Distributed vending machine

학번	이름
201511238	허준호
201911147	강숙영
201711320	오준엽
201511218	이지영
201710397	이지훈

Index

1. JUnit Test Case
2. JavaScript Unit Test Case
3. Run Windows
4. Demo Video

1. Start Order

Use case	1. Start Order
Actor	User
Purpose	자판기 사용을 시작한다.
Overview	자판기의 상품을 고르기 위해 카드를 투입한다.
Type	evident
Cross Reference	System Functions : 1.1, 1.2 Use Case : “Check Card Input”
Pre-Requisites	자판기의 전원이 켜져 있어야 한다. 전원이 켜져 있어야 한다.
Typical Courses of Events	(AU) : Actor User, (S) : System 1.(AU) 카드리더기에 카드를 투입한다. 2.(S) Window-2(상품 선택 화면)을 출력한다. 3.(S) “Input choice”로 넘어간다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	all line-a. (AU) 카드리더기에서 카드를 제거한다. all line-b. (S) 모든 작업을 중단한다 all line-c. (S) Window-24(카드 제거 안내 화면)을 출력한다 all line-d. (S) Window-1(대기 화면)을 출력한다.

1. Start Order


```java
@Test
public void getDrinkList() throws IOException {
    URL obj = null;

    Map<String, Object> params = new HashMap<String, Object>();
    params.put("INDEX", INDEX);

    StringBuilder postData = new StringBuilder();
    for(Map.Entry<String,Object> param : params.entrySet()) {
        if(postData.length() != 0) postData.append('&');
        postData.append(param.getKey());
        postData.append('=');
        postData.append(param.getValue());
    }
    byte[] postDataBytes = postData.toString().getBytes( charsetName: "UTF-8");

    obj = new URL( spec: "http://localhost:8080/MainProject/drink/getDrinkList");

    HttpURLConnection con = (HttpURLConnection)obj.openConnection();
    con.setRequestMethod("POST");
    con.setRequestProperty("Content-Length", String.valueOf(postDataBytes.length));
    con.setDoOutput(true);
    con.getOutputStream().write(postDataBytes);

    BufferedReader in = new BufferedReader(
            new InputStreamReader(con.getInputStream(), charsetName: "UTF-8"));
    String inputLine;
    StringBuffer response = new StringBuffer();
    while((inputLine = in.readLine()) != null){
        response.append(inputLine);
    }
    in.close();

    String result = response.toString();
}
```

2. Input Choice

Use case	2.Input Choice
Actor	User
Purpose	사용자가 상품을 선택한다.
Overview	사용자가 선택한 상품의 재고가 현재 자판기에 존재하는지 확인한다.
Type	evident
Cross Reference	System Functions : 2.1, 2.2, 3.2 Use Case : “Check chosen item Stock”, “Payment”
Pre-Requisites	자판기의 전원이 켜져 있어야 한다. 결제 가능한 카드가 투입되어 있어야 한다.
Typical Courses of Events	(AU) : Actor User, (S) : System 1.(AU) 사용자가 Window-2(상품 선택 화면)에서 원하는 상품의 버튼을 선택한다. 2.(S) 현재 자판기에서 재고를 확인한다. 3.(S) 재고가 존재한다면 “Payment”로 넘어간다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	3-a. (S) 현재 자판기에 상품이 존재하지 않는다. 3-b. (S) “Check chosen item Stock”으로 넘어간다. all line-a. (AU) 카드리더기에서 카드를 제거한다. all line-b. (S) 모든 작업을 중단한다. all line-c. (S) Window-24(카드 제거 안내 화면)을 출력한다. all line-d. (S) Window-1(대기화면)을 출력한다.

2. Input Choice


```java
@Test
public void buyDrink() throws IOException {
    URL obj = null;

    Map<String, Object> params = new HashMap<~>();
    params.put("INDEX", INDEX);
    params.put("D_SEQ", SEQ);

    StringBuilder postData = new StringBuilder();
    for(Map.Entry<String,Object> param : params.entrySet()) {
        if(postData.length() != 0) postData.append('&');
        postData.append(param.getKey());
        postData.append('=');
        postData.append(param.getValue());
    }
    byte[] postDataBytes = postData.toString().getBytes( charsetName: "UTF-8");

    obj = new URL( spec: "http://localhost:8080/MainProject/drink/buyDrink");

    HttpURLConnection con = (HttpURLConnection)obj.openConnection();
    con.setRequestMethod("POST");
    con.setRequestProperty("Content-Length", String.valueOf(postDataBytes.length));
    con.setDoOutput(true);
    con.getOutputStream().write(postDataBytes);

    BufferedReader in = new BufferedReader(
            new InputStreamReader(con.getInputStream(), StandardCharsets.UTF_8));
    String inputLine;
    StringBuffer response = new StringBuffer();
    while((inputLine = in.readLine()) != null){
        response.append(inputLine);
    }
    in.close();

    String result = response.toString();
}
```

3. Check Chosen Item Stock

Use case	3. Check Chosen Item Stock
Actor	System
Purpose	사용자가 선택한 상품의 재고가 다른 자판기에 존재하는지 확인한다.
Overview	다른 모든 자판기에 사용자가 선택한 상품의 재고가 있는지 확인을 요청하고 정보를 받는다.
Type	evident
Cross Reference	System Functions : 2.2, 2.3, 2.4 Use Case : “Answer Stock Info”, “Determine location”
Pre-Requisites	현재 자판기에 상품 재고가 존재하지 않는다.
Typical Courses of Events	(AU) : Actor User, (AD) : Actor other DVMs (S) : System 1.(S) AD의 location,상품의 quantity를 요청한다. 2. (AD) “Answer Stock Info”을 통해 위치, 상품 재고 정보를 받는다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	2-a. 재고가 없다 2-b. Window-23(모든 재고 소진 화면)을 출력한다. all line-a. (AU) 카드리더기에서 카드를 제거한다. all line-b. (S) 모든 작업을 중단한다. all line-c. (S) Window-24(카드 제거 안내 화면)을 출력한다. all line-d. (S) Window-1(대기화면)을 출력한다.

4. Answer Chosen Item Stock

Use case	4. Answer Chosen Item Stock
Actor	System
Purpose	자판기의 재고와 위치 정보를 알려준다.
Overview	전달받은 상품의 재고를 확인 후 해당 정보를 자판기 위치와 함께 요청한 자판기에 보내준다.
Type	hidden
Cross Reference	System Functions : //2.2,// 2.3 //Use Case : “Check chosen item Stock”//
Pre-Requisites	(AD)으로 부터 재고 확인 요청이 있어야한다.
Typical Courses of Events	(AU) : Actor User, (AD) : Actor other DVM, (S) : System 1.(S) “Check chosen item Stock ”을 통해 요청된 상품의 재고를 확인한다. 2.(S) 확인한 재고 정보를 자신의 위치정보와 함께 (AD)로 전달한다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	N/A

4. Answer Chosen Item Stock


```java
@Test
public void getDrinkInfoFromOtherDVM() throws IOException {
    URL obj = null;

    Map<String, Object> params = new HashMap<~>();
    params.put("INDEX", INDEX);
    params.put("D_NAME", D_NAME);

    StringBuilder postData = new StringBuilder();
    for(Map.Entry<String,Object> param : params.entrySet()) {
        if(postData.length() != 0) postData.append('&');
        postData.append(param.getKey());
        postData.append('=');
        postData.append(param.getValue());
    }
    byte[] postDataBytes = postData.toString().getBytes( charsetName: "UTF-8");

    obj = new URL( spec: "http://localhost:8080/MainProject/drink/getDrinkInfoFromOtherDVM");

    HttpURLConnection con = (HttpURLConnection)obj.openConnection();
    con.setRequestMethod("POST");
    con.setRequestProperty("Content-Length", String.valueOf(postDataBytes.length));
    con.setDoOutput(true);
    con.getOutputStream().write(postDataBytes);

    BufferedReader in = new BufferedReader(
            new InputStreamReader(con.getInputStream(), StandardCharsets.UTF_8));
    String inputLine;
    StringBuffer response = new StringBuffer();
    while((inputLine = in.readLine()) != null){
        response.append(inputLine);
    }
    in.close();

    String result = response.toString();
}
```

Other Java Project


```java
@SuppressWarnings("unchecked")
public class DrinkDaoImpl extends JdbcDaoSupport implements DrinkDao{
    public List<Map<String, Object>> getDrinkList(int index) {
        final String query = "select u.SEQ, u.NAME, z.PRICE, z.STOCK from drink_info as u join drink_stoc
        return (List<Map<String, Object>>) getJdbcTemplate().query(query, new PSSetDao.PSSForInt(index),
    }
    public int searchDrink(String dName) {
        final String query = "select SEQ from drink_info where REPLACE(NAME, ' ', '')=?";
        return (Integer) getJdbcTemplate().query(query, new PSSetDao.PSSForString(dName), new ResultDao.R
    }
    public int addNewDrink(String dName) {
        final String query = "INSERT INTO drink_info(NAME) VALUES(?)";
        return (Integer) getJdbcTemplate().update(query, new PSSetDao.PSSForString(dName));
    }
    public int addNewDrinkM(int index, int dPrice) {
        final String query = "INSERT INTO drink_stock(DVM_SEQ, DRINK_SEQ, PRICE, STOCK) VALUES(1, ?, ?, 0
        return (Integer) getJdbcTemplate().update(query, new PSSetDao.PSSForDoubleInt(index, dPrice));
    }
    public int changeDrinkM(int index, int dPrice, int stock) {
        final String query = "INSERT INTO drink_stock(DVM_SEQ, DRINK_SEQ, PRICE, STOCK) VALUES(1, ?, ?, ?
        return (Integer) getJdbcTemplate().update(query, new PSSetDao.PSSForFiveInt(index, dPrice, stock,
    }
    public int checkDrink(int seq, int index) {
        final String query = "select STOCK from drink_stock where DVM_SEQ=? and DRINK_SEQ=?";
        return (Integer) getJdbcTemplate().query(query, new PSSetDao.PSSForDoubleInt(index, seq), new Res
    }
    public Map<String, Object> checkDrinkOtherDVM(String name, int index) {
        final String query = "select u.DVM_SEQ, u.STOCK, z.LONGITUDE, z.LATITUDE from drink_stock as u jo
        return (Map<String, Object>) getJdbcTemplate().query(query, new PSSetDao.PSSForTripleInt(index, i
    }
    public int buyDrink(int seq, int index) {
        final String query = "update drink_stock set STOCK=STOCK-1 where DVM_SEQ=? and DRINK_SEQ=?";
        return (Integer) getJdbcTemplate().update(query, new PSSetDao.PSSForDoubleInt(index, seq));
    }
    public boolean checkPreCode(String code, int index){
        final String query = "select SEQ from pre_code as a where a.TARGET_DVM=? and a.CODE=? and a.STATE
        return (Integer) getJdbcTemplate().update(query, new PSSetDao.PSSForIntString(index, code)) > 0;
    }

    public String getName(int seq){
        final String query = "select NAME from drink_info where SEQ=?";
        return (String) getJdbcTemplate().query(query, new PSSetDao.PSSForInt(seq), new ResultDao.RSEForS
    }
}
```

1. Start Order

JUnit Test Case

JavaScript Unit Test Case

Run Windows

Demo Video

Use case	1. Start Order
Actor	User
Purpose	자판기 사용을 시작한다.
Overview	자판기의 상품을 고르기 위해 카드를 투입한다.
Type	evident
Cross Reference	System Functions : 1.1, 1.2 Use Case : “Check Card Input”
Pre-Requisites	자판기의 전원이 켜져 있어야 한다. 전원이 켜져 있어야 한다.
Typical Courses of Events	(AU) : Actor User, (S) : System 1.(AU) 카드리더기에 카드를 투입한다. 2.(S) Window-2(상품 선택 화면)을 출력한다. 3.(S) “Input choice”로 넘어간다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	all line-a. (AU) 카드리더기에서 카드를 제거한다. all line-b. (S) 모든 작업을 중단한다 all line-c. (S) Window-24(카드 제거 안내 화면)을 출력한다 all line-d. (S) Window-1(대기 화면)을 출력한다.

1. Start Order

*** Card API Add**

```
9   window.onkeydown = function (e : KeyboardEvent ) {
10      if (e.code == 'KeyC') location.href='../window2?INDEX='+ls;
11  }
```

11. Check Precode

Use case	11. Check Precode
Actor	System
Purpose	입력된 선결제 코드와 생성된 선결제 코드의 일치 여부를 확인한다.
Overview	입력된 선결제 코드를 다른 모든 DVM에게 보낸 뒤 일치한다는 답변이 오면 상품을 배출한다.
Type	evident
Cross Reference	System Functions : 3.5, 3.6, 3.7, 3.8 Use Case : “Serve Item”, “Answer Precode Info”, “Input Precode”
Pre-Requisites	자판기의 전원이 켜져 있어야 한다.
Typical Courses of Events	(AU) : Actor User, (AD) : Actor other DVMs (S) : System 1.//(S) “Input Precode”에서 입력 받은 선결제 코드를 (AD)에게 보낸다. 2.(AD) “Answer Precode Info”를 통해 선결제 코드 일치 여부를 전달한다. 3.(S) 선결제 코드가 일치한다.// 4.(S) 코드 생성시 안내된 DVM이 현재 DVM이다. 5.(S) “Serve Item”을 진행한다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	3-a. (S) 선결제 코드가 불일치한다. 3-b. (S) Window-19(잘못된 입력 화면)을 출력한다. 4-a. (S) 현재 DVM과 선결제시 안내된 DVM이 아니다. 4-b. (S) Window-22(잘못된 DVM 화면)을 출력한다.

11. Check Precode

*** DB check Add**

```
function checkNextPage() {
    let code = document.getElementsByClassName( classNames: 'input_code')[0].value;
    let randS = [8, 19, 22];
    let randomN = Math.floor( x: Math.random()*3);
    console.log(randS[randomN]);
    switch (randS[randomN]){
        case 8:
            let str = "../window"+randS[randomN]+"?PageNum=3&Number=" + code ;
            location.href=str;
            break;
        case 19:
            alert("잘못된 입력입니다.");
            break;
        case 22:
            alert("이 기기에서는 해당 선결제 코드를 사용할 수 없습니다.");
            break;
    }
}
```

3. Check Chosen Item Stock

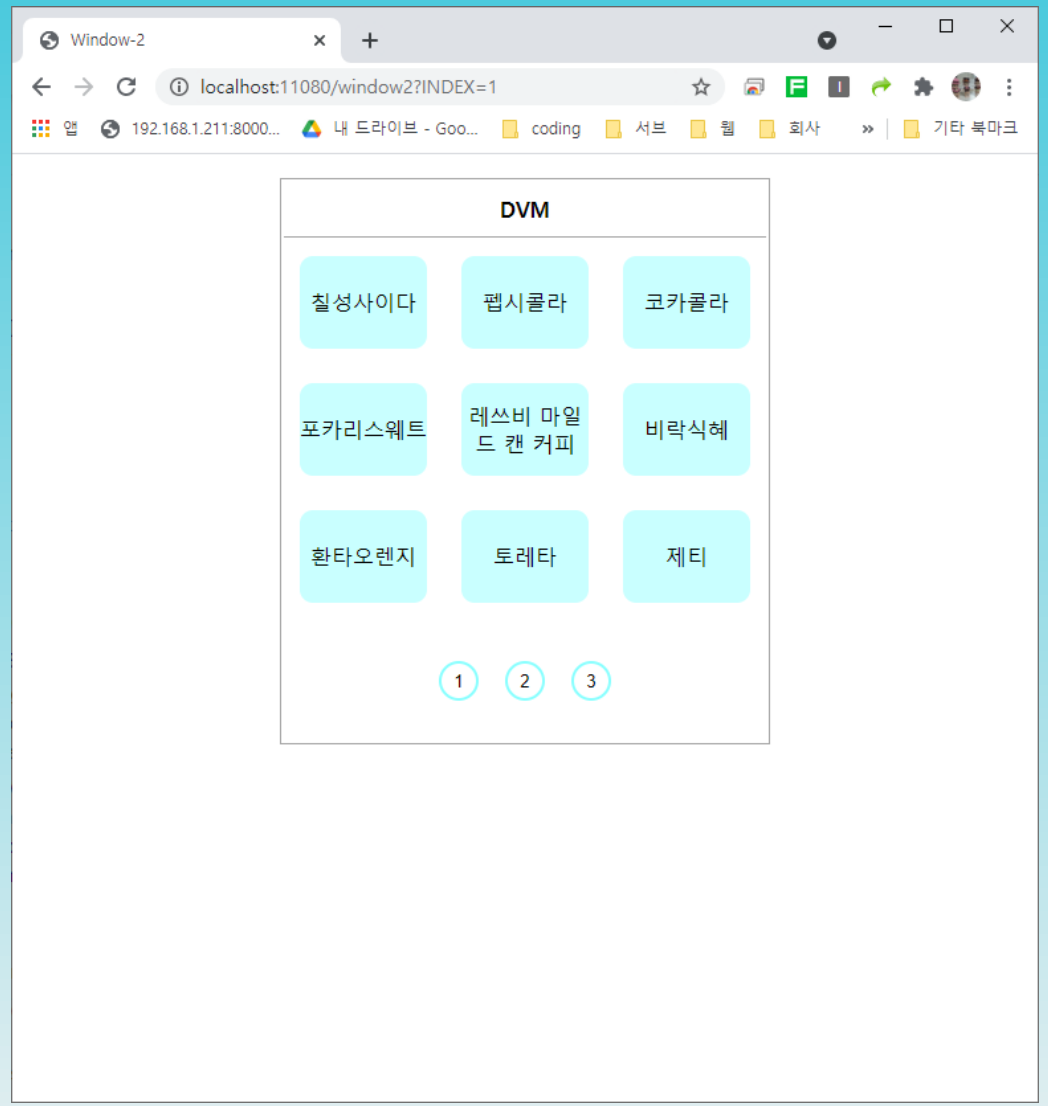
Use case	3. Check Chosen Item Stock
Actor	System
Purpose	사용자가 선택한 상품의 재고가 다른 자판기에 존재하는지 확인한다.
Overview	다른 모든 자판기에 사용자가 선택한 상품의 재고가 있는지 확인을 요청하고 정보를 받는다.
Type	evident
Cross Reference	System Functions : 2.2, 2.3, 2.4 Use Case : “Answer Stock Info”, “Determine location”
Pre-Requisites	현재 자판기에 상품 재고가 존재하지 않는다.
Typical Courses of Events	(AU) : Actor User, (AD) : Actor other DVMs (S) : System 1.(S) AD의 location,상품의 quantity를 요청한다. 2. (AD) “Answer Stock Info”을 통해 위치, 상품 재고 정보를 받는다.
Alternative Courses of Events	N/A
Exceptional Courses of Events	2-a. 재고가 없다 2-b. Window-23(모든 재고 소진 화면)을 출력한다. all line-a. (AU) 카드리더기에서 카드를 제거한다. all line-b. (S) 모든 작업을 중단한다. all line-c. (S) Window-24(카드 제거 안내 화면)을 출력한다. all line-d. (S) Window-1(대기화면)을 출력한다.

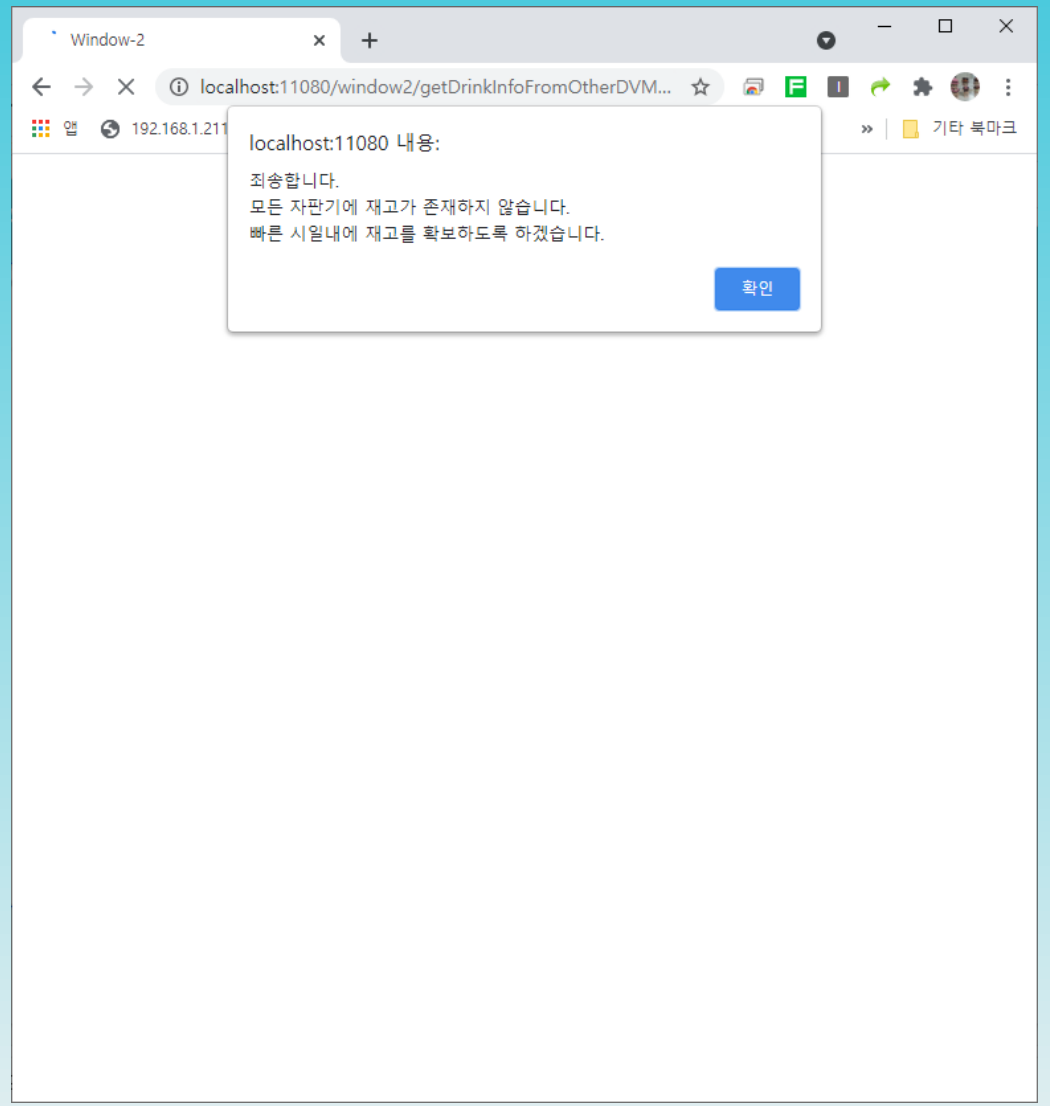
3. Check Chosen Item Stock

* 함수화

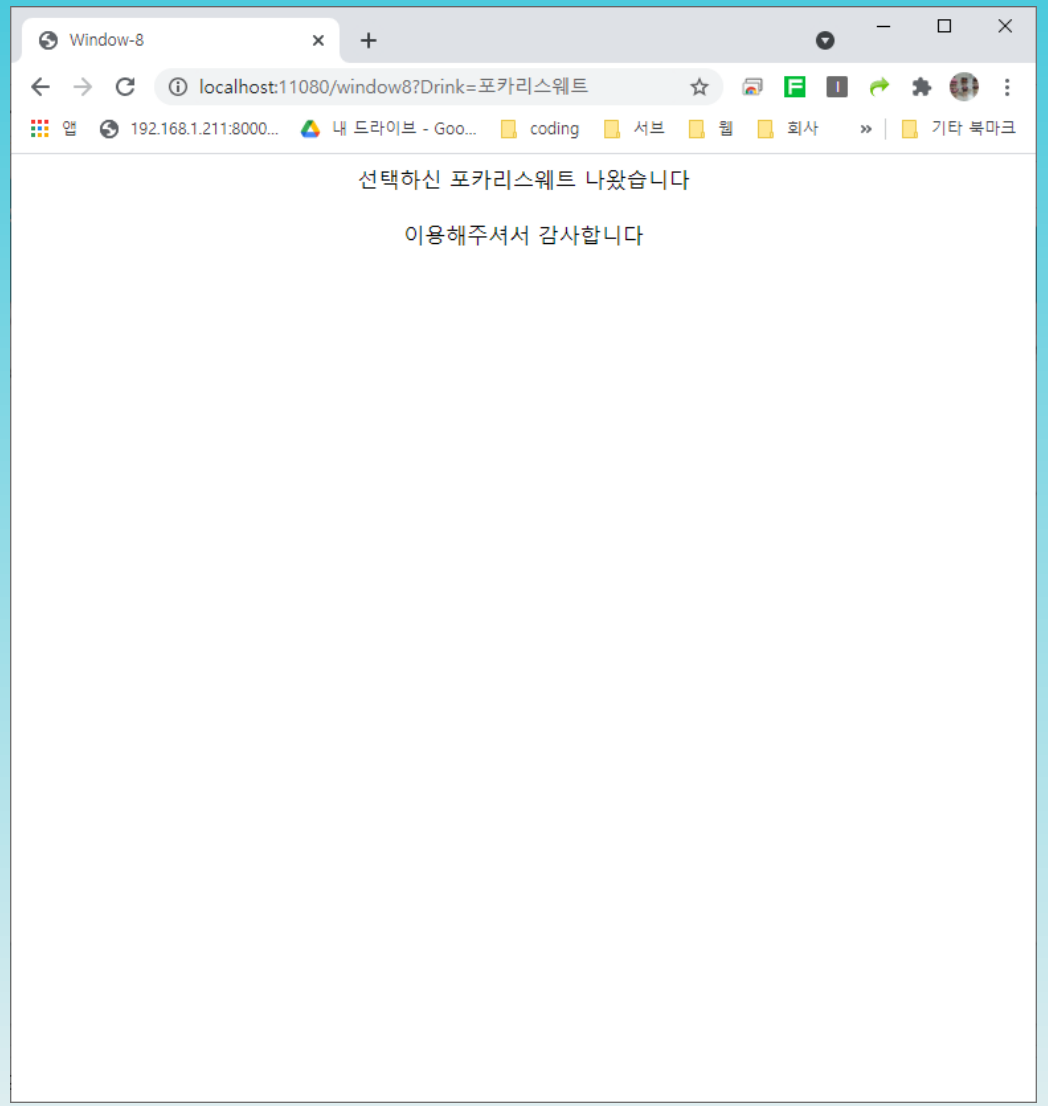
```
87      var Other = document.getElementById( elementId: "Other");
88      try {
89          Other = JSON.parse(Buy.innerHTML).R;
90          location.href = '../window4?LONGITUDE='+Other.LONGITUDE+'&LATITUDE='+Other.LATITUDE;
91      }catch (e) {
92          if (Buy.innerHTML.indexOf( searchElement: "<")>=0)
93              alert("죄송합니다.\n모든 자판기에 재고가 존재하지 않습니다.\n빠른 시일내에 재고를 확보하도록 하겠습니다.");
94      }
```

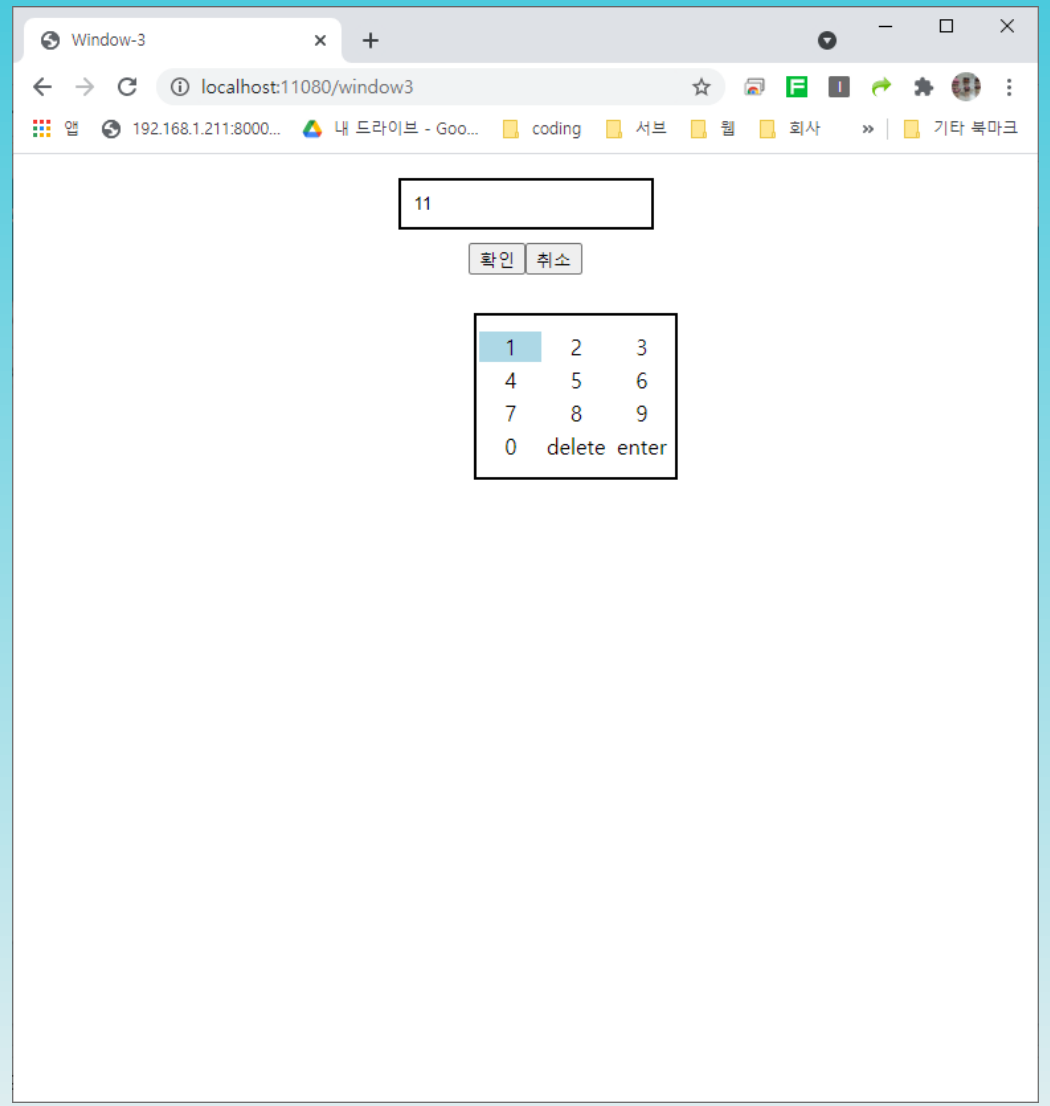
Run Windows



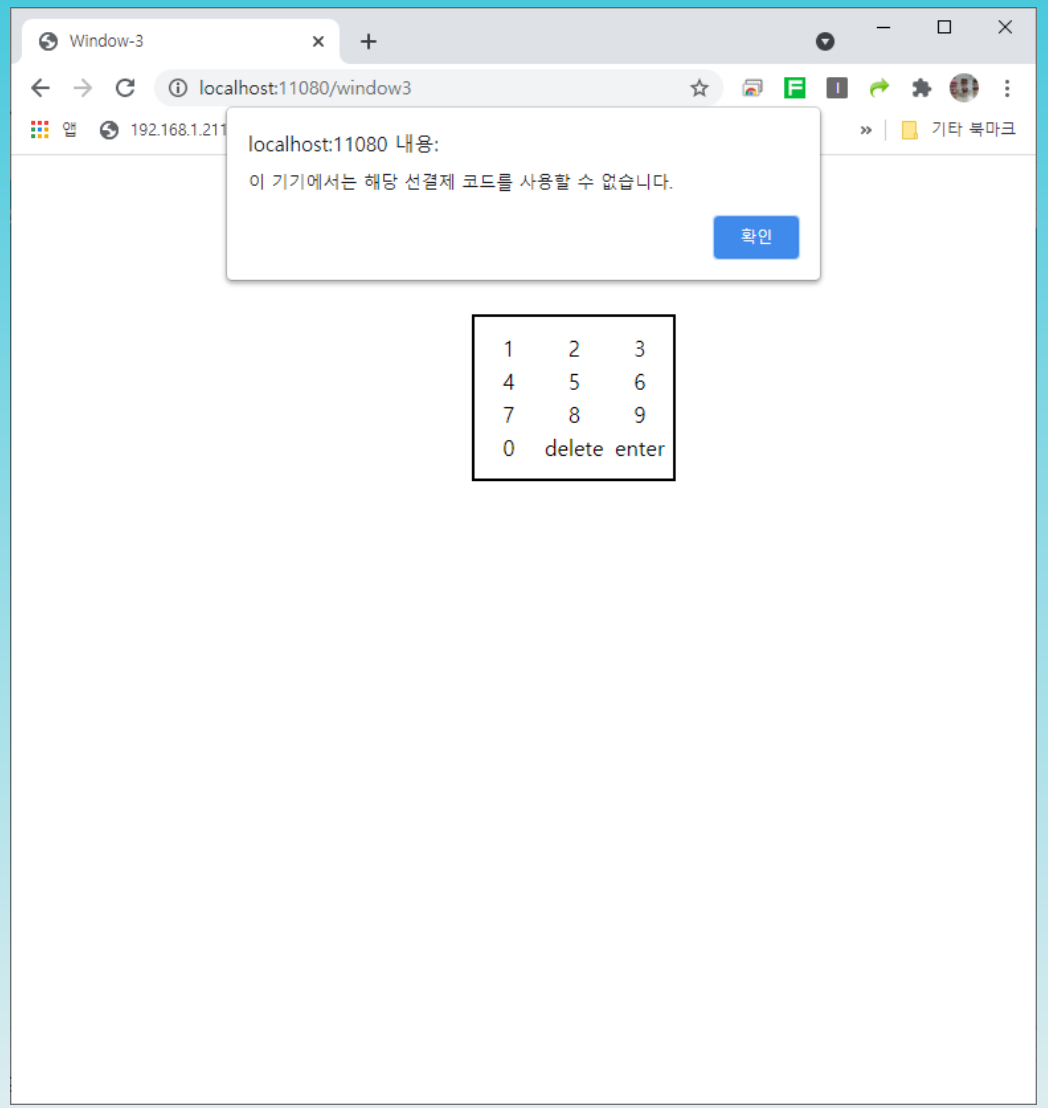


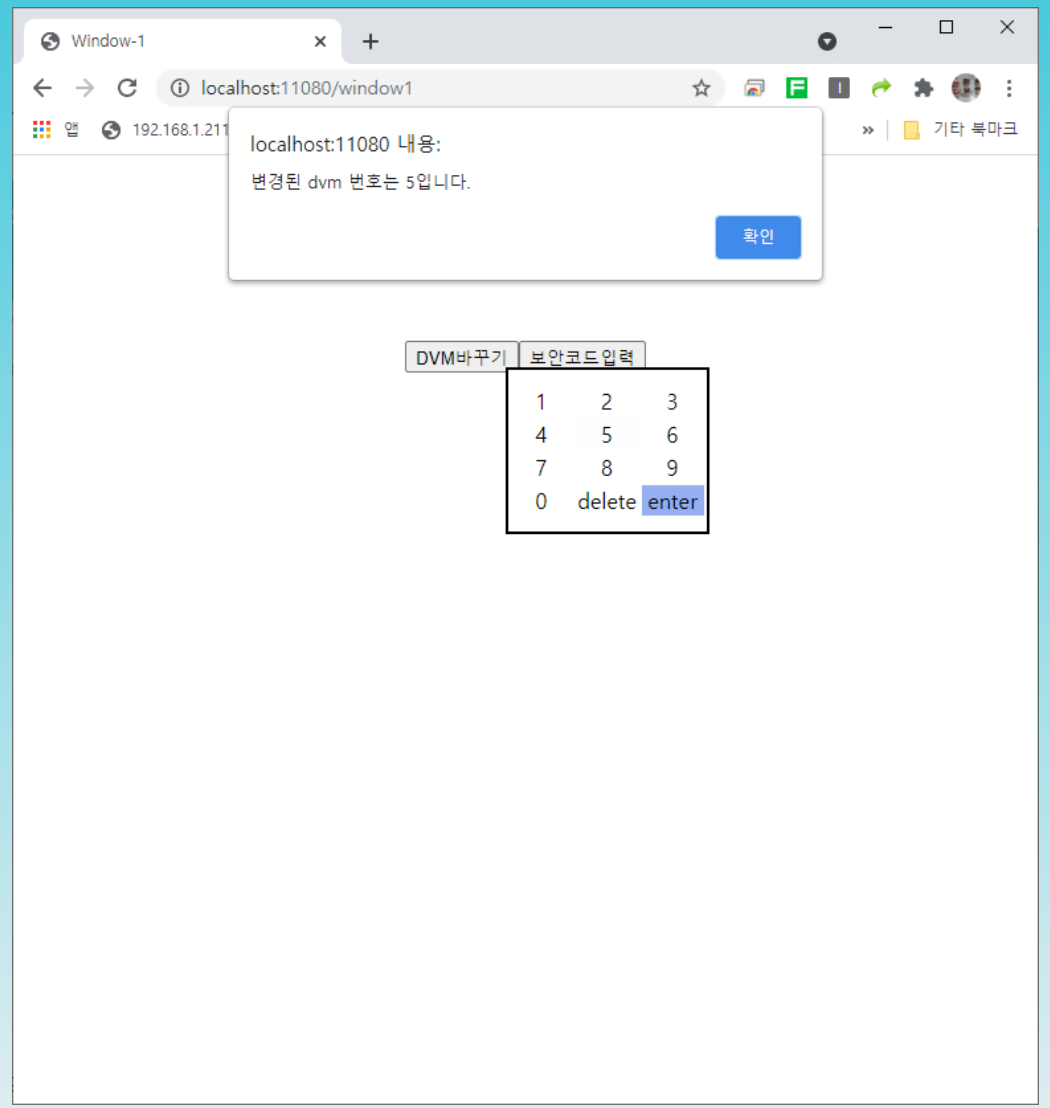
Run Windows



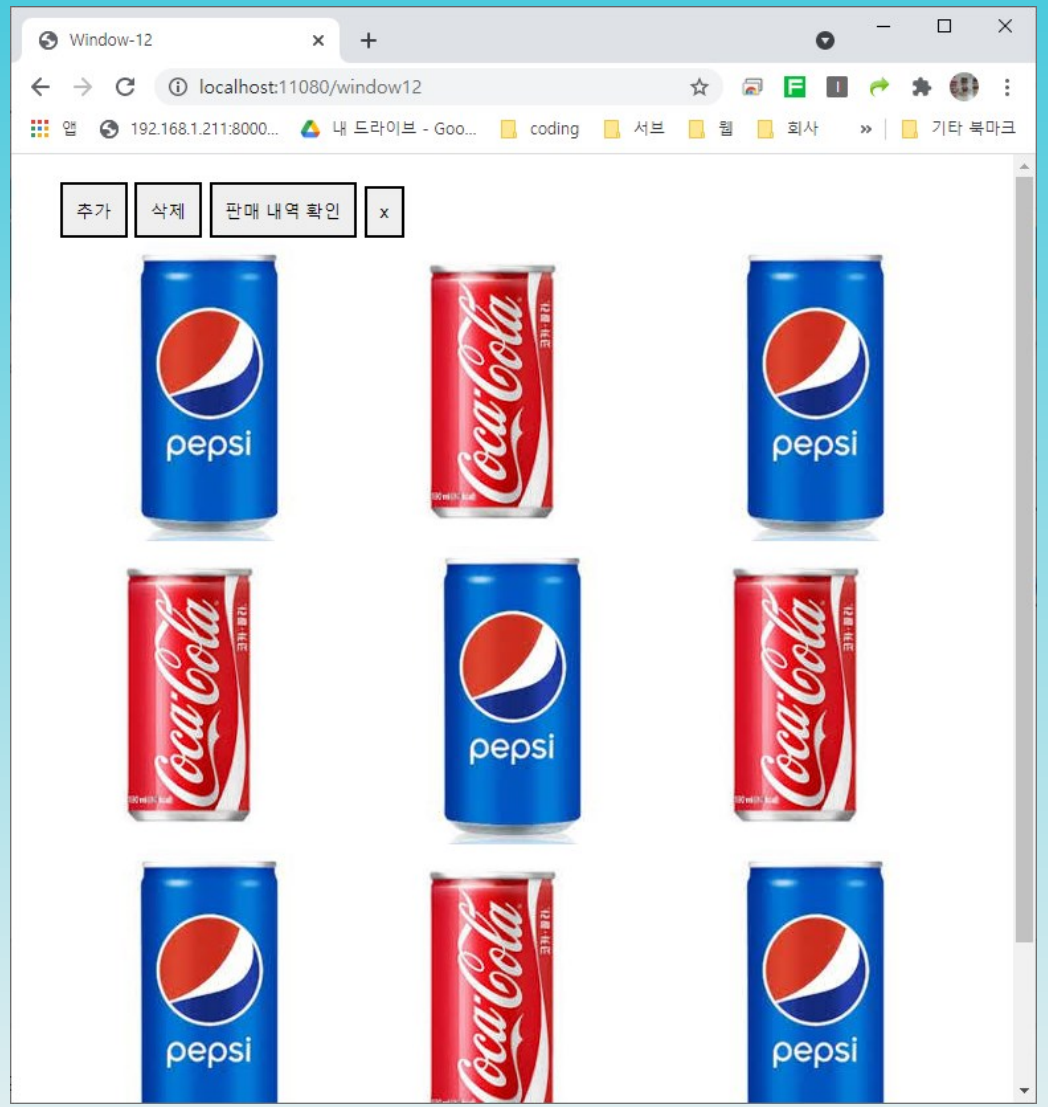


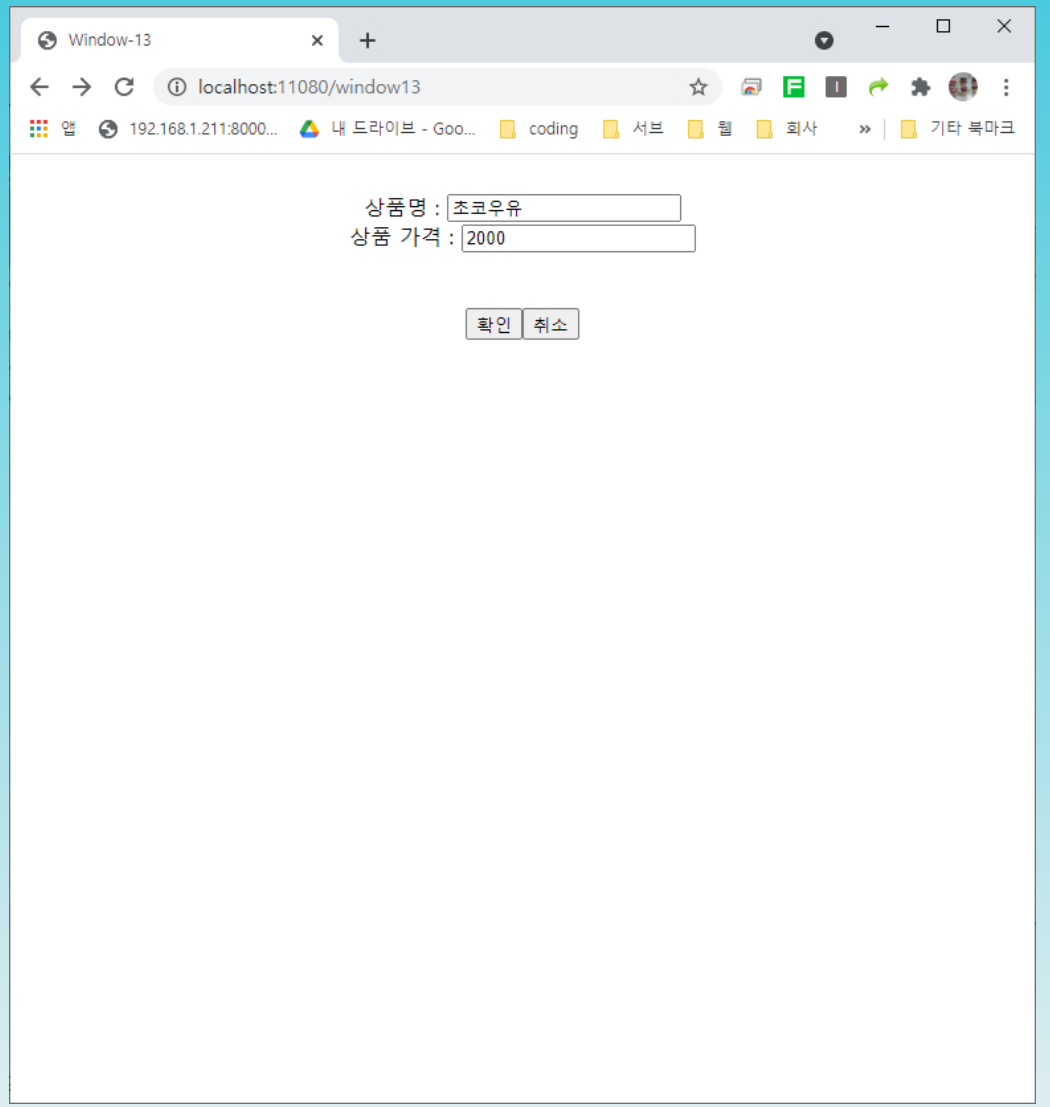
Run Windows



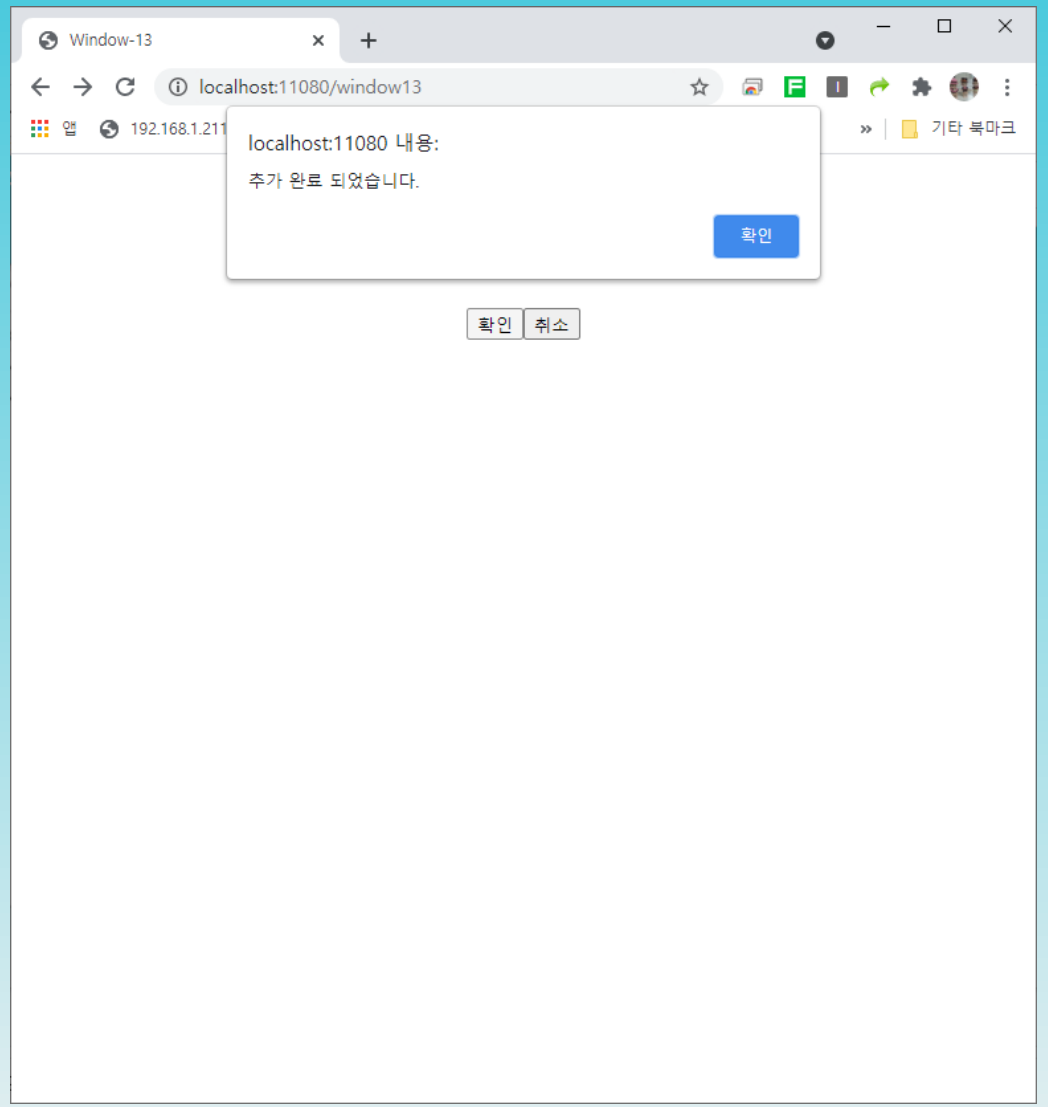


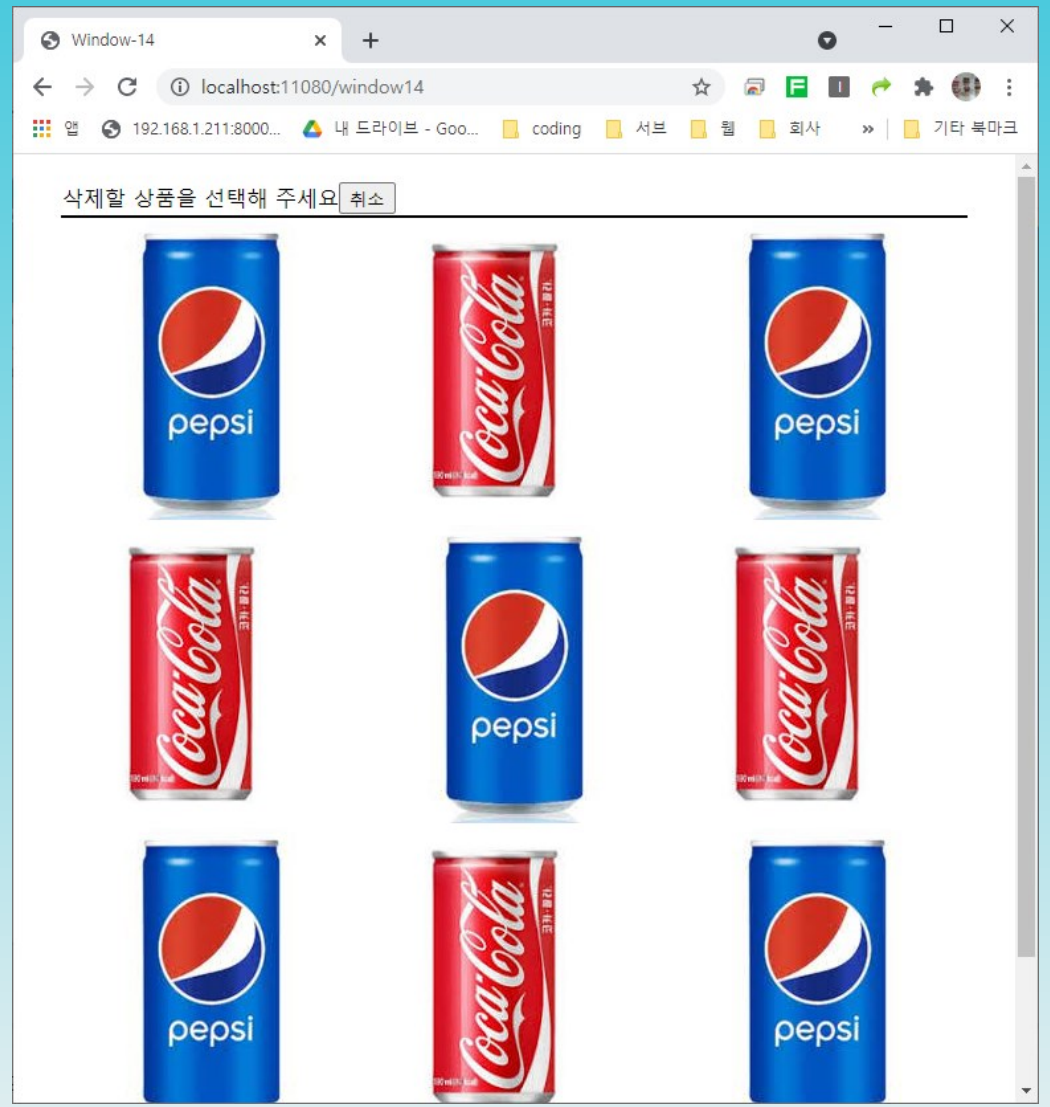
Run Windows



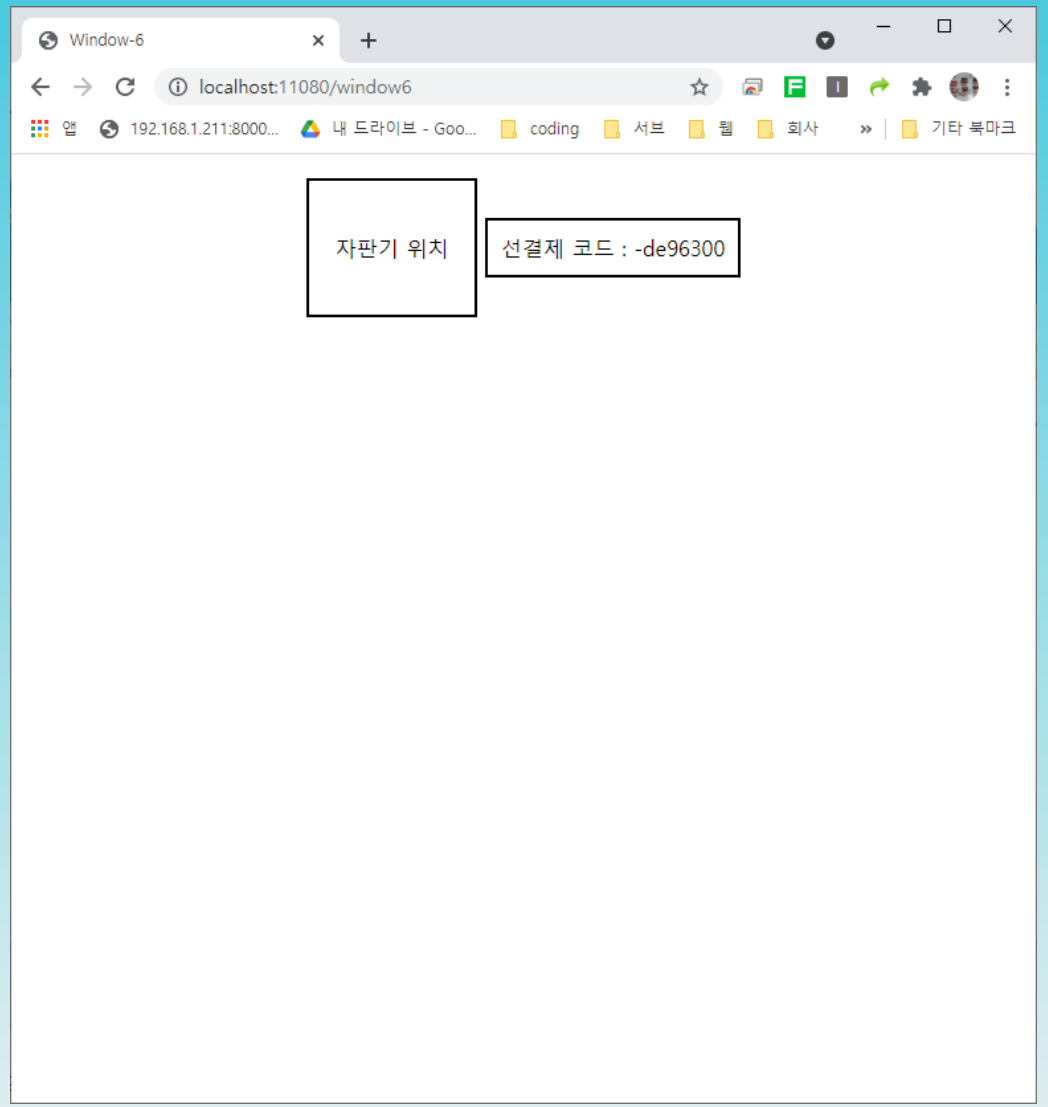


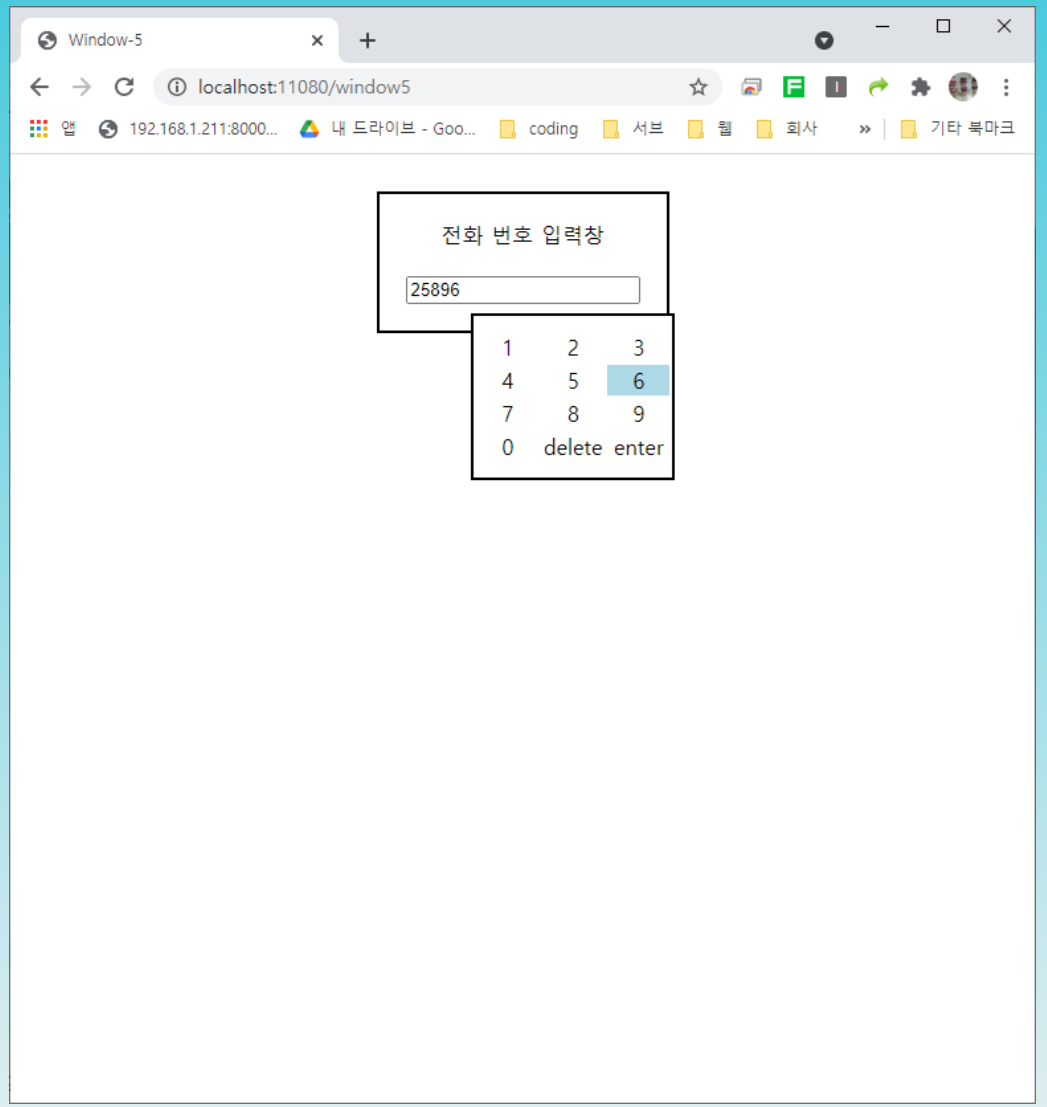
Run Windows



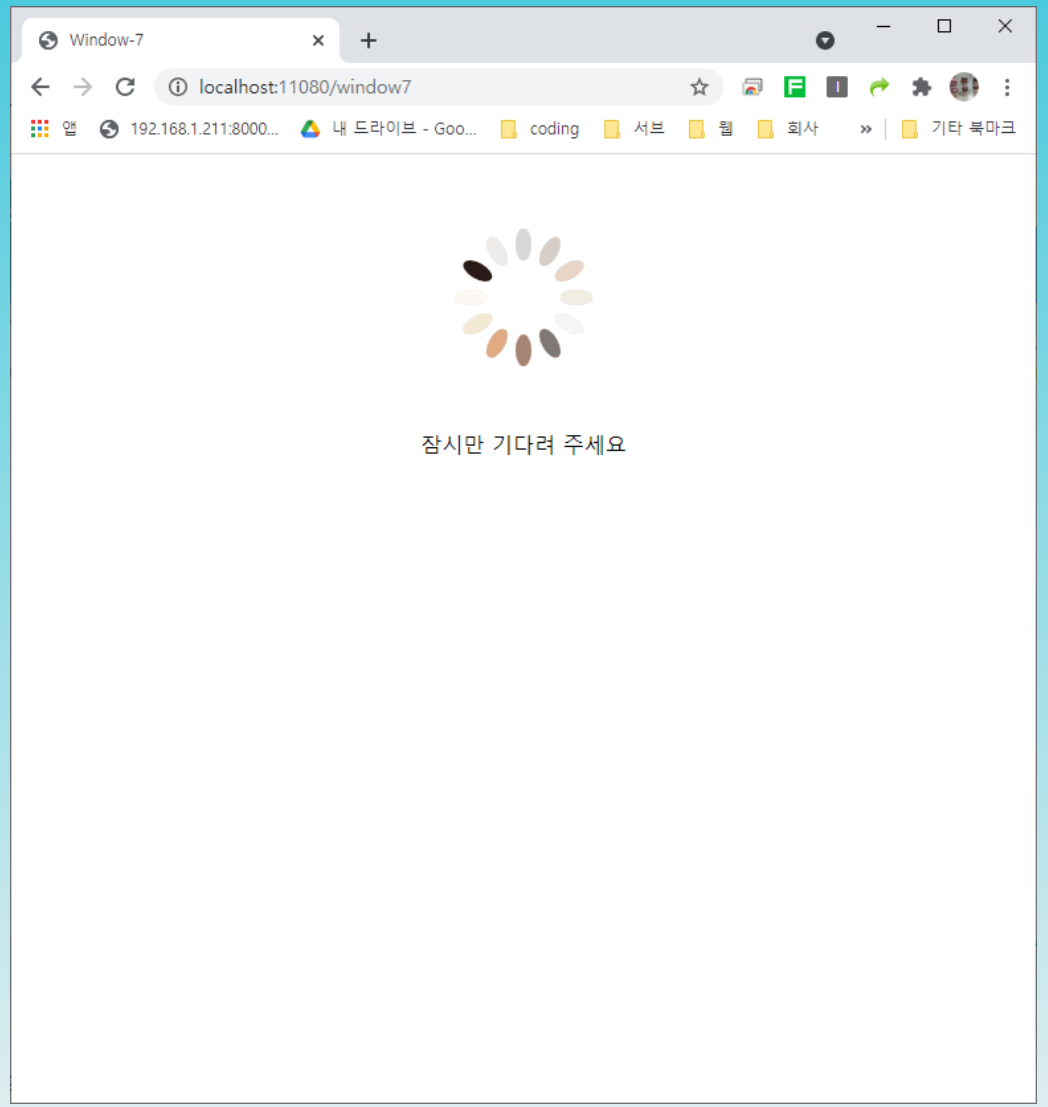


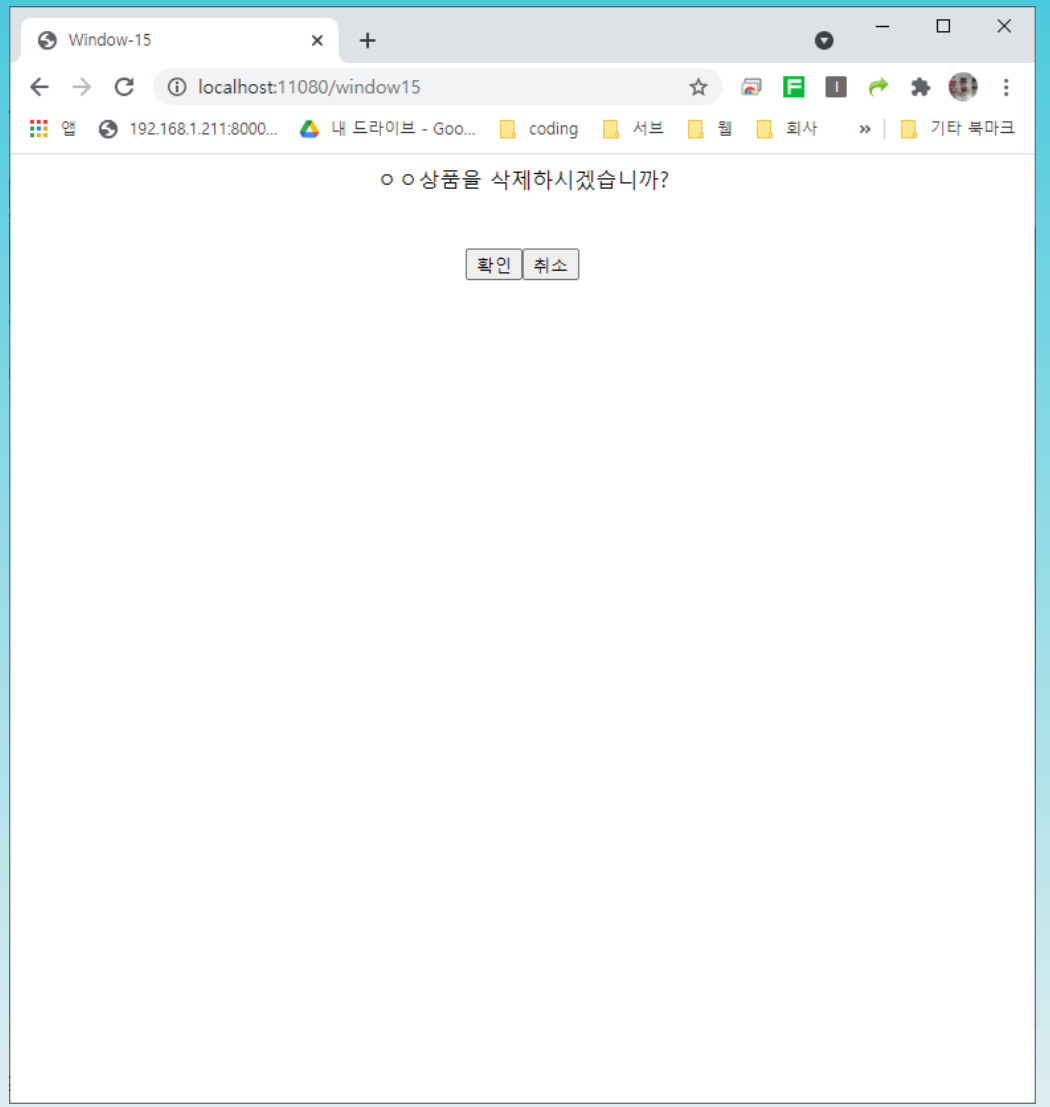
Run Windows





Run Windows





Demo Video

https://youtu.be/SgjkygEiQ-Q

감사합니다