**사원 이혜균 Q & A Board 결과 문서**

1. **Database**

DB Table – BOARD(게시판 본문 글)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 속성 명 | COUNT | TITLE | NAME | CATEGORY | CONTENTS | INPUTDAY | INPUTPW |
| 속성 타입 | Int | Text | Text | Text | Text | Text | Text |
| 기타 | PK, Not null | Not null | Not null | Not null | Not null | Not null | FK, Not null |

DB Table – COMMENT(댓글)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 속성 명 | COUNT | TITLE | NAME | CONTENTS | INPUTDAY | INPUTPW | REFCNT |
| 속성 타입 | Int | Text | Text | Text | Text | Text | Int |
| 기타 | PK, Not null | Not null | Not null | Not null | Not null | Not null | FK, Not null |

* 처음 DB 설계에서 여러 속성이 증가 했습니다. 우선 BOARD에는 CATEGORY, INPUTDAY, INPUTPW가 추가가 되었으며 각각 글의 카테고리, 쓴 날짜 또는 수정 날짜, 각 글의 비밀번호를 의미합니다. 또한 COMMENT 역시 INPUTDAY와 INPUTPW가 새롭게 삽입 되었습니다. 또한 SQLite는 외래키 개념이 존재하지 않아서 완벽한 외래키 기능을 구현하지 못했습니다.

1. **주요 기능**

**2-1. 기존 기획서 내 기능.**

* 게시물 삽입 : HTML, Jquery를 이용해서 Textbox나 Select Box에 있는 속성들을 받아서 DB에 저장.
* 게시물 수정 : 값을 받는 것은 삽입과 동일, Update query 사용.
* 게시물 삭제 : 해당 게시물 삭제.
* 댓글 창 열기 : location.search, parsing을 통해 본문의 COUNT(댓글 : REFCNT)를 받고 새 창을 염.
* 댓글 삽입 & 수정 & 삭제 : 게시물과 동일하나 where REFCNT=? 이용.
* CSS & Jquery를 이용한 User Interface 디자인.

**2-2. 추가된 기능**

* 검색 기능 : Option값과 Text를 받아서 검색. Select 쿼리에 ‘where Option=Text 을 사용.’
* 카테고리 기능 : 질문하는 글에 대한 큰 주제. 카테고리에 따라 검색이 가능하며 입력을 할 때 select box에 있는 카테고리 영역을 선택해야 함.
* 페이징 기능 : 모든 게시물을 한번에 보여줄 수 없기 때문에 10개 단위로 게시물을 나누어서 사용자에게 제공을 함.(완변히 구현하지 못함.)
* 본문 창 & 댓글 창 디자인 : CSS와 Jquery를 이용하여 디자인 삽입.

1. **보충 사항.**

우선 가장 아쉬운 것은 페이징 기능이 미완성이라는 것 입니다. 현재 완성이 된 것은 모든 select 결과물을 보여주는 것이 아닌 10개 단위로 나누어서 보여주며 이것에 따른 페이지 개수 구현까지만 되어 있습니다. 그러나 추가로 페이지 블록 단위로 앞으로 가는 기능이나 뒤로 가는 기능까지 구현이 되어야 모든 기능이 제대로 구현이 되어야 하는데 블록 단위의 동적 생성 알고리즘을 제대로 이해하지 못해서 구현을 하지 못했습니다.

다음으로 잘 안된 것은 작성한 소스코드의 가독성이 많이 떨어진다는 것 입니다. 중복된 소스 코드가 많많 쓸 때 없는 코드 역시 많았습니다. 소스 코드를 체계적으로 구현을 한 것이 아니라 즉흥적으로 코드를 작성하고 삽입을 시켰기 때문에 코드를 분석하는데 있어서 매우 힘이 듭니다. 이러한 것은 지금처럼 규모가 작은 작업일 경우에는 크게 상관이 없지만 규모가 조금만 커져도 큰 문제가 될 것으로 생각이 됩니다.

1. **이슈사항**

* **innerHTML vs html()**

코드를 살펴보면 동적으로 html 코드를 추가하는 부분에 있어서 Jquery의 html() 메소드가 아니라 innerHTML을 사용했음을 볼 수 있습니다. 두 개의 기능은 html 코드를 추가한다는 것에서 동일합니다.

제가 innerHTML을 사용한 이유는 바로 속도의 차이 때문입니다. 결론부터 언급을 한다면 innerHTML보다 html()의 수행 속도가 더 빠릅니다. 그 이유는 innerHTML의 랜더링 속도가 html() 메소드보다 더 빠르기 때문입니다. 성능 테스트를 한 홈페이지에 대해서는 아래에 링크를 걸었습니다.

링크 : <http://jsperf.com/innerhtml-vs-jquery-html-vs-jquery-append>

1. **마무리**

HTML, Javascript, CSS, Jquery 등을 새롭게 배울 수 있는 기회였습니다. 특히 CSS와 Jquery는 많이 알지 못했는데 이번 기회를 통해서 이 언어들의 편의성 등을 알게 되었습니다. 또한 HTML과 Javascript 역시 미처 알지 못했던 것을 다시 알 수 있게 되어서 좋았습니다.