**🎮 Switch Manga API**

디지털 만화/망가 플랫폼을 위한 RESTful API 서버

**📋 프로젝트 개요**

**Switch Manga**는 전 세계 일본 망가 배급을 목표로 하는 종합 디지털 만화 플랫폼입니다. 8년 된 레거시 시스템을 현대화하여 3개 하드코딩 출판사에서 수백 개의 출판사로 확장할 수 있도록 설계되었습니다.

**🎯 핵심 기능**

* **출판사 관리**: 다중 출판사 콘텐츠 관리
* **시리즈/볼륨 관리**: 만화 시리즈 및 권호 관리
* **사용자 관리**: 회원 가입, 인증, 권한 관리
* **수익 공유**: 출판사 70% / 플랫폼 30% 수익 배분
* **Action Viewer**: 자동 패널 네비게이션 기능 (차별화 요소)

**🏗️ 기술 스택**

* **Backend**: Spring Boot 3.x, Java 17
* **Database**: MariaDB 11
* **Container**: Docker (PHP 8.2-FPM, Nginx 1.27)
* **ORM**: Spring Data JPA / Hibernate
* **Build**: Gradle
* **Documentation**: Swagger/OpenAPI 3.0

**🚀 시작하기**

**사전 요구사항**

* Java 17 이상
* Gradle 8.x
* MariaDB 11.x
* Docker & Docker Compose (선택사항)

**설치 방법**

**1. 저장소 클론**

git clone https://github.com/your-org/switch-manga-api.git

cd switch-manga-api

**2. 데이터베이스 설정**

**로컬 환경 (개발)**

CREATE DATABASE switchmanga CHARACTER SET utf8mb4 COLLATE utf8mb4\_unicode\_ci;

CREATE USER 'switchmanga'@'localhost' IDENTIFIED BY '2025';

GRANT ALL PRIVILEGES ON switchmanga.\* TO 'switchmanga'@'localhost';

FLUSH PRIVILEGES;

**3. 애플리케이션 설정**

application.yml 파일에서 프로파일 선택:

* **local**: 로컬 개발 환경
* **prod**: GCP 운영 환경

**4. 빌드 및 실행**

# 프로젝트 빌드

./gradlew clean build

# 애플리케이션 실행

./gradlew bootRun

# 또는 JAR 파일 실행

java -jar build/libs/switch-manga-api-0.0.1-SNAPSHOT.jar

**5. API 문서 접근**

애플리케이션 실행 후:

* Swagger UI: http://localhost:8080/swagger-ui.html
* OpenAPI Spec: http://localhost:8080/v3/api-docs

**📚 API 엔드포인트**

**🧑 User API (/api/v1/users)**

사용자 관리 및 인증 기능

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| GET | /api/v1/users | 전체 회원 조회 |
| GET | /api/v1/users/{id} | 회원 상세 조회 |
| GET | /api/v1/users/email/{email} | 이메일로 조회 |
| GET | /api/v1/users/username/{username} | 사용자명으로 조회 |
| GET | /api/v1/users/role/{role} | 권한별 조회 |
| GET | /api/v1/users/status/{status} | 상태별 조회 |
| GET | /api/v1/users/active/verified | 활성+인증 회원 조회 |
| GET | /api/v1/users/search/email?email={query} | 이메일로 검색 |
| GET | /api/v1/users/search/username?username={query} | 사용자명으로 검색 |
| GET | /api/v1/users/count/role/{role} | 권한별 회원 수 |
| GET | /api/v1/users/count/status/{status} | 상태별 회원 수 |
| POST | /api/v1/users | 회원 가입 |
| PUT | /api/v1/users/{id} | 회원 정보 수정 |
| PATCH | /api/v1/users/{id}/password | 비밀번호 변경 |
| PATCH | /api/v1/users/{id}/role | 권한 변경 |
| PATCH | /api/v1/users/{id}/status | 상태 변경 |
| PATCH | /api/v1/users/{id}/verify | 이메일 인증 |
| PATCH | /api/v1/users/{id}/last-login | 마지막 로그인 시간 업데이트 |
| DELETE | /api/v1/users/{id} | 회원 삭제 |

**📝 User 요청 예시**

POST /api/v1/users

{

"username": "john\_doe",

"email": "john@example.com",

"password": "securePassword123",

"role": "READER",

"status": "ACTIVE"

}

**📚 Publisher API (/api/v1/publishers)**

출판사 관리 기능

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| GET | /api/v1/publishers | 전체 출판사 조회 |
| GET | /api/v1/publishers/{id} | 출판사 상세 조회 |
| GET | /api/v1/publishers/active | 활성화된 출판사만 조회 |
| GET | /api/v1/publishers/country/{country} | 국가별 출판사 조회 |
| POST | /api/v1/publishers | 출판사 생성 |
| PUT | /api/v1/publishers/{id} | 출판사 수정 |
| DELETE | /api/v1/publishers/{id} | 출판사 삭제 |

**📝 Publisher 요청 예시**

POST /api/v1/publishers

{

"name": "Shueisha",

"country": "Japan",

"contactEmail": "contact@shueisha.co.jp",

"isActive": true

}

**📖 Series API (/api/v1/series)**

만화 시리즈 관리 기능

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| GET | /api/v1/series | 전체 시리즈 조회 |
| GET | /api/v1/series/{id} | 시리즈 상세 조회 |
| GET | /api/v1/series/publisher/{publisherId} | 출판사별 시리즈 조회 |
| GET | /api/v1/series/active | 활성화된 시리즈만 조회 |
| GET | /api/v1/series/status/{status} | 상태별 시리즈 조회 |
| GET | /api/v1/series/search?title={query} | 제목으로 검색 |
| GET | /api/v1/series/search/author?author={query} | 작가로 검색 |
| POST | /api/v1/series?publisherId={id} | 시리즈 생성 |
| PUT | /api/v1/series/{id} | 시리즈 수정 |
| PATCH | /api/v1/series/{id}/publisher | 출판사 변경 |
| DELETE | /api/v1/series/{id} | 시리즈 삭제 |

**📝 Series 요청 예시**

POST /api/v1/series?publisherId=1

{

"title": "One Piece",

"author": "Eiichiro Oda",

"description": "해적왕을 꿈꾸는 루피의 모험",

"genre": "Adventure, Fantasy",

"status": "ONGOING",

"isActive": true

}

**📗 Volume API (/api/v1/volumes)**

만화 권호 관리 기능

| **Method** | **Endpoint** | **Description** |
| --- | --- | --- |
| GET | /api/v1/volumes | 전체 Volume 조회 |
| GET | /api/v1/volumes/{id} | Volume 상세 조회 |
| GET | /api/v1/volumes/series/{seriesId} | 시리즈별 Volume 조회 |
| GET | /api/v1/volumes/series/{seriesId}/ordered | 시리즈별 Volume 조회 (권수 순) |
| GET | /api/v1/volumes/series/{seriesId}/number/{volumeNumber} | 특정 시리즈의 특정 권 조회 |
| GET | /api/v1/volumes/active | 활성화된 Volume만 조회 |
| GET | /api/v1/volumes/series/{seriesId}/active | 시리즈별 활성 Volume 조회 |
| GET | /api/v1/volumes/search?title={query} | 제목으로 검색 |
| GET | /api/v1/volumes/isbn/{isbn} | ISBN으로 조회 |
| GET | /api/v1/volumes/series/{seriesId}/count | 시리즈별 Volume 개수 |
| POST | /api/v1/volumes?seriesId={id} | Volume 생성 |
| PUT | /api/v1/volumes/{id} | Volume 수정 |
| PATCH | /api/v1/volumes/{id}/series | 시리즈 변경 |
| DELETE | /api/v1/volumes/{id} | Volume 삭제 |

**📝 Volume 요청 예시**

POST /api/v1/volumes?seriesId=1

{

"title": "One Piece Vol. 1",

"volumeNumber": 1,

"isbn": "978-4-08-872754-7",

"publishDate": "1997-12-24",

"price": 4.99,

"pageCount": 200,

"isActive": true

}

**🗂️ 데이터베이스 스키마**

**주요 테이블**

* **bt\_users**: 사용자 정보
* **bt\_publishers**: 출판사 정보
* **bt\_series**: 만화 시리즈 정보
* **bt\_volumes**: 만화 권호 정보
* **bt\_books**: 전자책 파일 정보
* **bt\_orders**: 주문 정보
* **bt\_sales**: 매출 정보

전체 스키마는 SwitchManga\_Database\_Schema.sql 참조

**🔧 개발 가이드**

**프로젝트 구조**

src/

├── main/

│ ├── java/com/switchmanga/api/

│ │ ├── controller/ # REST 컨트롤러

│ │ ├── service/ # 비즈니스 로직

│ │ ├── repository/ # 데이터 접근 계층

│ │ ├── entity/ # JPA 엔티티

│ │ ├── dto/ # 데이터 전송 객체

│ │ └── config/ # 설정 파일

│ └── resources/

│ ├── application.yml # 애플리케이션 설정

│ └── static/ # 정적 리소스

└── test/ # 테스트 코드

**개발 원칙**

1. **모듈화**: "Lego block" 아키텍처로 재사용 가능한 컴포넌트 구성
2. **분리**: 개발/운영 환경 엄격한 분리
3. **백업**: Git 버전 관리 및 롤백 전략
4. **호환성**: 레거시 자산 활용 및 하위 호환성 유지
5. **보안**: 로그인 시스템 재구축 (진행 중)

**코딩 스타일**

* Java: Google Java Style Guide 준수
* Git Commit: [Conventional Commits](https://www.conventionalcommits.org/) 사용
* 주석: 복잡한 비즈니스 로직에만 명확한 주석 작성

**🧪 테스트**

# 전체 테스트 실행

./gradlew test

# 특정 테스트 클래스 실행

./gradlew test --tests UserServiceTest

# 통합 테스트

./gradlew integrationTest

**🚀 배포**

**Docker로 배포**

# Docker 이미지 빌드

docker build -t switch-manga-api:latest .

# Docker 컨테이너 실행

docker run -p 8080:8080 \

-e SPRING\_PROFILES\_ACTIVE=prod \

switch-manga-api:latest

**GCP 배포**

현재 운영 서버:

* IP: 34.64.84.117
* 포트: 8081
* 환경: GCP (Google Cloud Platform)

# 운영 프로파일로 실행

java -jar -Dspring.profiles.active=prod \

switch-manga-api-0.0.1-SNAPSHOT.jar

**📊 모니터링**

**헬스 체크**

# 애플리케이션 상태 확인

curl http://localhost:8080/actuator/health

# 상세 정보

curl http://localhost:8080/actuator/info

**로깅**

* 로그 레벨: DEBUG (개발), INFO (운영)
* 로그 위치: logs/application.log
* Hibernate SQL 로깅: 활성화 (개발 환경)

**🛣️ 로드맵**

**Phase 1: 출판사 CMS 완료 ✅**

* 대시보드 기능
* 주문 관리 (검색/필터링)
* 매출/수익 추적 (통계/그래프)
* 시리즈 관리 (수익 상태 포함)
* 책 업로드 워크플로우 (ZIP 파일 추출/정리)

**Phase 2: Super Admin CMS 개발 🔄**

* 관리자 대시보드
* 전체 출판사 관리
* 시스템 모니터링

**Phase 3: 모바일 앱 통합 📅**

* Flutter 기반 모바일 앱
* **Action Viewer** 통합 (핵심 차별화 기능)
* 사용자 경험 최적화

**주요 마일스톤**

* 🎯 2024년 12월: 프로토타입 완료 (Action Viewer 포함 필수)
* 🚀 2025년 3월: 첫 프로덕션 버전 출시

**🤝 기여하기**

1. Fork the repository
2. Create your feature branch (git checkout -b feature/AmazingFeature)
3. Commit your changes (git commit -m 'Add some AmazingFeature')
4. Push to the branch (git push origin feature/AmazingFeature)
5. Open a Pull Request

**📄 라이선스**

This project is proprietary software. All rights reserved.

**📞 연락처**

**프로젝트 관리자**:

* GitHub: [@your-username](https://github.com/your-username)
* Email: your-email@example.com

**이슈 보고**:

* GitHub Issues: [Issues](https://github.com/your-org/switch-manga-api/issues)

**🙏 감사의 말**

* Spring Boot 팀
* MariaDB 커뮤니티
* Docker 커뮤니티

<div align="center"> <strong>🎮 Switch Manga - 전 세계 망가를 하나로</strong> </div>