CodeIT Sprint: Spring - 2nd Project
| 2025.04.16-2025.05.12

The 2nd CodeIT Sprint project

MONEW

모두의 뉴스

PRESENTED BY TEAM 8

프로젝트 소개

- 다양한 뉴스 API를 활용하여 최신 뉴스 콘텐츠 제공
- 사용자 간 의견을 나눌 수 있는 소통 중심의 소셜 기능 탑재
- 개인 관심사 기반 맞춤형 뉴스 추천 기능을 통해 정보 탐색 효율 향상

개발 환경 및 사용 스택

GitHub | Discord | Notion

Swagger | Figma | Postman

Git | AWS | Docker

SpringBoot | PostgreSQL | MongoDB

CodeRabbit | Codecov

프로젝트 구조


monewteam08
main
controller
service
repository
entity
dto
mapper
config
exception
event
scheduler
util
resources
application.yml
application-dev.yml
application-prod.yml
test
controller
service
repository

팀원 소개 및 역할

한상은

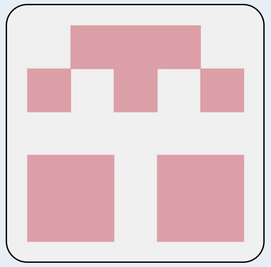
팀장, CI/CD 파이프라인,
사용자 관리, 활동 내역 관리

신은섭

DB 관리
뉴스 기사 관리

박태식

관심사 관리, 시연 영상



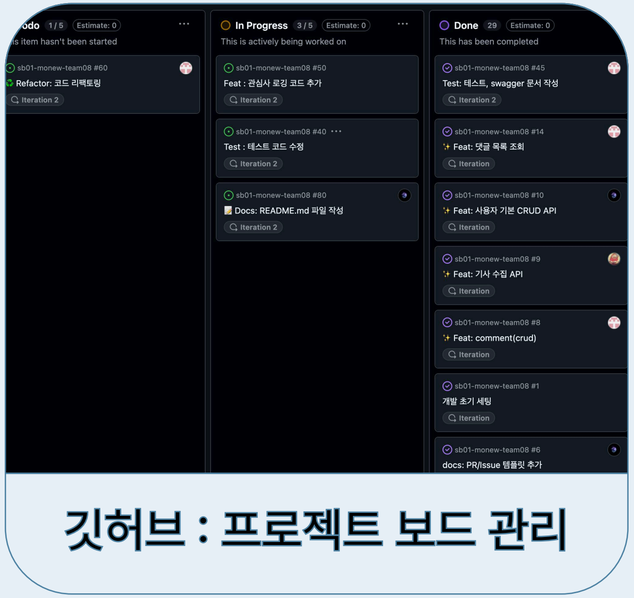
김창우

테스트 관리
댓글 관리, 알림 관리

협업 방식

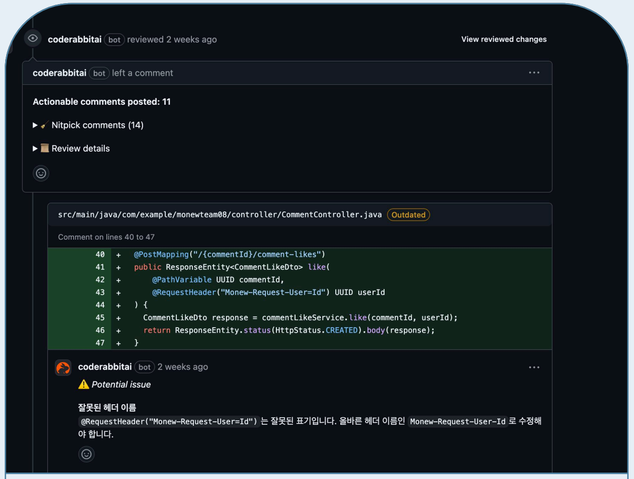
- 노션 이용한 희의록 관리
- 깃허브 프로젝트 보드를 이용한 일정 관리
- 매일 9시 데일리 스크럼 : 오늘 할일 공유
- 매일 6시 PR 리뷰 및 질의응답 : 코드 리뷰 진행
- 코드리뷰 자동화(Coderabbit) 활용

노션 : 회의록 작성



깃허브 : 프로젝트 보드 관리

데일리 스크럼 진행



코드리뷰 자동화 : Coderabbit

API 명세서

관심사 관리 관심사 관련 API

- GET /api/interests 관심사 목록 조회
- POST /api/interests 관심사 등록
- DELETE /api/interests/{interestId} 관심사 물리 삭제
- PATCH /api/interests/{interestId} 관심사 정보 수정
- POST /api/interests/{interestId}/subscriptions 관심사 구독
- DELETE /api/interests/{interestId}/subscriptions 관심사 구독 취소

뉴스 기사 관리 뉴스 기사 관리 API

- GET /api/articles 기사 목록 조회
- DELETE /api/articles/{articleId} 기사 논리 삭제
- POST /api/articles/{articleId}/article-views 기사 뷰 등록
- DELETE /api/articles/{articleId}/hard 기사 물리 삭제
- GET /api/articles/restore 뉴스 복구
- GET /api/articles/sources

댓글 관리 댓글 관리 API

- GET /api/comments 댓글 목록 조회
- POST /api/comments 댓글 등록
- DELETE /api/comments/{commentId} 댓글 논리 삭제
- PATCH /api/comments/{commentId} 댓글 정보 수정
- POST /api/comments/{commentId}/comment-likes 좋아요
- DELETE /api/comments/{commentId}/comment-likes 좋아요 취소
- DELETE /api/comments/{commentId}/hard 댓글 물리 삭제

사용자 관리 사용자 관련 API

- POST /api/users 회원가입
- DELETE /api/users/{userId} 사용자 논리 삭제
- PATCH /api/users/{userId} 사용자 정보 수정
- DELETE /api/users/{userId}/hard 사용자 물리 삭제
- POST /api/users/login 로그인

사용자 활동 내역 관리 사용자 활동 내역 관련 API

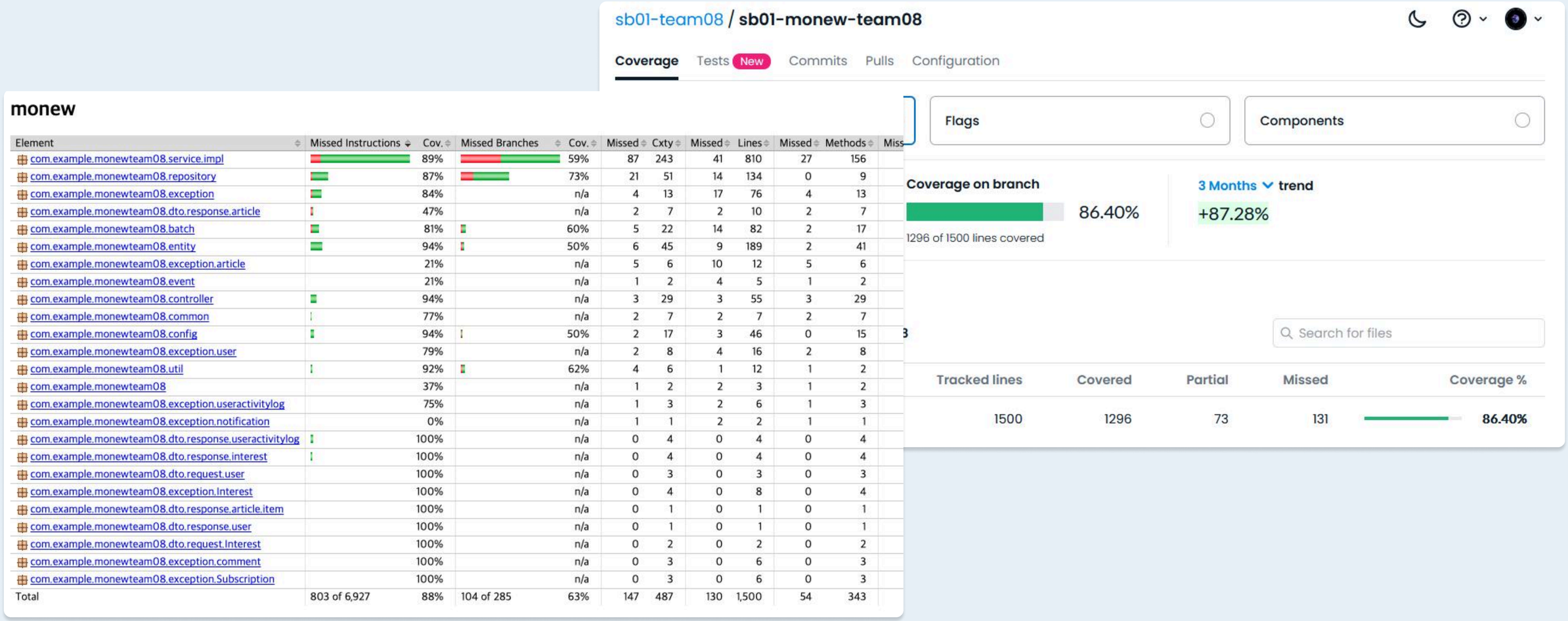
- GET /api/user-activities/{userId} 사용자 활동 내역 조회

알림 관리 알림 관리 API

- GET /api/notifications 알림 목록 조회
- PATCH /api/notifications 전체 알림 확인
- PATCH /api/notifications/{notificationId} 알림 확인

TDD 주도 개발

87%에 달하는 높은 테스트 커버리지 달성



sb01-team08 / sb01-monew-team08

Coverage | Tests New | Commits | Pulls | Configuration

Flags | Components

Coverage on branch 86.40%

1296 of 1500 lines covered

3 Months trend +87.28%

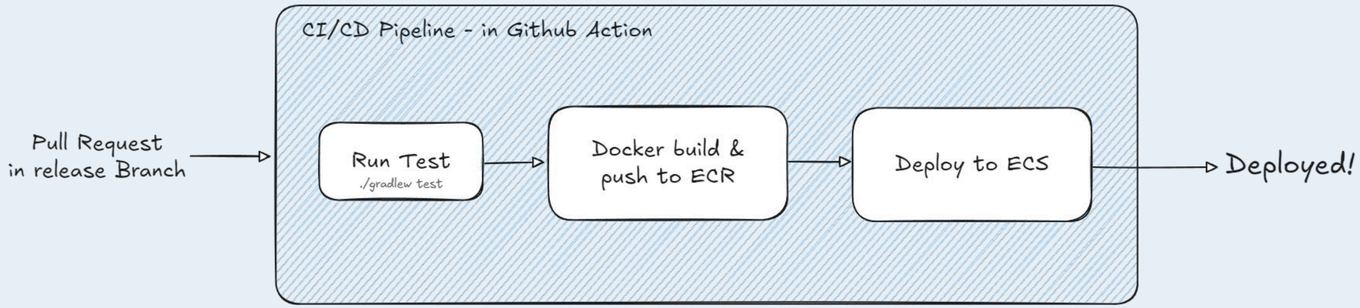
Search for files

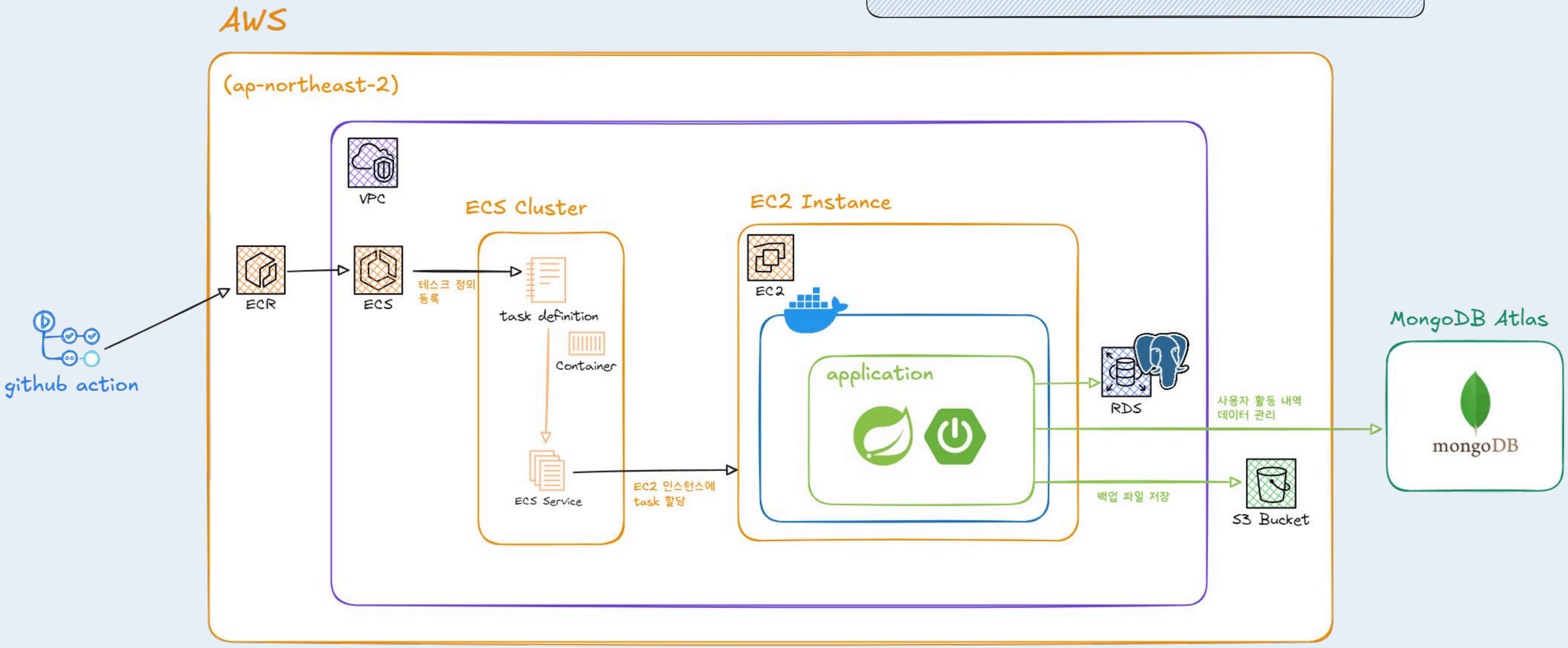
Tracked lines	Covered	Partial	Missed	Coverage %
1500	1296	73	131	86.40%

monew

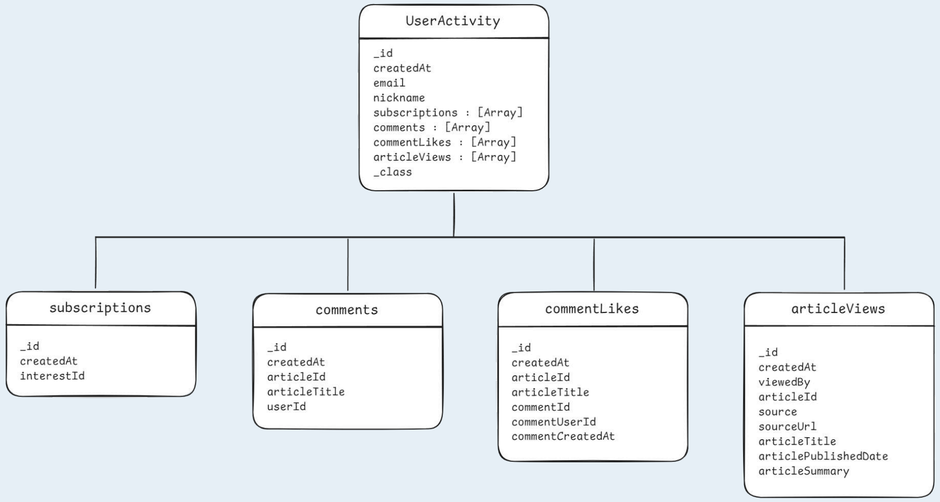
Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Miss
com.example.monewteam08.service.impl		89%		59%	87	243	41	810	27	156	
com.example.monewteam08.repository		87%		73%	21	51	14	134	0	9	
com.example.monewteam08.exception		84%		n/a	4	13	17	76	4	13	
com.example.monewteam08.dto.response.article		47%		n/a	2	7	2	10	2	7	
com.example.monewteam08.batch		81%		60%	5	22	14	82	2	17	
com.example.monewteam08.entity		94%		50%	6	45	9	189	2	41	
com.example.monewteam08.exception.article		21%		n/a	5	6	10	12	5	6	
com.example.monewteam08.event		21%		n/a	1	2	4	5	1	2	
com.example.monewteam08.controller		94%		n/a	3	29	3	55	3	29	
com.example.monewteam08.common		77%		n/a	2	7	2	7	2	7	
com.example.monewteam08.config		94%		50%	2	17	3	46	0	15	
com.example.monewteam08.exception.user		79%		n/a	2	8	4	16	2	8	
com.example.monewteam08.util		92%		62%	4	6	1	12	1	2	
com.example.monewteam08		37%		n/a	1	2	2	3	1	2	
com.example.monewteam08.exception.useractivitylog		75%		n/a	1	3	2	6	1	3	
com.example.monewteam08.exception.notification		0%		n/a	1	1	2	2	1	1	
com.example.monewteam08.dto.response.useractivitylog		100%		n/a	0	4	0	4	0	4	
com.example.monewteam08.dto.response.interest		100%		n/a	0	4	0	4	0	4	
com.example.monewteam08.dto.request.user		100%		n/a	0	3	0	3	0	3	
com.example.monewteam08.exception.Interest		100%		n/a	0	4	0	8	0	4	
com.example.monewteam08.dto.response.article.item		100%		n/a	0	1	0	1	0	1	
com.example.monewteam08.dto.response.user		100%		n/a	0	1	0	1	0	1	
com.example.monewteam08.dto.request.Interest		100%		n/a	0	2	0	2	0	2	
com.example.monewteam08.exception.comment		100%		n/a	0	3	0	6	0	3	
com.example.monewteam08.exception.Subscription		100%		n/a	0	3	0	6	0	3	
Total	803 of 6,927	88%	104 of 285	63%	147	487	130	1,500	54	343	

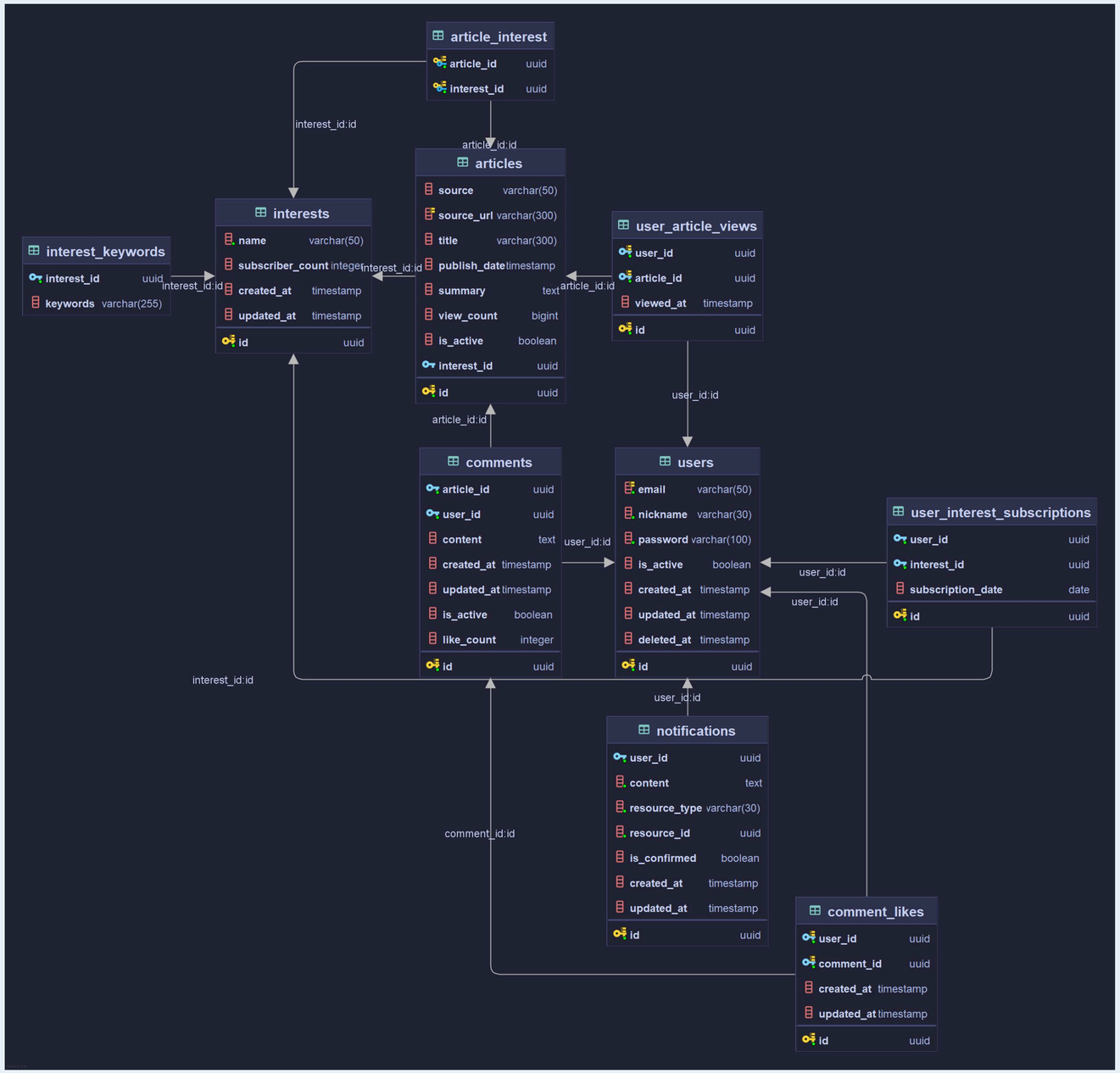
배포 파이프라인


CI/CD Pipeline - in Github Action
Pull Request
in release Branch
Run Test
./gradlew test
Docker build &
push to ECR
Deploy to ECS
Deployed!


AWS
(ap-northeast-2)
VPC
ECS Cluster
EC2 Instance
ECR
ECS
테스크 정의
등록
task definition
Container
ECS Service
EC2 인스턴스에
task 할당
EC2
application
RDS
사용자 활동 내역
데이터 관리
백업 파일 저장
S3 Bucket
MongoDB Atlas
mongoDB
github action

ERD 다이어그램


UserActivity
_id
createdAt
email
nickname
subscriptions : [Array]
comments : [Array]
commentLikes : [Array]
articleViews : [Array]
_class
subscriptions
_id
createdAt
interestId
comments
_id
createdAt
articleId
articleTitle
userId
commentLikes
_id
createdAt
articleId
articleTitle
commentId
commentUserId
commentCreatedAt
articleViews
_id
createdAt
viewedBy
articleId
source
sourceUrl
articleTitle
articlePublishedDate
articleSummary


article_interest
article_id uuid
interest_id uuid
interest_id:id
article_id:id
articles
source varchar(50)
source_url varchar(300)
title varchar(300)
publish_date timestamp
summary text
view_count bigint
is_active boolean
interest_id uuid
id uuid
interests
name varchar(50)
subscriber_count integer
created_at timestamp
updated_at timestamp
id uuid
interest_keywords
interest_id uuid
keywords varchar(255)
user_article_views
user_id uuid
article_id uuid
viewed_at timestamp
id uuid
user_id:id
comments
article_id uuid
user_id uuid
content text
created_at timestamp
updated_at timestamp
is_active boolean
like_count integer
id uuid
users
email varchar(50)
nickname varchar(30)
password varchar(100)
is_active boolean
created_at timestamp
updated_at timestamp
deleted_at timestamp
id uuid
user_interest_subscriptions
user_id uuid
interest_id uuid
subscription_date date
id uuid
notifications
user_id uuid
content text
resource_type varchar(30)
resource_id uuid
is_confirmed boolean
created_at timestamp
updated_at timestamp
id uuid
comment_likes
user_id uuid
comment_id uuid
created_at timestamp
updated_at timestamp
id uuid

시연 영상

시연 링크

트러블 슈팅#1

활동 내역 관련 N+1 문제 및 과도한 쿼리 호출

발생 원인

MongoDB는 변동이 적은 정적인 데이터만 저장,
자주 바뀌는 동적인 데이터(좋아요 수, 조회 수 등)은 요청 시점에 직접 조회하여 응답.
사용자 활동 내역 조회 과정에서 과도한 쿼리 요청 및 N+1 문제가 발생함

해결 방법

1. ID 리스트 기반 일괄 조회로 쿼리 호출 구조 변경
2. N+1이 생기는 데이터는 Fetch Join을 통해 즉시 로딩 처리

개선 사항

- 쿼리 호출 구조를 변경하며 약 65%(20→7)의 쿼리 요청 수 개선
- 관심사 키워드에서 발생하던 N+1 문제를 Fetch Join을 통해 해결

배운 점

데이터 설계와 쿼리 전략을 함께 고려하는 것이 중요하다는 점을 다시금 깨달았다.
특히, 원하는 기능을 구현하기 위해 시스템적으로 어떤 데이터 흐름과
조회 방식이 필요한지를 정확히 파악하는 역량의 중요성을 체감했다.

댓글 좋아요 중복 방지 처리

발생 원인

user_id와 comment_id 조합에 대한 DB 유니크 제약이 없음

서비스 로직에서 중복 요청에 대한 검증 로직 미흡

해결 방법

1. 데이터베이스 테이블에 (user_id, comment_id)에 대한 UNIQUE 제약 추가
2. 서비스 레이어에서 좋아요 요청 시 중복 여부를 먼저 확인하는 로직 보완

개선 사항

- API 설계를 idempotent하게 개선하여 같은 요청이 여러번 들어와도
  시스템 상태가 일관되도록 구현
- 데이터 무결성을 로직에만 의존하지 않고, DB제약 조건으로 이중 방어

배운 점

비즈니스 로직 외에도 DB 수준에서 무결성 보장이 중요함
RESTful API는 반복 호출에 대한 동일한 결과를 보장하도록
idmpotent하게 설계해야 함

트러블 슈팅#2

네이버 기사 수신 시 Spring에서 최신 기사를 받아오지 못하는 문제

발생 원인

UriComponentsBuilder.toUriString() 사용 후
RestTemplate.exchange()에서 전달된 URI를 다시 인코딩
→ 쿼리 파라미터가 이중 인코딩되어 네이버 서버가 요청을 다르게 처리함

해결 방법

1. toUriString() 대신 .build().toUri()로 URI를 생성해 이중 인코딩 문제를 방지
2. 직접 URI 작성 후 URI.create()으로 변환

개선 사항

- 1번 방법으로 문제 해결

  Spring 권장 방식이고 URI.getQuery() 등을 통해 파라미터를 검증할 수 있기 때문

배운 점

라이브러리의 내부 동작을 이해하는 것이 중요함

React 새로고침 시 화이트페이지(404) 발생

발생 원인

React Router 설정이 서버와 연동되지 않아,
새로고침시 서버가 해당 경로를 인식하지 못함.

해결 방법

1. BrowserRouter 대신 HashRouter 적용
2. 서버 경로 요청을 클라이언트 라우터로 연결

개선 사항

- 2번 방법으로 문제 해결
  서버가 먼저 API로 판단해 404를 리턴하지 않고, React SPA를 열어준 다음
  클라이언트 라우팅이 완료된 후 필요한 시점에 API를 호출하도록 흐름을 수정

배운 점

프론트엔드와 같이 소통하며 개발한다면 HashRouter 훅, 설정 연동을 하면 좀 더 좋은 방향으로 해결 가능할 것이나, 현재는 소통이 불가능하고,
프론트페이지를 수정하면 가능하기는 하지만 백엔드를 배우는 입장에서는 백엔드 내에서 개선하는 것이 좋다고 생각하여 적용

후기

✅ 잘한 점

CI/CD, S3 연동, 커서 페이지네이션 등 실무에
가까운 기술 스택을 적극 도입

GitHub Actions 기반 자동 테스트 및 배포
파이프라인을 통해 개발 → 배포 전 과정 자동화 경험

테스트 코드를 적극적으로 사용해서 안정적인 개발을 한 점
(전체 코드 커버리지 87%)

⚠️ 아쉬운 점

인덱스를 활용해보고 싶었는데,
시간 상 바로 MongoDB로 전환하게 된 점

코드래빗 세팅 적용이 잘 안돼서
기대하던 만큼의 활용을 하지 못했던 점 (그래도 도움은 많이 됨)

💡 느낀 점

기능 구현보다도 사전 설계와 팀 내 커뮤니케이션의 중요성을 깊이 체감함

익숙하지 않은 기술 스택도 직접 다뤄보며, 실무에 필요한 역량과 자신감을 쌓을 수 있었음

무엇보다도, 협업 프로젝트를 통해 개발을 단순 구현이 아닌 실제 서비스 운영 관점에서 바라보게 됨

Q & A

THANK YOU

배포링크