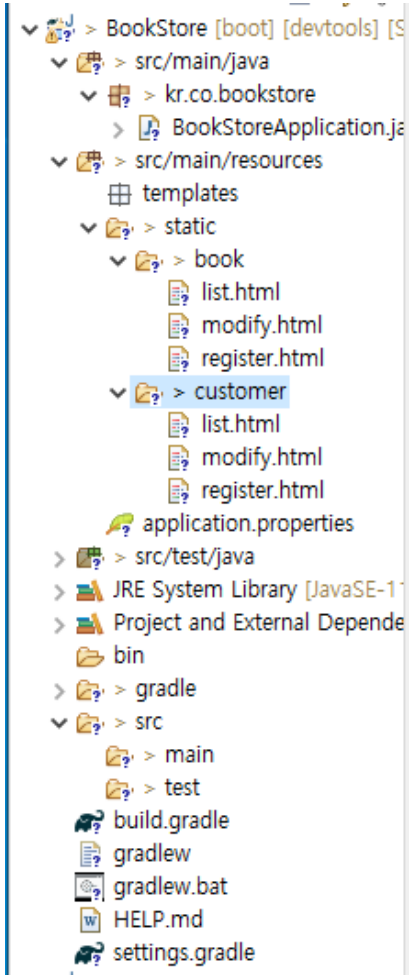
스프링 프로그래밍 응용

스프링 프로그래밍 응용

심규영

목차

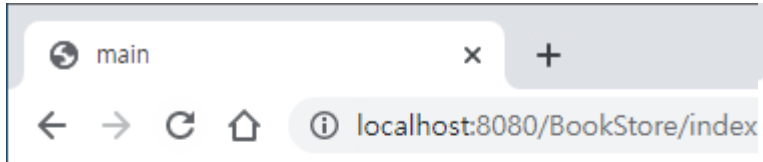
1) 프로젝트 생성 및 구현



- ProjectName : BookStore
- WAS : Tomcat 9(Spring boot app)
- DB : java2_bookstore
- Framework : Spring(SpringBoot), Mybatis
- Tools : Eclipse, Workbench

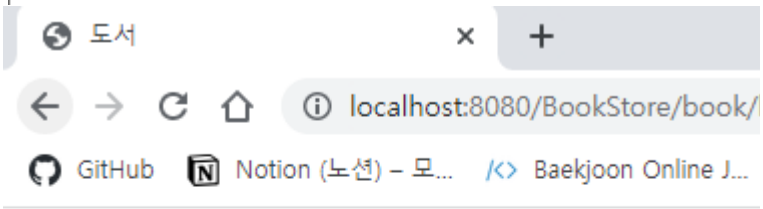
2) 화면 구현

- 메인(index)
- book(도서) 목록, 등록, 수정 화면 구현
- customer(고객) 목록, 등록, 수정 화면 구현



BookStore

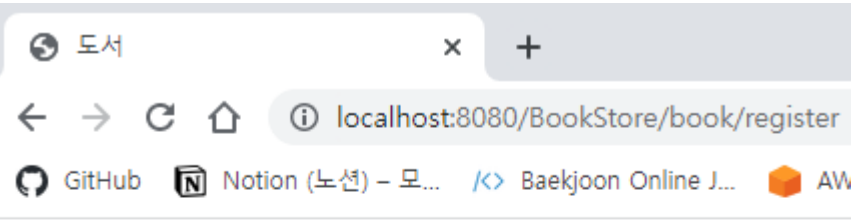
도서목록 고객목록



도서목록

처음으로 도서등록

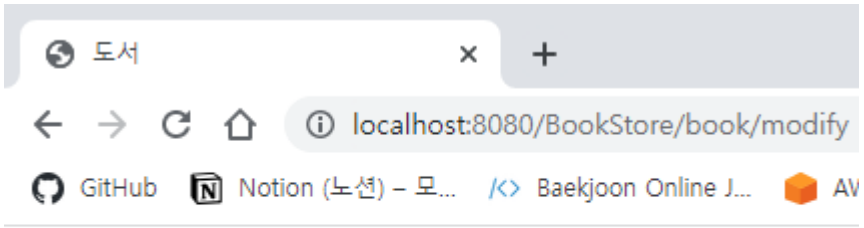
도서번호	도서명	출판사	가격	관리
1	축구의 역사	굿스포츠	7,000	수정 삭제



도서등록

처음으로 도서목록

도서명	
출판사	
가격	
	등록

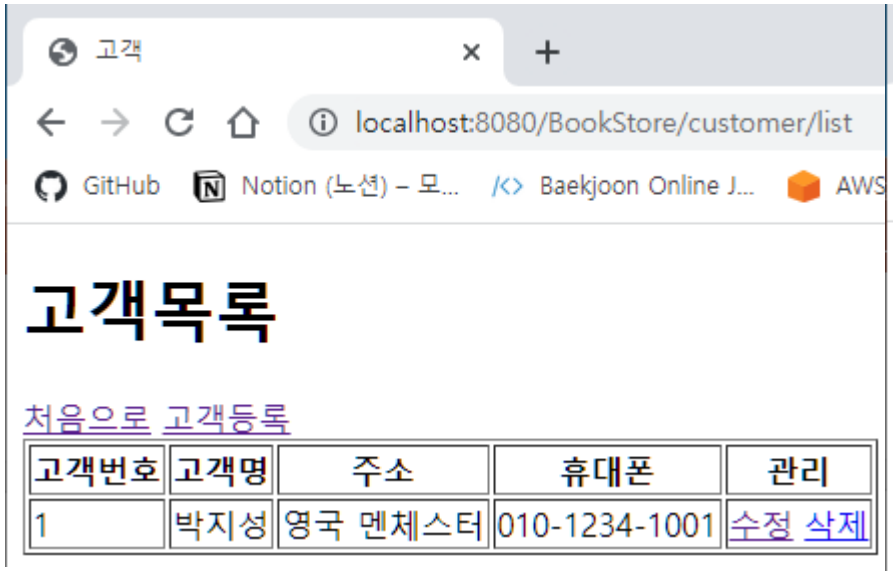


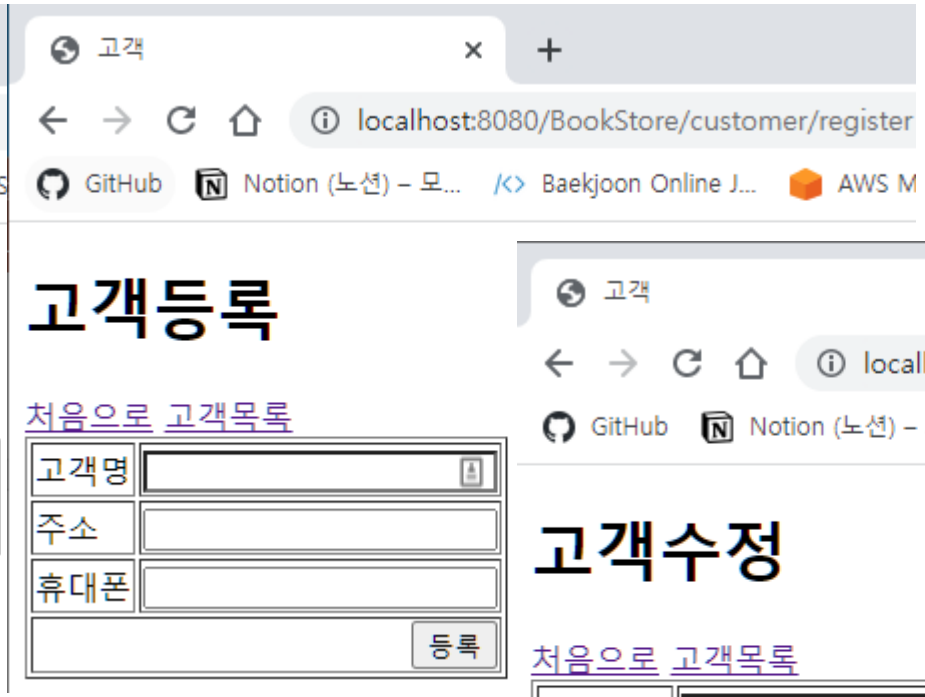
도서수정

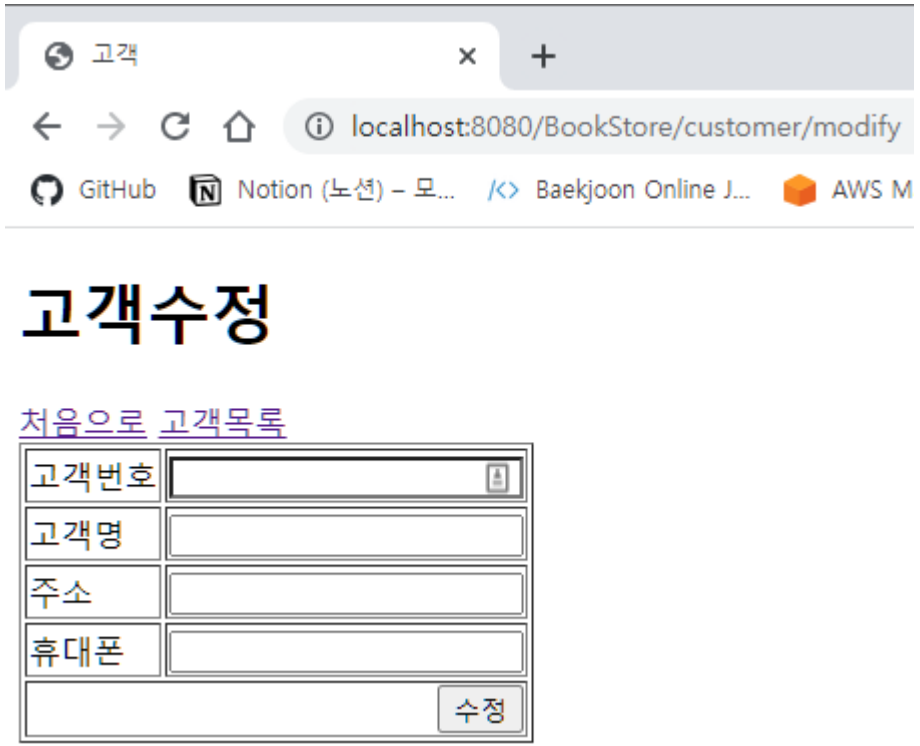
처음으로 도서목록

도서번호	
도서명	
출판사	
가격	
	수정

2) 화면 구현


고객
localhost:8080/BookStore/customer/list
GitHub
Notion (노션) – 모...
Baekjoon Online J...
AWS
고객목록
처음으로 고객등록
고객번호 고객명 주소 휴대폰 관리
1 박지성 영국 멘체스터 010-1234-1001 수정 삭제


고객
localhost:8080/BookStore/customer/register
GitHub
Notion (노션) – 모...
Baekjoon Online J...
AWS M
고객등록
처음으로 고객목록
고객명
주소
휴대폰
등록


고객
localhost:8080/BookStore/customer/modify
GitHub
Notion (노션) – 모...
Baekjoon Online J...
AWS M
고객수정
처음으로 고객목록
고객번호
고객명
주소
휴대폰
수정

3) 기능 구현

-기본 설정
- VO 설정
- book.xml 설정

application.properties

```
server.servlet.context-path=/BookStore
server.port=8080
spring.thymeleaf.cache=false

# Mybatis 설정
spring.datasource.url=jdbc:mysql://127.0.0.1:3306/java2_bookstore
spring.datasource.username=root
spring.datasource.password=1234
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

#MyBatis Mapper 경로설정 -> Ch06Application 클래스 상단에 @MapperScan("kr
mybatis.mapper-locations=classpath:mappers/**/*.xml
```

```
@Getter
@Setter
@AllArgsConstructor
@NoArgsConstructor
@ToString
public class BookVO {
    private int bookId;
    private String bookName;
    private String publisher;
    private int price;
}
```

```
@Getter
@Setter
@AllArgsConstructor
@NoArgsConstructor
@ToString
public class CustomerVO {
    private int custId;
    private String name;
    private String address;
    private String phone;
}
```

book.xml

```
-//mybatis.org//DTD Mapper 3.0//EN (doctype)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="book">
    <insert id="insertBook">
        INSERT INTO `book`(`bookName`,`publisher`,`price`) VALUES (#
    </insert>
    <select id="selectBook" resultType="kr.co.bookstore.VO.BookVO">
        SELECT * FROM `book` WHERE `bookId`=#{bookId};
    </select>
    <select id="selectBooks" resultType="kr.co.bookstore.VO.BookVO">
        SELECT * FROM `book`;
    </select>
    <update id="updateBook">
        UPDATE `book` SET
            `bookName`=#{bookName},
            `publisher`=#{publisher},
            `price`=#{price}
        WHERE `bookId`=#{bookId};
    </update>
    <delete id="deleteBook">
        DELETE FROM `book` WHERE `bookId`=#{bookId};
    </delete>
</mapper>
```

3) 기능 구현

- customer.xml 설정
- BookService 설정

customer.xml

```
-//mybatis.org//DTD Mapper 3.0//EN (doctype)
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "https://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="customer">
    <insert id="insertCustomer">
        INSERT INTO `customer`(`name`,`address`,`phone`) VALUES (#{name}
    </insert>
    <select id="selectCustomer" resultType="kr.co.bookstore.VO.BookVO">
        SELECT * FROM `customer` WHERE `custId`=#{custId};
    </select>
    <select id="selectCustomers" resultType="kr.co.bookstore.VO.BookVO">
        SELECT * FROM `customer`;
    </select>
    <update id="updateCustomer">
        UPDATE `customer` SET
            `name`=#{name},
            `address`=#{address},
            `phone`=#{phone}
        WHERE `custId`=#{custId};
    </update>
    <delete id="deleteCustomer">
        DELETE FROM `customer` WHERE `custId`=#{custId};
    </delete>
</mapper>
```

```
@Service
public class BookService {

    @Autowired
    private BookDAO dao;

    // service

    // create
    public void insertBook(BookVO vo) {
        dao.insertBook(vo);
    }

    // read
    public BookVO selectBook(int bookId) {
        return dao.selectBook(bookId);
    }
    public List<BookVO> selectBooks() {
        return dao.selectBooks();
    }

    // upload
    public void updateBook(BookVO vo) {
        dao.updateBook(vo);
    }

    // delete
    public void deleteBook(int bookId) {
        dao.deleteBook(bookId);
    }
}
```

3) 기능 구현

- BookDAO 설정
- BookController 설정

```java
@Repository
public class BookDAO {

    @Autowired
    private SqlSessionTemplate mybatis;

    // create
    public void insertBook(BookVO vo) {
        mybatis.insert("book.insertBook", vo);
    }

    // read
    public BookVO selectBook(int bookId) {
        return mybatis.selectOne("book.selectBook", bookId);
    }
    public List<BookVO> selectBooks() {
        return mybatis.selectList("book.selectBooks");
    }

    // upload
    public void updateBook(BookVO vo) {
        mybatis.update("book.updateBook", vo);
    }

    // delete
    public void deleteBook(int bookId) {
        mybatis.delete("book.deleteBook", bookId);
    }
}
```

BookController.java

```java
@Controller
public class BookController {

    @Autowired
    private BookService service;

    // list

    @GetMapping("/book/list")
    public String list(Model model) {
        List<BookVO> books = service.selectBooks();
        model.addAttribute("books", books);
        return "/book/list";
    }

    // reigster

    @GetMapping("/book/register")
    public String register(Model model) {
        BookVO book = new BookVO();
        model.addAttribute("book", book);
        return "/book/register";
    }

    @PostMapping("/book/register")
    public String register(BookVO vo) {
        service.insertBook(vo);
        return "redirect:/book/list";
    }
```

```java
// modify

@GetMapping("/book/modify")
public String modify(int bookId, Model model) {
    BookVO book = service.selectBook(bookId);
    model.addAttribute("book", book);
    return "/book/modify";
}

@PostMapping("/book/modify")
public String modify(BookVO vo) {
    service.updateBook(vo);
    return "redirect:/book/list";
}

// delete
@GetMapping("/book/delete")
public String delete(int bookId) {
    service.deleteBook(bookId);
    return "redirect:/book/list";
}
```

3) 기능 구현

- CustomerService 설정
- CustomerDAO 설정

```java
@Service
public class CustomerService {

    @Autowired
    private CustomerDAO dao;

    // service

    // create
    public void insertCustomer(CustomerVO vo) {
        dao.insertCustomer(vo);
    }

    // read
    public CustomerVO selectCustomer(int custId) {
        return dao.selectCustomer(custId);
    }
    public List<CustomerVO> selectCustomers() {
        return dao.selectCustomers();
    }

    // update
    public void updateCustomer(CustomerVO vo) {
        dao.updateCustomer(vo);
    }

    // delete
    public void deleteCustomer(int custId) {
        dao.deleteCustomer(custId);
    }
}
```

```java
@Repository
public class CustomerDAO {

    @Autowired
    private SqlSessionTemplate mybatis;

    // create
    public void insertCustomer(CustomerVO vo) {
        mybatis.insert("customer.insertCustomer", vo);
    }

    // read
    public CustomerVO selectCustomer(int custId) {
        return mybatis.selectOne("customer.selectCustomer", custId);
    }
    public List<CustomerVO> selectCustomers() {
        return mybatis.selectList("customer.selectCustomers");
    }

    // update
    public void updateCustomer(CustomerVO vo) {
        mybatis.update("customer.updateCustomer", vo);
    }

    // delete
    public void deleteCustomer(int custId) {
        mybatis.delete("customer.deleteCustomer", custId);
    }
}
```

3) 기능 구현

- CustomerController 설정
- Main페이지 설정

CustomerController.java

```java
@Controller
public class CustomerController {

    @Autowired
    private CustomerService service;

    // list

    @GetMapping("/customer/list")
    public String list(Model model) {
        List<CustomerVO> customers = service.selectCustomers();
        model.addAttribute("customers", customers);
        return "/customer/list";
    }

    // reigster

    @GetMapping("/customer/register")
    public String register(Model model) {
        CustomerVO customer = new CustomerVO();
        model.addAttribute("customer", customer);
        return "/customer/register";
    }

    @PostMapping("/customer/register")
    public String register(CustomerVO vo) {
        service.insertCustomer(vo);
        return "redirect:/customer/list";
    }
```

```java
// modify

@GetMapping("/customer/modify")
public String modify(int custId, Model model) {
    CustomerVO customer = service.selectCustomer(custId);
    model.addAttribute("customer", customer);
    return "/customer/modify";
}

@PostMapping("/customer/modify")
public String modify(CustomerVO vo) {
    service.updateCustomer(vo);
    return "redirect:/customer/list";
}

// delete
@GetMapping("/customer/delete")
public String delete(int custId) {
    service.deleteCustomer(custId);
    return "redirect:/customer/list";
}
```

```java
@Controller
public class MainController {

    @GetMapping(value = {"/","/index"})
    public String index() {
        return "/index";
    }

}
```

3) 기능 구현

- book/list
- book/modify

```
<body>
    <h1>도서목록</h1>

    <a th:href="@{../index}">처음으로</a>
    <a th:href="@{./register}">도서등록</a>

    <table border="1">
        <tr>
            <th>도서번호</th>
            <th>도서명</th>
            <th>출판사</th>
            <th>가격</th>
            <th>관리</th>
        </tr>
        <tr th:each="book:${books}">
            <td th:text="${book.bookId}">1</td>
            <td th:text="${book.bookName}">축구의 역사</td>
            <td th:text="${book.publisher}">굿스포츠</td>
            <td th:text="${book.price}">7,000</td>
            <td>
                <a th:href="@{./modify(bookId=${book.bookId})}">수정</a>
                <a th:href="@{./delete(bookId=${book.bookId})}">삭제</a>
            </td>
        </tr>
    </table>
</body>
```

```
<form th:action="@{./modify}" th:object="${book}" method="post">
    <table border="1">
        <tr>
            <td>도서번호</td>
            <td>
                <input type="number" th:field="*{bookId}" readonly="readonly">
            </td>
        </tr>
        <tr>
            <td>도서명</td>
            <td><input type="text" th:field="*{bookName}"></td>
        </tr>
        <tr>
            <td>출판사</td>
            <td><input type="text" th:field="*{publisher}"></td>
        </tr>
        <tr>
            <td>가격</td>
            <td><input type="number" th:field="*{price}"></td>
        </tr>
        <tr>
            <td colspan="2" align="right"><input type="submit" value="수정"></td>
        </tr>
    </table>
</form>
```

3) 기능 구현

- book/register
- customer/list

```
<form th:action="@{/book/register}" th:object="${book}" method="post">
    <table border="1">
        <tr>
            <td>도서명</td>
            <td><input type="text" th:field="*{bookName}"></td>
        </tr>
        <tr>
            <td>출판사</td>
            <td><input type="text" th:field="*{publisher}"></td>
        </tr>
        <tr>
            <td>가격</td>
            <td><input type="number" th:field="*{price}"></td>
        </tr>
        <tr>
            <td colspan="2" align="right"><input type="submit" value="등록"></td>
        </tr>
    </table>
</form>
```

```
<h1>고객목록</h1>

<a th:href="@{../index}">처음으로</a>
<a th:href="@{./register}">고객등록</a>

<table border="1">
    <tr>
        <th>고객번호</th>
        <th>고객명</th>
        <th>주소</th>
        <th>휴대폰</th>
        <th>관리</th>
    </tr>
    <tr th:each="customer:${customers}">
        <td th:text="${customer.custId}">1</td>
        <td th:text="${customer.name}">박지성</td>
        <td th:text="${customer.address}">영국 멘체스터</td>
        <td th:text="${customer.phone}">010-1234-1001</td>
        <td>
            <a th:href="@{./modify(custId=${customer.custId})}">수정</a>
            <a th:href="@{./delete(custId=${customer.custId})}">삭제</a>
        </td>
    </tr>
</table>
```

3) 기능 구현

- customer/modify
- customer/register

```
<form th:action="@{./modify}" th:object="${customer}" method="post">
    <table border="1">
        <tr>
            <td>고객번호</td>
            <td>
                <input type="text" th:field="*{custId}" readonly="readonly">
            </td>
        </tr>
        <tr>
            <td>고객명</td>
            <td><input type="text" th:field="*{name}"></td>
        </tr>
        <tr>
            <td>주소</td>
            <td><input type="text" th:field="*{address}"></td>
        </tr>
        <tr>
            <td>휴대폰</td>
            <td><input type="text" th:field="*{phone}"></td>
        </tr>
        <tr>
            <td colspan="2" align="right"><input type="submit" value="수정"></td>
        </tr>
    </table>
</form>
```

```
<form th:action="@{./register}" th:object="${customer}" method="post">
    <table border="1">
        <tr>
            <td>고객명</td>
            <td><input type="text" th:field="*{name}"></td>
        </tr>
        <tr>
            <td>주소</td>
            <td><input type="text" th:field="*{address}"></td>
        </tr>
        <tr>
            <td>휴대폰</td>
            <td><input type="text" th:field="*{phone}"></td>
        </tr>
        <tr>
            <td colspan="2" align="right"><input type="submit" value="등록"></td>
        </tr>
    </table>
</form>
```

4) 실행

- 메인 페이지
- 도서 목록, 등록, 수정, 삭제 기능 동작 화면

도서목록

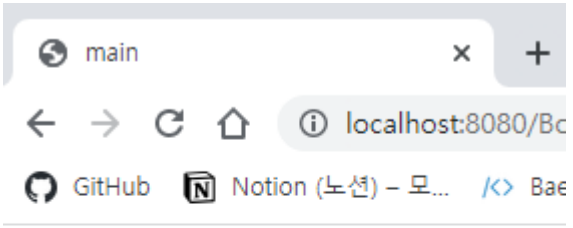
처음으로 도서등록

도서번호	도서명	출판사	가격	관리
1	축구의 역사	굿스포츠	7000	수정 삭제
2	축구아는 여자	나무수	13000	수정 삭제
3	축구의 이해	대한미디어	22000	수정 삭제
4	골프 바이블	대한미디어	35000	수정 삭제
5	피겨 교본	굿스포츠	8000	수정 삭제
6	역도 단계별기술	굿스포츠	6000	수정 삭제
7	야구의 추억	이상미디어	20000	수정 삭제
8	야구를 부탁해	이상미디어	13000	수정 삭제
9	올림픽 이야기	삼성당	7500	수정 삭제
10	Olympic Champions	Pearson	13000	수정 삭제
11	스포츠의학	한솔의학서적	0	수정 삭제
13	도서이름2	출판사2	20000	수정 삭제

도서등록

처음으로 도서목록

도서명	도서이름1
출판사	출판사1
가격	10000
	등록



BookStore

도서목록 고객목록

11	스포츠의학	한솔의학서적	0	수정 삭제
13	도서이름2	출판사2	20000	수정 삭제
15	도서이름1	출판사1	10000	수정 삭제

도서수정

처음으로 도서목록

도서번호	15
도서명	도서이름3
출판사	출판사3
가격	15000
	수정

11	스포츠의학	한솔의학서적	0	수정 삭제
13	도서이름2	출판사2	20000	수정 삭제
15	도서이름3	출판사3	15000	수정 삭제

10	Olympic Champions	Pearson	13000	수정 삭제
11	스포츠의학	한솔의학서적	0	수정 삭제
15	도서이름3	출판사3	15000	수정 삭제

4) 실행

- 고객 목록, 등록, 수정, 삭제 기능 동작 화면

고객목록

처음으로 고객등록

고객번호	고객명	주소	휴대폰	관리
1	박지성	영국 맨체스타	000-5000-0001	수정 삭제
2	김연아	대한민국 서울	000-6000-0001	수정 삭제
3	장미란	대한민국 강원도	000-7000-0001	수정 삭제
4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
8	사람이름2	마산	010-2234-5678	수정 삭제

고객등록

처음으로 고객목록

고객명	사람이름1
주소	창원시
휴대폰	010-1234-5678
	등록

4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
8	사람이름2	마산	010-2234-5678	수정 삭제
11	사람이름1	창원시	010-1234-5678	수정 삭제

고객수정

처음으로 고객목록

고객번호	11
고객명	사람이름3
주소	부산시
휴대폰	010-4321-8765
	수정

4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
8	사람이름2	마산	010-2234-5678	수정 삭제
11	사람이름3	부산시	010-4321-8765	수정 삭제

3	장미란	대한민국 강원도	000-7000-0001	수정 삭제
4	추신수	미국 클리블랜드	000-8000-0001	수정 삭제
11	사람이름3	부산시	010-4321-8765	수정 삭제