14강 계좌 상세보기

1. MyBatis 매퍼 만들기

account.xml

```
<select id="findByIdWithUser" resultType="shop.mtcoding.bankapp.dto.account.AccountDetailRespDto">
    select at.id, at.number, at.balance, at.user_id, (select fullname from user_tb ut where ut.id = at.user_id) fullname
    from account_tb at
    where at.id = #{id}
</select>
```

history.xml

```
<select id="findByGubun" resultType="shop.mtcoding.bankapp.dto.history.HistoryRespDto">
    <if test="gubun == 'withdraw'">
        SELECT ht.id, ht.amount, ht.w_balance balance, wat.number sender, 'ATM' receiver, ht.created_at
        FROM history_tb ht
        INNER JOIN account_tb wat
        ON ht.w_account_id = wat.id
        WHERE ht.w_account_id = #{accountId}
    </if>

    <if test="gubun == 'deposit'">
        SELECT ht.id, ht.amount, ht.d_balance balance, 'ATM' sender, dat.number receiver, ht.created_at
        FROM history_tb ht
        INNER JOIN account_tb dat
        ON ht.d_account_id = dat.id
        WHERE ht.d_account_id = #{accountId}
    </if>

    <if test="gubun == 'all'">
        SELECT
        ht.id,
        ht.amount,
        nvl(wat.number, 'ATM') sender,
        nvl(dat.number, 'ATM') receiver,
        case when ht.w_account_id = #{accountId} then ht.w_balance
             when ht.d_account_id = #{accountId} then ht.d_balance
             end as balance,
        ht.created_at
        FROM history_tb ht
        LEFT OUTER JOIN account_tb wat
        ON ht.w_account_id = wat.id
        LEFT OUTER JOIN account_tb dat
        ON ht.d_account_id = dat.id
        WHERE ht.w_account_id = #{accountId} or ht.d_account_id = #{accountId};
    </if>
</select>
```

2. 응답 DTO 만들기

```
package shop.mtcoding.bankapp.dto.account;

import lombok.Getter;
import lombok.Setter;

@Setter
@Getter
public class AccountDetailRespDto {
    private Integer id;
    private String number;
    private Long balance;
    private Integer userId;
    private String fullname;
}
```

```
package shop.mtcoding.bankapp.dto.history;

import java.sql.Timestamp;

import lombok.Getter;
import lombok.Setter;

@Setter
@Getter
public class HistoryRespDto {
    private Integer id;
    private Long amount;
    private Long balance;
    private String sender;
    private String receiver;
    private Timestamp createdAt;
}
```

3. Repository 만들기

AccountRepository.java

```
public AccountDetailRespDto findByIdWithUser(int id);
```

HistoryRepository.java

```
public List<HistoryRespDto> findByGubun(@Param("gubun") String gubun,
@Param("accountId") int accountId);
```

4. Controller 만들기

AccountController.java

```
@GetMapping("/account/{id}")
public String detail(@PathVariable int id, @RequestParam(name = "gubun",
defaultValue = "all") String gubun,
        Model model) {
    // 1. 인증체크
    User principal = (User) session.getAttribute("principal");
    if (principal == null) {
        return "redirect:/loginForm";
    }

    // 2. 레파지토리 호출 (메서드를 3개 or 마이바티스 동적쿼리)
    AccountDetailRespDto aDto = accountRepository.findByIdWithUser(id);
    if (aDto.getUserId() != principal.getId()) {
        throw new CustomException("해당 계좌를 볼 수 있는 권한이 없습니다",
HttpStatus.FORBIDDEN);
    }
    List<HistoryRespDto> hDtoList = historyRepository.findByGubun(gubun, id);

    model.addAttribute("aDto", aDto);
    model.addAttribute("hDtoList", hDtoList);

    return "account/detail";
}
```

5. JSP 뷰 완성하기

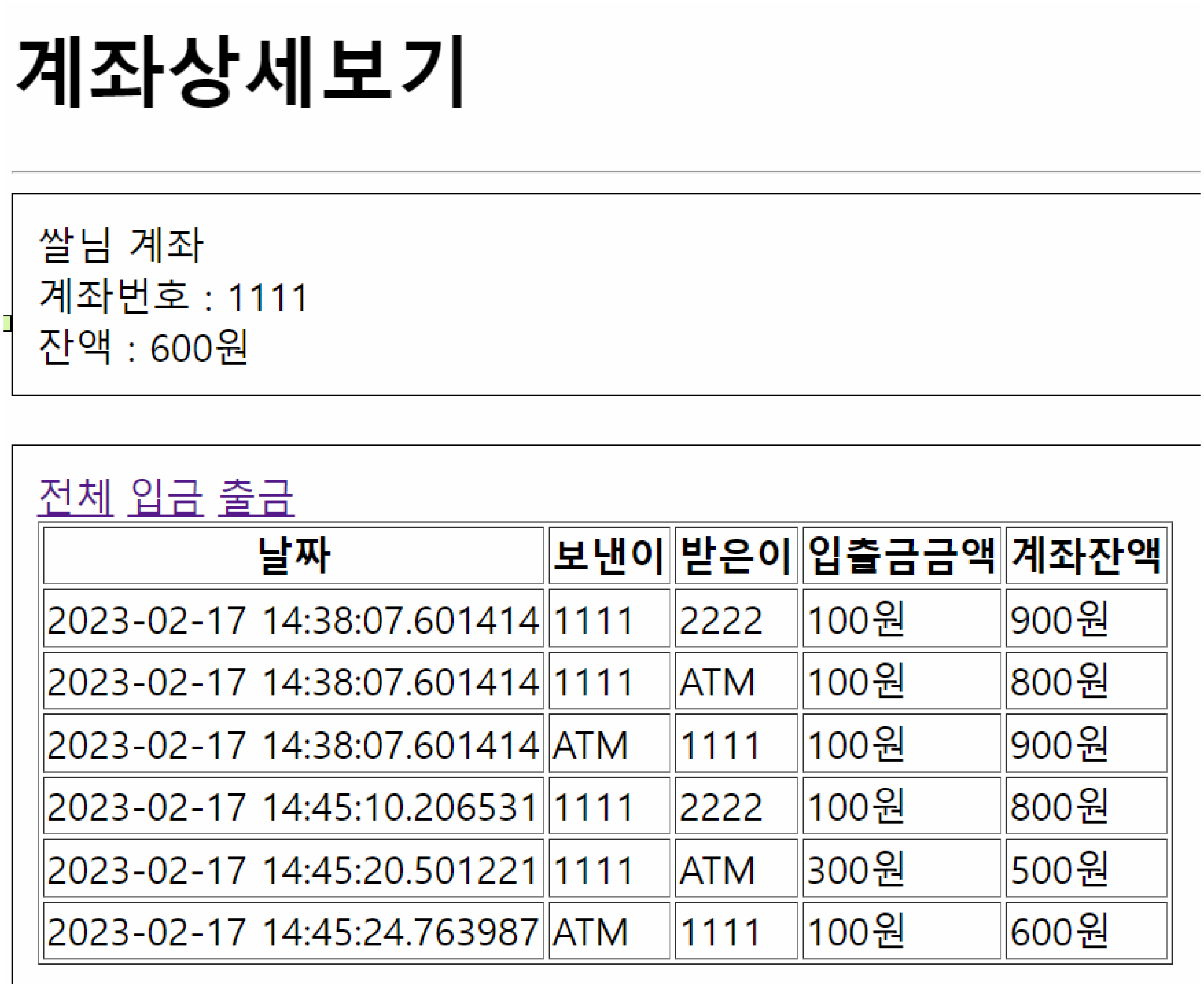
```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"
%>
<%@include file="../layout/header.jsp" %>

<h1>계좌상세보기</h1>
<hr />
<div class="user-box">
    ${aDto.fullname}님 계좌<br />
    계좌번호 : ${aDto.number}<br />
    잔액 : ${aDto.balance}원
</div>
<div class="list-box">
    <a href="/account/${aDto.id}?gubun=all">전체</a>
    <a href="/account/${aDto.id}?gubun=deposit">입금</a>
    <a href="/account/${aDto.id}?gubun=withdraw">출금</a>
    <br />
    <table border="1">
        <thead>
            <tr>
                <th>날짜</th>
                <th>보낸이</th>
                <th>받은이</th>
                <th>입출금금액</th>
                <th>계좌잔액</th>
            </tr>
        </thead>
        <tbody>
            <c:forEach items="${hDtoList}" var="history">
                <tr>
                    <td>${history.createdAt}</td>
                    <td>${history.sender}</td>
                    <td>${history.receiver}</td>
                    <td>${history.amount}원</td>
                    <td>${history.balance}원</td>
                </tr>
            </c:forEach>

        </tbody>
    </table>
</div>

</body>
</html>
```

6. 실행 (테스트)

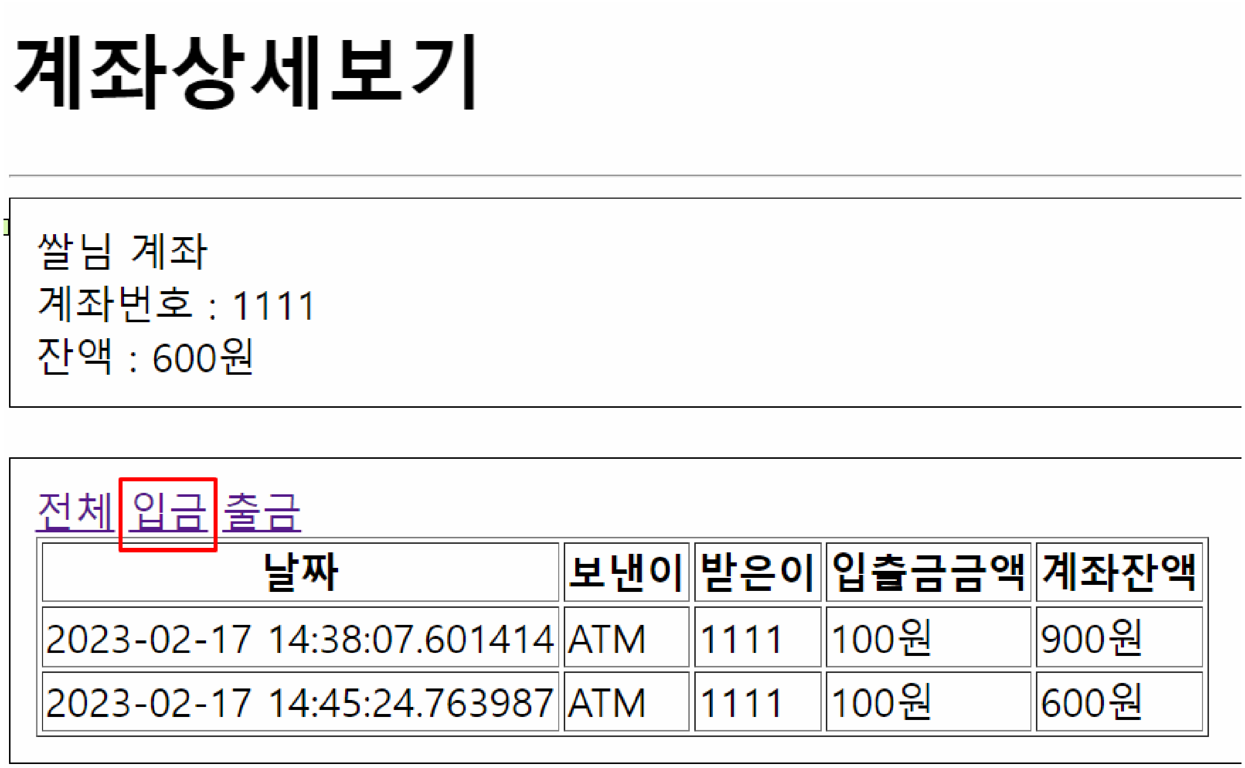


계좌상세보기

쌀님 계좌
계좌번호 : 1111
잔액 : 600원

전체 입금 출금

날짜	보낸이	받은이	입출금금액	계좌잔액
2023-02-17 14:38:07.601414	1111	2222	100원	900원
2023-02-17 14:38:07.601414	1111	ATM	100원	800원
2023-02-17 14:38:07.601414	ATM	1111	100원	900원
2023-02-17 14:45:10.206531	1111	2222	100원	800원
2023-02-17 14:45:20.501221	1111	ATM	300원	500원
2023-02-17 14:45:24.763987	ATM	1111	100원	600원

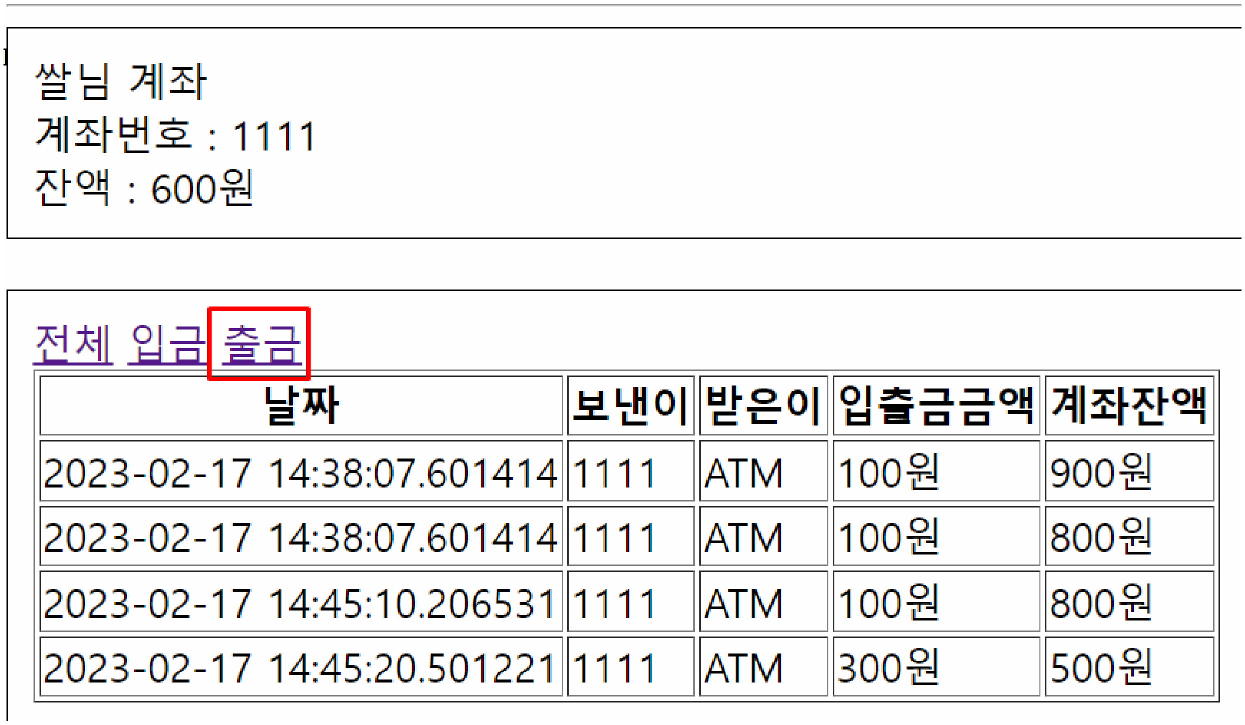


계좌상세보기

쌀님 계좌
계좌번호 : 1111
잔액 : 600원

전체 입금 출금

날짜	보낸이	받은이	입출금금액	계좌잔액
2023-02-17 14:38:07.601414	ATM	1111	100원	900원
2023-02-17 14:45:24.763987	ATM	1111	100원	600원



쌀님 계좌
계좌번호 : 1111
잔액 : 600원

날짜	보낸이	받은이	입출금금액	계좌잔액
2023-02-17 14:38:07.601414	1111	ATM	100원	900원
2023-02-17 14:38:07.601414	1111	ATM	100원	800원
2023-02-17 14:45:10.206531	1111	ATM	100원	800원
2023-02-17 14:45:20.501221	1111	ATM	300원	500원