Story24 운영자 매뉴얼

2020-08-21

문서 이력 정보

Ver	주요 제/개정 내용	작성일	작성자	승인자	시행일
0.1	신규작성	2020.08.21	이원도		
0.4	현행화	2020.11.23	이원도		
0.41	그룹코드 등록쿼리 추가	2020.11.26	이원도		

목 차

1.개발 환경 구축

1.1. Front

https://gitlab.kotech.co.kr/root/story24_react/-/blob/develop/README.md

1.1.1 Install node.js

https://nodejs.org/ko/download/releases/

공식 홈페이지 이동 후 version v12.13.0 다운로드 클릭 후 페이지이동
node-v12.13.0-x64.msi 클릭하여 다운로드 후 설치

1.1.2 install yarn

공식사이트에서 yarn 다운로드 후 설치

https://classic.yarnpkg.com/en/docs/install#windows-stable

1.1.3 Install git bash

본 매뉴얼은 Windows 초기설정 상황에서 설치과정이 작성됨
https://gitforwindows.org/

공식 홈페이지에서 Donwload 클릭 후 installer 다운로드 후 설치

```
kotech@kotechPC MINGW64 ~
$
```

위 디렉토리부터 작업 시작

1.1.4 make project

```
mkdir workspace
cd workspace
git clone https://gitlab.kotech.co.kr/root/story24_react.git
```

깃클론 완료 후 프로젝트 생성 확인 및 프로젝트 파일내부 접근

```
ls
cd story24_react/
```

1.1.5 download visual studio

Visual studio 설치 과정은 1.2.3 의 설명 참고

1.1.6 노드모듈 설치

```
yarn install
```

1.1.7 개발 모드 구동

Story24_react 폴더 하위에 .env.development 파일 생성 후 아래와 같은 내용으로 저장 (story24yes24stor 오타 아님)

```
API_URL=http://182.237.86.231:8080          (backend test 주소)
PORT=3003
DOMAIN_URL=http://story24-local.yes24.com:3003
ANONY_SECRET=testsecret
ANONY_KEY=story24yes24stor
```

다시 git bash (cmd) 창에서 명령어 입력

추가로 노드 모듈설치

```
yarn global add win-node-env
```

개발모드 구동

```
yarn dev:node
```

1.1.8 운영모드 구동

BUILD

```
yarn build
```

운영모드 구동

```
yarn prod:node
```

1.1.9 노드모듈 추가

개발시 노드모듈 추가

```
yarn add 모듈명
```

노드모듈 추가 후 package.json 파일에서 확인 가능

노드모듈 추가시 어떤 모듈인지 사전에 공유하는 것을 권장

노드 모듈 추가 후 git 에서 내려받을 때 yarn install 명령어 입력 후 서버 가동

1.1.10 Docker 환경

도커 컨테이너 내부에서는 다음 명령어를 써서 실행됨

```
#stage 서버의 경우
pm2-runtime ecosystem.config.js --only story24_stage

#실서버의 경우
pm2-runtime ecosystem.config.js --only story24_prod
```

환경은 다음과 같이 설정됨

```
#ecosystem.config.js 에서
{
  name: 'story24_stage',
  script: 'server.js',

// Options reference: https://pm2.keymetrics.io/docs/usage/application-declaration/

    args: 'one two',
    instances: 1,
    autorestart: true,
    watch: false,
    max_memory_restart: '1G',
    env: {

# NODE_ENV 는 development 와 production 중에서만 선택가능하기 때문에,
production 으로 세팅하고
# 주요 정보는 .env.production 이 아니라 여기에 설정

      NODE_ENV: 'production',
      API_URL: 'https://story24-api-stage.yes24.com/backend',
      PORT: 3004,
      DOMAIN_URL: 'https://story24-stage.yes24.com'
    },
  }
```

1.2. Backend and CMS

본 매뉴얼은 Windows 초기설정 상황에서 설치과정이 작성됨
Backend 와 CMS 프로젝트 구성 자체에 공통적인 부분이 많아 통합 작성

1.2.1 Install git bash

https://gitforwindows.org/

공식 홈페이지에서 Donwload 클릭 후 installer 다운로드 후 설치

```
kotech@kotechPC MINGW64 ~
$
```

위 디렉토리부터 작업 시작

참고
Git bash 가 아닌 다른 유틸리티 툴을 사용해도 무방함
Ex) cmder 등등

1.2.2 make project

```
mkdir workspace
cd workspace
git clone https://gitlab.kotech.co.kr/root/story24-cms_java
git clone https://gitlab.kotech.co.kr/root/story24_java_project
```

1.2.3 install visual studio

https://code.visualstudio.com/

Download for Windows 클릭 후 installer 다운로드 후 설치
"기타:" 하단에 있는 체크박스 전부 클릭하는 것이 사용하기 편함
(미설치 시 code 명령어 별도로 세팅해야함)

미설치 시 참고
https://stackoverflow.com/questions/29971053/how-to-open-visual-studio-code-from-the-command-line-on-

osx#:~:text=Open%20Visual%20Studio%20Code%20and,Select%20that%20options.&text=That's%20it.,Now%20open%20your%20terminal%20type.

1.2.4 install visual studio

visual studio 환경변수를 읽기 위해서 git bash 재시작

```
cd ~/workspace/story24-cms_java
code .

cd ~/workspace/story24_java_project
code .
```

각각의 디렉토리에서 "code ." 명령어를 실행하면 visual studio 위에 각각의 프로젝트가 열리게 된다 아래의 과정부터는 각각 동일함

1.2.5 install jdk

open jdk 1.8.0
https://github.com/ojdkbuild/ojdkbuild

jdk 14.0.1
https://www.oracle.com/java/technologies/javase/jdk14-archive-downloads.html

open jdk 의 경우 READMD.md 에 java-1.8.0-openjdk-1.8.0.265-1.b01.ojdkbuild.windows.x86_64.msi 를 설치

jdk 의 경우 Windows x64 Installer 를 설치

주의
Language Support for Java(TM) by Red Hat 이 2020 년 7 월 22 일부터 업데이트되면서 **jdk 11 이상을 가지고 있어야함**

open jdk 1.8.0
C:₩Program Files₩ojdkbuild

jdk 14.0.1
C:₩Program Files₩Java₩jdk-14.0.1

1.2.6 install jdk

Extension 메뉴 열기 -> 단축키 (Ctrl + Shift + X)
visual studio code 에서 아래의 extension 을 설치

Java Extension Pack

Spring Boot Extension Pack

Lombok Annotations Support for VS Code

1.2.7 setting jdk

좌측 상단 메뉴
File > Preferences > Settings 클릭
jdk 검색 후 settings.json 열기

settings.json

```
{
  "java.home": "C:₩₩Program Files₩₩Java₩₩jdk-14.0.1",
  "java.configuration.runtimes": [
    {
      "name": "Java8",
      "path": "C:₩₩Program Files₩₩₩ojdkbuild₩₩java-1.8.0-openjdk-1.8.0.265-1",
    },
    {
      "name": "Java14",
      "path": "C:₩₩Program Files₩₩Java₩₩jdk-14.0.1",
      "default": true
    }
  ]
}
```

jdk 는 위 룰에 맞춰서 사용자환경에 맞도록 디렉토리 세팅할 것
default 로 jdk 11 이상의 버전으로 설정하고 jdk 8 을 등록해놔야함
자바 프로젝트 내 설정파일에 의해 jdk 8 로 실행됨

1.2.8 start project

Run 메뉴 열기 -> 단축키 (Ctrl + shift + D)
visual studio code 에서 Run 메뉴에서 "create a launch.json file" 을 클릭하여 launch.json 생성

Select Environment 에서 "Java" 검색 후 선택

우측 하단 팝업

Run/Debug feature requires Java language server to run in Standard mode. Do you want to switch it to Standard mode now?

"Yes" 클릭

주의
JDK 설정이 안되있을 경우 "Yes"를 클릭하더라도 실행되지 않음

1.2.9 test

cms
http://127.0.0.1:9000/cms/

backend
http://127.0.0.1:8080/swagger-ui.html#/common-controller

2.Front 공통 영역

2.1. 프로젝트 구조

- React 기반 SSR 프레임웍 nexjs 의 npx(npm executable)명령에 의한 기본 스캐폴딩 (참고: https://nextjs.org/learn/basics/create-nextjs-app/setup)
- api: 백엔드 API 서버를 호출하는 모듈
- components: react 컴포넌트
- constants: 상수값 관리
- cypress: e2e 테스트도구
- lib: 주요 자체 모듈
- pages: URL 로 라우팅되는 페이지
- public: 검색엔진 대응 robot.txt 및 구글웹마스터 인증용
- static: 이미지, 폰트, scss, js 등 정적 리소스, 뷰어관련 파일

- store: 리덕스 스토어 관련 파일
- utils: 유틸리티성 파일

2.2. 환경변수 및 설정

- 개발환경 환경변수는 dotenv 모듈이 server.js 가 실행되는 시점에 package.json 에 명시된 스크립트에 따라 .env 파일에서 로드함
- 운영환경 환경변수는 pm2 의 설정파일인 ecosystem.config.js 파일에서 로드함
- 빌드시점의 설정은 nextjs 의 설정파일인 next.config.js 에서 로드함
- 빌드 타겟폴더는 개발환경은 .next, 운영환경은 dist

2.3. 기본페이지 및 라우팅 구조

- next 프레임웤 페이지기반 라우팅 MSPA(Multi Single Page Application)
- pages 폴더 하위에 폴더명 및 파일명으로 라우팅됨
- 페이지 구조는 기본적으로 메뉴구성을 따름: 분야별(story), 채널(channel), 펜살롱(pensalon), 24.1(hertz), 보관함(locker), 검색(search), 마이페이지(mypage) 등

2.4. 핵심공통모듈 및 nextjs SSR/CSR 프로세스

- npx 에 의해 스캐폴딩된 기본 nextjs 프로젝트는 최초 진입점이 개별 페이지(pages/index.js)
- server.js: 익명토큰 등의 요구사항에 대응해 생성한 커스텀서버 server.js 에서 - next(node-express)서버 를 실행하는 게 최초 런타임 진입점. node-express 서버가 받은 request 를 next 앱의 리퀘스트 핸들러가 처리
- 커스텀 next 앱의 애플리케이션 디폴트 진입점인 _app.js 를 래핑한 lib/with-redux-store 래핑함수가 next 어플리케이션 레벨 최초 진입점이 됨. 여기서 최상위 리액트 컴포넌트의 properties 를 nextjs 의 API 인 getInitialProps 를 오버라이딩해 최초 initialize 함
- 공통묘듈 lib/with-redux-store.js: 넥스트 어플리케이션의 최초 진입점이자 최상위 공통모듈로써, 다음 작업 등이 수행됨

1. redux 스토어 초기화

2. 토큰/서비스쿠키/익명토큰 등 인증 관련 값, native 앱인터페이스 관련 값 쿠키 처리
3. 서버사이드 API 서버 콜의 공통헤더 세팅

- 즉 SSR request 가 들어오면, server.js 의 next requestHandler => with-redux-store 의 getInitialProps => 페이지별 getInitialProps 순으로 실행.
- 공통/페이지별 getInitialProps 는 서버사이드/클라이언트사이드 양측에서 모두 실행되므로 작업시 주의요함.
- with-redux-store 의 공통 getInitialProps 실행후 페이지별 getInitialProps 가 실행되어 페이지별로 내려줄 props 를 세팅.
- 공통모듈 pages/_app.js: 커스텀 next 앱의 애플리케이션 디폴트 진입점. 리덕스 스토어 Provider 와 로컬스토리지-리덕스스토어 간 동기화 모듈인 redux-persist 의 PersistGate 컴포넌트로 애플리케이션을 래핑한다. 그외 센트리셋업 등 클라이언트 사이드 공통 처리를 할 수 있다.
- 공통모듈 pages/_document.js: nextjs 가 SSR 로 생성하는 HTML document 의 공통코드 처리. meta 태그로 document 헤더를 셋팅하거나 script 태그를 사용하여 react 내부로 컴파일 되지 않는 전역함수를 정의할 수 있다.
- 공통모듈 pages/_error.js: nextjs 의 에러핸들링 공통페이지. 에러가 발생한곳에서 throw 되는 error 에 따라 커스터마이징한 페이지를 노출할 수 있다.

- SSR 과 CSR 발생시점

SSR: 직접 URL 입력하여 페이지로드, 새로고침, location.href, 웹뷰를 통한 url 로드
CSR: <Link>, <a>태그 등의 클릭, Router.push 를 통한 라우팅

2.5. 로그인 및 인증관련 구성

2.6. 리덕스 스토어 구성

- 대메뉴별+기능별 복합 state 구성
- 개별 대메뉴의 상태들을 store/index.js 에서 combine 함. 로컬스토리지와 동기화되는 persistance 상태들도 해당파일에서 설정함

- 상태는 코드의 간결함과 유지보수를 위해 최대한 flat 하게 구성함
- 상태의 불변성관리를 위해 immer 사용

2.7. 주요 UI 컴포넌트(components/)

- common/Layout.js: 웹어플리케이션의 공통 layout 을 잡는다. 헤더, 푸터, 바텀툴바, 토스트 등 전체 페이지 공통 컴포넌트
- common/Image.js: 에러핸들러와 타입별 디폴트 이미지를 prop 으로 갖는 공통 이미지 컴포넌트
- common/Toasts.js: 화면 전역에서 알럿등의 용도로 사용되는 컴포넌트
- common/layerpopup/ReplyPopup:

2.8. 주요 노드 모듈

- swiper
- lodash
- axios
- redux-persist
- react-modal

3.CMS 공통 영역

3.1. CMS 소스 공통 (토큰 암호화)

- 로그인시에 사용자 정보를 암호화하여 COOKIE 로 세팅한다.

로그인시에 토큰 생성 (AuthController.java login 메서드)

```
/** 로그인 데이터 조회 **/
LoginUser loginUser = authService.getUser( input );
/** 토큰 생성 **/
String authorization = userComponent.createAuthorization( loginUser );

/** TOKEN COOKIE 등록 **/
CommonUtils.addCookie( Constrants.AUTHORIZATION , authorization,
Duration.ofMinutes( this.token_time_minute ) );
```

인증토큰 생성 (UserComponent.java createAuthorization 메서드)

```
StringBuilder result = new StringBuilder();
result.append( Constrants.AUTHORIZATION_PREFIX  );
loginUser.setAdminPwd( null );
/**  토큰유효시간 갱신 **/
loginUser.renewExpireTime( Duration.ofMinutes( this.token_expire_time_minute )  );

String aes = aes256Encoder.encodeJson( loginUser );
result.append( aes );
return result.toString();
```

로그인 정보를 JSON 파싱이후에 암호화 (Aes256Encoder.java encodeJson 메서드)

```
try {
        String json = jackson_object_mapper.writeValueAsString( obj );
        return this.encode( json );
}catch( Exception e) {
        throw new RuntimeException( "encodeJson Failed..", e );
}
```

AES 256 암호화 (Application.yml)

```
aes.attribute.aes_256_key : STORY_24_CMS_AES_KEY
```

3.2. CMS 소스 공통 (토큰 복호화, LOCAL THREAD)

- 로그인된 사용자의 COOKIE 토큰을 복호화한다.

토큰정보를 복호화 이후 LOCAL THREAD 에 세팅한다. (CookieInterceptor.java preHandle 메서드)

```
/** 인증 정보  **/
String authorization = CommonUtils.getCookie( Constrants.AUTHORIZATION  );

if( StringUtils.isEmpty( authorization ) ) {
  /** 로그인 되지않은 사용자  **/
  throw new ErrorMessageException("user.unauthorized", HttpStatus.UNAUTHORIZED );
```

```
}

/** JWT 토큰을 파싱하여 사용자 정보를 조회한다. **/
LoginUser login_user = userComponent.getLoginUser( authorization );

if( ObjectUtils.isEmpty( login_user ) ) {
   /** 유효하지 않은 토큰 형식  **/
   throw new ErrorMessageException("token.notvalid", HttpStatus.UNAUTHORIZED );
}
if( login_user.isExpired() ) {
   /** 사용자 정보가 존재하지 않거나 유효하지 않은 토큰 **/
   throw new ErrorMessageException("token.expired", HttpStatus.UNAUTHORIZED );
}
/** LOCAL THREAD 에 대한 객체를 초기화 한다. **/
LocalThread.init( login_user );
```

@Controller @Service 에서 LOCAL THREAD 활용 (AuthController.java login 메서드)

```
if( LocalThread.isLogin() ) {
   String adminId = LocalThread.getLoginAdminId();
   login.setAdminId( adminId );
   /** COOKIE 가 존재할때. **/
   return "auth/oldLogin";
}else {
   String adminId = CommonUtils.getCookie( Constrants.REMEMBER_ME_NAME );
   login.setAdminId( adminId );
   /** COOKIE 가 존재하지 않을때**/
   return "auth/newLogin";
}
```

Mybatis Mapper XML 에서 LOCAL THREAD 활용 (RoleMapper.xml addPrgm 메서드)
LocalThread.xml 에서 해당 설정이 정리되어있다

```
/* RoleMapper.addPrgm */
INSERT INTO TS_cms_prgm_mng
 (
 prgm_seq
 ,prgm_nm
 ,URL
 ,method
```

```
 ,prgm_descr
 ,del_yn
 ,reg_mem_no
 ,reg_dts
 ,reg_ip
 ,updt_mem_no
 ,updt_dts
 ,updt_ip
)
VALUES
(
 #{prgmSeq}
 ,#{prgmNm}
 ,#{url}
 ,#{method}
 ,#{prgmDescr}
 ,#{delYn}
 ,<include refid="localThread.ADMIN_NO"/>
 ,NOW()
 ,<include refid="localThread.MEM_IP"/>
 ,<include refid="localThread.ADMIN_NO"/>
 ,NOW()
 ,<include refid="localThread.MEM_IP"/>
)
```

3.3. CMS 소스 공통 (SPRING AOP)

- SPRING AOP 기능을 활용하여 OOP 기능을 SPRING CONTROLLER, SERVICE 에서 사용한다.

SPRING CONTROLLER 중 모든 @RequestMapping 메서드 전 처리기

(Story24Aspect.java beforeController 메서드)

1. 비로그인 사용자 접근제한 처리
2. 로그인 토큰의 유효시간 체크
3. HTTP METHOD, HTTP URI 에 대한 사용자 권한 유무 체크

```
if( !LocalThread.isLogin() ) {
   /** 로그인 되지않은 사용자  **/
   throw new ErrorMessageException("user.unauthorized", HttpStatus.UNAUTHORIZED );
}
```

```
LoginUser loginUser = LocalThread.getLoginUser();
if( ObjectUtils.isEmpty( loginUser ) ) {
   /** 유효하지 않은 토큰 형식  **/
   throw new ErrorMessageException("token.notvalid", HttpStatus.UNAUTHORIZED );
}
if( loginUser.isExpired() ) {
   /** 사용자 정보가 존재하지 않거나 유효하지 않은 토큰 **/
   throw new ErrorMessageException("token.expired", HttpStatus.UNAUTHORIZED );
}
if( StreamUtils.isMatchUri( "/*/modal/**", "/*/pop/**" ).test( uri )   ) {
   /** MODAL 과 POPUP 은 로그인 유무만 체크한다. **/
   return;
}
if( optionalAuthority.isPresent() ) {
   /** @Authority 어노테이션의 권한이 존재하는지 확인  **/
   boolean has_role = StreamUtils.toStream( optionalAuthority.get().roles() )
                                 .filter( ObjectUtils::isNotEmpty )
                                 .filter( this.getAuthorityPredicate.apply( loginUser ) )
                                 .findFirst()
                                 .isPresent();
   if( !has_role ) {
      /** 사용자가 접근할수 없는 URI 오류 처리  **/
      throw new ErrorMessageException("user.forbidden", HttpStatus.FORBIDDEN, method, uri);
   }
} else {
   /** URI , METHOD 로 접근 가능한 ROLE 들을 조회한다.  **/
   UrlRoles urlRoles = Optional.ofNullable( roleComponent.getUrlRoles( method, uri ) )
                               .filter( UrlRoles::hasUrlRole )
                               .orElseThrow( () ->
                        /** 존재하지 않는 URL 접근  오류 처리  **/
                        new ErrorMessageException("url.notfound", HttpStatus.NOT_FOUND, method, uri)
                               );
   /** 접근한 사용자의 ROLE 체크 **/
   Optional.ofNullable( loginUser )
           .map( LoginUser::getRoles )
           .filter( login_role -> urlRoles.enableAccess( login_role )  )
           .orElseThrow( () ->
           /** 사용자가 접근할수 없는 URI 오류 처리  **/
           new ErrorMessageException("user.forbidden", HttpStatus.FORBIDDEN, method, uri)
           );
}
```

SPRING SERVICE 중 모든 @ValidBizAuth 가 존재하는 메서드 전 처리기
(Story24Aspect.java beforeBizAuth 메서드)

비즈니스 권한 (회원정보 조회, 캐시지급, 환불) 이 존재하는 사용자인지 확인한다.
SPRING SERVICE 에서 사용자에 대한 비즈니스 권한이 필요한경우
아래와 같이 ANNOTATION 를 추가해준다.

1. 회원정보 조회 권한 : @ValidBizAuth(types={ BIZ_AUTH_CD.MEM_INFO_VIEW })
2. 캐시지급 권한: @ValidBizAuth(types={ BIZ_AUTH_CD.CASH_GIVE })
3. 환불 권한: @ValidBizAuth(types={ BIZ_AUTH_CD.SRLZTNWORK_RETURN }

```
MethodSignature signature = (MethodSignature) joinPoint.getSignature();
Method method = signature.getMethod();
ValidBizAuth biz_auth_tag = method.getAnnotation( ValidBizAuth.class );
/** 유효성 체크를 할 비즈니스 권한 CD 들 **/
List<BIZ_AUTH_CD> biz_auth_cds = Optional.of( biz_auth_tag )
if( CollectionUtils.isEmpty( biz_auth_cds ) ) {
   return;
}
/** 로그인 사용자의 어드민 번호 **/
Integer adminNo = LocalThread.getLoginAdminNo();
/** 사용자의 BIZ 권한 조회  **/
Optional< Map<String,String> > user_authority = this.userComponent.getUserAuthority( adminNo );
if( !user_authority.isPresent() ) {
   /** 사용자의 BIZ 권한이 존재하지 않음. **/
   throw new ErrorMessageException("auth.none.user", HttpStatus.BAD_REQUEST);
}
/** 회원정보 조회 권한 여부 **/
boolean enableMemInfoView = UserComponent.isEnableBizAuth.apply( user_authority ,
"memInfoViewYn" );
/**  캐시지급 권한 여부 **/
boolean enableCashGive = UserComponent.isEnableBizAuth.apply( user_authority , "cashGiveYn" );
/**  환불 권한 여부 **/
boolean enableSrlztnworkReturn = UserComponent.isEnableBizAuth.apply( user_authority ,
"srlztnworkReturnYn" );

..... 이하 생략
```

SPRING SERVICE 중 모든 @RemoveTag 가 존재하는 메서드 후 처리기
(Story24Aspect.java afterReturnRemoveTag 메서드)

응답값중에 TAG 를 삭제할 컬럼이 존재하는경우 html TAG 를 삭제한다.
Ex) @RemoveTag(columns = { "target_ttl","ttl" })
응답값 중에서 target_ttl, ttl 에 대해서 HTML TAG 삭제 처리한다

```
MethodSignature signature = (MethodSignature) joinPoint.getSignature();
Method method = signature.getMethod();
RemoveTag remove_tag = method.getAnnotation( RemoveTag.class );
/** REMOVE TAG 대상이 되는 컬럼들을 조회  **/
List< String > target_colunm = Optional.ofNullable( remove_tag )
                                        .map( RemoveTag::columns )
                                        .map( Arrays::stream )
                                        .orElseGet( Stream::empty )
                                        .filter( StringUtils::isNotEmpty )
                                        .distinct()
                                        .collect( Collectors.toList() );
if( returnValue == null ) { return; }
if( CollectionUtils.isEmpty( target_colunm )  ) { return; }
try {
  if( returnValue instanceof  Map  ) {
    this.doRemoveTag( ( Map<String,Object> ) returnValue ,target_colunm );
  }
  if( returnValue instanceof PagingResult ) {
    PagingResult<?> paginResult = (PagingResult<?>) returnValue;
    List<?> list = paginResult.getData();
    this.doRemoveTagList( list ,target_colunm );
  }
  if( returnValue instanceof List ) {
    this.doRemoveTagList( (List<?>) returnValue ,target_colunm );
  }
}catch( Exception e) {
  log.error("[Remove Tag] {}" , e);
}
```

3.4. CMS 소스 공통 (COOKIE INTERCEPTOR)

- 로그인 토큰에 대해서 복호화, 유효성 체크, LOCAL THREAD 를 생성 및 폐기한다.

COOKIE INTERCEPTOR 전 처리기 (CookieInterceptor.java preHandle 메서드)

```
/** 인증 정보  **/
String authorization = CommonUtils.getCookie( Constrants.AUTHORIZATION   );
if( StringUtils.isEmpty( authorization ) ) {
  /** 로그인 되지않은 사용자  **/
  throw new ErrorMessageException("user.unauthorized", HttpStatus.UNAUTHORIZED );
}
/** JWT 토큰을 파싱하여 사용자 정보를 조회한다. **/
LoginUser login_user = userComponent.getLoginUser( authorization );
if( ObjectUtils.isEmpty( login_user ) ) {
  /** 유효하지 않은 토큰 형식  **/
  throw new ErrorMessageException("token.notvalid", HttpStatus.UNAUTHORIZED );
}
if( login_user.isExpired() ) {
  /** 사용자 정보가 존재하지 않거나 유효하지 않은 토큰 **/
  throw new ErrorMessageException("token.expired", HttpStatus.UNAUTHORIZED );
}
/** LOCAL THREAD 에 대한 객체를 초기화 한다. **/
LocalThread.init( login_user );
Optional.ofNullable( login_user )
            .map( user -> userComponent.createAuthorization( user )    )
            .ifPresent( new_authorization -> {
            /** 쿠키에 인증 정보 세팅  **/
            CommonUtils.addCookie( Constrants.AUTHORIZATION
                                  ,new_authorization
                                  ,Duration.ofMinutes( this.token_time_minute ) );
            });
return true;
```

COOKIE INTERCEPTOR 후 처리기 (CookieInterceptor.java afterCompletion 메서드)

```
try {
  /** LOCAL THREAD 에 대한 객체를 삭제한다. **/
  LocalThread.destory();
}catch(Exception e) {
}
```

3.5. CMS 소스 공통 (MENU INTERCEPTOR)

- CMS HTTP 요청시에 현재 메뉴 정보를 세팅한다.

MENU INTERCEPTOR 후 처리기 (MenuInterceptor.java preHandle 메서드)

```
if( !CommonUtils.isResponseJsp( request )  ) {
   /** 응답 VIEW 가 AJAX or HTTP 통신인 경우 **/
   return true;
}
/** REQUEST 정보 **/
String method = request.getMethod();
String uri = StringUtils.replaceOnce( request.getRequestURI(),   request.getContextPath() , "");
/** 해당프로그램의 시퀀스 세팅 **/
this.setAttributeFunction.apply( request , Constrants.META_PRGM_SEQ_NAME )
                          .accept( () -> this.roleComponent.getPrgmSeq( method, uri ) );
/** LV1 메뉴 시퀀스 세팅  **/
this.setAttributeFunction.apply( request , Constrants.META_LV1_MENU_SEQ_NAME )
                          .accept( () -> request.getParameter( Constrants.META_LV1_MENU_SEQ_NAME ) );
/** LV2 의 메뉴 시퀀스 세팅  **/
this.setAttributeFunction.apply( request,   Constrants.META_LV2_MENU_SEQ_NAME )
                          .accept( () -> request.getParameter( Constrants.META_LV2_MENU_SEQ_NAME ) );
/** LV3 의 메뉴 시퀀스 세팅  **/
this.setAttributeFunction.apply( request,   Constrants.META_LV3_MENU_SEQ_NAME )
                         .accept( () -> request.getParameter( Constrants.META_LV3_MENU_SEQ_NAME ) );
/** 현재 메뉴의 대상 권한 (파워구분) **/
                       this.setAttributeFunction.apply( request,   Constrants.META_TARGET_ROLE_NAME )
                        .accept( () -> request.getParameter( Constrants.META_TARGET_ROLE_NAME ) );
```

4.BACKEND 공통 영역

4.1. BACKEND 소스 공통 (API INTERCEPTOR)

-BACKEND API 요청시에 인증토큰에 대한 처리를 수행한다.

API INTERCEPTOR 전 처리기 (ApiInterceptor.java preHandle 메서드)

1. 인증토큰 의 유효성 체크
2. LOCAL THREAD 초기화

```
/** 요청 method 정보  **/
String method = request.getMethod();
String uri = request.getRequestURI();
if( StringUtils.equals( method, HttpMethod.OPTIONS.name() ) ) {
  /** CORS 시에 OPTIONS 로 서버요청 체크시에 처리   **/
  return true;
}
try {
  API_AUTH_TYPE api_auth_type = this.doCheckToken( request ,response );
  log.info("############### {} {}", method, uri);
  log.info("## API_AUTH_TYPE={}", api_auth_type);
}catch(ErrorMessageException eme ) {
  throw eme;
}catch(Exception e){
}finally{
  /** LOCAL THREAD 에 대한 객체를 초기화 한다. **/
  LocalThread.init();
}
```

인증토큰 세부 처리 (ApiInterceptor.java doCheckToken 메서드)

```
... 이전 생략

if( StringUtils.startsWith( authorization, EnumCodes.LOGIN_TOKEN_PREFIX ) ) {
  /** 서비스쿠키와 토큰의 유효성 체크 **/
  String jwt = StringUtils.replaceOnce( authorization, EnumCodes.LOGIN_TOKEN_PREFIX , "");
  API_AUTH_TYPE api_auth_type = jwtUtil.validtionLoginToken( jwt, serviceCookies );
  if( API_AUTH_TYPE.LOGIN == api_auth_type ) {
    /** 로그인 토큰이 유효한경우  **/
    response.setHeader( EnumCodes.AUTHORIZATION_NAME, EnumCodes.LOGIN_TOKEN_PREFIX + jwt);
    return API_AUTH_TYPE.LOGIN;
```

```
  }else if( API_AUTH_TYPE.FALSIFY_COOKIE == api_auth_type ) {
    /** 위조된 토큰인경우  **/
    response.setHeader( EnumCodes.AUTHORIZATION_NAME, null );
    /** 변조된 서비스쿠키로 접근하였습니다. **/
    throw new ErrorMessageException("9900");
  }
}
try {
  /** 서비스 쿠키로 신규 로그인 토큰을 생성한다. **/
  String login_token = jwtUtil.createLoginJwt( request );
  response.setHeader( EnumCodes.AUTHORIZATION_NAME, EnumCodes.LOGIN_TOKEN_PREFIX +
login_token );
  return API_AUTH_TYPE.LOGIN;
}catch( ErrorMessageException em ) {
  throw em;
}catch( Exception e ) {}

... 이하생략
```

4.2. BACKEND 소스 공통 (LOCAL THREAD)

-현재 로그인한 사용자의 정보를 조회하는 LOCAL THREAD 를 활용한다.

@Controller @Service 에서 LOCAL THREAD 활용 (CommonController.java coupons 메서드)

```
String memNo = String.valueOf(   LocalThread.getLoginMemNo() );
/** memNo = "256242";   **/
List<Map<String,Object>> list = restComponent.getCouponList(memNo, useYn);
```

Mybatis Mapper XML 에서 LOCAL THREAD 활용 (CommonMapper.xml addMemRecentlyViewLog 메서드)

LocalThread.xml 에서 해당 설정이 정리되어있다

```
/* CommonMapper.addMemRecentlyViewLog */
INSERT INTO ts_mem_recently_view_work
(
 mem_no
 ,info_gb
```

```
 ,tar_info_pk
 ,del_yn
 ,reg_mem_no
 ,reg_dts
 ,reg_ip
 ,updt_mem_no
 ,updt_dts
 ,updt_ip
)
VALUES
(
 <include refid="localThread.MEM_NO"/>
 ,#{infoGb}
 ,#{tarInfoPk}
 ,'N'
 ,<include refid="localThread.MEM_NO"/>
 ,NOW()
 ,<include refid="localThread.MEM_IP"/>
 ,<include refid="localThread.MEM_NO"/>
 ,NOW()
 ,<include refid="localThread.MEM_IP"/>
 )
 ON DUPLICATE KEY UPDATE
 del_yn = 'N'
 ,updt_dts = NOW()
```

4.3. BACKEND 소스 공통 (SPRING AOP)

- SPRING AOP 기능을 활용하여 OOP 기능을 SPRING CONTROLLER, SERVICE 에서 사용한다.

SPRING CONTROLLER 중 모든 메서드 전 처리기 (Story24Aspect.java aroundController 메서드)

API 를 수행하기 전에 사용자가 API 사용 권한이 있는지 확인한다.

```
/** Method 호출 **/
MethodSignature signature = (MethodSignature) joinPoint.getSignature();
Method method = signature.getMethod();
/** 로그인 체크 annotation **/
LoginOnly loginOnly =   method.getAnnotation( LoginOnly.class );
```

```
/** 비로그인 annotation **/
NotLogin notLogin = method.getAnnotation( NotLogin.class );
/** 로그인 여부 **/
Boolean isLogin = LocalThread.isLogin();
/** FRONT 에서 발급된 토큰이 존재하는지 여부 **/
Boolean isAnymous = LocalThread.isAnymous();
if( BooleanUtils.isTrue( isLogin ) ) {
   /** Controller 메서드 수행  **/
   return doProceedController(joinPoint, method);
} else {
   if( loginOnly != null ) {
      /** 로그인하지 않는 사용자 접근 **/
      throw new NotLoginException();
   }
   if( notLogin == null && BooleanUtils.isNotTrue( isAnymous )  ) {
      /** 잘못된 접근 **/
      throw new NotAnymousException();
   }
   /** Controller 메서드 수행  **/
   return doProceedController(joinPoint, method);
}
```

@LoginOnly ANNOTATION (StoryOnController.java settings 메서드)

SPRING CONTROLLER 메서드 에 세팅해서 로그인 사용자만 사용할 수 있는 API 로 지정한다.

```
@LoginOnly
@TargetCode("so.cont")
@CrossOrigin("*")
@PostMapping(value="/settings", consumes="application/json")
@ApiOperation("[등록기능은 POST /co/usersetting 로 대체 가능  응답값이 필요한지 문의 필요.]스토리온 - 메뉴 설정")
 public ResponseEntity<LinkedHashMap<String, Object>> settings(@RequestBody
List<HashMap<String, Object>> option                    // 메뉴 설정 리스트
                                                                    ){
 ... 이하생략
}
```

4.4. BACKEND 소스 공통 (샘플)

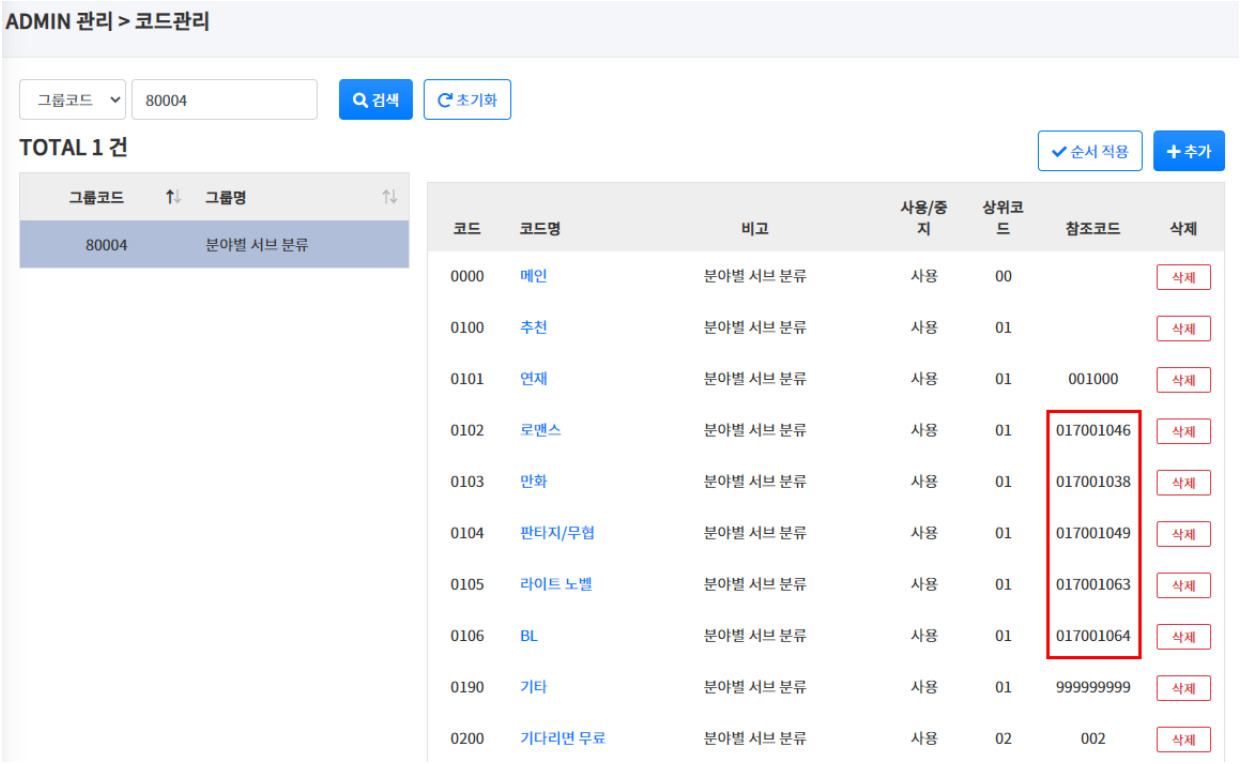
-BACKEND API 요청시에 인증토큰에 대한 처리를 수행한다.

```
/** 응답 VIEW 가 AJAX or HTTP 통신인 경우 **/
```

5.분야별 메뉴 설정(추가) 관리

5.1. Ebook 카테고리 추가 및 카테고리 변경

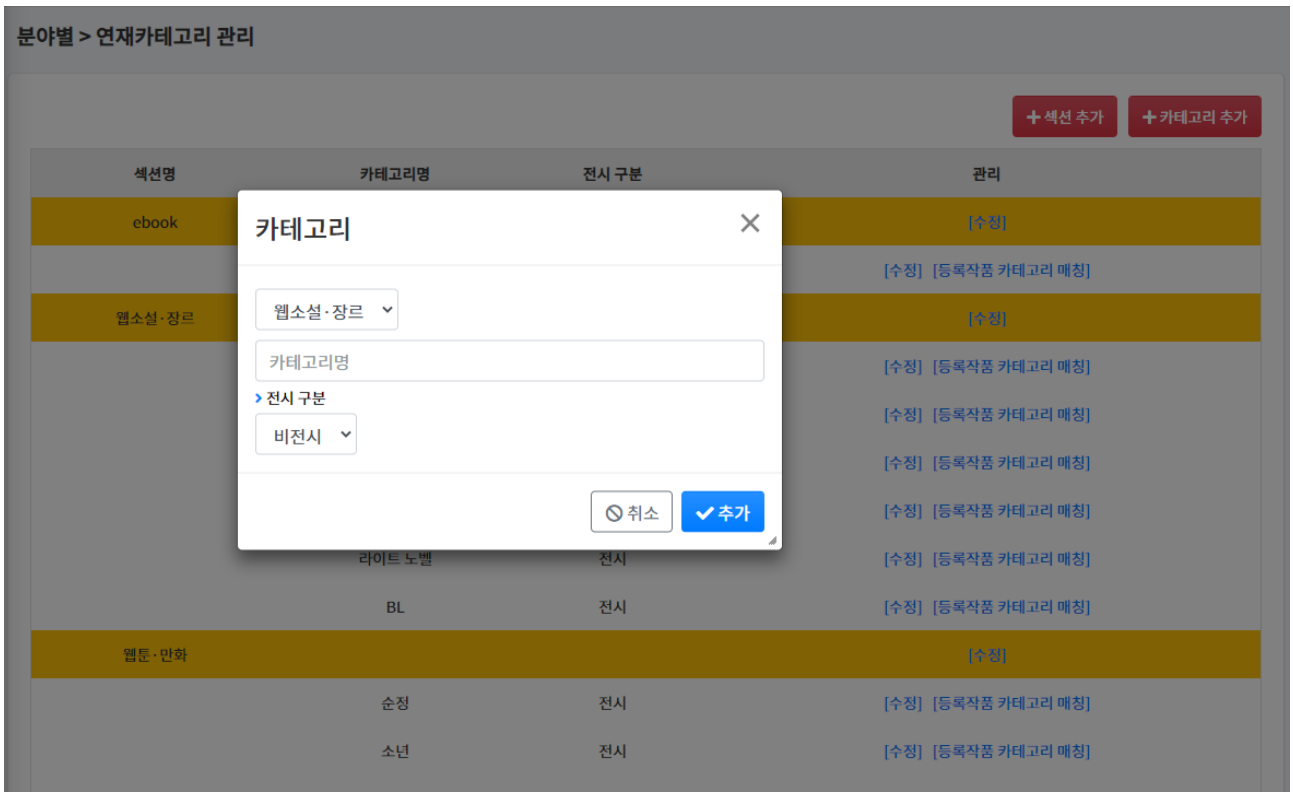
- CMS] ADMIN 관리>코드관리 메뉴 진입
- 그룹코드 8004 조회



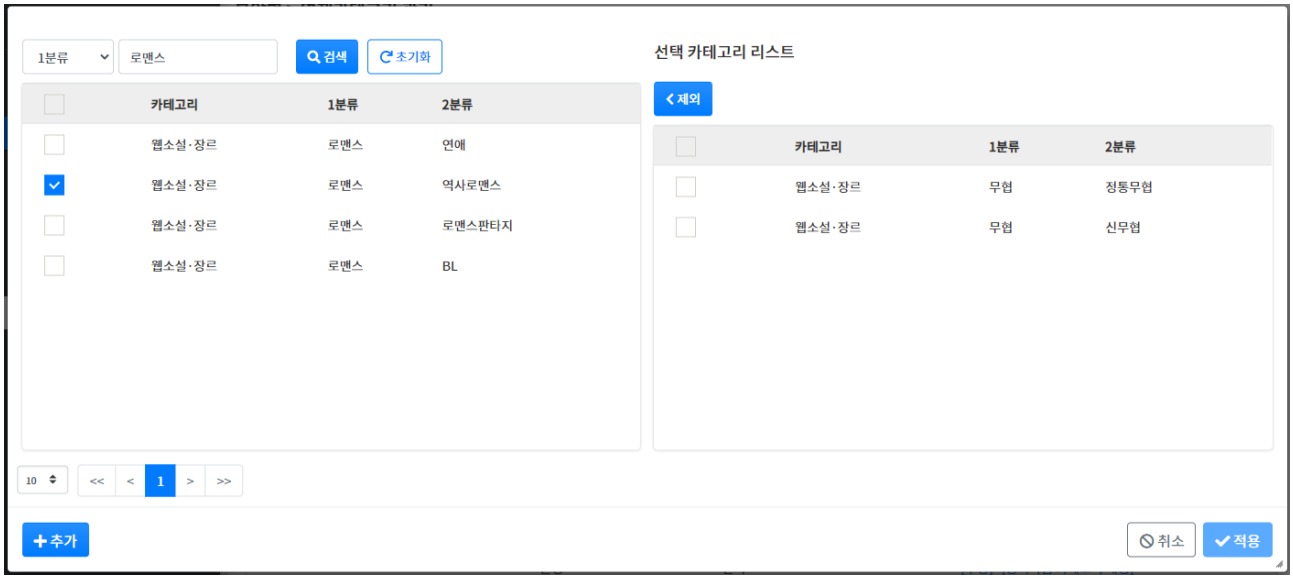
- 상위코드 그룹별 순서 변경후 순서 적용
- 전시 분류를 변경하고자 할 경우 참조코드를 서비스 하고자 하는 YES24 전시분류 코드로 변경

5.2. 웹소설/웹툰만화 카테고리 추가 및 카테고리 변경

- CMS] 분야별 > 연재카테고리관리 메뉴 진입
- 카테고리 추가



-등록작품 카테고리 매칭

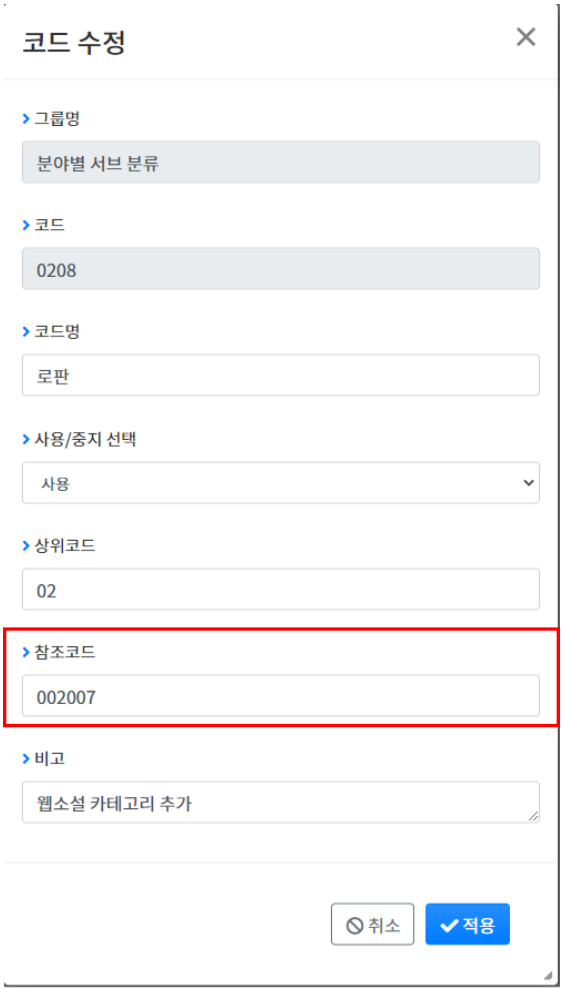


- 2 분류를 지정하면 2 분류에 속하는 3 분류의 모든 카테고리는 story24 대상 분류로 함께 지정 됨.
- 3 분류가 추가된 경우 해당 2 분류 삭제 후 재지정

- [CMS] ADMIN 관리> 코드관리 이동
- 그룹코드 8004 조회

0200	기다리면 무료	분야별 서브 분류	사용	02	002	삭제
0201	연재	분야별 서브 분류	사용	02	002	삭제
0202	로맨스	분야별 서브 분류	사용	02	002001	삭제
0203	판타지	분야별 서브 분류	사용	02	002002	삭제
0204	무협	분야별 서브 분류	사용	02	002003	삭제
0205	미스터리	분야별 서브 분류	사용	02	002004	삭제
0206	라이트 노벨	분야별 서브 분류	사용	02	002005	삭제
0207	BL	분야별 서브 분류	사용	02	002006	삭제

상위코드 기준 순서 변경 후 순서 적용
전시 분류를 변경하고자 할 경우 참조 코드를 연재작품관리의 카테고리 번호로 변경



- 신규 메뉴 추가시 카테고리 추가 후 신규 메뉴 코드 등록과 신규 카테고리 코드를 참조코드에 등록

5.3. 메뉴별 필터 설정

-Ts_filter_set 구성

컬렴명	설명
filter_seq	필터 순차번호
filter_tar_gb	필터대상구분[80062] (01 랭킹,02 분야별,03 작가의펜, 04 24.1, 09 보관함)
main_cat	메인 카테고리
main_cat_nm	메인 카테고리 명
sub_cat	서브 카테고리
sub_cat_nm	서브 카테고리 명
dtl_cat	상세 카테고리
dtl_cat_nm	상세 카테고리 명
filter_type	필터유형[80063]
filter_type_nm	필터 유형 명
filter_cd	필터 코드
filter_expo_nm	필터 노출 명
expo_sort	노출 순서
del_yn	삭제여부

- 필터 유형 구분[80063]

코드	필터유형 명	필터 사용 값
01	메뉴	해당 상세코드는 group_code 80064
02	작품타입	해당 상세코드는 group_code 80017
03	업데이트알림	해당 상세코드는 group_code 80064
04	작품이용	해당 상세코드는 group_code 80027
05	카테고리	해당 상세코드는 group_code 80068
06	프로모션 타입	해당 상세코드는 group_code 80067
07	프로모션 타입	tg_gl_cate.gb=ARTICLE_TYPE
08	연재 타입	해당 상세코드는 group_code 80065
09	분야	해당 상세코드는 group_code 80079
10	공개여부	해당 상세코드는 group_code 80065
11	작품이용	해당 상세코드는 group_code 80079
12	신고여부	해당 상세코드는 group_code 80081
13	국가	해당 상세코드는 group_code 80066

- 필터 등록 예시

1 Ebook 랭킹 :

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
```

```
filter_cd, filter_expo_nm, del_yn , expo_sort )
select '01' as filter_tar_gb , cmmn_cd,   null     as sub_cat, null as sub_cat_nm, null as dtl_cat, '05' as
filter_type ,
cat_cd as filter_cd, cat_nm as filter_expo_nm
, 'N', cd.expo_sort
 from ts_cmmn_cd   cd
 join ts_cat_mng cm on cm.uppr_cat = concat('0',cmmn_cd)
 where grp_cd ='80003' and cmmn_cd between '02' and '03';
```

2 분야별 연재 프로모션 (07)

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '02' as filter_tar_gb , cd2.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '07' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.cmmn_cd ='0101'
 where cd1.grp_cd ='80068' ;
```

3 분야별 ebook 메뉴

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn ,expo_sort)
select '02' as filter_tar_gb , cd1.uppr_cd , cd1.cmmn_cd as sub_cat, cd1.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '05' as filter_type ,
m01.disp_no as filter_cd, disp_nm as   filter_expo_nm,
'N',   disp_prir
from ts_cmmn_cd cd1
  join td_tsmdi01m m01 on m01.high_cate_no = cd1.rfrc_cd and   m01.del_yn='N' and m01.use_yn='Y'
and m01.cate_dep=4
 where cd1.grp_cd ='80004' and   cd1.uppr_cd='01' and cd1.cmmn_cd between '0102' and '0106'
 order by sub_cat, disp_prir ;
```

3 분야별 ebook 기타

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat,dtl_cat_nm, filter_type,
filter_cd, filter_expo_nm, del_yn   ,expo_sort)
with data1 as (
select disp_no, disp_nm, disp_prir
```

```
  from   td_tsmdi01m where high_cate_no = '017001'
  and del_yn='N' and use_yn='Y' and sum_yn='Y'
  and disp_no not in ( select rfrc_cd from ts_cmmn_cd cd3 where cd3.grp_cd ='80004' and
cd3.uppr_cd='01' and cd3.cmmn_cd between '0102' and '0106' )
)
  select
      '02' as filter_tar_gb , cd2.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm ,
d1.disp_no as dtl_cat, d1.disp_nm as dtl_cat_nm , '05' as filter_type ,
    m012.disp_no as   filter_cd , m012.disp_nm as filter_expo_nm, 'N' , m012.disp_prir
from ts_cmmn_cd cd1
    join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.cmmn_cd ='0190'
    join data1 as d1
    join td_tsmdi01m m012 on m012.high_cate_no = d1.disp_no and m012.del_yn='N' and
m012.use_yn='Y' and m012.sum_yn='Y'
 where cd1.grp_cd ='80004' and   cd1.uppr_cd='01' and cd1.cmmn_cd ='0190'
 order by d1.disp_prir ,m012.disp_prir;
```

3 분야별 웹 소설 연재 카테고리

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn ,expo_sort)
select '02' as filter_tar_gb , cd1.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '05' as filter_type ,
cd1.rfrc_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
    join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.cmmn_cd ='0201'
 where cd1.grp_cd ='80004' and   cd1.uppr_cd='02'   and cd1.cmmn_cd between '0202' and '0207'   ;
 ;
```

4 분야별 웹소설 연재 프로모션

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
with data1 as (
select cmmn_cd,cmmn_cdnm,grp_cd   ,expo_sort from ts_cmmn_cd where grp_cd ='80027' and
cmmn_cd !=3
union
```

```
select '0', '일반' , 80027 , 0 from dual
)
select '02' as filter_tar_gb , cd2.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '06' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from data1 cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.uppr_cd='02' and cd2.cmmn_cd between '0201'
and '0207'
 where cd1.grp_cd ='80027' order by cd2.expo_sort ,cd1.expo_sort ;
```

5 분야별 웹소설 연재 타입

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '02' as filter_tar_gb , cd2.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '08' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.uppr_cd='02' and cd2.cmmn_cd between '0202'
and '0207'
 where cd1.grp_cd ='80067' order by cd2.expo_sort ,cd1.expo_sort ;
```

6 분야별 웹툰만화 연재 카테고리

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn ,expo_sort)
select '02' as filter_tar_gb , cd1.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '05' as filter_type ,
cd1.rfrc_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
   join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.cmmn_cd ='0301'
 where cd1.grp_cd ='80004' and   cd1.uppr_cd='03'   and cd1.cmmn_cd between '0302' and '0307'   ;
```

7 분야별 웹툰만화 프로모션

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
```

```
with data1 as (
select cmmn_cd,cmmn_cdnm,grp_cd   ,expo_sort from ts_cmmn_cd where grp_cd ='80027' and
cmmn_cd !=3
union
select '0', '일반' , 80027 , 0 from dual
)
select '02' as filter_tar_gb , cd2.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '06' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from data1 cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.uppr_cd='03' and cd2.cmmn_cd between '0301'
and '0307'
 where cd1.grp_cd ='80027' order by cd2.expo_sort ,cd1.expo_sort ;
```

8 분야별 웹툰만화 연재 타입

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '02' as filter_tar_gb , cd2.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '08' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.uppr_cd='03' and cd2.cmmn_cd between '0302'
and '0307'
 where cd1.grp_cd ='80067' order by cd2.expo_sort ,cd1.expo_sort ;
```

9 분야별 웹툰만화 연재 타입

```
insert into TS_filter_set( filter_tar_gb, main_cat, sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '02' as filter_tar_gb , cd2.uppr_cd , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '08' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.uppr_cd='03' and cd2.cmmn_cd between '0302'
and '0307'
```

```
 where cd1.grp_cd ='80067' order by cd2.expo_sort ,cd1.expo_sort ;
```

9 채널예스 분야 (09) [명사의서재 제외]

```
insert into TS_filter_set( filter_tar_gb, main_cat,   sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   ,expo_sort)
with data1 as (
 SELECT cd_no, left(nm,2)   as nm FROM TG_GL_CD WHERE GB='ARTICLE_TYPE' AND DEL_YN='N' AND
DPTH=2
  )
  select
      '02' as filter_tar_gb , cd1.uppr_cd , cd1.cmmn_cd as sub_cat, cd1.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '09' as filter_type ,
    cd_no as   filter_cd , nm as filter_expo_nm, 'N' ,cd_no
from ts_cmmn_cd cd1
join data1 as d1
 where cd1.grp_cd ='80004' and   cd1.uppr_cd='05'    and cd1.cmmn_cd between '0501' and '0503'
 order by cd1.expo_sort ,cd_no ;
```

10 채널예스 카테고리 (05)

```
insert into TS_filter_set( filter_tar_gb, main_cat,   sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   ,expo_sort)
with data1 as (
 SELECT PAT_NO, cate_no, ttl ,sort   FROM TG_GL_CATE WHERE   DEL_YN='N'
  )
  select
      '02' as filter_tar_gb , cd1.uppr_cd , cd1.cmmn_cd as sub_cat, cd1.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '05' as filter_type ,
    cate_no as   filter_cd , ttl as filter_expo_nm, 'N' ,sort
from ts_cmmn_cd cd1
     -- join ts_cmmn_cd cd2 on cd2.grp_cd = '80004' and cd2.cmmn_cd ='0401'
   join data1 as d1   on d1.pat_no=cd1.rfrc_cd
 where cd1.grp_cd ='80004' and   cd1.uppr_cd='05'    and cd1.cmmn_cd between '0501' and '0503'
 and cate_no not in (2454)
 order by cd1.expo_sort ,sort ;
```

11 스냅 메뉴

```
insert into TS_filter_set( filter_tar_gb, main_cat,   sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   ,expo_sort)
with data1 as (
   SELECT     STEP1,STEP2, MENUNM ,sort
   FROM ti_ist24_user_menu_info WHERE 1=1
 AND LINKYN='Y' AND STEP2>0
   )
   select
        '02' as filter_tar_gb , cd1.uppr_cd , cd1.cmmn_cd as sub_cat, cd1.cmmn_cdnm as sub_cat_nm , null
as dtl_cat, '01' as filter_type ,
     STEP2 as   filter_cd , MENUNM as filter_expo_nm, 'N' ,sort
from ts_cmmn_cd cd1
join data1 as d1   on d1.step1 = cd1.rfrc_cd
 where cd1.grp_cd ='80004' and   cd1.uppr_cd='06'     and cd1.cmmn_cd between '0601' and '0604'
 order by cd1.expo_sort ,sort ;
```

12 보관함 작품이용 (04) 지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)   ;
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '04' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
   join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01')
 where cd1.grp_cd ='80017'   and cd1.cmmn_cd in ('1','2')
 order by cd2.expo_sort, cd1.expo_sort ;
```

13 보관함 작품타입 (02) 지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '02' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
   join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01','05','06')
```

```
 where cd1.grp_cd ='80015'
 order by cd2.expo_sort, cd1.expo_sort ;
```

14 보관함 연재 작품이용 유무료 (11) 지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '11' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01')
 where cd1.grp_cd ='80079'
 order by cd2.expo_sort, cd1.expo_sort ;
```

15 보관함 업데이트알림 (03) 지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '03' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01')
 where cd1.grp_cd ='80064'
 order by cd2.expo_sort, cd1.expo_sort ;
```

16 보관함 보관함 프로모션 타입 (06) 지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
with data1 as (
select cmmn_cd,cmmn_cdnm,grp_cd   ,expo_sort from ts_cmmn_cd where grp_cd ='80027' and
cmmn_cd !=3
union
select '0', '일반' , 80027 , 0 from dual
```

```
)
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '06' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from data1 cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01','06')
 where cd1.grp_cd ='80027'
 order by cd2.expo_sort, cd1.expo_sort ;
```

17 보관함 업데이트알림 (03) 지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '03' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01')
 where cd1.grp_cd ='80064'
 order by cd2.expo_sort, cd1.expo_sort ;
```

18 보관함 연재타입 지정 (08)지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '08' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01')
 where cd1.grp_cd ='80067'
 order by cd2.expo_sort, cd1.expo_sort ;
```

19 보관함 ebook 카테고리(05) 지정

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type,
filter_cd, filter_expo_nm, del_yn   , expo_sort)
select '09' as filter_tar_gb , '09' , cd2.cmmn_cd as sub_cat, cd2.cmmn_cdnm as sub_cat_nm , null as
dtl_cat, '08' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80005' and cd2.del_yn ='N'   and cd2.cmmn_cd in('01')
 where cd1.grp_cd ='80067'
 order by cd2.expo_sort, cd1.expo_sort ;
```

20 펜살롱 공개여부 등록

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type, filter_cd,
filter_expo_nm, del_yn   , expo_sort)
 select '03' as filter_tar_gb , '03' AS main_cat , '01' as sub_cat, '작가의룸 글관리 발행' as sub_cat_nm ,
null as dtl_cat, cd2.cmmn_cd as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80063' and cd2.del_yn ='N'   and cd2.cmmn_cd in('10')
 where cd1.grp_cd ='80065'
 order by cd2.expo_sort, cd1.expo_sort ;
```

21 펜살롱 신고여부 등록

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type, filter_cd,
filter_expo_nm, del_yn   , expo_sort)
 select '03' as filter_tar_gb , '03' AS main_cat , '01' as sub_cat, '작가의룸 글관리 발행' as sub_cat_nm ,
null as dtl_cat, '12' as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80063' and cd2.del_yn ='N'   and cd2.cmmn_cd in('12')
 where cd1.grp_cd ='80081'
 order by cd2.expo_sort, cd1.expo_sort ;
```

21 24.1 필터 항목 공개여부 등록

```
insert into TS_filter_set( filter_tar_gb, main_cat,sub_cat,sub_cat_nm, dtl_cat, filter_type, filter_cd,
filter_expo_nm, del_yn   , expo_sort)
 select '04' as filter_tar_gb , '04' AS main_cat , '01' as sub_cat, '24.1 나만의 hz' as sub_cat_nm , null as
dtl_cat, cd2.cmmn_cd as filter_type ,
cd1.cmmn_cd as   filter_cd , cd1.cmmn_cdnm as filter_expo_nm, 'N', cd1.expo_sort
from ts_cmmn_cd cd1
  join ts_cmmn_cd cd2 on cd2.grp_cd = '80063' and cd2.del_yn ='N'   and cd2.cmmn_cd in('10')
 where cd1.grp_cd ='80065'
 order by cd2.expo_sort, cd1.expo_sort ;
```

22 기존 분야별 필터 복사 예시

```
insert into filter_tar_gb(, filter_tar_gb, main_cat, main_cat_nm, sub_cat, sub_cat_nm, filter_type,
filter_type_nm, filter_cd, filter_expo_nm, expo_sort, del_yn)
select   filter_tar_gb, main_cat, main_cat_nm, {2level 메뉴코드 ex)'0208'}, {2level 메뉴명 ex)'로판 3'},
filter_type, filter_type_nm, filter_cd, filter_expo_nm, expo_sort, del_yn from ts_filter_set
where filter_tar_gb='02' and main_cat='02' and sub_cat='0202'   and del_yn='N' ;
```

분야별 메뉴 추가시 절차 및 유의사항

1. 연재카테고리 생성
2. 연재 카테고리 매핑
3. 공통코드(80004) 메뉴용용 코드 추가 (서비스 대상 연재카테고리 필수 입력) : 설정완료시까지 미사용으로 지정
4. 분야별 전시관리 전시내용 설정
5. 사용 필터 지정 (Database ts_filter_set 데이터 추가)
6. 통합검색 페이지 필터지정 등 관련 프로그램 수정
7. 설정 완료 후 공통 코드 메뉴 미사용을 사용으로 전환

6.CMS 메뉴 추가 절차

6.1. 프로그램 등록

- 신규 프로그램 정보를 등록 한다. (ts_cms_prgm_mng)

컬럼명	설명
prgm_seq	프로그램 순차번호

prgm_nm	프로그램 명
URL	url
method	메소드[70002]
prgm_descr	프로그램 설명
oprtn_status_gb	운영 현황 구분
del_yn	삭제여부
prgm_seq	프로그램 순차번호
prgm_nm	프로그램 명
URL	url
method	메소드[70002]
prgm_descr	프로그램 설명
oprtn_status_gb	운영 현황 구분
del_yn	삭제여부

- 신규 프로그램 등록 예시

```
insert into ts_cms_prgm_mng(prgm_nm,url,method,prgm_descr,del_yn )
values('스토리 ON>콘텐츠 추가/수정','/storyon/mStoryonContAdd','GET','스토리 ON>콘텐츠
추가/수정' ,'N' , 'O'  );
```

6.2. 메뉴 셋 등록

- 신규 메뉴 정보를 등록 한다.(ts_cms_menu_set)

컬럼명	설명
menu_seq	메뉴 순차번호
menu_nm	메뉴 명
param1	파라미터 1
param2	파라미터 2
param3	파라미터 3
font_awesome	fontawesome (https://fontawesome.com/icons?d=gallery) 에서 선택 적용. 1level 메뉴에서만 사용됨
uppr_menu_seq	상위 메뉴 순차번호
menu_lvl	메뉴 레벨
menu_tar_yn	메뉴 대상 여부
oprtn_status_gb	운영 현황 구분 (O 운영, S 현황)
expo_sort	노출 순서
prgm_seq	프로그램 순차번호

menu_set	메뉴 세트
del_yn	삭제여부

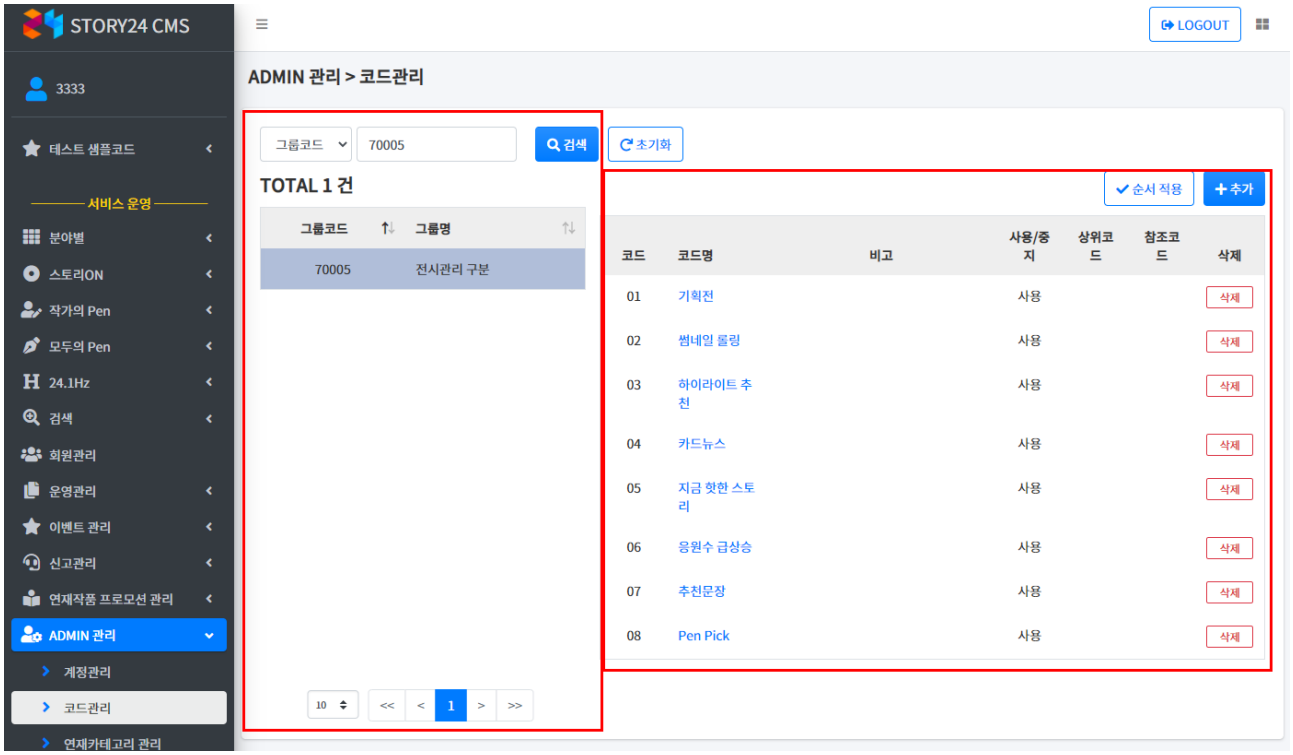
- 신규 메뉴 등록 예시

```
-- 1 level
insert into TS_cms_menu_set(menu_nm, font_awesome, uppr_menu_seq, menu_lvl, menu_tar_yn,
oprtn_status_gb, expo_sort, prgm_seq, del_yn)
values('1level menu 추가','fa-address-book',null,1,'Y','O',99,null, 'N' ) ;

-- 2 level
insert into TS_cms_menu_set(menu_nm, font_awesome, uppr_menu_seq, menu_lvl, menu_tar_yn,
oprtn_status_gb, expo_sort, prgm_seq, del_yn)
values('2level menu 추가',null,{1level menu 번호},2,'Y','O',99,{TS_cms_prgm_mng.prgm_seq}, 'N' ) ;
```

7.공통코드 관리

7.1. UI 공통



7.2. 그룹코드

컬럼명	설명
그룹코드(grp_cd)	그룹코드. 상세코드 조회 기준 key 값
그룹명(grp_cdnm)	그룹코드에 대한 명칭,설명
그룹코드설명 (grp_cd_descr)	그룹코드에 대한 상세 부연설명
상위코드(uppr_cd)	코드간 종속관계가 있는경우 참고할 수 있도록 추가 지정 (예:분야 메인코드, 분야 서브코드 또는 게시판코드, 게시판 header 코드)

-주요 공통 코드

그룹코드	그룹코드명
70001	관리자 권한
70002	메소드
70003	메인 전시항목
70004	관리자 데이터 조회건수
70005	전시관리 구분
70006	연결링크
70007	전시여부
70008	스토리온 검색어 구분
70009	검색키워드 선택구분
70010	검색 작품 선정 기준
70011	cms 데이터 검색어 구분
70012	24.1 주파수 검색어
70013	유료화차 지정 구분
70017	관리자 제한사유
70018	환불 사유
70020	연재상태구분
70021	프로모션 상태구분
70022	카테고리 상태
70023	작가 활동구분
70024	확인여부
70025	작가승인 반려사유
70026	cms 관리자 활동내역
70027	CMS 검색 작품선정 대상
70028	cms 검색 작품 목록 구분
70029	작가 활동제한사유

70030	분야별 전시관리 설정
70031	캐시 결제 OS
70033	환불상태
80001	영화지정구분
80002	스토리온 카테고리
80003	분야별 메인 분류
80004	분야별 서브 분류
80005	보관함 카테고리
80006	노출제한 ebook 전시분류 코드
80008	카테고리그룹
80009	북클럽 하위 카테고리
80010	연재 요일
80011	코너 외 섹션
80012	뷰어 종류
80013	요청상태
80015	연재작품타입
80017	대여 소장 구분
80018	전시 자료 구분
80023	포스 주제
80024	포스트 상세주제
80027	이용권 구분
80030	회원 지정정보구분
80031	정보구분
80032	사용자 액션 구분
80034	팔로잉 정보구분
80035	포스트 전시관리 대상 구분
80036	포스트 전시관리 대상 전시타입
80037	상품추천구분
80038	랭킹주기
80039	포스트배너구분
80040	모두의펜 동영상 링크구분
80041	게시판 구분
80042	게시판 머리말
80046	콘텐츠 추천 구분
80047	링크정보 검색구분
80050	랭킹구분
80051	랭킹주기
80052	캐시소멸구분

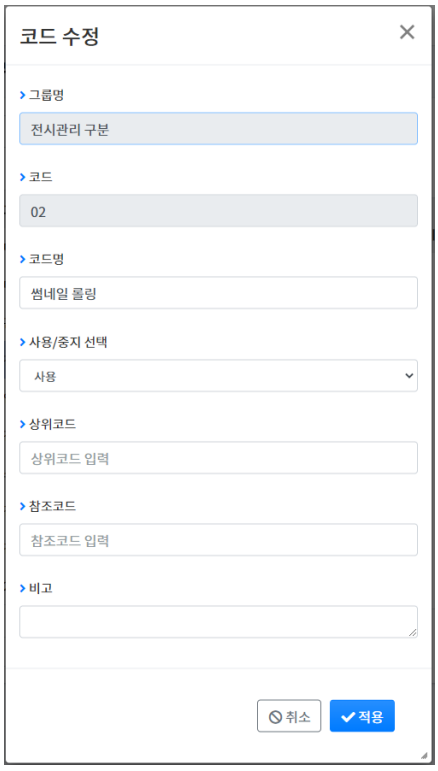
80053	콘텐츠 노출지정 작품노출구분
80055	랭킹 산출 기준
80061	데이터 정열순서
80062	필터대상구분
80063	필터 타입
80064	구독설정여부(알림)
80065	공개여부
80066	아이스타일 스트리트뷰 나라명
80067	연재타입
80068	프로모션타입(일반,선연재,무료)
80069	캐시충전방법
80072	결제상태
80073	작가구분
80074	캐시구분(s 캐시,p 캐시)
80075	신고하기 사유
80076	연재작품 읽기 방향
80077	연재작품 서비스 상태
80078	연재작품 작품관리 상태
80079	작품이용(유료무료)
80080	자동충전 대상 잔여 캐시
80081	필터 신고여부
80090	파일등록구분
80096	회원 알림 구분
80097	이벤트 팝업 알림 구분
80098	공유 타입
80099	공유 대상
80113	테그 분류 구분
80115	작가의펜 작가 배경색
80116	작가의펜 보관함 정보 구분
80131	sey 코인 지급 사유
80132	코인 지급 사용구분
80133	쿠폰 구분
80134	쿠폰 발급 구분
80135	코인 지급처
80136	쿠폰상태
80137	캐시상태
80201	스토리온 전시순서구분
80206	상세보기 타입

80207	스토리온 제목 배경 색상
80300	예스 24 작가 구분
90004	포스트 메인 노출 플래그(독자입장)
90005	본인 포스트 플래그 종류(작가입장)
90006	포스트작가 상태
90007	포스트 타입
90008	작가의펜 고정메뉴
90009	포스트 배너 플래그 구분
90010	작가의룸 tab 구분
90011	배경 사용 구분(컬러, 이미지)
90012	작가의 펜 정보 구분
90013	모두의펜 고정메뉴

-그롭코트 추가 쿼리 샘플

```
insert into TS_cmmn_grp_cd ( grp_cd,grp_cdnm,grp_cd_descr,uppr_cd,del_yn,reg_dts)
 values( '80004','분야별 서브 분류','분야별 서브 그룹코드를 정의 함','80003','N' ,now() );
```

7.3. 상세코드



구분	설명

그룹코드	상세코드의 그룹 코드
코드명	상세코드 코드명
사용/중지 선택	사용 중지여부 지정. 기존 사용한 코드의 값은 유효하나 신규로 해당 코드를 지정할 수 없음
상위코드(uppr_cd)	해당 코드값 간의 구분을 위하여 상위구분 값 지정 (예:게시판 header 지정)
참조코드(rfrc_cd)	해당 코드의 연결 key 값의 지정이 필요한 경우 사용 (예:분야서브분류의 참조코드로 콘텐츠별 연결 코드값 지정)
정열순서(expo_sort)	노출 순서

- 코드 설정값의 노출 순서를 변경하고자 하는 경우 마우스 dragAnddrop 후 순서적용 클릭하여 변경

8.소스 배포 절차

8.1. 개발 소스 stage 반영

```
git pull origin stage

git merge dev

git push origin stage
```

8.2. Stage 소스 prod 반영

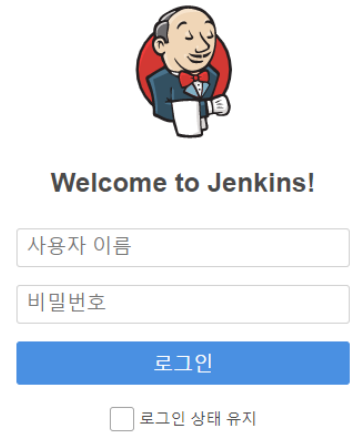
```
git pull origin prod

git merge stage

git push origin prod
```

8.3. Prod Jenkins 빌드

- http://jenkins.story24.yes24.com 접속
- 사이트 로그인 (id:kotech / pwd:kotech1234)

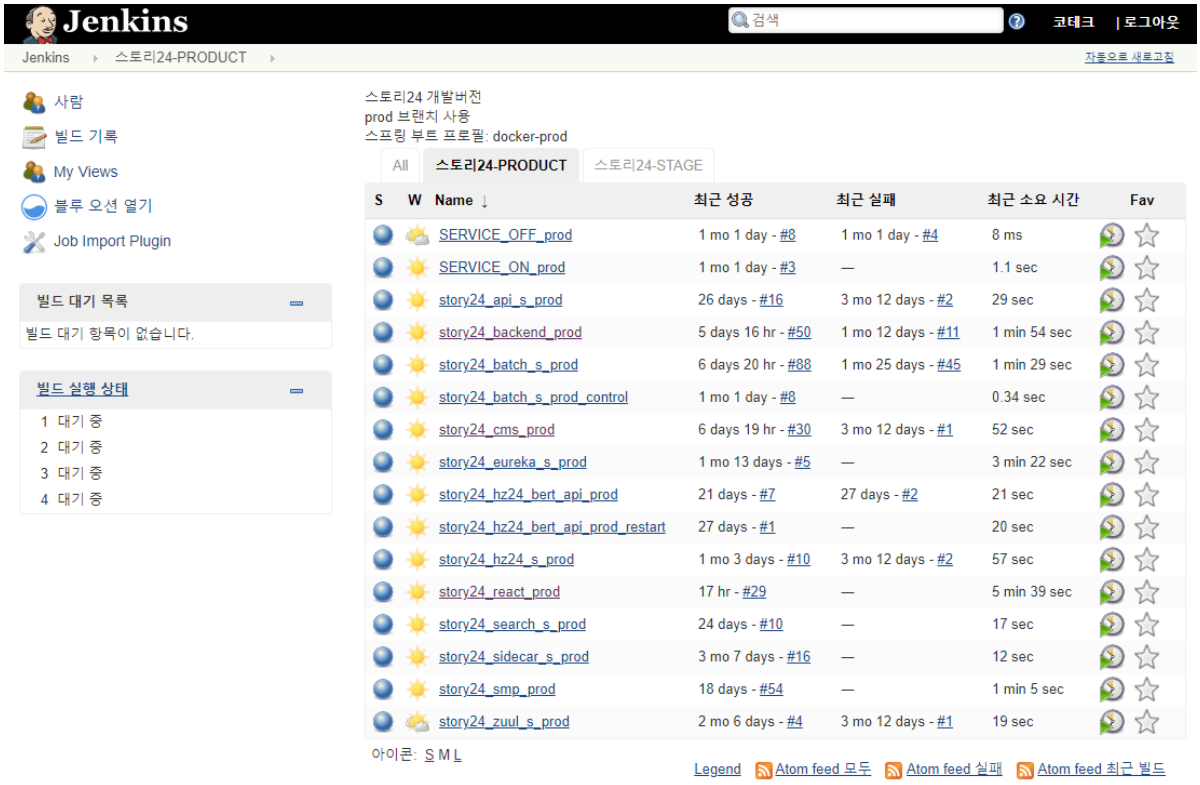


- 빌드대상 name 클릭

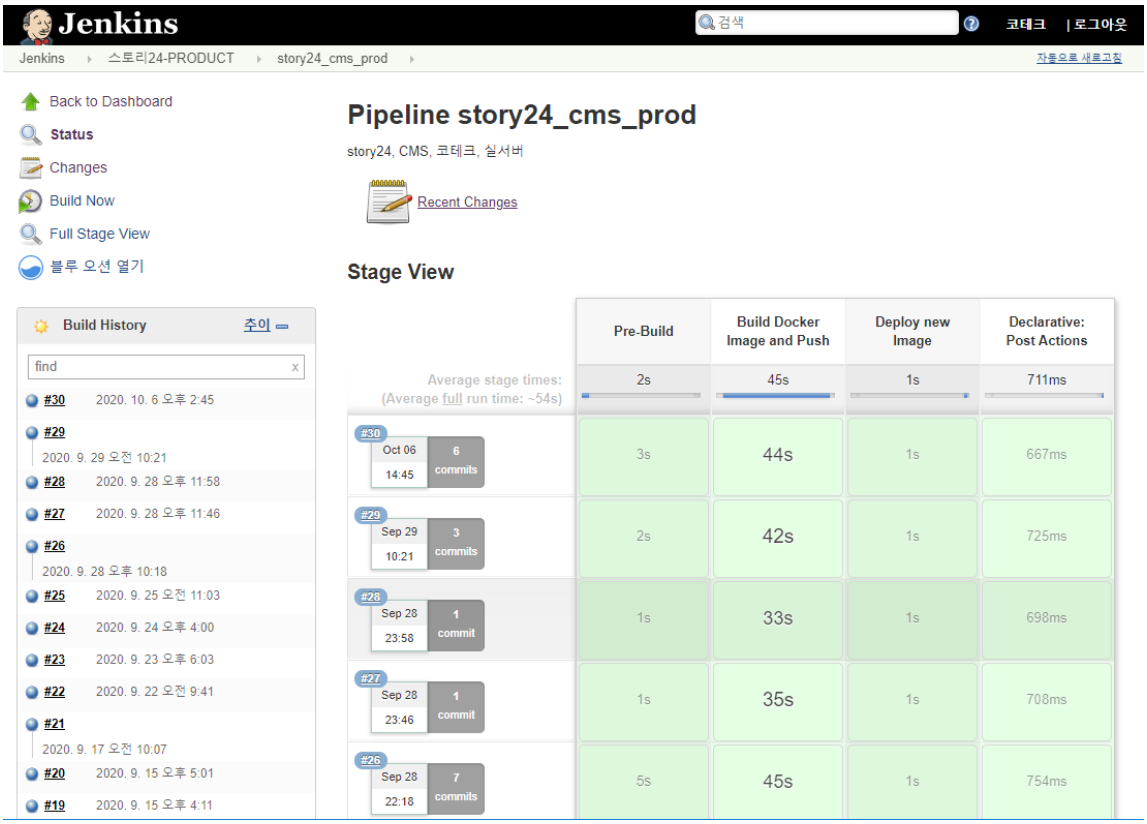
Story24_backend_prod : story24 backend 서비스 live

Story24_cms_prod : story24 관리자 서비스 live

Story24_react_prod : story24 front 서비스 live



- Build Now 클릭


Jenkins
검색
코테크
| 로그아웃
Jenkins ▸ 스토리24-PRODUCT ▸ story24_cms_prod ▸
자동으로 새로고침
Back to Dashboard
Status
Changes
Build Now
Full Stage View
블루 오션 열기
Build History
추이
find
#30 2020. 10. 6 오후 2:45
#29 2020. 9. 29 오전 10:21
#28 2020. 9. 28 오후 11:58
#27 2020. 9. 28 오후 11:46
#26 2020. 9. 28 오후 10:18
#25 2020. 9. 25 오전 11:03
#24 2020. 9. 24 오후 4:00
#23 2020. 9. 23 오후 6:03
#22 2020. 9. 22 오전 9:41
#21 2020. 9. 17 오전 10:07
#20 2020. 9. 15 오후 5:01
#19 2020. 9. 15 오후 4:11
Pipeline story24_cms_prod
story24, CMS, 코테크, 실서버
Recent Changes
Stage View
Pre-Build
Build Docker Image and Push
Deploy new Image
Declarative: Post Actions
Average stage times: (Average full run time: ~54s)
2s
45s
1s
711ms
#30 Oct 06 14:45 6 commits 3s 44s 1s 667ms
#29 Sep 29 10:21 3 commits 2s 42s 1s 725ms
#28 Sep 28 23:58 1 commit 1s 33s 1s 698ms
#27 Sep 28 23:46 1 commit 1s 35s 1s 708ms
#26 Sep 28 22:18 7 commits 5s 45s 1s 754ms