|  |
| --- |
| TaskQoS FAQ 프로젝트 인수인계 문서 |
| |  |  | | --- | --- | | **작 성 자** | **구 민 주** | | **작성기간** | **2025.03.24~2024.03.27.** | |
|  |
|  |

The information contained is proprietary and confidential.

Do not duplicate without permission.

NSTEK Technologies Ltd. NSTEK Technologies Inc

Junbgu bldg 5F, 329, Yeongdong-daero, Gangnam-gu,

Seoul, Korea NSTEK Inc

TEL : +82)02.6207.0880 Fax : 82)02.6207.0888

Preface

The 21st century has become an era where a digital network can overcome time and space.

Our current environment is challenging us with a new paradigm and with help of the Internet,

we’re one step closer to reach the goal.

We are making network and Internet based solutions that are high in portability and scalability.

As a young start-up company, NSTEK will not hesitate to face the difficult challenges.

Our belief that the most profound solution is in a human mind will carry us through the 21st century.

NSTEK Technologies Ltd.

5F Jung Bu building, 329 Young-Dong Street, Kang Nam-GU, Seoul, Korea NSTEK Inc.

TEL: + 82)02.6207.0880 Fax: + 82)02.6207.0888

About this Document

This application note is what maps out the \_\_\_ Development Blueprint for replacing \_\_\_

on Traffic Management System.

Related Documents

Reserved

This Document

This following is a brief description of the contents of each section:

Reserved

Table of Contents

[1. 프로젝트 개요 4](#_Toc193989187)

[1.1. 프로젝트 주요 목적 4](#_Toc193989188)

[1.2. 주요 기능 4](#_Toc193989189)

[1.3. 사용 언어 4](#_Toc193989190)

[1.4. 프로젝트 경로 및 문서 위치 4](#_Toc193989191)

[1.5. DB 5](#_Toc193989192)

[1.5.1. 챕터, 서브챕터, 섹션, 서브섹션 ERD 5](#_Toc193989193)

[1.5.2. ＤＢ　출력　화면　및　구성　요소 6](#_Toc193989194)

[1.5.3. 테이블 설명 6](#_Toc193989195)

[1.5.4. 버전 관리 시스템 7](#_Toc193989196)

[2. 시스템 구조 및 파일 설명 8](#_Toc193989197)

[2.1. 전체 폴더 구조 8](#_Toc193989198)

[2.2. 주요 파일 및 폴더 상세 설명 9](#_Toc193989199)

[3. 주요 기능 및 코드 분석 12](#_Toc193989200)

[3.1. 버전 관리 시스템 12](#_Toc193989201)

[3.1.1. 버전 관리 핵심 필드 12](#_Toc193989202)

[3.1.2. 버전 UI 및 사용자 인터페이스 13](#_Toc193989203)

[3.1.3. 버전 처리 플로우 14](#_Toc193989204)

[3.1.4. 버전 관리 고려사항 및 제한사항 15](#_Toc193989205)

[3.2. 계층적 구조 관리 16](#_Toc193989206)

[3.2.1. 4단계 구조 설계 16](#_Toc193989207)

# 프로젝트 개요

## 프로젝트 주요 목적

* 계층적 구조(챕터-서브챕터-섹션-서브섹션)를 가진 FAQ  고객응대 페이지 관리
* 버전 관리를 통한 FAQ 콘텐츠의 이력 추적 및 관리
* 관리자가 FAQ 내용을 쉽게 생성, 수정, 삭제할 수 있는 인터페이스 제공

## 주요 기능

1. **콘텐츠 관리**

* FAQ 항목 생성/수정/삭제
* 계층적 구조(챕터-서브챕터-섹션-서브섹션) 관리
* WYSIWYG 에디터를 통한 내용 편집

1. **버전 관리**

* FAQ 콘텐츠의 버전별 관리
* 이전 버전으로의 롤백 가능
* 버전 간 변경 이력 추적

1. **구조화된 데이터 관리**

* 체계적인 카테고리 구조
* 직관적인 사용자 인터페이스
* 데이터베이스 기반의 안정적인 저장 시스템

## 사용 언어

* 프론트엔드: JavaScript, HTML5, CSS3
* 백엔드: PHP 7.x
* 데이터베이스: MySQL
* 캐싱 시스템: Redis
* 데이터 형식: JSON, CSV

## 프로젝트 경로 및 문서 위치

* 프로젝트 URL : http://fuelcelldr.nstek.com:11180/FAQ/
* 프로젝트 서버 경로 : /home/nstek/h2\_system/patch\_active/FAQ
* 인수인계 문서 위치 : /home/nstek/h2\_system/patch\_active/FAQ/doc

## DB

### 챕터, 서브챕터, 섹션, 서브섹션 ERD

텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

### ＤＢ　출력　화면　및　구성　요소

* 챔터　목록（클릭시　서브챕터와　서브챕터의　버전　나옴）

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

* 각　섹션　안　컨텐츠



### 테이블 설명

데이터베이스는 계층적 구조로 설계되어 있으며, 4개의 주요 테이블로 구성

1. **CHAPTERS (대분류)**

* CHAP\_ID: 기본키, 자동 증가
* CHAP\_NAME: 챕터 이름
* 최상위 카테고리로서 FAQ의 대분류를 관리

1. **SUBCHAPTERS (중분류)**

* SUB\_CHAP\_ID: 기본키, 자동 증가
* CHAP\_ID: CHAPTERS 테이블 참조 (외래키)
* POSITIONS: 서브챕터의 위치 정보
* VERSIONS: 버전 관리 번호
* SUB\_CHAP\_NAME: 서브챕터 이름
* SUB\_CHAP\_DESC: 서브챕터 설명

1. **SECTIONS (소분류)**

* SEC\_ID: 기본키, 자동 증가
* CHAP\_ID: CHAPTERS 테이블 참조 (외래키)
* SUB\_CHAP\_ID: SUBCHAPTERS 테이블 참조 (외래키)
* SEC\_NAME: 섹션 이름
* SEC\_DESC: 섹션 설명
* VERSION: 버전 정보

1. **SUBSECTIONS (실제 콘텐츠)**

* SUB\_SEC\_ID: 기본키, 자동 증가
* SEC\_ID: SECTIONS 테이블 참조 (외래키)
* SUB\_SEC\_NAME: 서브섹션 이름

1. SUB\_SEC\_CONTENT: FAQ 실제 내용

* VERSION: 버전 정보

### 버전 관리 시스템

1. **버전 관리 구조**
2. SUBCHAPTERS 테이블의 VERSIONS
3. SECTIONS 테이블의 VERSION
4. SUBSECTIONS 테이블의 VERSION
5. 이 세 필드가 연동되어 전체 FAQ의 버전을 관리
6. **버전 관리 방식**
7. 새로운 버전 생성 시, 기존 데이터를 복사하여 새로운 버전 번호 부여
8. 각 계층(SUBCHAPTERS, SECTIONS, SUBSECTIONS)의 버전이 동기화되어 관리

# 시스템 구조 및 파일 설명

## 전체 폴더 구조

├── **/home/nstek/h2\_system/patch\_active/FAQ**

├── index.html # FAQ 메인 페이지

├── nav.php # 네비게이션 컴포넌트

├── faq.css # FAQ 스타일시트

├── faq.js # FAQ 핵심 관리 모듈

├── faq\_form.php # FAQ 작성/수정 폼

├── faq\_content.js # FAQ 콘텐츠 관리 모듈

├── faq\_submit.php # FAQ 데이터 저장 및 새 버전 생성

├── faq\_delete.php # FAQ 삭제 처리

├── debug.php # 디버깅파일

├── **데이터 관리**

├── faq\_db.php # DB 연결 및 공통 함수

├── faq\_content.php # FAQ 목록 관리

├── faq\_view.php # FAQ 상세 조회

├── faq\_update.php # FAQ 업데이트 처리

├── fetch\_sections.php # 섹션 데이터 조회 API

├── fetch\_subchapters.php # 하위 챕터 데이터 조회 API

├── fetch\_subsections.php # 하위 섹션 데이터 조회 API

├── get\_faq\_view.php # FAQ 상세 내용 조회 API

├── **에디터**

├── editor/

│ ├── faq\_editor.js # SunEditor 설정 및 이미지 업로드 관리

│ ├── img\_upload.php # 이미지 업로드 처리

├── **스타일**

│ ├── css/ # 스타일시트 디렉토리

├── js/

│ ├── toggleDetails.js # 메뉴 토글 기능

├── uploads/ # FAQ 이미지 저장소

├── **인증**

├── auth/

│ ├── auth.php # 인증 및 권한 관리

## 주요 파일 및 폴더 상세 설명

├── **/home/nstek/h2\_system/patch\_active/FAQ**

├── index.html               # FAQ 메인 페이지

│- PHP와 HTML이 혼합된 메인 페이지

│- 주요 기능:

│    - FAQ 메뉴 구조 (FAQ/Menu 탭 시스템)

│    - 실시간 FAQ 콘텐츠 로딩 (loadFAQContent 함수)

│    - 관리자 전용 수정/삭제 버튼 관리

│    - 버전별 페이지네이션 UI

│- 특이사항:

│    - 동적으로 생성되는 hidden input 필드로 버전 관리

│    - 첫 번째 FAQ 항목 자동 로드 기능

│

├── nav.php                  # 네비게이션 컴포넌트

│- 공통 네비게이션 컴포넌트

│- 시스템 전역 메뉴 제공

│

├── faq.css                  # FAQ 스타일시트

│- 공통 스타일시트

│

├── faq.js                   # FAQ 핵심 관리 모듈

│   - FAQ 관리 시스템의 프론트엔드 핵심

│   - 주요 모듈:

│   - chapterManager: 챕터/서브챕터 선택 관리

│   - sectionManager: 섹션/서브섹션 동적 관리

│   - 이벤트 처리:

│   - 폼 초기화 및 검증

│   - 섹션/서브섹션 추가/삭제

│   - 에디터 초기화

│

├── faq\_form.php        # FAQ 폼 PHP

│    - FAQ 작성/수정 폼

│    - 다중 섹션/서브섹션 지원

│    - 버전 관리 인터페이스

│

├── faq\_content.js           # FAQ 콘텐츠 관리 모듈

│    - FAQ 콘텐츠 관리 핵심 스크립트

│    - 주요 함수:

│    - toggleInputFieldVisibility: 입력 필드 표시/숨김 처리

│    - handleChapterChange: 챕터 변경 시 서브챕터 동적 로드

│    - handleSubChapterChange: 서브챕터 변경 및 버전 옵션 처리

│    - loadFAQContent: AJAX 기반 FAQ 콘텐츠 로드

│    - 버전 관리:

│     - 버전별 UI 업데이트

│     - 페이지네이션 이벤트 처리

│     - 버전 전환 시 콘텐츠 동적 갱신

│

├── faq\_submit.php      # FAQ 제출

│    - FAQ 데이터 저장

│    - 새 버전 생성 처리

│

├── faq\_delete.php      # FAQ 삭제

│    - FAQ 삭제 처리

│    - 관련 이미지 정리

│

├── debug.php # 디버깅파일

│

├── **데이터 관리**

├── faq\_db.php          # DB 연결 및 공통 함수

├── faq\_content.php     # FAQ 목록 관리

│  - FAQ 목록을 불러와서 표시

│  - 서브챕터 조회 및 관련 데이터 로드

│  - AJAX를 활용한 동적 콘텐츠 업데이트

│  - 서브챕터 버전 그룹화 쿼리

│  - 버전별 UI 생성

│

├── faq\_view.php        # FAQ 상세 조회

│  - 버전 유효성 검증, 버전별 데이터 처리

│  - createVersionMapping(): UI/DB 버전 매핑

│  - 섹션/서브섹션 계층 조회

│

├── faq\_update.php      # FAQ 업데이트 처리

│- 트랜잭션 기반 데이터 처리:

│  - 서브챕터 정보 업데이트

│  - 섹션/서브섹션 동시 업데이트

│- 버전 관리:

│  - 현재 버전 데이터만 수정

│  - 섹션/서브섹션 버전 동기화+

│

├── fetch\_sections.php  # 섹션 정보 가져오는 PHP

│     - 섹션 데이터 조회 API

│     - AJAX 요청 처리

│

├── fetch\_subchapters.php # 하위 챕터 정보 가져오는 PHP

│    - 서브챕터 데이터 조회 API

│    - 버전별 데이터 처리

│

├──  fetch\_subsections.php # 하위 섹션 정보 가져오는 PHP

│    - 서브섹션 데이터 조회 API

│

├──  get\_faq\_view.php    # FAQ 상세 내용 조회 API

│     - FAQ 상세 내용 조회 API

│     - 버전별 데이터 처리

│

├── **에디터**

├── editor/

│   ├── faq\_editor.js   # SunEditor 설정

│   │    - 이미지 업로드 처리

│   │    - 다중 에디터 인스턴스

│   ├── img\_upload.php  # 이미지 처리

│   │    - 파일 업로드 검증

│   │    - 자동 파일명 생성

│

├── **스타일**

├── css/                # 스타일시트 디렉토리

├── js/

│   ├── toggleDetails.js # 메뉴 토글 기능

│   ├── uploads/            # FAQ 이미지 저장소

│

├── **인증**

├── auth/

│   ├── auth.php        # 인증 및 권한 관리

# 주요 기능 및 코드 분석

## 버전 관리 시스템

### 버전 관리 핵심 필드

1. **버전 관리를 위한 테이블 구조**

SUBCHAPTERS (

    POSITIONS int NOT NULL,    -- 동일 콘텐츠 그룹 식별자

    VERSIONS int NOT NULL,     -- 버전 번호

    CHAP\_ID int NOT NULL      -- 소속 챕터

)

SECTIONS (VERSION)

SUBSECTIONS (VERSION)

1. **POSITIONS의 역할**

* 동일한 콘텐츠의 여러 버전을 그룹화
* 같은 POSITIONS 값을 가진 레코드들은 동일 콘텐츠의 다른 버전을 의미

1. **버전 생성 매커니즘**

// 1. 기존 POSITION 유지하면서 새 버전 생성

$stmt = $dbconnect->prepare("

    SELECT POSITIONS, MAX(VERSIONS) as max\_version

    FROM SUBCHAPTERS

    WHERE CHAP\_ID = ? AND POSITIONS = ?

");

// 2. 동일 POSITION에 새 버전 추가

INSERT INTO SUBCHAPTERS

(CHAP\_ID, POSITIONS, VERSIONS, SUB\_CHAP\_NAME)

VALUES

(?, ?, ?, ?)  // POSITIONS는 유지, VERSIONS만 증가

1. **버전 생성 매커니즘**

-- 예시 데이터

CHAP\_ID | POSITIONS | VERSIONS | SUB\_CHAP\_NAME

1 | 1 | 1 | 'FAQ 초기버전'

1 | 1 | 2 | 'FAQ 수정버전'

1 | 1 | 3 | 'FAQ 최신버전'

1 | 2 | 1 | '다른 FAQ 초기버전'

* POSITIONS=1인 레코드들은 동일 FAQ의 버전들
* POSITIONS=2는 다른 FAQ의 시작을 의미

1. **새 FAQ vs 버전 업데이트**

if ($mode === 'create') {

// 새 FAQ 생성 시: 새로운 POSITION 할당

$new\_position = "SELECT MAX(POSITIONS) + 1 FROM SUBCHAPTERS WHERE CHAP\_ID = ?";

} else {

// 기존 FAQ 수정 시: 기존 POSITION 유지, VERSION만 증가

$current\_position = "SELECT POSITIONS FROM SUBCHAPTERS WHERE SUB\_CHAP\_ID = ?";

}

### 버전 UI 및 사용자 인터페이스

1. 버전 번호 표시 매핑

- 데이터베이스 버전 번호와 ui 표시 번호의 분리

// 버전 매핑을 위한 함수

function createVersionMapping($db\_versions) {

$mapping = [];

$ui\_version = 1; // UI에 보여줄 버전 번호는 항상 1부터 시작

// DB 버전을 순서대로 정렬

sort($db\_versions);

// 각 DB 버전에 대해 UI 버전 매핑

foreach ($db\_versions as $db\_version) {

$mapping[$ui\_version] = $db\_version;

$ui\_version++;

}

return $mapping;

}

1. 페이지네이션 구현

- 버전 간 이동을 위한 UI 구현

<!-- 페이지네이션 -->

<div class="pagination">

<?php foreach ($version\_mapping as $ui\_ver => $db\_ver): ?>

<a href="?mode=view&chap\_id=<?php echo $current\_chap\_id; ?>&sub\_chap\_id=<?php echo $current\_sub\_chap\_id; ?>&position=<?php echo $current\_position; ?>&version=<?php echo $db\_ver; ?>"

class="page-number <?php echo ($db\_ver == $current\_version) ? 'active' : ''; ?>">

#<?php echo $ui\_ver; ?>

</a>

<?php endforeach; ?>

</div>

1. 버전 선택 옵션

- 새 버전 생성 시 사용자 인터페이스

<!-- 버전 추가 옵션 -->

<div class="version-option" style="display:none; margin-top: 10px;">

<div class="version-checkbox-group">

<input type="checkbox" id="has\_version" name="has\_version">

<label for="has\_version">서브챕터 #버전 추가</label>

<span id="version\_number\_display" class="version-number"></span>

</div>

<input type="hidden" id="sub\_chapter\_number" name="sub\_chapter\_number">

</div>

### 버전 처리 플로우

1. 새 버전 생성 시 데이터 흐름

- 기존 버전 정보 조회 → 최대 버전 계산 → 새 버전 생성 → 관련 섹션/서브섹션 생성

// 기존 서브챕터 선택 시

$subChapId = $subChapterOption;

// 현재 서브챕터의 정보 조회

$stmt = $dbconnect->prepare("

SELECT POSITIONS, SUB\_CHAP\_NAME

FROM SUBCHAPTERS

WHERE SUB\_CHAP\_ID = ? AND CHAP\_ID = ?

");

$stmt->bind\_param("ii", $subChapId, $chapId);

$stmt->execute();

$subChapInfo = $stmt->get\_result()->fetch\_assoc();

// 같은 챕터, 같은 POSITION 내에서 최대 VERSION 조회

$stmt = $dbconnect->prepare("

SELECT MAX(VERSIONS) as max\_version

FROM SUBCHAPTERS

WHERE CHAP\_ID = ? AND POSITIONS = ?

");

$stmt->bind\_param("ii", $chapId, $subChapInfo['POSITIONS']);

$stmt->execute();

$maxVersion = $stmt->get\_result()->fetch\_assoc()['max\_version'];

// 새로운 버전으로 서브챕터 추가

$newVersion = $maxVersion + 1;

$stmt = $dbconnect->prepare("

INSERT INTO SUBCHAPTERS

(CHAP\_ID, SUB\_CHAP\_NAME, SUB\_CHAP\_DESC, POSITIONS, VERSIONS)

VALUES (?, ?, ?, ?, ?)

");

1. 버전별 컨텐츠 로딩

- 버전 파라미터에 따라 적절한 데이터 로드

// 섹션 및 서브섹션 정보 가져오기 (버전에 따라 제한)

$query = "

SELECT s.\*

FROM SECTIONS s

WHERE s.SUB\_CHAP\_ID = ? AND s.VERSION = ?

";

$stmt = $dbconnect->prepare($query);

$stmt->bind\_param("ii", $current\_sub\_chap\_id, $current\_version);

$stmt->execute();

### 버전 관리 고려사항 및 제한사항

1) 버전 관리의 범위

* 서브챕터, 섹션, 서브섹션 레벨에서 버전 관리
* 이미지 파일명에도 버전 정보 포함 (chap{$chap\_id}\_sub{$sub\_chap\_id}\_v{$version}\_{$timestamp}.{$file\_extension})

2) 버전 번호 표시 전략

* 내부 DB 버전(VERSIONS)과 UI 표시 버전(1부터 시작하는 연속 번호) 분리
* 사용자는 항상 순차적 버전 번호 확인 가능

3) 버전 간 데이터 독립성

* 각 버전은 독립적인 데이터 세트 유지
* 한 버전 수정 시 다른 버전에 영향 없음

4) 성능 및 스토리지 고려사항

* 모든 버전 데이터를 별도 저장하므로 스토리지 사용량 증가
* 버전 증가에 따른 쿼리 최적화 필요성

5) 버전 브랜치 및 병합 제한

* 현재 구현은 선형 버전 관리만 지원 (버전 분기 및 병합 미지원)
* 버전 간 콘텐츠 복사만 가능, 차등 업데이트 기능 없음

## 계층적 구조 관리

### 4단계 구조 설계

1) 계층 구조

CHAPTERS (챕터)

└── SUBCHAPTERS (서브챕터 - 버전 관리 단위)

└── SECTIONS (섹션)

└── SUBSECTIONS (서브섹션 - 실제 컨텐츠 포함)

2) 계층별 관계 및 쿼리 구조

// 계층적 데이터 조회 예시

$query = "

SELECT s.\*

FROM SECTIONS s

WHERE s.SUB\_CHAP\_ID = ? AND s.VERSION = ?

";

$stmt = $dbconnect->prepare($query);

$stmt->bind\_param("ii", $current\_sub\_chap\_id, $current\_version);

$stmt->execute();

$sections = $stmt->get\_result();

$sections\_data = [];

while ($section = $sections->fetch\_assoc()) {

$section\_id = $section['SEC\_ID'];

$subsections\_query = "

SELECT \*

FROM SUBSECTIONS

WHERE SEC\_ID = ? AND VERSION = ?

";

// ... 중첩 쿼리로 서브섹션 조회

}