

4) 전체 구조도 (System Architecture)

① HTML/CSS – JSP – DB 데이터 흐름도 (Data Flow)

데이터는 **Client(브라우저) ↔ Server(Tomcat/JSP) ↔ Database(MySQL)** 순서로 흐릅니다.

1. Client (View & Event):

- **UI 렌더링:** HTML5와 CSS3(Flexbox, Variables)를 사용하여 반응형 레이아웃과 다크/라이트 모드 테마를 제공합니다.
- **사용자 인터랙션:** script.js가 버튼 클릭, 드래그 앤 드롭, 키보드 입력(검색) 이벤트를 감지합니다.
- **데이터 전송:**
 - **Form Submit:** 로그인, 할 일 추가 시 사용 (POST 방식).
 - **AJAX (Fetch API):** 화면 깜빡임 없이 처리해야 하는 순서 변경(**Reorder**), 메모 저장, 완료 항목 일괄 삭제 요청 시 비동기 통신을 사용합니다.
 - **DOM 조작:** 실시간 검색(**Filtering**) 시 서버 요청 없이 자바스크립트로 즉시 요소를 숨기거나 보여줍니다(display: none/flex).

2. Server (Controller & Logic):

- **Session Management:** index.jsp 진입 시 session.getAttribute("userID")를 확인하여 비로그인 사용자를 차단(Redirect)합니다.
- **Action Handling (todoAction.jsp):** 클라이언트의 요청(action 파라미터: insert, delete, updateStatus, reorder, updateMemo, clearDone)을 분기 처리합니다.
- **JDBC:** DBConnection.java를 통해 MySQL과 연결하고 PreparedStatement로 SQL을 실행합니다.

3. Database (Storage):

- **MySQL:** 회원 정보와 투두 데이터를 영구 저장하며, ON DELETE CASCADE를 통해 참조 무결성을 유지합니다.

② 페이지 연결 흐름도 (Page Sitemap)

- **진입 (Entry):** login.jsp
 - └─ 회원가입: register.jsp → (성공 시) login.jsp
 - └─ 로그인 성공: index.jsp (메인 페이지)
- **메인 로직 (index.jsp ↔ todoAction.jsp):**
 - └─ 할 일 추가: todoAction.jsp?action=insert → index.jsp (새로고침)
 - └─ 상태 변경/삭제: todoAction.jsp → index.jsp
 - └─ 순서 변경/메모: todoAction.jsp (AJAX 백그라운드 처리)
 - └─ 완료 일괄 삭제: todoAction.jsp?action=clearDone → index.jsp

- 종료: logoutAction.jsp (세션 만료) → login.jsp

③ 메인 UI 구조 (Main UI Structure)

사용자 경험(UX)을 고려하여 상단 정보 - 도구 모음 - 목록 순으로 배치하였습니다.

구역 (Area)	구성 요소 (Components)	주요 특징 (Features)
Header	환영 문구, 테마 토글 스위치, 로그아웃	localStorage를 이용해 다크 모드 설정을 유지함.
Input	할 일 입력창, Add 버튼	autofocus 속성으로 페이지 로드 시 즉시 입력 가능.
Toolbar	검색창, 검색 버튼, 완료 삭제 버튼	검색 시 입력창 위치가 고정(Sticky)되어 UI 흔들림 방지.
List	할 일 목록 (), 기능 버튼 그룹	드래그 앤 드롭 핸들링, 메모장(Toggle), 수정/삭제/체크 버튼.

5) DB 설계 (Database Design)

회원별로 독립적인 할 일 관리가 가능하도록 1:N 관계로 설계하였으며, 드래그 앤 드롭 순서 저장을 위한 컬럼을 포함합니다.

① 테이블 구조 (ERD & Table Schema)

- Database: MySQL (todo_db)

- **Charset:** utf8mb4 (이모지 😊 지원)

1. USERS 테이블 (사용자 정보)

컬럼명 (Column)	데이터 타입	키 (Key)	Null 여부	설명 (Description)
username	VARCHAR(50)	PK	NN	사용자 아이디 (고유 식별자)
password	VARCHAR(50)		NN	사용자 비밀번호

2. TODOS 테이블 (할 일 데이터)

컬럼명 (Column)	데이터 타입	키 (Key)	기본값 (Default)	설명 (Description)
id	INT	PK	Auto Inc	할 일 고유 번호 (자동 증가)
username	VARCHAR(50)	FK	NN	작성자 ID (users 테이블 참조)
content	VARCHAR(200)		NN	할 일 내용
status	ENUM		'TODO'	진행 상태 ('TODO' 또는 'DONE')

seq	INT		0	드래그 정렬 순서 (낮을수록 상단 배치)
memo	TEXT		NULL	상세 메모 (길이 제한 없는 텍스트)
created_at	TIMESTAMP		NOW()	생성 일시

(NN: Not Null, PK: Primary Key, FK: Foreign Key)

② 주요 컬럼 및 제약조건 설명

1. **seq** (순서 컬럼):

- 용도: 사용자가 드래그 앤 드롭으로 재정렬한 순서를 DB에 영구 저장하기 위해 사용합니다.
- 동작: 조회 시 ORDER BY seq ASC, id ASC로 쿼리하여 사용자가 지정한 순서대로 출력합니다.

2. **status** (상태 컬럼):

- 타입: ENUM('TODO', 'DONE')을 사용하여 정해진 값 외에 다른 데이터가 들어오는 것을 원천 차단하여 데이터 무결성을 높였습니다.

3. **memo** (메모 컬럼):

- 타입: TEXT 타입을 사용하여 긴 내용의 메모도 잘리지 않고 저장할 수 있도록 설계했습니다.

4. **Foreign Key (username) & Cascade:**

- 제약조건: FOREIGN KEY (username) REFERENCES users(username) ON DELETE CASCADE
- 설명: users 테이블의 username을 참조합니다. **ON DELETE CASCADE** 옵션을 적용하여, 회원이 탈퇴(삭제)할 경우 해당 회원이 작성한 모든 할 일 데이터도 자동으로 삭제되도록 설계하였습니다. 이는 고아 레코드(Orphan Data)가 남는 것을 방지합니다.