 **True**이면 3을 진행하고 그렇지 않다면 대안상황으로 이동한다.  4. **transaction\_status = 2**로 변경하여. ‘거래 수락’ 상태로. 만든다.  5. **buyer\_id**, 현재시간, **reqAmount** 등을 이용하여 해시값을 생성하고 **transaction\_hash**에 저장한다.  6. **sendEmail(buyer, transaction)**호출하여 T-2를 이행한다. |
| T-4 | 최종 승인 메일을 받은 구매자(Buyer)가 거래에 대한 최종 승인 버튼을 클릭한다. | **finalAccept(transaction\_id, hash)**을 호출하여 다음을 실행한다.  1. 구매자에게 **buyerValidationTest(…)**을,  판매자에게 **sellerValidationTest(…)**을 실행한다.  2. **buyerValidationTest(…)**에서 문제가 발생하였다면 대안 상황 S-1을 실행한다.  3. **sellerValidationTest(…)**에서 문제가 발생하였다면 대안 상황. S-2을 실행한다.  4. **transaction\_status = 3**로 변경하여. ‘거래 중’ 상태로 만든다.  **sendEmail(transaction)**함수를 호출하여 구매자와 판매자에게 다음과 같은 이메일을 보낸다.  [“거래가 최종 승인되어 진행되고 있습니다. 아래 링크로. 접속하여 거래 상태를 확인해 주세요.”]  5. **checkTimeLimit(timeLimit = 60)**을 호출하여 제한 시간 내(60분)에 승인이. 되지 않을 시에 대안 상황 S-3을 실행한다. |
| T-5 | 웹 서버에서 릴레이 개방 시그널을 전송한다. | 1. **isChannelOpen = True**로 바꾼다.  2. **relayOpenReq(buyer, seller)**을 호출하여 릴레이 채널 개방 시그널을(**openRelay**) HTTP프로토콜을 이용하여 JSON 형태로 전송한다. |

**\* 대안 흐름**

|  |  |  |
| --- | --- | --- |
| S-1 | buyerVaildationTest(buyer, reqAmount)  : 구매자의 부족한 배터리 용량이 구매하고자 하는 전력량을 수용할 수 없는 경우 | 1. 웹 서버에서 **alertError(buyer)**을 호출하여 구매자에게 다음과 같은 팝업 알림을 띄운다.  [“구매 불가 : 충전 가능한 배터리 용량이 구매하고자 하는 전력량 보다 적습니다.”]  2. **transaction\_status = 0**으로 바꾼 뒤 거래 프로세스를 종료한다. |
| S-2 | sellerValidationTest(seller, transaction)  1. 판매자의 배터리 충전량이 판매 전력량보다 적을 경우  2. 판매자의 초당 생산량이 초당 소모량보다 적을 경우 | 1-1). 웹 서버에서 **alertError(seller)**를 호출하여 판매자에게 다음과 같은 팝업 알림을 띄운다.  [“판매 불가 : 배터리의 잔량이 판매 전력량보다 적습니다.”]  1-2). **transaction\_status = 0**으로 바꾼 뒤 거래 프로세스를 종료한다.  2-1). 웹 서버에서 **alertError(seller)**를 호출하여 판매자에게 다음과 같은 팝업 알림을 띄운다.  [“판매 불가 : 전력 생산량이 총 소비량보다 적습니다.”]  2-2) **transaction\_status = 0**으로 바꾼 뒤 거래 프로세스를 종료한다. |
| S-3 | timeLimitError(buyer)  : 구매자가 제한시간(60분)이내로 최종 승인에 대한 응답이 없을 경우 | 1. **sendErrorEmail(seller, buyer);**를 호출하여 구매자와 판매자에게 다음과 같은 이메일을 전송한다.  [“구매자가 제한시간 내에 현재 거래에 대한 최종승인을 하지 않아 자동으로 거래가 취소되었습니다. ”]  2. **transaction\_status = 0**으로 바꾼 뒤 거래 프로세스를 종료한다. |

**\* 사전/사후 조건**

**- 사전 조건**

1. 로그인
2. 판매글 등록

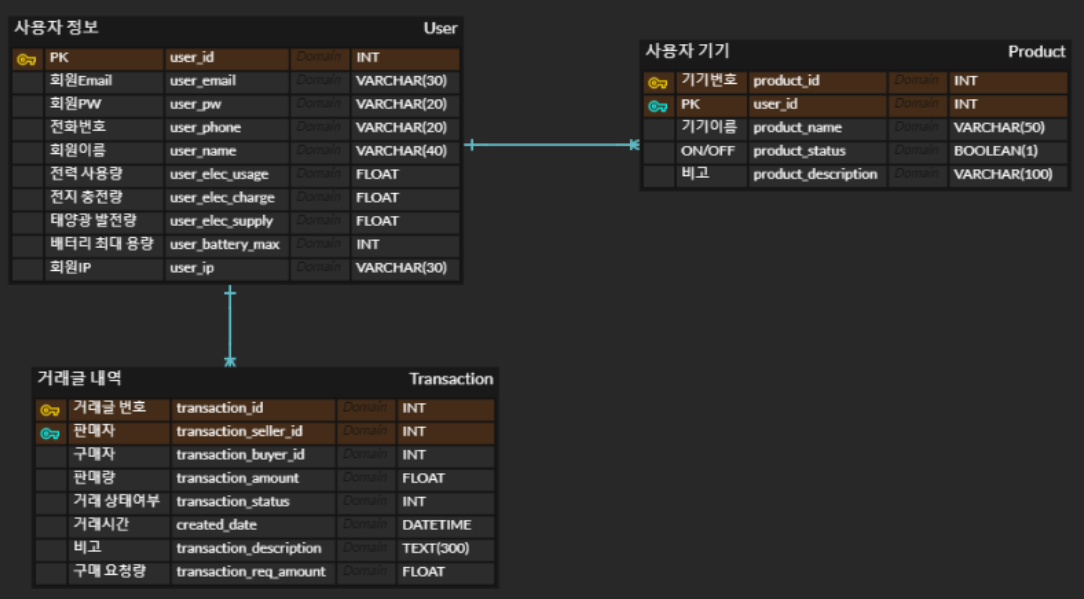
**- 사후 조건**

1. 거래액 반영
2. 거래자 및 판매자 거래 유효성 검사(실제로 목표한 송전량이 송전됐는지, 수신됐는지 확인)
3. 거래 내용을 DB에 저장

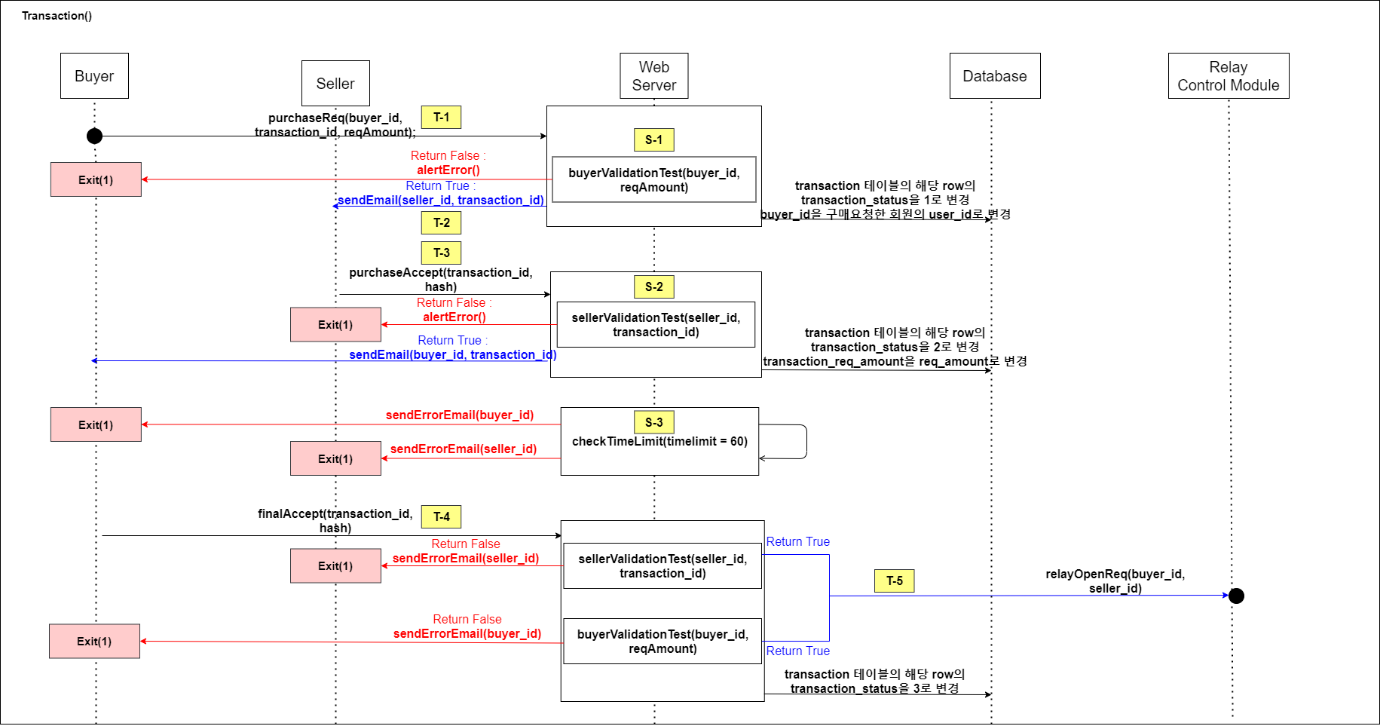
**Data Map**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Use case 명** | | | 전력 거래 | | |
| **Use case 설명** | | | 구매자와 판매자 사이의 전력 거래를 중개한다.. | | |
| **Data Name** | | | **Type** | **Size** | **Description** |
| transActionFrom  (Object) | | ID | Int | 20 | 거래 ID |
| buyer | Int | 20 | 게시글 생성자 ID |
| seller | Int | 20 | 거래 신청자 ID |
| amount | Float | 8 | 총 거래량 |
| reqAmount | Float | 8 | 구매 요청량 |
| CreateTStamp | DateTime | 4 | 거래 게시 시점 |
| endTStamp | DateTime | 4 | 거래 종료 시점 |
| startTStamp | DateTime | 4 | 거래 시작 시점 |
| User  (Object) | | ID | Int | 20 | 사용자 ID |
| eCharge | Float | 8 | 현재 보유 전기량 |
| batteryMax | Float | 8 | 배터리 용량 |
| leftover | Float | 8 | 잔여 배터리량 |
| Consumption\_SEC | Float | 8 | 초당 전력 소모량 |
| Supply\_SEC | Float | 8 | 초당 전력 생산량 |
| isChannelOpen | | | BOOLEAN | 1 | 릴레이 개폐 여부 |
| timeLimit | | | Int | 4 | 응답 대기 시간 |
| openRelay  (Object) | ID | | String | 20 | 거래 ID |
| buyerID | | String | 20 | 구매자 ID |
| sellerID | | String | 20 | 판매자 ID |
| amount | | Int | 4 | 전력 거래량 |
| targetGate | | Int | 4 | 개방할 릴레이 채널 포트 번호 |

**DB ER Diagram**



**Sequence Diagram**

****

**Class Diagram**

|  |  |
| --- | --- |
| **클래스** | **설명** |
| Main | 거래 중계를 위한 클래스 |
| Web Server | DB와 Client를 연결하기 위한 클래스 |
| Seller | 판매자의 정보를 관리하는 클래스 |
| Buyer | 구매자의 정보를 관리하는 클래스 |

**1. Main**

1) 개요

Main 클래스에서는 Transaction Object, User Object에서 각 정보를 얻어와 릴레이를 컨트롤하는 아두이노와 통신을 위한 함수 **relayOpenReq(buyer\_id, seller\_id)**를 사용해 구매자와 판매자간의 거래를 진행한다. 그리고 구매자가 오랫동안 거래 승인을 하지 않는 경우를 위해 **checkTimeLimit(timeLimit)**함수를 두어 지정해둔 **timeLimit**이 넘어가게되면 자동으로 거래를 취소시키는 **timeLimitError(buyer\_id)**함수를 사용하여 거래를 원활하게 한다

2) 연산

|  |  |  |  |
| --- | --- | --- | --- |
| **연산** | **파라미터** | **반환 값** | **설명** |
| relayOpenReq(  buyer\_id,  seller\_id  ) | INT, INT | Boolean | 구매자와 판매자의 거래를 위한 릴레이 채널을 열어주는 함수 |
| checkTimeLimit(  timeLimit  ) | INT | Boolean | 지정해둔 timeLimit을 넘겼는지 확인하는 함수 |
| timeLimitError(  buyer\_id  ) | INT | Boolean | timeLimit을 넘겼다면 현재 transaction에 대한 거래를 종료시키는 함수 |

**2. Transaction**

1) 개요

buyer의 구매 요청에 대한 buyer의 유효성 검사를 진행하고, buyer가 거래를 진행할 수 있는 상태인지를 확인한다. buyer에 대한 검사를 진행한 후 seller가 거래를 진행할 수 있는 상태인지를 확인한다. 앞의 과정에서 buyer or seller가 거래를 진행할 수 없는 상태가 된다면 거래를 진행할 수 없는 User에게 거래 불과 alert창을 띠운다. 거래를 진행하는 경우 거래 수락 or 구매 요청에 대한 확인 Email을 보내고, 최종 승인 Email을 보낸다. 이때 최종 승인 Email에 승인 응답이 없는 경우 거래 취소 Email을 보낸다.

2) 연산

|  |  |  |  |
| --- | --- | --- | --- |
| 연산 | 파라미터 | 반환 값 | 설명 |
| purchaseReq(  seller\_id,  transaction\_id,  reqAmount  ) | INT, INT, INT | x | buyer의 구매 요청에 대한 buyer의 유효성 검사 및 진행 |
| buyerValidationTest(  buyer\_id,  reqAmount  ) | INT, INT | Boolean | buyer가 사고자 하는 전력의 양이 buyer의 배터리의 총량보다 적거나 같은지를 확인 |
| sellerValidationTest(  seller\_id,  transaction\_id  ) | INT, INT | Boolean | 1. seller의 배터리 충전량이 판매 전력량보다 크거나 같은지 확인  2. seller의 초당 생산량이 초당 소모량보다 많은지 확인 |

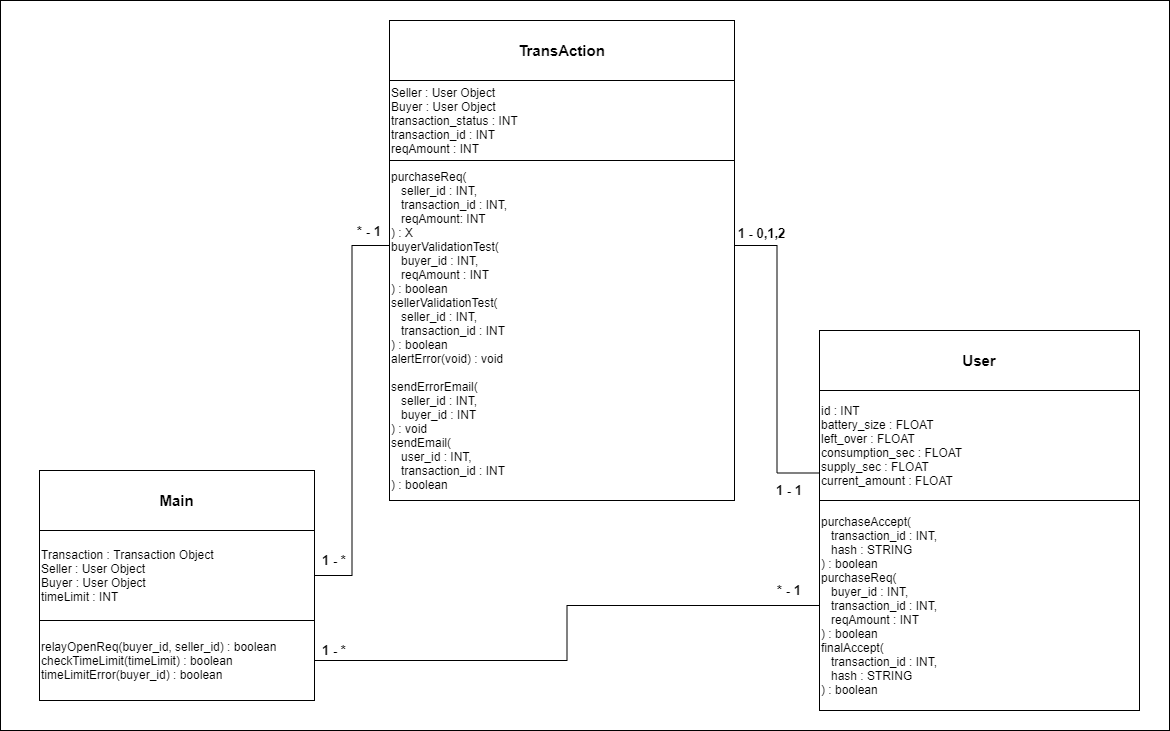
**3. User**

1) 개요

User 클래스에서는 구매를 원할 시, transActionReq 메서드를 실행하여 거래 요청을 보내고, transActionApprove 메서드를 통하여 거래 수락을 하고, transActionExecute 메서드를 통하여 최종 승인한다.

2) 연산

|  |  |  |  |
| --- | --- | --- | --- |
| **연산** | **파라미터** | **반환 값** | **설명** |
| purchaseReq(  buyer\_id,  transaction\_id,  reqAmount  ) | INT, INT, INT | Boolean | 거래 요청을 개시하는 함수, transaction 클래스 정보를 갱신 하고,  buyerValidation과 sendEmail을 호출한다. |
| purchaseAccept (  transaction\_id,  hash  ) | INT, STRING | Boolean | 거래 수락시 시행되는 함수, sellerValidation 을 시행하고, transaction\_staus를 갱신한다. |
| finalAccept (  transaction\_id,  hash  ) | INT, STRING | Boolean | 최종 승인 통보 시, 시행되는 함수, sendEmail, sellerValidation, buyerValidation을 호출하고, transaction\_status를 갱신한다. |

****

**필요 부품 목록**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 품명 | 규격 | 단가 | 개수 | 수량 | 참조사이트 |
| Raspberry-Pi 3 | model B | 51,150 | 1 | 51,150 | <http://item.gmarket.co.kr/DetailView/Item.asp?goodscode=911065717&GoodsSale=Y&jaehuid=200002657&service_id=pcd> |
| micro sd card | 16gb | 7,900 | 1 | 7,900 | https://www.devicemart.co.kr/goods/view?no=12237605 |
| USB 2.0 와이파이 동글 | 802.11n 150Mbps U | 16,350 | 1 | 16,350 | https://www.lotteon.com/p/product/LO1035194830?sitmNo=LO1035194830\_1035194831&ch\_no=100173&ch\_dtl\_no=1000548&dp\_infw\_cd=CHT |
| Arduino Uno (R3) | 5V | 21,000 | 1 | 21,000 | https://www.devicemart.co.kr/goods/view?no=34404 |
| Arduino USB 케이블 | 1.5m | 700 | 1 | 700 | https://www.devicemart.co.kr/goods/view?no=1330154 |
| Arduino 아답터 | 12V/1.5A | 5,900 | 1 | 5,900 | <https://www.devicemart.co.kr/goods/view?no=1330492> |
| 아두이노 무선 wifi 모듈 | 1MB | 3,500 | 1 | 3,500 | https://www.icbanq.com/P008165046?gclid=CjwKCAjwu5CDBhB9EiwA0w6sLQpMmywVzmcQJgDZHhtFmbv0jyG4OhCP1WcDJOB3Olk\_JQbaYeB0-xoChhEQAvD\_BwE |
| 태양 전지 판 | 5V 400mA | 6,500 | 2 | 13,000 | <https://www.devicemart.co.kr/goods/view?no=12372315> |
| 릴레이 1ch | 5V | 1,200 | 3 | 3,600 | <http://www.11st.co.kr/products/2516095658?trTypeCd=21&trCtgrNo=585021> |
| 릴레이 2ch | 5V | 1,430 | 1 | 1,430 | <http://www.11st.co.kr/products/2042999716?trTypeCd=21&trCtgrNo=585021> |
| ACS712 | 5A | 2,140 | 4 | 2,140 | <http://www.11st.co.kr/products/2044896518?utm_medium=%EA%B2%80%EC%83%89&gclid=CjwKCAjwu5CDBhB9EiwA0w6sLdFUNmB_kwCQl3BBxBnQIw_bCKEPKtgGHl_nImdQLKfV5NQyv6XsYRoCwG4QAvD_BwE&utm_source=%EA%B5%AC%EA%B8%80_PC_S_%EC%87%BC%ED%95%91&utm_campaign=%EA%B5%AC%EA%B8%80%EC%87%BC%ED%95%91PC+%EC%B6%94%EA%B0%80%EC%9E%91%EC%97%85&utm_term=> |
| T형 커넥터 | 1.5~2.5mm | 100 | 5 | 500 | <https://www.techshenzhen.com/goods/goods_view.php?goodsNo=1000000699> |
| LED 전구 | 3mm 5V-9V | 500 | 2 | 1000 | <http://itempage3.auction.co.kr/DetailView.aspx?itemno=B710081058&BCODE=BN00251428&ck=&sn=&igaw_eng=&airbridge_referrer=&appgate_guid=> |
| 인두기 납땜 세트 | ZD-70S | 17,180 | 1 | 17,180 | <https://front.wemakeprice.com/product/1719202844?utm_source=google_ss&utm_medium=cpc&utm_campaign=r_sa&gclid=CjwKCAjwu5CDBhB9EiwA0w6sLQr16LyiMY3Rkrc3uUYCXDYsYohY94wlzZibO4unEdK_ccg5sB2eUxoCiwMQAvD_BwE> |
| 점퍼케이블 | 10CM 암-숫 | 600 | 1 | 600 | <http://www.11st.co.kr/products/2421954336?trTypeCd=21&trCtgrNo=585021> |
| C-Type 커넥터 | USB 3.1 | 500 | 1 | 500 | <http://www.11st.co.kr/products/3299301424?utm_medium=%EA%B2%80%EC%83%89&utm_source=%EA%B5%AC%EA%B8%80_PC_S_%EC%87%BC%ED%95%91&utm_campaign=%EA%B5%AC%EA%B8%80%EC%87%BC%ED%95%91PC+%EC%B6%94%EA%B0%80%EC%9E%91%EC%97%85&utm_term=> |
| 5핀 커넥터 | 5핀 | 500 | 1 | 500 | <http://www.11st.co.kr/products/2815395131?utm_medium=%EA%B2%80%EC%83%89&utm_source=%EA%B5%AC%EA%B8%80_PC_S_%EC%87%BC%ED%95%91&utm_campaign=%EA%B5%AC%EA%B8%80%EC%87%BC%ED%95%91PC+%EC%B6%94%EA%B0%80%EC%9E%91%EC%97%85&utm_term=> |
| USB 커넥터 | 2A | 800 | 3 | 2,400 | <http://shopping.interpark.com/product/productInfo.do?prdNo=4581105238> |
| 합계 | | | | 164,370 |  |