Distributed Vending Machine

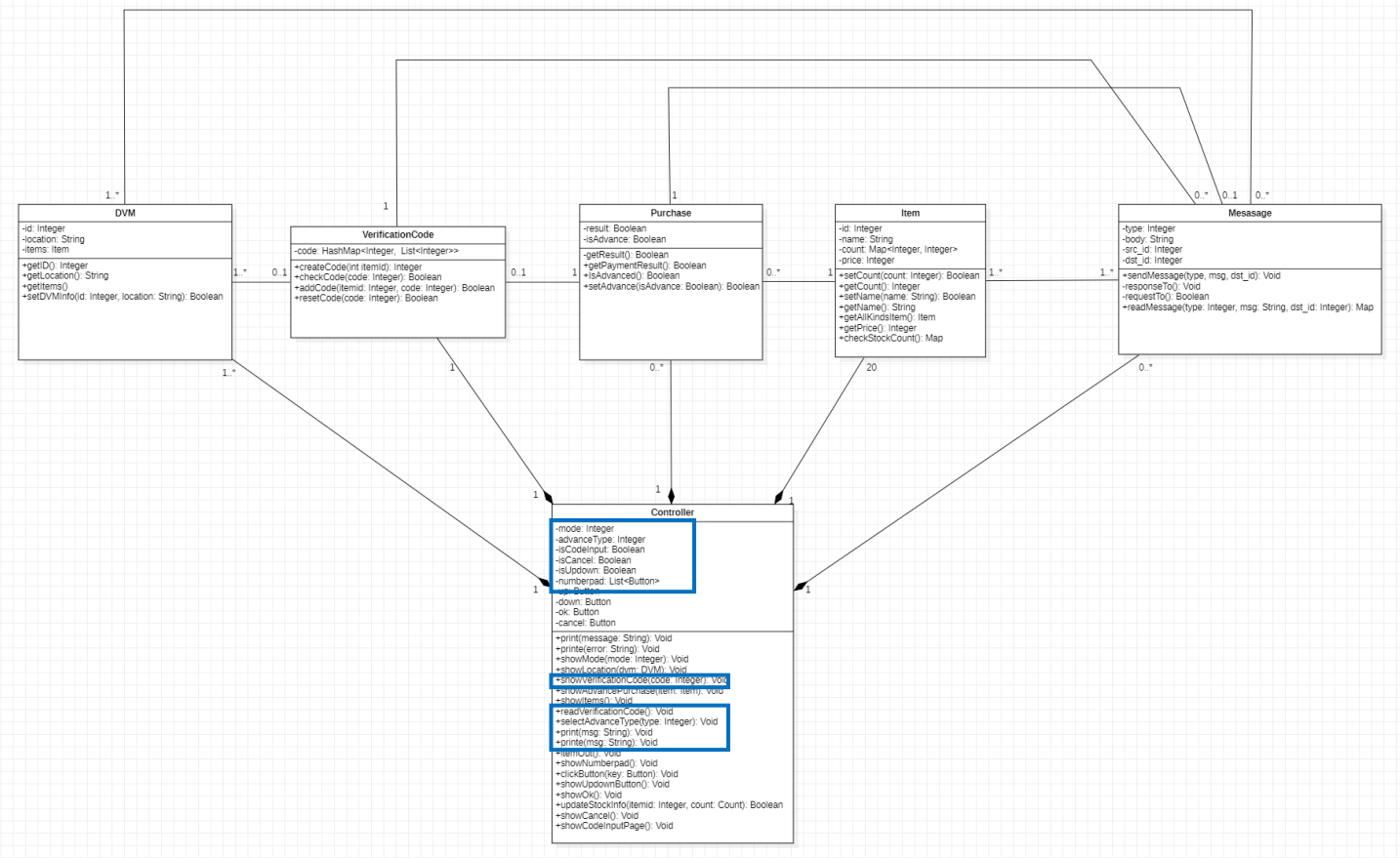
OOPT : OOI (The 1st Cycle)

TEAM 1
팀장 : 정연수
팀원 : 김민환 이승헌 조벽정 황유란

INDEX

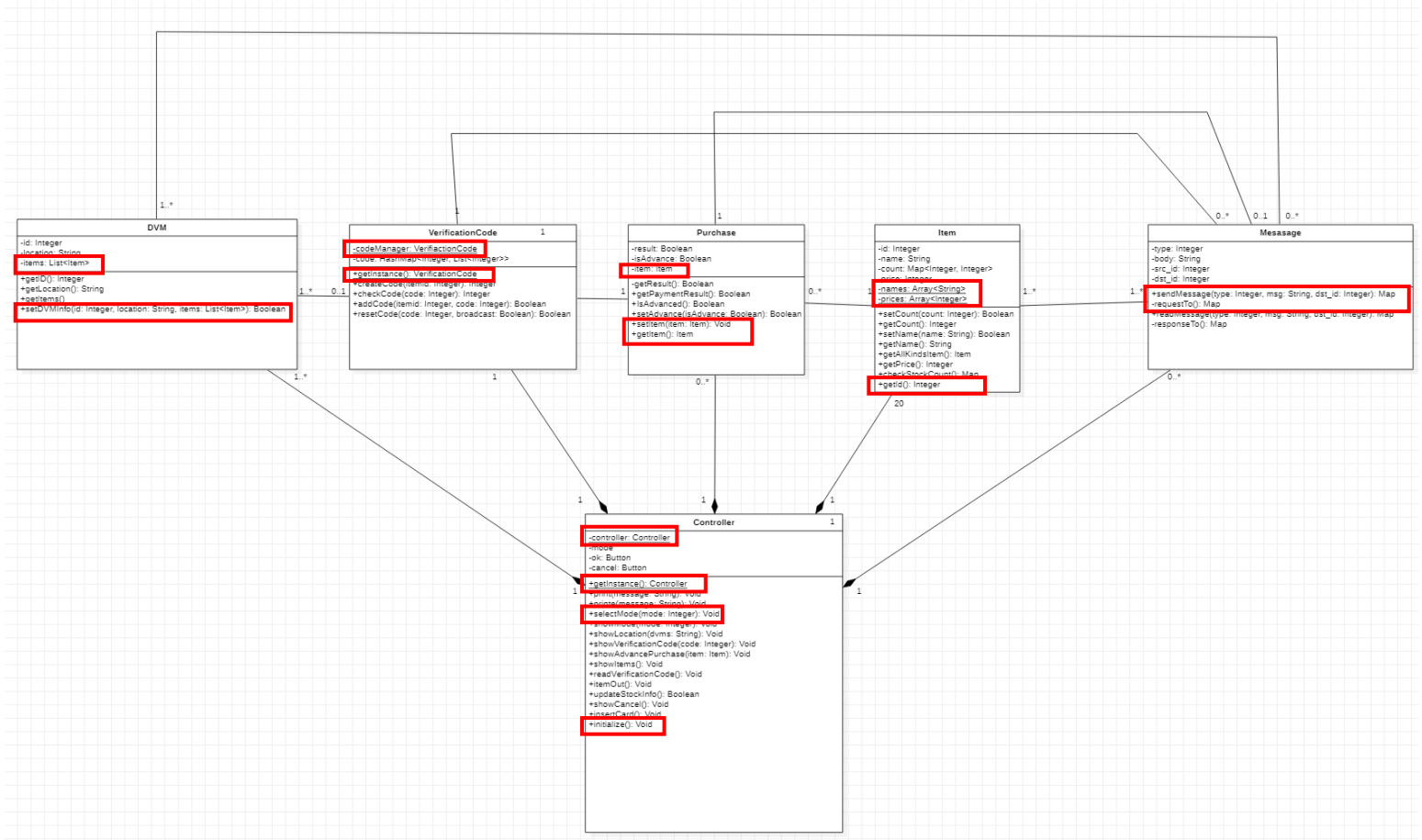
2051 Class Diagram & Sequence Diagram

수정 전



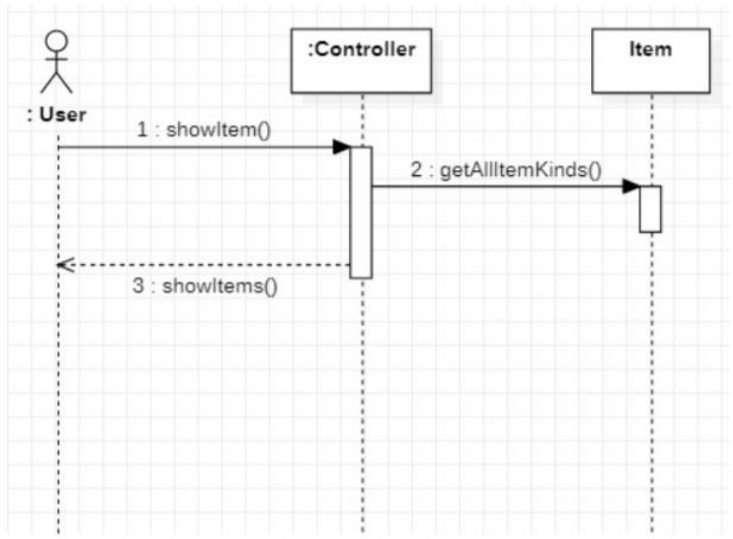
2051 Class Diagram & Sequence Diagram

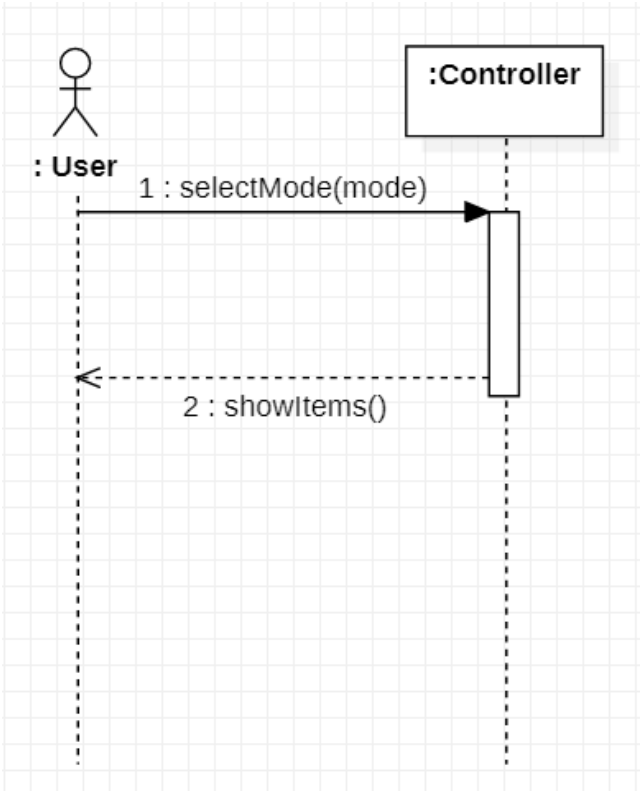
수정 후



2051 Class Diagram & Sequence Diagram

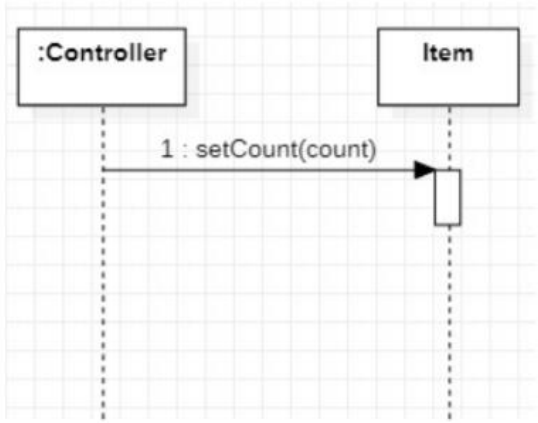
(4) Show Item

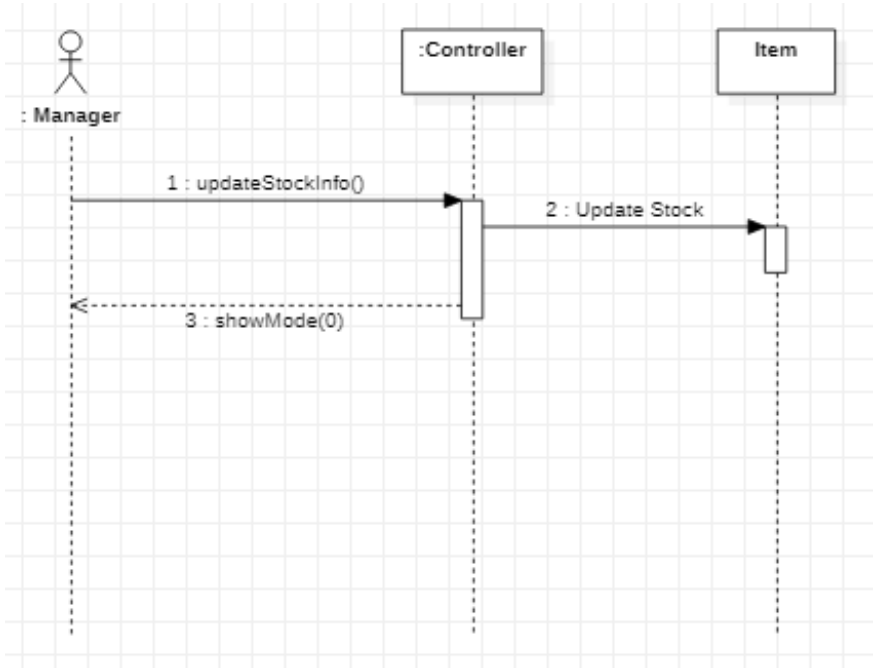




2051 Class Diagram & Sequence Diagram

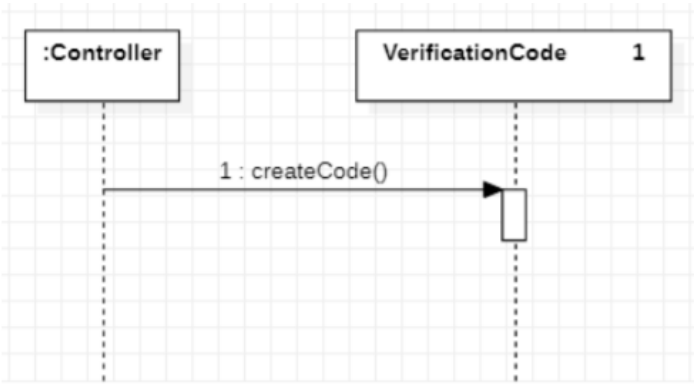
(7) Update Stock

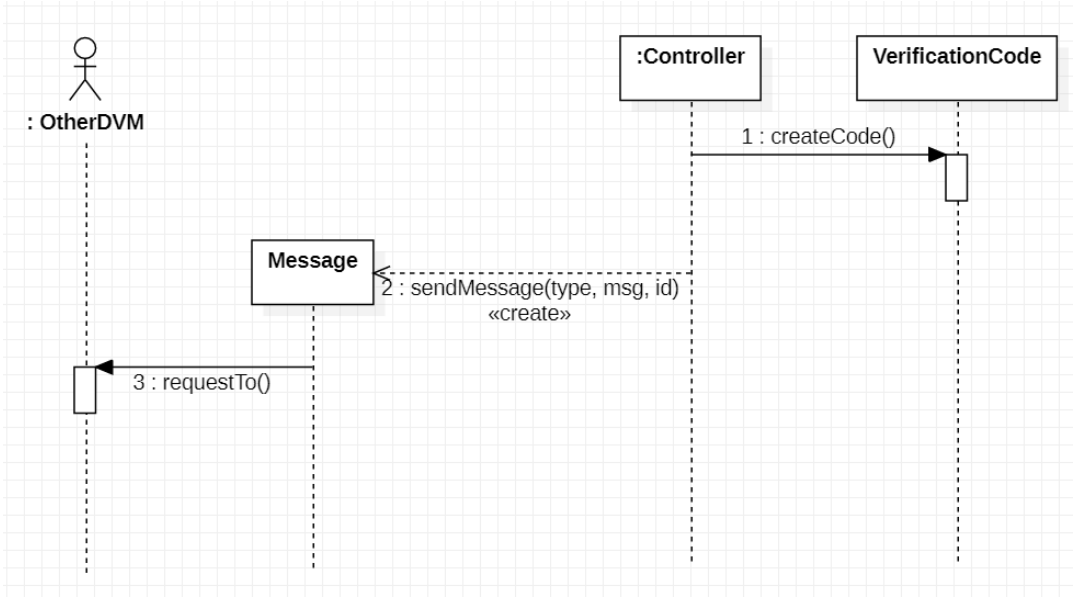




2051 Class Diagram & Sequence Diagram

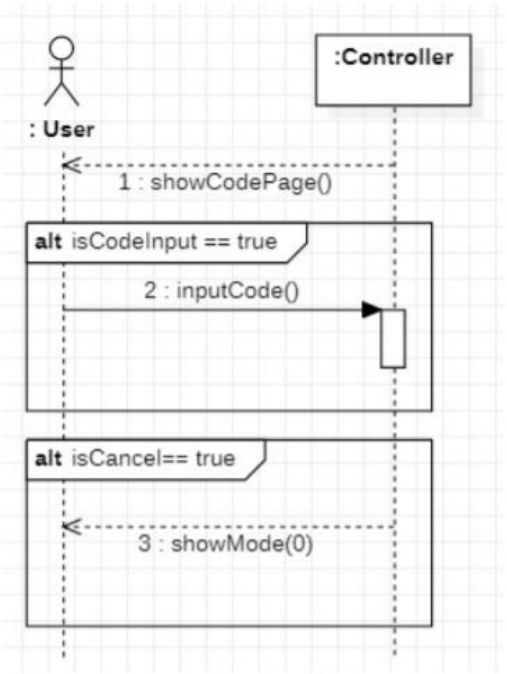
(13) Create Verification Code

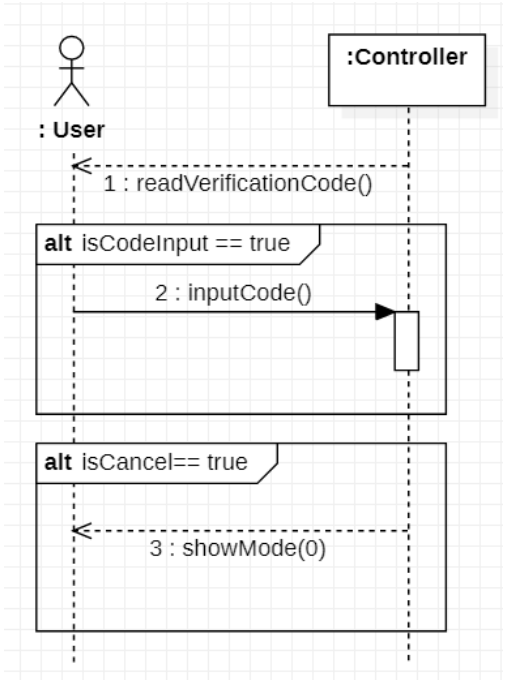




2051 Class Diagram & Sequence Diagram

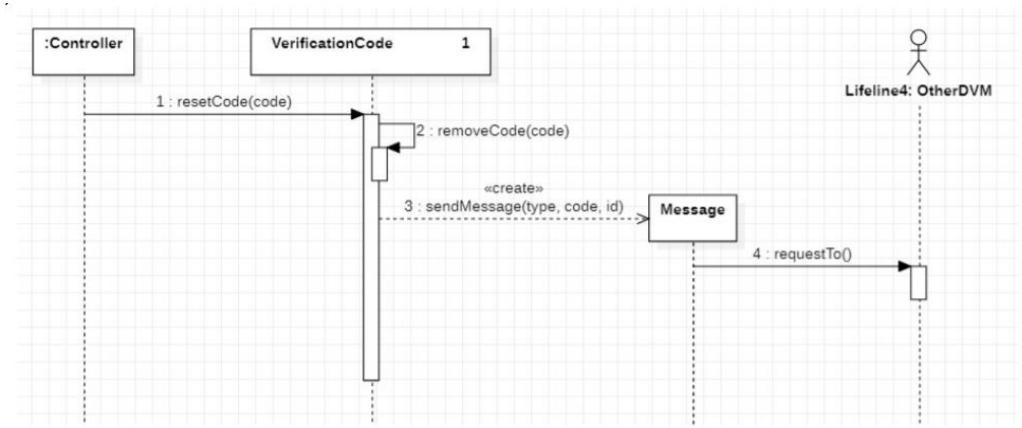
(16) Read Verification Code

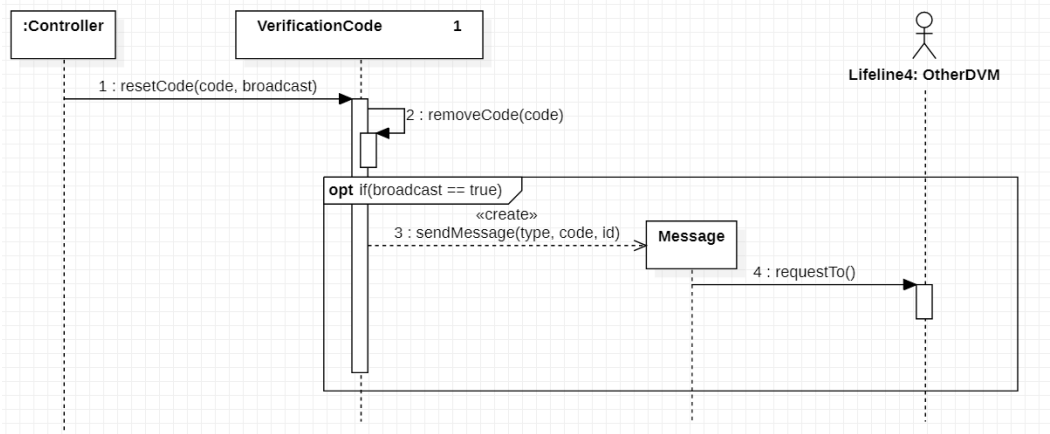




2051 Class Diagram & Sequence Diagram

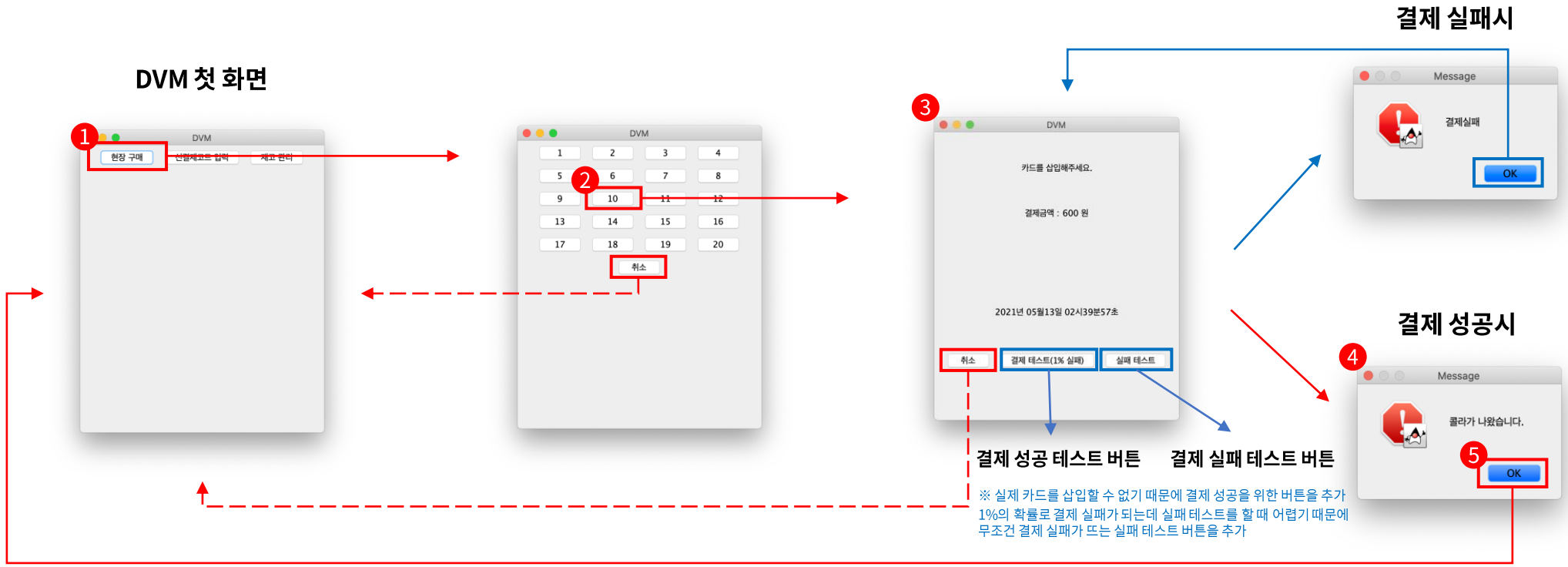
(18) Reset Verification Code





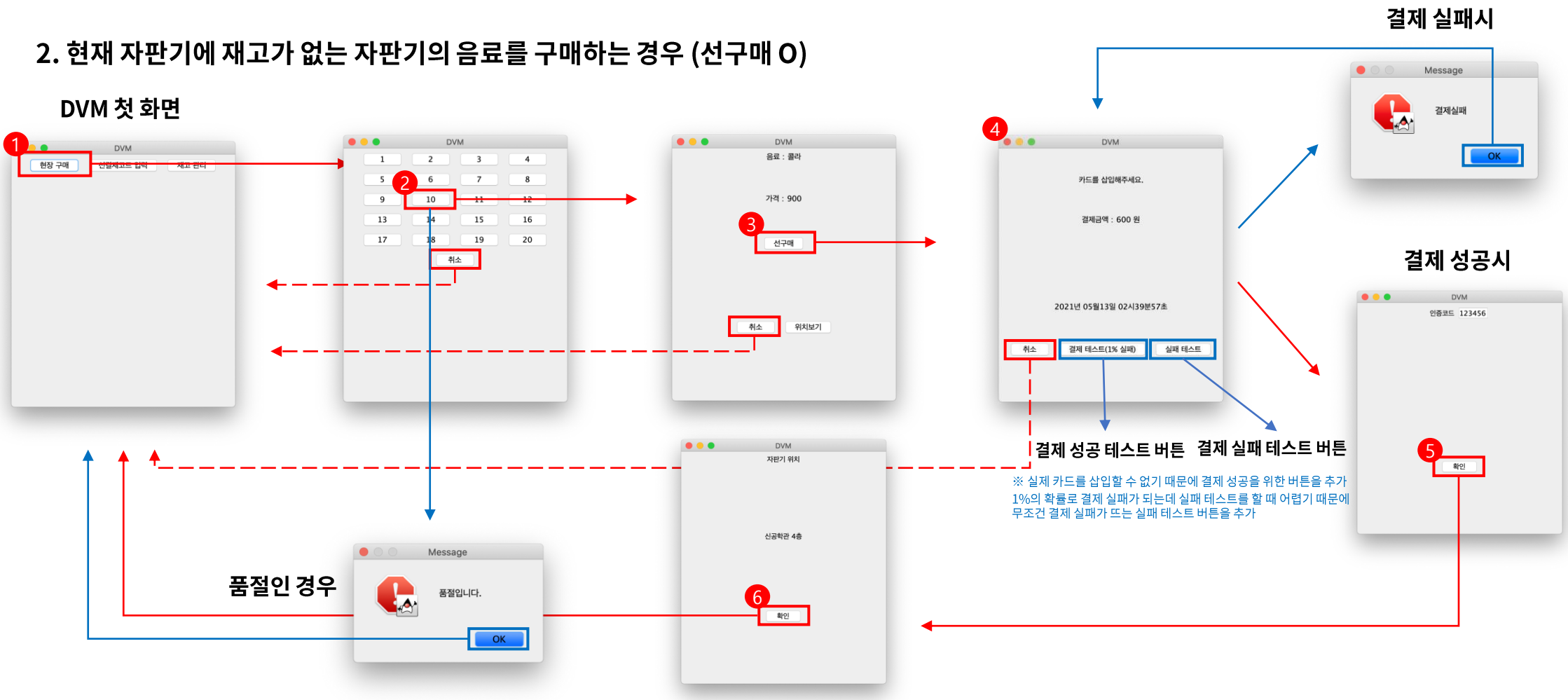
2052 Implement Windows (사용자 매뉴얼)

1. 현재 자판기에 재고가 있는 자판기의 음료를 구매하는 경우



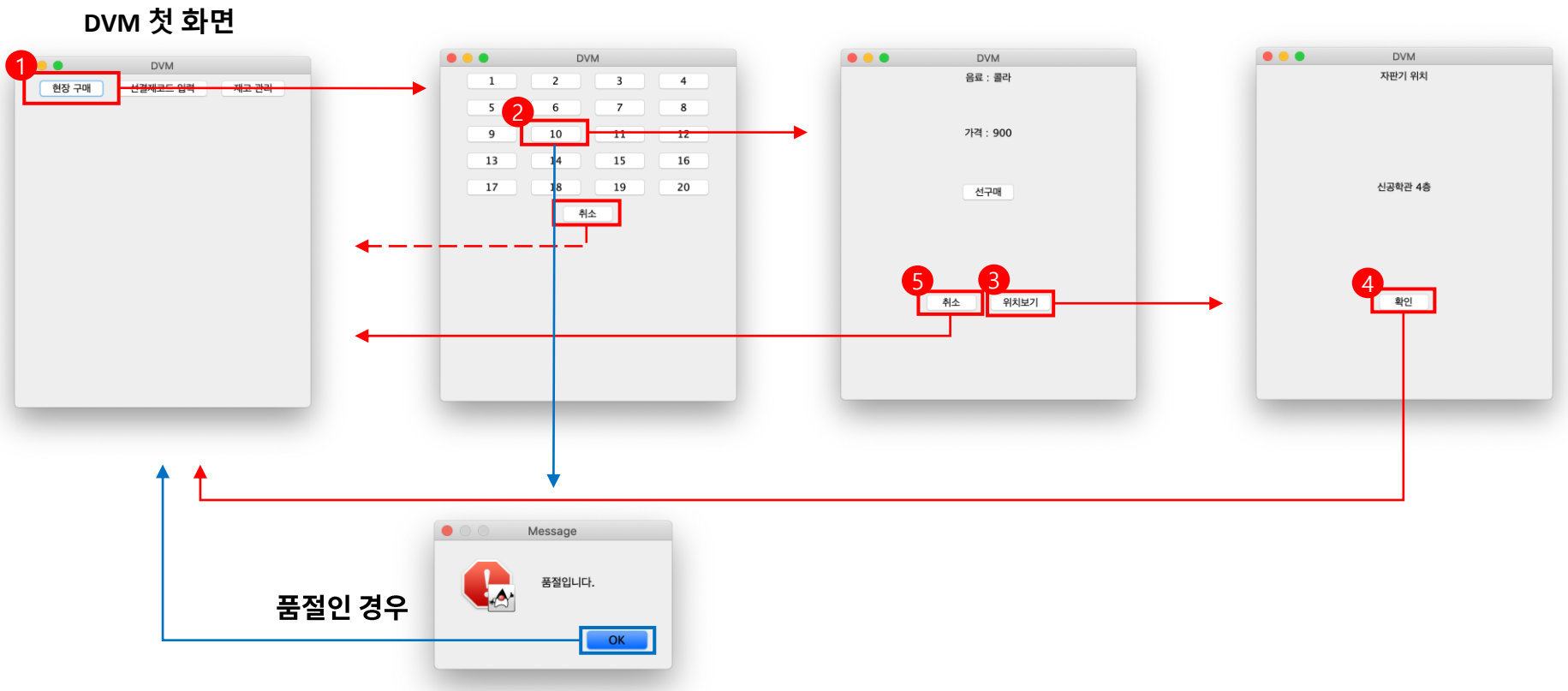
2052 Implement Windows (사용자 매뉴얼)

2. 현재 자판기에 재고가 없는 자판기의 음료를 구매하는 경우 (선구매 O)



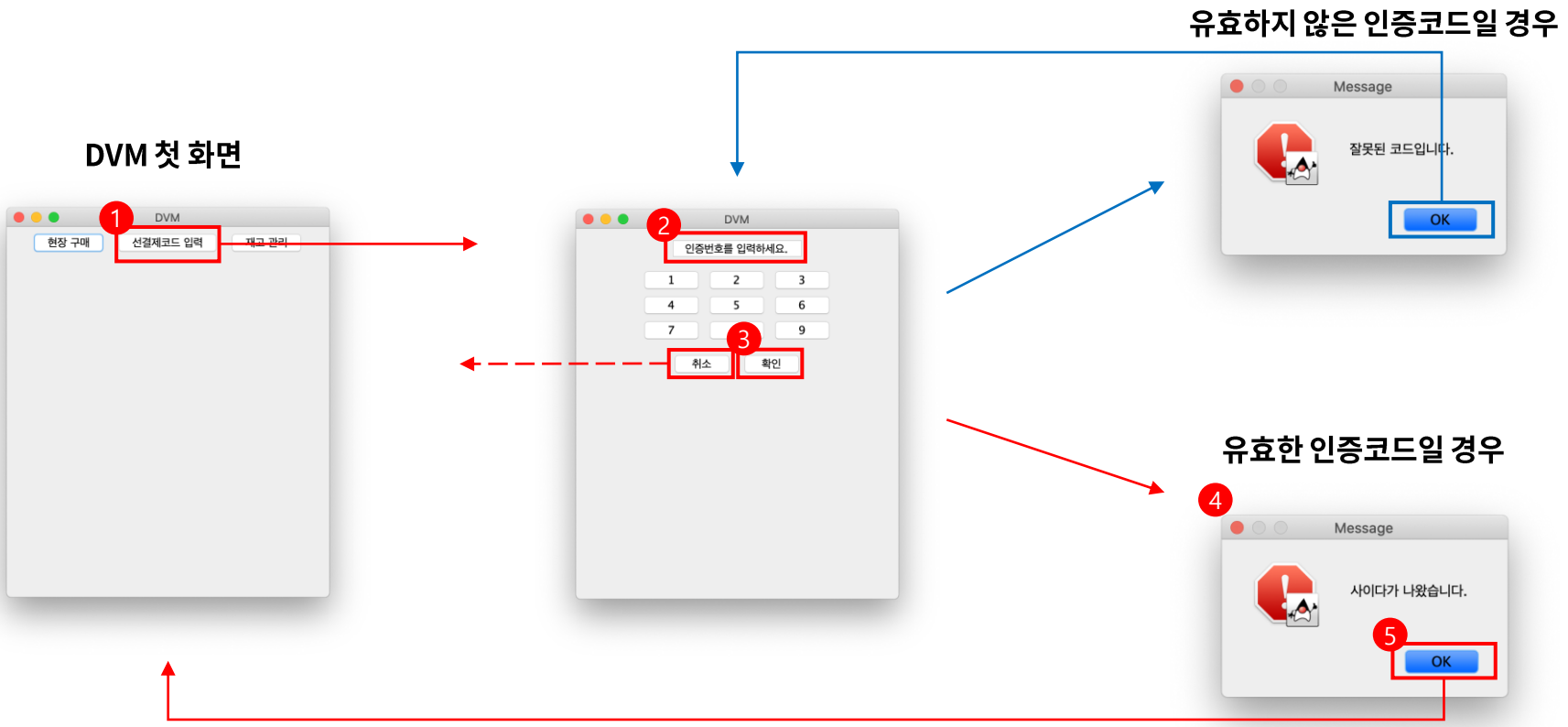
2052 Implement Windows (사용자 매뉴얼)

3. 현재 자판기에 재고가 없는 자판기의 음료를 구매하는 경우 (선구매 X)



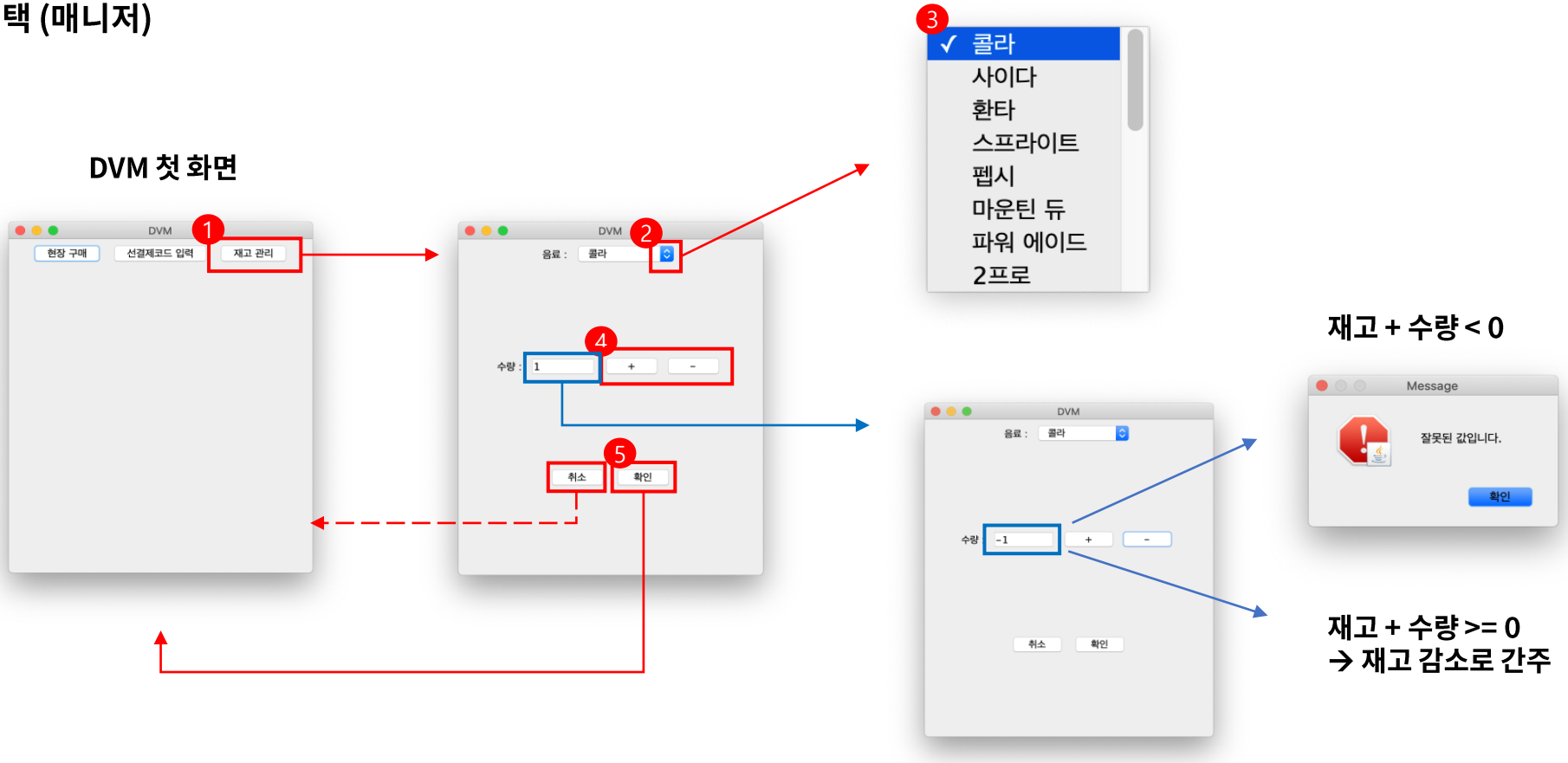
2052 Implement Windows (사용자 매뉴얼)

4. 선결제 코드 입력



2052 Implement Windows (사용자 매뉴얼)

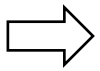
5. 재고관리 선택 (매니저)

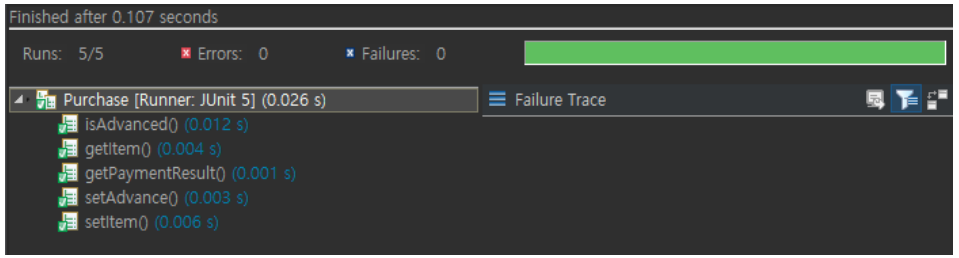


2055 Write Unit Test Code & 2061 Unit Testing

Purchase

```java
@Test
public void getPaymentResult() {
    Random random = new Random();
    assertNotNull(random);
}

@Test
public void isAdvanced() {

}

@Test
public void setAdvance() {
    this.isAdvance = true;
    assertTrue(isAdvance);

    this.isAdvance = false;
    assertFalse(isAdvance);
}

@Test
public void setItem() {
    Item setItem = new Item();
    assertNotNull(setItem);
    this.item = setItem;
    assertEquals(this.item, setItem);
}

@Test
public void getItem() {
    assertNotNull(this.item);
}
```

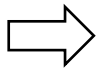


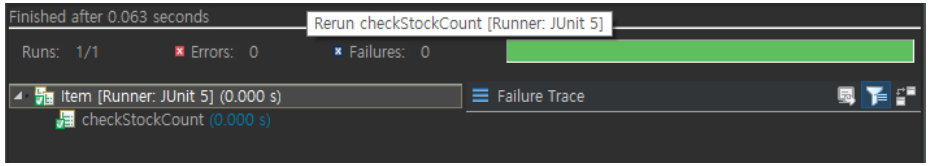


2055 Write Unit Test Code & 2061 Unit Testing

Item

```
@Test
public void checkStockCount() {
    Map<Integer, Integer> map=new HashMap<Integer, Integer>(); // <DVM_ID, 음료개수>
    Map<Integer, String> tmap = new HashMap<Integer, String>();
    if(getCount() <= 0) {
        Message msg = new Message();
        assertNotNull(msg);
        //broadcast
        tmap = new HashMap<>();
        assertNotNull(tmap);
        if(tmap.size() != 0) {
            String responseMessage = tmap.get(2);
            assertNotNull(responseMessage);
            if(responseMessage.contains("#")) {
                String[] msg_body = tmap.get(2).split("#");
                for(int i=0;i<msg_body.length;i++) {
                    String targetItemId_str = msg_body[i].split("\\^")[0];
                    String targetCode_str = msg_body[i].split("\\^")[1];
                    map.put(Integer.parseInt(targetItemId_str), Integer.parseInt(targetCode_str));
                }
            } else {
                String targetItemId_str = responseMessage.split("\\^")[0];
                String targetCode_str = responseMessage.split("\\^")[1];
                map.put(Integer.parseInt(targetItemId_str), Integer.parseInt(targetCode_str));
            }
        }
    }
    else {
        map.put(0,getCount());
    }
}
```





2055 Write Unit Test Code & 2061 Unit Testing

Message

```
@Test
public void requestTo() {
    List<STUB> stubs = Controller.getInstance().getStubs();
    Map<Integer, String> response = null;
    /* broadcast */
    int dst_id = 3;
    assertNotNull(dst_id);
    if(dst_id == 0) {
        //
        response = new HashMap<Integer, String>();
        for(int i = 0; i < 4; i++) {
            STUB stub = stubs.get(i);
            Map<Integer, String> retFromDVM = stub.readMessage(this.type, this.body, 0);
            assertNotNull(retFromDVM);
            if(retFromDVM.get(2) != null) {
                int tCount = Integer.parseInt(retFromDVM.get(2).split("\\^")[1]);
                if(tCount > 0) {
                    if(response.get(2) == null) {
                        response.put(2, (retFromDVM.get(2) + "#"));
                    } else {
                        String old = response.get(2);
                        response.put(2, old + (retFromDVM.get(2) + "#"));
                    }
                } else {
                    if(response.get(2) == null)
                        response.put(2, "0^0#");
                }
            } else if (retFromDVM.get(3) != null) {
                // success 만 올거임.

                if(!retFromDVM.get(3).contentEquals("success")) {
                    //메시지 전달 실패이고
                    //^ 조차 없으면(메시지 요청이 아니면)
                    if(!retFromDVM.get(3).contains("^"))
                        i--;
                }
            } else if (retFromDVM.get(4) != null) {
                // success 만 올거임.
                if(!retFromDVM.get(4).contentEquals("success")) {
                    //메시지 전달 실패이고
                    //^ 조차 없으면(메시지 요청이 아니면)
                    if(!retFromDVM.get(4).contains("^"))
                        i--;
                }
            } else if (retFromDVM.get(6) != null) {
                if(response.get(6) == null) {
                    response.put(6, (retFromDVM.get(6) + "#"));
                } else {
                    String old = response.get(6);
                    response.put(6, old + (retFromDVM.get(6) + "#"));
                }
            } else if (retFromDVM.get(8) != null) {
                int tCount = Integer.parseInt(retFromDVM.get(2));
                if(tCount > 0) {
                    if(response.get(8) == null) {
                        response.put(8, (retFromDVM.get(8) + "#"));
                    } else {
                        String old = response.get(8);
                        response.put(8, old + (retFromDVM.get(8) + "#"));
                    }
                }
            } else {
                response.put(0, "Unknown");
            }
        }
    } else {
        response = new HashMap<>();

        if(response.get(2) != null) {
            // Nothing just return
        } else if (response.get(3) != null) {
            // Nothing just return
        } else if (response.get(4) != null) {
            // Nothing just return
        } else if (response.get(6) != null) {
            // Nothing just return
        } else if (response.get(8) != null) {
            // Nothing just return
        } else {
            response.put(0, "Unknown");
        }
    }
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Message

```
@Test
public void responseTo() {
    HashMap<Integer, String> result = new HashMap<>();
    List<Item> items = Controller.getInstance().getItems();
    int type = 3;
    assertNotNull(type);
    switch(type)
    {
    //재고 요청
    case 1:
    {
        Integer targetItemId = Integer.parseInt(body);
        Item tarItem = items.get(targetItemId);
        assertNotNull(tarItem);
        String message = "0^" + tarItem.getCount();
        result.put(2, message);
        break;
    }
    //선결제 코드 등록
    case 3:
    {
        String body = "0^0";
        assertNotNull(body);
        if(body.contentEquals("success")) {
            break;
        }
        String targetItemId_str = body.split("\\^")[0];
        String targetCode_str = body.split("\\^")[1];

        Integer targetItemId = Integer.parseInt(targetItemId_str);
        Integer targetItemCode = Integer.parseInt(targetCode_str);
        VerificationCode.getInstance().addCode();
        result.put(3, "success");
        break;
    }
```

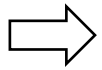
```
    //선결제 코드 삭제
    case 4:
    {
        String body = "test";
        assertNotNull(body);
        if(body.contentEquals("success")) {
            break;
        }
        Integer targetItemCode = Integer.parseInt(body);
        VerificationCode.getInstance().resetCode();
        result.put(4, "success");
        break;
    }
    //주소 요청
    case 5:
    {
        String message = ("0^신공학관1층");
        result.put(6, message);
        break;
    }
    //음료 판매 확인 : 음료 누가 파나요?~
    case 7:
    {
        Integer targetItemId = Integer.parseInt(body);

        Item tarItem = items.get(targetItemId);
        assertNotNull(tarItem);
        int count = tarItem.getCount();

        String message = (count == 0 ? "-1" : ("0"));

        result.put(8, message);
        break;
    }
    default:
    {
        result.put(-1, "unknown");
        break;
    }

    }
}
```

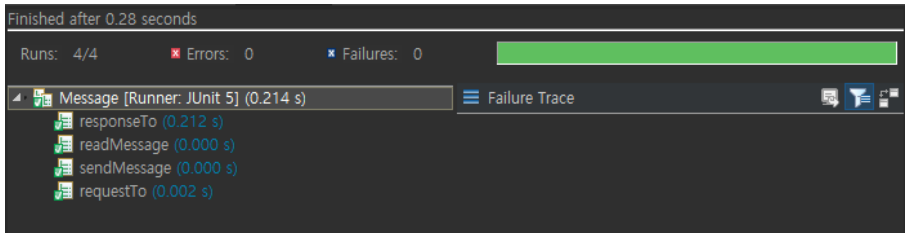
2055 Write Unit Test Code & 2061 Unit Testing

Message

```java
@Test
public void sendMessage() {
    // TODO implement here
    this.type = 0;
    this.body = "test message";
    this.dst_id = 3;
    //나는 항상 0이다.
    this.src_id = 0;
}
```

```java
@Test
public void readMessage() {
    // TODO implement here
    //출발지는 이제 목적지가 됨
    this.src_id = 0;

    /* generate message */
    this.type = 1;
    this.body = "Test Message";
    this.dst_id = 3;
}
```





2055 Write Unit Test Code & 2061 Unit Testing

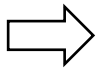
Verification Code

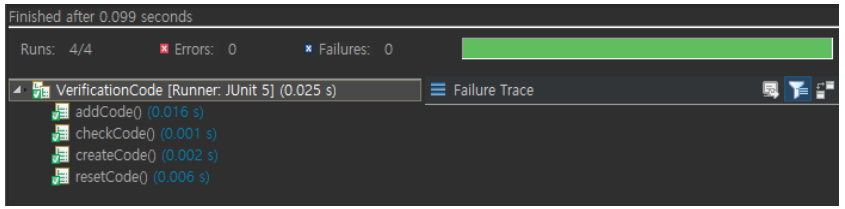
```
@Test
public void createCode() {
    Random rand = new Random();
    Integer code = rand.nextInt(999999);

    //Test code
    assertNotNull(rand);
    assertNotNull(code);
    assert(code.toString().length() == 6);
}

@Test
public void checkCode() {
    // TODO implement here
    Integer code = 123456;
    Set<Integer> itemIds = codeList.keySet();
    Iterator<Integer> iter = itemIds.iterator();

    assertNotNull(code);
    assertNotNull(itemIds);
    assertNotNull(iter);

    int isDone = -1;
    while(iter.hasNext()) {
        Integer key = iter.next();
        ArrayList<Integer> itemCodes = codeList.get(key);

        assertNotNull(key);
        assertNotNull(itemCodes);

        for(int i = 0; i < itemCodes.size(); i++) {
            if(itemCodes.get(i).equals(code)) {
                isDone = i;
                assert(0 <= isDone);
            }
        }
        if(isDone > -1)
            assert(-1 < isDone);
            break;
    }
}

@Test
public void addCode() {
    // TODO implement here
    Integer itemid = new Integer(1);
    Integer code = new Integer(123456);

    assertNotNull(itemid);
    assertNotNull(code);

    try {
        ArrayList<Integer> oldList = (codeList.get(itemid) == null ? new ArrayList<Integer>() : codeList.get(itemid));

        assertNotNull(oldList);

        oldList.add(code);
        codeList.put(itemid, oldList);
        System.out.println("codes: " + codeList);

    } catch (Exception e) {
        System.err.println("Error occured while adding code");

    }
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Verification Code

```java
@Test
public void resetCode() {
    //argument
    Integer code = new Integer(123456);
    Boolean broadcast = true;

    try {
        Set<Integer> itemIds = codeList.keySet();
        Iterator<Integer> iter = itemIds.iterator();

        assertNotNull(itemIds);
        assertNotNull(iter);

        boolean isDone = false;
        int itemid = -1;
        while(iter.hasNext()) {
            Integer key = iter.next();
            ArrayList<Integer> itemCodes = codeList.get(key);

            assertNotNull(key);
            assertNotNull(itemCodes);

            for(int i = 0; i < itemCodes.size(); i++) {
                if(itemCodes.get(i).equals(code)) {
                    itemid = i;
                    itemCodes.remove(i);
                    codeList.put(i, itemCodes);
                    isDone = true;
                    assertTrue(isDone);
                    break;
                }
            }
            if(isDone) {
                assertTrue(isDone);
                break;
            }
        }
        /* broadcast */
        try {
            if(broadcast) {
                Message net = new Message();
                //0^123456 : 콜라에 대한 123456 삭제
                assertNotNull(net);
                net.sendMessage();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        //return true;
    }catch(NullPointerException e) {
        System.err.println("There is no code: " + code);
        //return false;
    }catch(Exception e) {
        System.err.println("Error occured while removing testCode + " +e.getMessage());
        //return false;
    }
}
```





2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void initialize() {
    this.stubs = new ArrayList<>();
    this.dvm = new ArrayList<>();
    this.Items = new ArrayList<>();

    for(int i = 0; i < 20; i++) {
        Item item = new Item();
        assertNotNull(item);
        this.Items.add(item);
    }

    String[] locations = {"신공학관 1층", "새천년관1층", "학생회관1층", "법학관1층", "도서관1층"};
    //층 DVM은 5대. local id = 0;

    for(int i = 0; i < 5; i++) {
        ArrayList<Item> stubItems = new ArrayList<>();
        for(int si = 0; si < 20; si++) {
            stubItems.add(null);
        }
        int totalCount = 0;
        Random rnd = new Random();
        while(totalCount < 7) {
            int cur = rnd.nextInt(20);
            if(stubItems.get(cur) != null) {
                continue;
            } else {
                int count = rnd.nextInt(9)+1;
                Item item = new Item();
                assertNotNull(item);
                stubItems.set(cur, item);
                totalCount++;

                if(i == 0) {

                    Item oldItem = this.Items.get(cur);
                    assertNotNull(oldItem);
                    oldItem.setCount(i, oldItem.getCount() + count);
                }
            }
        }
        DVM curDVM = new DVM(i, locations[i], stubItems);
        assertNotNull(curDVM);
        dvm.add(curDVM);
        /*
         * STUB
         */
        if(i != 0) {
            STUB curStub = new STUB(i, locations[i], stubItems);
            assertNotNull(curStub);
            stubs.add(curStub);
        }
    }

    for(int i = 1; i < 5; i++) {
        Message message = new Message();
        for(int j = 0; j < 20; j++) {
            Map<Integer, String> result = new HashMap<>();
            assertNotNull(result);

            if(result.get(2) == null)
                continue;

            Integer itemCount = Integer.parseInt(result.get(2).split("\\^")[1]);
            Items.get(j).setCount(i, itemCount);
        }
    }

    JFrame UI = new JFrame();
    UI.setTitle("DVM");
    UI.setSize(370,450);
    UI.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    selectMode();
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void showItems() {
    JFrame UI = new JFrame();
    JPanel showItems = new JPanel();

    showItems.setLayout(new GridLayout());

    JPanel showItemUnder = new JPanel();
    showItemUnder.setSize(370,450);

    showItemUnder.setLayout(new FlowLayout());
    JButton[] n = new JButton[21];

    for(int i=0;i<=20;i++) {
        if (i == 20) {
            n[i] = new JButton();
            n[i].setText("취소");
        } else {
            n[i] = new JButton();
            n[i].setText(Integer.toString(i+1));
        }
        showItemUnder.add(n[i]);
    }

    showItems.add(showItemUnder);

    UI.add(showItems);
    showItems.setVisible(true);

    for(int i=0;i<=20;i++) {
        n[i].addActionListener( new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {

                String itemNum = e.getActionCommand();
                //  테스트 위한 가상 변수
                int exit1 = 0;
                int exit2= 1;

                if(itemNum.equals("취소")) { //  취소 클

                    showItems.setVisible(false);
                    showCancel();

                } else {
                    int itemid_int = Integer.parseInt(itemNum)-1;
                    currentItem = Items.get(itemid_int);

                    int stockOK = -1;
                    if(currentItem.getCount() == 0) {
                        HashMap<Integer, Integer> otherStocks = new HashMap<>();
                        assertNotNull(otherStocks);
                        Set<Integer> keys = otherStocks.keySet();
                        Iterator<Integer> iter = keys.iterator();

                        while(iter.hasNext()) {
                            Integer iCount = iter.next();
                            if(iCount != 0) {
                                stockOK = 1;
                                break;
                            }
                        }
                    } else {
                        stockOK = 0;
                    }

                    if (stockOK == 0) {
                        showItems.setVisible(false);
                        purchase = new Purchase();
                        insertCard();
                    } else if (stockOK == 1) {
                        showItems.setVisible(false);
                        purchase = new Purchase();
                        showAdvancePurchase();
                    } else {
                        showItems.setVisible(false);
                        printe();
                        showCancel();
                    }
                }
            }
        });
    }
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void selectMode() {
    JFrame UI = new JFrame();
    JPanel Panel = new JPanel();
    assertNotNull(UI);
    assertNotNull(Panel);
    Integer mode = 0;
    if(Panel != null) {
        Panel.setVisible(false);
        System.out.println(mode + " 클릭");
    }
    if(mode == 0)
        showMode();
    else if(mode == 1) {
        showItems();
    }
    else if(mode == 2) {
        readVerificationCode();
    }
    else if(mode == 3) {
        updateStockInfo();
    }
}
```

```
@Test
public void print() {
    String message = "inform message";
    JOptionPane.showMessageDialog(null,
            message, "Message",
            JOptionPane.INFORMATION_MESSAGE);
}

@Test
public void printe() {
    String error="error message";
    JOptionPane.showMessageDialog(null,
            error, "Message",
            JOptionPane.ERROR_MESSAGE);
}

@Test
public void setScreen() {   //  완
    Integer mode = 1;
    JFrame UI = new JFrame();
    UI.setTitle("DVM");
    UI.setSize(370,450);
    UI.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    showMode();

    if (mode == 2) {
        //showVerificationCode(UI, veriCode);
    }
}
```

```
@Test
public void itemOut() {
    JOptionPane.showMessageDialog(null,
            ("음료가 나왔습니다."), "Message",
            JOptionPane.ERROR_MESSAGE);
    // TODO implement here
    currentItem = null;
    purchase = null;
}



@Test
public void showCancel() {
    JFrame UI = new JFrame();
    showMode();
    currentItem = null;
    purchase = null;
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void updateStockInfo() {
    JFrame UI = new JFrame();
    JPanel manageStock = new JPanel();
    manageStock.setLayout(new GridLayout(3,1));
    JPanel manageItems = new JPanel();
    JPanel manageEtc = new JPanel();
    JPanel manageButton = new JPanel();
    JPanel updown = new JPanel(new GridLayout(1,0));
    JLabel itemName = new JLabel();
    String itemList[] = Item.names;
    JComboBox JItemList = new JComboBox<String>(itemList);
    itemName.setText("음료 : ");
    JLabel itemCount = new JLabel();
    JTextField countItem = new JTextField(6);
    itemCount.setText("수량 : ");
    countItem.setText("1");
    JButton up = new JButton("+");
    JButton down = new JButton("-");

    cancel = new JButton("취소");
    ok = new JButton("확인");

    manageItems.add(itemName);
    manageItems.add(JItemList);
    manageEtc.add(itemCount);
    manageEtc.add(countItem);
    manageEtc.add(updown);
    manageButton.add(cancel);
    manageButton.add(ok);
    updown.add(up);
    updown.add(down);
    manageStock.add(manageItems);
    manageStock.add(manageEtc);
    manageStock.add(manageButton);
    UI.add(manageStock);
    manageStock.setVisible(true);

    cancel.addActionListener( new ActionListener(){
        @Override

        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            manageStock.setVisible(false);
            showCancel();
        }
```

```
    });
    up.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            int count = Integer.parseInt(countItem.getText());
            count += 1;
            countItem.setText(Integer.toString(count));
        }
    });
    down.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            int count = Integer.parseInt(countItem.getText());
            count -= 1;
            countItem.setText(Integer.toString(count));
        }
    });

    ok.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            Item selectedItem = Items.get(JItemList.getSelectedIndex());
            int count = Integer.parseInt(countItem.getText());
            int oldCount = selectedItem.getCount();
            if(oldCount + count < 0)
                printe();
            else {
                selectedItem.setCount(0, selectedItem.getCount() + count);
                manageStock.setVisible(false);
                showMode();
            }

        }

    });
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void insertCard() {  //
    JFrame UI = new JFrame();
    SimpleDateFormat format2 = new SimpleDateFormat ( "yyyy년 MM월dd일 HH시mm분ss초");
    Date time = new Date();
    String time2 = format2.format(time);
    if(currentItem == null) {
        selectMode();
        System.out.println("음료가 선택되지 않았습니다.");
        return;
    }
    int itemPrice = currentItem.getPrice();
    JPanel insertCard = new JPanel();
    insertCard.setLayout(new GridLayout(4,0));
    JPanel buttonPanel = new JPanel(new FlowLayout());
    JPanel itemAndPrice = new JPanel(new FlowLayout());
    JLabel insert = new JLabel();
    JLabel price = new JLabel();
    JLabel setPrice = new JLabel();
    JLabel date = new JLabel();
    cancel = new JButton("취소");
    JButton purchaseResult = new JButton("결제 테스트(1% 실패)");
    JButton failTest = new JButton("실패 테스트");

    insert.setText("카드를 삽입해주세요.");
    price.setText("결제금액 :");
    setPrice.setText(Integer.toString(itemPrice) + " 원");
    date.setText(time2);
    itemAndPrice.add(price);
    itemAndPrice.add(setPrice);
    buttonPanel.add(cancel);
    buttonPanel.add(purchaseResult);
    buttonPanel.add(failTest);
    insertCard.add(insert);
    insertCard.add(itemAndPrice);
    insertCard.add(date);
    insertCard.add(buttonPanel);
    insert.setHorizontalAlignment(JLabel.CENTER);
    price.setHorizontalAlignment(JLabel.CENTER);
    date.setHorizontalAlignment(JLabel.CENTER);
    UI.add(insertCard);
    insertCard.setVisible(true);
    boolean validation = true;
```

```
    purchaseResult.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            boolean result = true;
            boolean result2 = true;

            if(result) {
                insertCard.setVisible(false);
                if(result2) {

                    int newCode = 123456;
                    Message message = new Message();
                    message.sendMessage();
                    showVerificationCode();
                }
                else {
                    int targetId = currentItem.getId();
                    Item item = Items.get(targetId);
                    item.setCount(0, item.getCount()-1);
                    itemOut();
                    showMode();
                }
            } else {
                printe();
            }
        }
    });
    failTest.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            printe();
        }
    });
    cancel.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            insertCard.setVisible(false);
            showCancel();
        }
    });
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void readVerificationCode() {
    JFrame UI = new JFrame();
    JPanel readVcode = new JPanel();
    JPanel txtPanel = new JPanel(new FlowLayout());
    JPanel button = new JPanel(new FlowLayout());
    JPanel numbers = new JPanel(new GridLayout(4,3));
    JTextField vcode = new JTextField(12);
    cancel = new JButton("취소");
    ok = new JButton("확인");
    JButton[] keypad = new JButton[10];

    vcode.setText("인증번호를 입력하세요.");
    txtPanel.add(vcode);
    button.add(cancel);
    button.add(ok);
    for(int i =1; i<=10; i++) {
        int realvalue = i;
        if(i == 10) {
            realvalue = 0;
        }
        keypad[realvalue] = new JButton(Integer.toString(realvalue));
        numbers.add(keypad[realvalue]);
        keypad[realvalue].addActionListener( new ActionListener(){
            @Override
            public void actionPerformed(ActionEvent e) {
                String preset = "인증번호를 입력하세요.";
                String vode = e.getActionCommand();
                if(vcode.getText().equals(preset)) {
                    vcode.setText(vode);
                } else if(!vcode.getText().equals(preset)) {
                    if(vcode.getText().length() < 6) {
                        vode = vcode.getText() + vode;
                        vcode.setText(vode);
                    }
                }
            }
        });
    }
```

```
    readVcode.add(txtPanel);
    readVcode.add(numbers);
    readVcode.add(button);
    UI.add(readVcode);
    readVcode.setVisible(true);
    vcode.setHorizontalAlignment(JLabel.CENTER);

    cancel.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            readVcode.setVisible(false);
            showCancel();
        }
    });

    ok.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            String preset = "인증번호를 입력하세요.";
            if(!vcode.getText().contentEquals(preset)) {
                int inputCode = 0;
                try {
                    inputCode = Integer.parseInt(vcode.getText());
                    int codeItem = 1;
                    if(codeItem > -1) {
                        readVcode.setVisible(false);
                        currentItem = Items.get(codeItem);
                        VerificationCode.getInstance().resetCode();
                        itemOut();
                        showMode();
                    } else {
                        printe();
                    }
                } catch (NumberFormatException er) {
                    printe();
                }
            } else {
                printe();
            }

        }
    });
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void showLocation() { // (DVM dvm)
    JFrame UI = new JFrame();
    String dvm_ids="2^";
    //JFrame showLocation2 = new JFrame();
    JPanel showLocation = new JPanel();
    setTitle("showLocation");
    setSize(370,450);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

    //Container c = getContentPane();
    showLocation.setLayout(new GridLayout(0,1));

    JPanel underLayer1 = new JPanel(new FlowLayout());
    JPanel underLayer2 = new JPanel(new FlowLayout());
    JPanel underLayer3 = new JPanel(new FlowLayout());

    JLabel locate = new JLabel("자판기 위치");
    JLabel locationSet = new JLabel();
    String dvmLocations = null;
    String[] dvms = null;
    if(dvm_ids != null) {
        if(dvmLocations == null)
            dvmLocations = "";
        dvms = dvm_ids.split("\\^");
    }

    String[] locations = {"신공학관 1층", "새천년관1층", "학생회관1층", "법학관1층", "도서관1층"};

    if(dvms != null) {
        for(int i = 0; i < dvms.length; i++) {
            Integer dvmId = Integer.parseInt(dvms[i]);
            dvmLocations += locations[dvmId];

            if(i < dvms.length-1)
                dvmLocations += ",";
        }
    } else {
        dvmLocations = "없음";
    }

    //      System.out.println("dvm location: " + this.dvm.get(dvm_id).getLocation());
    locationSet.setText(dvmLocations);
    ok = new JButton("확인");

    underLayer1.add(locate);
    underLayer2.add(locationSet);
    underLayer3.add(ok);

    showLocation.add(underLayer1);
    showLocation.add(underLayer2);
    showLocation.add(underLayer3);


    UI.add(showLocation);
    showLocation.setVisible(true);

    ok.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            showLocation.setVisible(false);
            showCancel();
        }
    });
}
```

```
@Test
public void showVerificationCode() { // (Integer Code)
    JFrame UI = new JFrame();
    Integer code = 123456;
    JPanel showVcode = new JPanel();
    showVcode.setLayout(new GridLayout(0,1));

    JPanel codeGrid = new JPanel(new FlowLayout());
    JPanel timeGrid = new JPanel(new FlowLayout());
    JPanel btnGrid = new JPanel(new FlowLayout());

    JLabel item = new JLabel("인증코드");
    JTextField cvcode = new JTextField(Integer.toString(code));
    JTextField pricep = new JTextField(6);
    ok = new JButton("확인");

    pricep.setText(Integer.toString(code));

    codeGrid.add(item);
    codeGrid.add(cvcode);

    btnGrid.add(ok);

    showVcode.add(codeGrid);
    showVcode.add(timeGrid);
    showVcode.add(btnGrid);

    UI.add(showVcode);
    showVcode.setVisible(true);

    ok.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            showVcode.setVisible(false);
            //current Item을 갖고 있는 dvm id 구하기
            HashMap<Integer, Integer> otherStocks = new HashMap<>();
            assertNotNull(otherStocks);
            int dvm_id = 1;

            String dvms = null;

            for(dvm_id = 1; dvm_id < 5; dvm_id++) {
                if(otherStocks.get(dvm_id) != null) {
                    if(otherStocks.get(dvm_id) > 0) {
                        if(dvms == null)
                            dvms = "";
                        dvms += dvm_id + "^";
                    }
                }
            }

            showLocation();
        }
    });
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

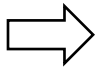
```
@Test
public void showAdvancePurchase() { // (Item item)
    JFrame UI = new JFrame();
    JPanel showAdvancePurchase1 = new JPanel(new GridLayout(3,1));
    JPanel itemPanel = new JPanel(new GridLayout(0,1));
    JPanel itemLabel = new JPanel(new FlowLayout());
    JLabel item = new JLabel("음료 :");
    JLabel itemName = new JLabel("콜라");
    itemLabel.add(item);
    itemLabel.add(itemName);

    JPanel priceLabel = new JPanel(new FlowLayout());
    JLabel price = new JLabel("가격 :");
    JLabel setPrice = new JLabel("700 원");
    priceLabel.add(price);
    priceLabel.add(setPrice);

    itemPanel.add(itemLabel);
    itemPanel.add(priceLabel);

    JPanel apbPanel = new JPanel(new FlowLayout());

    JButton AdvancePurcahaseButton = new JButton("선구매");
    apbPanel.add(AdvancePurcahaseButton);

    JPanel buttonPanel = new JPanel();
    cancel = new JButton("취소");
    JButton location = new JButton("위치보기");

    buttonPanel.add(cancel);
    buttonPanel.add(location);

    showAdvancePurchase1.add(itemPanel);
    showAdvancePurchase1.add(apbPanel);
    showAdvancePurchase1.add(buttonPanel);

    UI.add(showAdvancePurchase1);
    showAdvancePurchase1.setVisible(true);
```

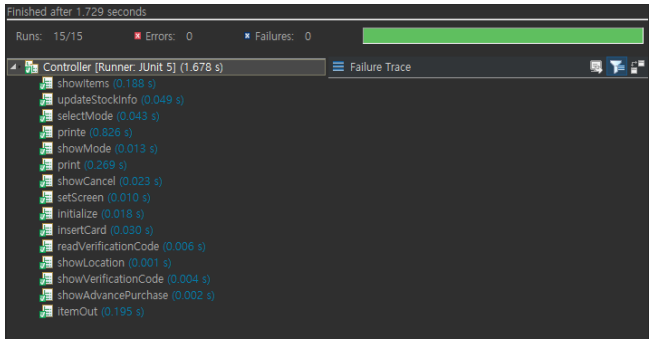
```
    //AdvancePurcahaseButton.setSize(100, 75);
    AdvancePurcahaseButton.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            showAdvancePurchase1.setVisible(false);
            insertCard();
        }
    });
    location.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            showAdvancePurchase1.setVisible(false);
            //current Item을 갖고 있는 dvm id 구하기
            HashMap<Integer, Integer> otherStocks = new HashMap<>();
            assertNotNull(otherStocks);
            int dvm_id = 1;

            String dvms = null;

            for(dvm_id = 1; dvm_id < 5; dvm_id++) {
                if(otherStocks.get(dvm_id) != null) {
                    if(otherStocks.get(dvm_id) > 0) {
                        if(dvms == null)
                            dvms = "";
                        dvms += dvm_id + "^";
                    }
                }
            }

            showLocation();
        }
    });

    cancel.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            showAdvancePurchase1.setVisible(false);
            showMode();
        }
    });
}
```

2055 Write Unit Test Code & 2061 Unit Testing

Controller

```
@Test
public void showMode() {
    // TODO implement here
    JFrame UI = new JFrame();

    JPanel showMode = new JPanel();
    showMode.setLayout(new FlowLayout());

    JButton goShowItem = new JButton("현장 구매");
    JButton goReadVcode = new JButton("선결제코드 입력");
    JButton goManageStock = new JButton("재고 관리");

    showMode.add(goShowItem);
    showMode.add(goReadVcode);
    showMode.add(goManageStock);

    UI.add(showMode);
    UI.setVisible(true);



    goShowItem.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            showMode.setVisible(false);
            System.out.println("1 클릭");
            showItems();
        }

    });
    goReadVcode.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            showMode.setVisible(false);
            System.out.println("2 클릭");
            readVerificationCode();
        }

    });
    goManageStock.addActionListener( new ActionListener(){
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            showMode.setVisible(false);
            System.out.println("3 클릭");
            manageStock();
        }

    });
}
```





2063 System Testing

System Test Case						
Seq	No	Func.	Use Case	Title	Description	Pass
1	1-1	R1.1	Set Up	Initializing Test - Get DVMs Id	시스템이 시작되었을 때 다른 DVM의 id를 받아왔는지 확인	P
2	1-2	R1.1	Set Up	Initializing Test - Get Other DVMs Stock	시스템이 시작되었을 때 다른 DVM의 재고를 받아왔는지 확인	P
3	1-3	R1.1	Set Up	Initializing Test - Check Stock Update	시스템이 시작되었을 때 현재 자판기의 재고가 업데이트 되었는지 확인	P
4	2-1	R1.2	Select Mode	Select Mode Test - Single Selection	User가 선택한 모드에 따라 정해진 메뉴가 실행되는지 확인	P
5	2-2	R1.2	Select Mode	Select Mode Test - Duplicate Selection	User가 여러개의 모드를 선택했을 때 현장구매모드가 실행되는지 확인	P
6	3-1	R1.3	Manage Stock	Manage Stock Test - Managing	Manager가 갱신한 재고 정보가 적용되는지 확인	P
7	3-2	R1.3	Manage Stock	Manage Stock Test – Check Over Updating	Manager가 총 8가지 이상의 음료를 갱신할 경우 에러가 나오는지 확인	P
8	4	R2.1	Show Item	Show Item Test - Display Item	모든 음료가 나오는지 확인	P
9	5	R2.2	Select Item	Select Item Test - Item Selection	User가 선택한 음료가 선택되는지 확인	P
10	6-1	R2.3	Check Stock Count	Check Stock Count Test - Item Sellable	User가 선택한 음료에 대해 현재 자판기에서 판매가능한지 확인	P

2063 System Testing

System Test Case						
Seq	No	Func.	Use Case	Title	Description	Pass
11	6-2	R2.3	Check Stock Count	Check Stock Count Test - Call “Purchase”	현재 자판기에서 판매 가능하다면 “Purchase”를 정상적으로 실행하는지 확인	P
12	6-3	R2.3	Check Stock Count	Check Stock Count Test - Other DVMs Sellable	통신을 통해 다른 자판기로부터 재고를 받아오는지 확인	P
13	6-4	R2.3	Check Stock Count	Check Stock Count Test - Call “Select Advance Payment”	다른 자판기로부터 재고가 있다는 응답을 받으면, “Select Advance Payment”를 정상적으로 실행하는지 확인	P
14	7	R2.4	Update Stock	Update Stock Test	음료 재고 정보를 주어진 값에 맞게 최신화 하는지 확인	P
15	8-1	R2.5	Purchase	Purchase Test - Subtracting Stock	결제가 성공했을 때 음료 재고를 맞게 줄이는지 확인	P
16	8-2	R2.5	Purchase	Purchase Test - Item Out	결제가 성공했을 때 해당 음료를 배출하는지 확인	P
17	8-3	R2.5	Purchase	Purchase Test - Show Message	결제가 실패했을 때, 결제 실패 메시지를 보여주는지 확인	P
18	9-1	R2.6	Advance Purchase	Advance Purchase Test - Code Result	선구매 결제에 성공했을 때, 선구매 코드를 받아오는지 확인	P
19	9-2	R2.6	Advance Purchase	Advance Purchase Test - Broadcasting	선구매 결제에 성공했을 때, 받아온 선구매 코드를 “Message Request”를 통해 Broadcast 하는지 확인	P
20	9-3	R2.6	Advance Purchase	Advance Purchase Test – Call “Location Inform”	선구매 결제에 성공했을 때, 해당 음료가 있는 자판기에 대해 “Inform Location”을 실행하는지 확인	P

2063 System Testing

System Test Case						
Seq	No	Func.	Use Case	Title	Description	Pass
21	9-4	R2.6	Advance Purchase	Advance Purchase Test - Show Code	선구매 결제에 성공했을 때, 생성된 선구매 코드를 보여주는지 확인	P
22	9-5	R2.6	Advance Purchase	Advance Purchase Test - Show Message	선구매 결제에 실패했을 때, 결제 실패 메시지를 보여주는지 확인	P
23	9-6	R2.6	Advance Purchase	Advance Purchase Test - Adding Stock	선구매 결제에 실패했을 때, 감소시켰던 재고 수량을 다시 증가시키는지 확인	P
24	10-1	R2.7	Check Payment	Check Payment Test - Check Result	결제 결과를 받아오는지 확인	P
25	10-2	R2.7	Check Payment	Check Payment Test - Show Message	정상적인 결제 카드가 아니라면 “잘못된 카드”라는 에러 메시지를 보여주는지 확인	P
26	11-1	R2.8	Select Advance Payment	Select Advance Payment Test – Stock Check	선구매를 선택한 음료에 대해 “Message Request”로 재고를 받아오는지 확인	P
27	11-2	R2.8	Select Advance Payment	Select Advance Payment Test – Subtracting Other DVMs Stock	선구매 결정한 음료의 재고가 남아있다면 “Message Request”로 재고를 감소시키는지 확인	P
28	11-3	R2.8	Select Advance Payment	Select Advance Payment Test – Call “Advance Purchase”	재고를 감소시킨 후 “Advance Purchase”를 실행하는지 확인	P
29	11-4	R2.8	Select Advance Payment	Select Advance Payment Test – Show Message and Return to “Select Mode”	선구매 결정한 음료의 재고가 없다면 품절메시지를 보여주고, “Select Mode”로 돌아가는지 확인	P
30	11-5	R2.8	Select Advance Payment	Select Advance Payment Test – Get Other DVMs Stock for Location Mode	위치확인 모드를 선택한 음료에 대해 “Message Request”로 재고를 받아오는지 확인	P

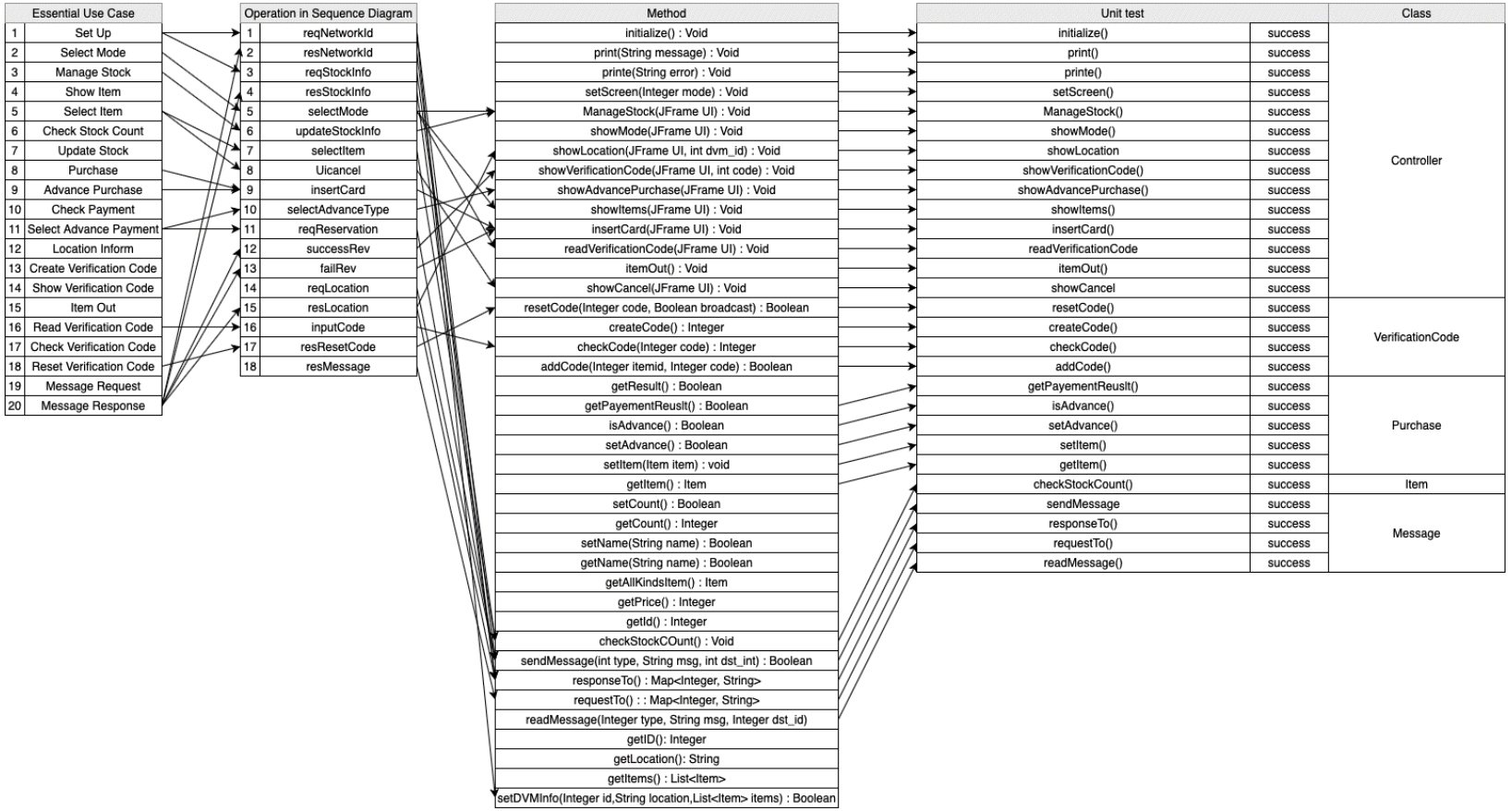
2063 System Testing

System Test Case						
Seq	No	Func.	Use Case	Title	Description	Pass
31	11-6	R2.8	Select Advance Payment	Select Advance Payment Test – Call “Location Inform”	위치확인 모드를 선택한 음료의 재고가 남아있다면 해당 음료에 대해 “Inform Location”을 실행하는지 확인	P
32	11-7	R2.8	Select Advance Payment	Select Advance Payment Test – Show Message and Return to “Select Mode”	위치확인 모드를 선택한 음료의 재고가 남아있지 않다면 품절메시지를 보여주고, “Select Mode”로 돌아가는지 확인	P
33	11-8	R2.8	Select Advance Payment	Select Advance Payment Test – Duplicate Selection	모드 선택과 취소를 둘다 누를 경우 사용자로부터 재입력 받는지 확인	P
34	12-2	R2.9	Inform Location	Inform Location Test	음료를 갖고있는 자판기에 대해 위치를 보여주는지 확인	P
35	13	R2.10	Create Verification Code	Create Verification Code Test	음료가 선택되어 있을 때, 새로운 선구매 코드를 생성하는지 확인	P
36	14	R2.11	Show Verification Code	Show Verification Code Test	“Create Verification Code”를 통해 생성된 선구매 코드를 보여주는지 확인	P
37	15	R2.12	Item Out	Item Out Test	음료가 선택되어 있고, 결제가 완료되었거나 유효한 선구매 코드를 입력했을 때만 음료를 제공하는지 확인	P
38	16-1	R3.1	Read Verification Code	Read Verification Code Test – Get Verifying Result	User가 입력한 선구매 코드 “Check Verification Code”에 전달하고 그 결과를 받는지 확인	P
39	16-2	R3.1	Read Verification Code	Read Verification Code Test – Call “Item Out”	기록된 코드가 유효하다면 “Item Out”을 실행하는지 확인	P
40	16-3	R3.1	Read Verification Code	Read Verification Code Test – Call “Reset Verification Code”	“Item Out”을 실행한 후에 “Reset Verification Code”를 실행하는지 확인	P

2063 System Testing

System Test Case						
Seq	No	Func.	Use Case	Title	Description	Pass
41	17	R3.2	Check Verification Code	Check Verification Code Test	시스템에 입력된 코드가 존재하는지 검사하는지 확인	P
42	18	R3.3	Reset Verification Code	Reset Verification Code Test	시스템에서 입력된 코드를 정상적으로 삭제하는지 확인	P
43	19	R4.1	Message Request	Message Request Test	보내고자 하는 메시지의 종류를 수신 대상에게 잘 보내는지 확인	P
44	20-1	R4.2	Message Response	Message Response Test – Response to Target	수신된 메시지에 따라 명시된 응답을 전송한 DVM에 정상적으로 보내는지 확인	P
45	20-2	R4.2	Message Response	Message Response Test – Exception to Target	알 수 없는 종류의 메시지를 수신한 경우, 알 수 없음을 응답하는지 확인	P

2066 Testing Traceability Analysis



Essential Use Case	
1	Set Up
2	Select Mode
3	Manage Stock
4	Show Item
5	Select Item
6	Check Stock Count
7	Update Stock
8	Purchase
9	Advance Purchase
10	Check Payment
11	Select Advance Payment
12	Location Inform
13	Create Verification Code
14	Show Verification Code
15	Item Out
16	Read Verification Code
17	Check Verification Code
18	Reset Verification Code
19	Message Request
20	Message Response

Operation in Sequence Diagram	
1	reqNetworkId
2	resNetworkId
3	reqStockInfo
4	resStockInfo
5	selectMode
6	updateStockInfo
7	selectItem
8	Uicancel
9	insertCard
10	selectAdvanceType
11	reqReservation
12	successRev
13	failRev
14	reqLocation
15	resLocation
16	inputCode
17	resResetCode
18	resMessage

Method
initialize() : Void
print(String message) : Void
printe(String error) : Void
setScreen(Integer mode) : Void
ManageStock(JFrame UI) : Void
showMode(JFrame UI) : Void
showLocation(JFrame UI, int dvm_id) : Void
showVerificationCode(JFrame UI, int code) : Void
showAdvancePurchase(JFrame UI) : Void
showItems(JFrame UI) : Void
insertCard(JFrame UI) : Void
readVerificationCode(JFrame UI) : Void
itemOut() : Void
showCancel(JFrame UI) : Void
resetCode(Integer code, Boolean broadcast) : Boolean
createCode() : Integer
checkCode(Integer code) : Integer
addCode(Integer itemid, Integer code) : Boolean
getResult() : Boolean
getPayementReuslt() : Boolean
isAdvance() : Boolean
setAdvance() : Boolean
setItem(Item item) : void
getItem() : Item
setCount() : Boolean
getCount() : Integer
setName(String name) : Boolean
getName(String name) : Boolean
getAllKindsItem() : Item
getPrice() : Integer
getId() : Integer
checkStockCOunt() : Void
sendMessage(int type, String msg, int dst_int) : Boolean
responseTo() : Map<Integer, String>
requestTo() : : Map<Integer, String>
readMessage(Integer type, String msg, Integer dst_id)
getID(): Integer
getLocation(): String
getItems() : List<Item>
setDVMInfo(Integer id,String location,List<Item> items) : Boolean

Unit test		Class
initialize()	success	Controller
print()	success	
printe()	success	
setScreen()	success	
ManageStock()	success	
showMode()	success	
showLocation	success	
showVerificationCode()	success	
showAdvancePurchase()	success	
showItems()	success	
insertCard()	success	
readVerificationCode	success	
itemOut()	success	
showCancel	success	
resetCode()	success	VerificationCode
createCode()	success	
checkCode()	success	
addCode()	success	
getPayementReuslt()	success	Purchase
isAdvance()	success	
setAdvance()	success	
setItem()	success	
getItem()	success	
checkStockCount()	success	Item
sendMessage	success	Message
responseTo()	success	
requestTo()	success	
readMessage()	success	

시연 동영상

감사합니다