뉴스 추천 서비스 테스트 케이스 설계서

1. 개요

1.1 테스트 목적

- 뉴스 추천 서비스의 기능 정상 동작 검증
- 개인화 추천 알고리즘의 정확성 검증
- API 응답 형식 및 에러 처리 검증
- 성능 및 보안 요구사항 충족 검증

1.2 테스트 범위

- **단위 테스트**: 개별 메서드 및 클래스 테스트
- **통합 테스트**: API 엔드포인트 및 서비스 간 연동 테스트
- **성능 테스트**: 응답 시간 및 처리량 테스트
- **보안 테스트**: 인증 및 권한 검증 테스트

2. 단위 테스트

2.1 FeedController 테스트

2.1.1 getUserFeed() 메서드 테스트

```
@ExtendWith(MockitoExtension.class)
class FeedControllerTest {

    @Mock
    private RecommendationService recommendationService;

    @InjectMocks
    private FeedController feedController;

    @Test
    @DisplayName("인증된 사용자의 개인화 피드 조회 성공")
    void getUserFeed_Success() {
        // Given
        String userIdStr = "123";
        int page = 0;
        int size = 21;
        List<FeedItemDto> expectedFeed = createMockFeedItems();

        when(recommendationService.getFeed(123L, page, size))
            .thenReturn(expectedFeed);

        // When
        ApiResponse<List<FeedItemDto>> response =
            feedController.getUserFeed(userIdStr, page, size);
```

```
        // Then
        assertThat(response.isSuccess()).isTrue();
        assertThat(response.getData()).isEqualTo(expectedFeed);
        assertThat(response.getData()).hasSize(15);

        verify(recommendationService).getFeed(123L, page, size);
    }

    @Test
    @DisplayName("잘못된 사용자 ID 형식으로 인한 실패")
    void getUserFeed_InvalidUserId_Failure() {
        // Given
        String invalidUserIdStr = "invalid";
        int page = 0;
        int size = 21;

        // When & Then
        assertThatThrownBy(() ->
            feedController.getUserFeed(invalidUserIdStr, page, size))
            .isInstanceOf(NumberFormatException.class);
    }

    @Test
    @DisplayName("기본 파라미터로 피드 조회")
    void getUserFeed_DefaultParameters() {
        // Given
        String userIdStr = "123";
        List<FeedItemDto> expectedFeed = createMockFeedItems();

        when(recommendationService.getFeed(123L, 0, 21))
            .thenReturn(expectedFeed);

        // When
        ApiResponse<List<FeedItemDto>> response =
            feedController.getUserFeed(userIdStr, 0, 21);

        // Then
        assertThat(response.isSuccess()).isTrue();
        verify(recommendationService).getFeed(123L, 0, 21);
    }
}
```

2.1.2 getUserFeedById() 메서드 테스트

```
@Test
@DisplayName("관리자용 특정 사용자 피드 조회 성공")
void getUserFeedById_Success() {
    // Given
    Long userId = 456L;
    int page = 1;
    int size = 10;
```

```
        List<FeedItemDto> expectedFeed = createMockFeedItems();

        when(recommendationService.getFeed(userId, page, size))
            .thenReturn(expectedFeed);

        // When
        ApiResponse<List<FeedItemDto>> response =
            feedController.getUserFeedById(userId, page, size);

        // Then
        assertThat(response.isSuccess()).isTrue();
        assertThat(response.getData()).isEqualTo(expectedFeed);

        verify(recommendationService).getFeed(userId, page, size);
    }
```

2.2 RecommendationService 테스트

2.2.1 개인화 피드 로직 테스트

```
@ExtendWith(MockitoExtension.class)
class RecommendationServiceImplTest {

    @Mock
    private VectorBatchService vectorBatchService;

    @Mock
    private UserPrefVectorRepository userPrefVectorRepository;

    @Mock
    private RecommendationNewsRepository newsRepository;

    @Mock
    private RecommendationProperties properties;

    @InjectMocks
    private RecommendationServiceImpl recommendationService;

    @Test
    @DisplayName("첫 페이지 개인화 추천 성공")
    void getFeed_FirstPage_PersonalizedRecommendation() {
        // Given
        Long userId = 123L;
        int page = 0;
        int size = 21;

        List<UserPrefVector> mockVectors = createMockUserPrefVectors();
        List<Long> mockNewsIds = createMockNewsIds();
        List<NewsEntity> mockNews = createMockNewsEntities();
```

```
when(vectorBatchService.upsert(userId)).thenReturn(CompletableFuture.compl
etedFuture(null));

when(userPrefVectorRepository.findTopByUserIdOrderByScoreDesc(eq(userId),
any(PageRequest.class)))
            .thenReturn(mockVectors);
        when(properties.getQuotas()).thenReturn(Arrays.asList(7, 5, 3));
        when(newsRepository.findLatestIdsByCategory(any(),
any(PageRequest.class)))
            .thenReturn(mockNewsIds);
        when(newsRepository.findByIdIn(any())).thenReturn(mockNews);

        // When
        List<FeedItemDto> result = recommendationService.getFeed(userId,
page, size);

        // Then
        assertThat(result).hasSize(15);

assertThat(result.get(0).getCategoryName()).isEqualTo(RecommendationCatego
ry.IT_SCIENCE);

        verify(vectorBatchService).upsert(userId);

verify(userPrefVectorRepository).findTopByUserIdOrderByScoreDesc(eq(userId
), any(PageRequest.class));
    }

    @Test
    @DisplayName("사용자 선호도 벡터가 없는 경우 빈 리스트 반환")
    void getFeed_NoUserPreference_EmptyList() {
        // Given
        Long userId = 123L;
        int page = 0;
        int size = 21;


when(vectorBatchService.upsert(userId)).thenReturn(CompletableFuture.compl
etedFuture(null));

when(userPrefVectorRepository.findTopByUserIdOrderByScoreDesc(eq(userId),
any(PageRequest.class)))
            .thenReturn(Collections.emptyList());

        // When
        List<FeedItemDto> result = recommendationService.getFeed(userId,
page, size);

        // Then
        assertThat(result).isEmpty();
    }
}
```

2.2.2 전체 뉴스 피드 로직 테스트

```
@Test
@DisplayName("나머지 페이지 전체 뉴스 최신순 조회")
void getFeed_OtherPages_LatestNews() {
    // Given
    Long userId = 123L;
    int page = 1;
    int size = 10;

    Page<NewsEntity> mockNewsPage = createMockNewsPage();


when(newsRepository.findAllByOrderByPublishedAtDesc(any(PageRequest.class)
))
        .thenReturn(mockNewsPage);

    // When
    List<FeedItemDto> result = recommendationService.getFeed(userId, page,
size);

    // Then
    assertThat(result).hasSize(10);

verify(newsRepository).findAllByOrderByPublishedAtDesc(any(PageRequest.cla
ss));
}
```

2.3 FeedMapper 테스트

```
class FeedMapperTest {

    @Test
    @DisplayName("NewsEntity를 FeedItemDto로 정상 변환")
    void toDto_Success() {
        // Given
        NewsEntity newsEntity = NewsEntity.builder()
            .newsId(12345L)
            .title("테스트 뉴스 제목")
            .press("테스트 언론사")
            .link("https://example.com/news/12345")
            .trusted(true)
            .publishedAt("2024-01-15T10:30:00")
            .createdAt(LocalDateTime.of(2024, 1, 15, 10, 35))
            .reporter("테스트 기자")
            .categoryName(RecommendationCategory.IT_SCIENCE)
            .dedupState(DedupState.REPRESENTATIVE)
            .imageUrl("https://example.com/images/12345.jpg")
            .oidAid("12345_67890")
            .updatedAt(LocalDateTime.of(2024, 1, 15, 10, 35))
```

```
                .build();

        // When
        FeedItemDto result = FeedMapper.toDto(newsEntity);

        // Then
        assertThat(result.getNewsId()).isEqualTo(12345L);
        assertThat(result.getTitle()).isEqualTo("테스트 뉴스 제목");
        assertThat(result.getPress()).isEqualTo("테스트 언론사");

assertThat(result.getCategoryName()).isEqualTo(RecommendationCategory.IT_S
CIENCE);

assertThat(result.getDedupState()).isEqualTo(DedupState.REPRESENTATIVE);

assertThat(result.getPublishedAt()).isEqualTo(LocalDateTime.of(2024, 1,
15, 10, 30));
    }

    @Test
    @DisplayName("잘못된 날짜 형식 처리")
    void toDto_InvalidDateFormat() {
        // Given
        NewsEntity newsEntity = NewsEntity.builder()
            .newsId(12345L)
            .title("테스트 뉴스")
            .publishedAt("invalid-date-format")
            .build();

        // When
        FeedItemDto result = FeedMapper.toDto(newsEntity);

        // Then
        assertThat(result.getPublishedAt()).isNull();
    }
}
```

3. 통합 테스트

3.1 API 엔드포인트 통합 테스트

```
@SpringBootTest(webEnvironment =
SpringBootTest.WebEnvironment.RANDOM_PORT)
@AutoConfigureTestDatabase
@TestPropertySource(properties = {
    "spring.jpa.hibernate.ddl-auto=create-drop"
})
class FeedControllerIntegrationTest {

    @Autowired
    private TestRestTemplate restTemplate;
```

```
    @Autowired
    private TestEntityManager entityManager;

    @Test
    @DisplayName("인증된 사용자 피드 조회 통합 테스트")
    void getUserFeed_Integration_Success() {
        // Given
        String jwtToken = generateValidJwtToken("123");
        HttpHeaders headers = new HttpHeaders();
        headers.setBearerAuth(jwtToken);

        // 테스트 데이터 설정
        setupTestData();

        // When
        ResponseEntity<ApiResponse> response = restTemplate.exchange(
            "/api/news/feed?page=0&size=21",
            HttpMethod.GET,
            new HttpEntity<>(headers),
            ApiResponse.class
        );

        // Then
        assertThat(response.getStatusCode()).isEqualTo(HttpStatus.OK);
        assertThat(response.getBody().isSuccess()).isTrue();

        List<FeedItemDto> feedItems = objectMapper.convertValue(
            response.getBody().getData(),
            new TypeReference<List<FeedItemDto>>() {}
        );

        assertThat(feedItems).hasSize(15);
    }

    @Test
    @DisplayName("인증되지 않은 사용자 요청 거부")
    void getUserFeed_Unauthorized_Rejected() {
        // Given
        HttpHeaders headers = new HttpHeaders();
        // JWT 토큰 없음

        // When
        ResponseEntity<ApiResponse> response = restTemplate.exchange(
            "/api/news/feed",
            HttpMethod.GET,
            new HttpEntity<>(headers),
            ApiResponse.class
        );

        // Then

assertThat(response.getStatusCode()).isEqualTo(HttpStatus.UNAUTHORIZED);
```

```
    }
}
```

3.2 서비스 간 연동 테스트

```
@SpringBootTest
@TestPropertySource(properties = {
    "spring.jpa.hibernate.ddl-auto=create-drop"
})
class RecommendationServiceIntegrationTest {

    @Autowired
    private RecommendationService recommendationService;

    @Autowired
    private UserPrefVectorRepository userPrefVectorRepository;

    @Autowired
    private RecommendationNewsRepository newsRepository;

    @Test
    @DisplayName("전체 추천 파이프라인 통합 테스트")
    void recommendationPipeline_Integration_Success() {
        // Given
        Long userId = 123L;

        // 사용자 선호도 벡터 설정
        setupUserPreferenceVectors(userId);

        // 뉴스 데이터 설정
        setupNewsData();

        // When
        List<FeedItemDto> result = recommendationService.getFeed(userId,
0, 21);

        // Then
        assertThat(result).hasSize(15);

        // 카테고리별 할당 검증
        Map<RecommendationCategory, Long> categoryCounts = result.stream()
            .collect(Collectors.groupingBy(FeedItemDto::getCategoryName,
Collectors.counting()));


assertThat(categoryCounts.get(RecommendationCategory.IT_SCIENCE)).isEqualT
o(7);

assertThat(categoryCounts.get(RecommendationCategory.ECONOMY)).isEqualTo(5
);
```

```
assertThat(categoryCounts.get(RecommendationCategory.SOCIETY)).isEqualTo(3
);
    }
}
```

4. 성능 테스트

4.1 API 응답 시간 테스트

```
@SpringBootTest(webEnvironment =
SpringBootTest.WebEnvironment.RANDOM_PORT)
class FeedControllerPerformanceTest {

    @Autowired
    private TestRestTemplate restTemplate;

    @Test
    @DisplayName("개인화 피드 응답 시간 테스트")
    void getUserFeed_Performance_WithinLimit() {
        // Given
        String jwtToken = generateValidJwtToken("123");
        HttpHeaders headers = new HttpHeaders();
        headers.setBearerAuth(jwtToken);

        // When
        long startTime = System.currentTimeMillis();

        ResponseEntity<ApiResponse> response = restTemplate.exchange(
            "/api/news/feed?page=0&size=21",
            HttpMethod.GET,
            new HttpEntity<>(headers),
            ApiResponse.class
        );

        long endTime = System.currentTimeMillis();
        long responseTime = endTime - startTime;

        // Then
        assertThat(response.getStatusCode()).isEqualTo(HttpStatus.OK);
        assertThat(responseTime).isLessThan(500); // 500ms 이내 응답
    }

    @Test
    @DisplayName("전체 뉴스 피드 응답 시간 테스트")
    void getUserFeed_LatestNews_Performance_WithinLimit() {
        // Given
        String jwtToken = generateValidJwtToken("123");
        HttpHeaders headers = new HttpHeaders();
        headers.setBearerAuth(jwtToken);

        // When
```

```
        long startTime = System.currentTimeMillis();

        ResponseEntity<ApiResponse> response = restTemplate.exchange(
            "/api/news/feed?page=1&size=21",
            HttpMethod.GET,
            new HttpEntity<>(headers),
            ApiResponse.class
        );

        long endTime = System.currentTimeMillis();
        long responseTime = endTime - startTime;

        // Then
        assertThat(response.getStatusCode()).isEqualTo(HttpStatus.OK);
        assertThat(responseTime).isLessThan(300); // 300ms 이내 응답
    }
}
```

4.2 동시 요청 처리 테스트

```
@Test
@DisplayName("동시 요청 처리 성능 테스트")
void concurrentRequests_Performance_Test() throws InterruptedException {
    // Given
    int numberOfThreads = 10;
    int requestsPerThread = 5;
    CountDownLatch latch = new CountDownLatch(numberOfThreads);
    AtomicInteger successCount = new AtomicInteger(0);
    AtomicInteger failureCount = new AtomicInteger(0);

    // When
    for (int i = 0; i < numberOfThreads; i++) {
        final int threadId = i;
        new Thread(() -> {
            try {
                for (int j = 0; j < requestsPerThread; j++) {
                    String jwtToken =
generateValidJwtToken(String.valueOf(threadId));
                    HttpHeaders headers = new HttpHeaders();
                    headers.setBearerAuth(jwtToken);

                    ResponseEntity<ApiResponse> response =
restTemplate.exchange(
                        "/api/news/feed?page=0&size=21",
                        HttpMethod.GET,
                        new HttpEntity<>(headers),
                        ApiResponse.class
                    );

                    if (response.getStatusCode() == HttpStatus.OK) {
                        successCount.incrementAndGet();
```

```
                    } else {
                        failureCount.incrementAndGet();
                    }
                }
            } finally {
                latch.countDown();
            }
        }).start();
    }

    latch.await(30, TimeUnit.SECONDS);

    // Then
    int totalRequests = numberOfThreads * requestsPerThread;
    assertThat(successCount.get()).isGreaterThan(totalRequests * 0.9); //
90% 이상 성공
    assertThat(failureCount.get()).isLessThan(totalRequests * 0.1); // 10%
미만 실패
}
```

5. 보안 테스트

5.1 인증 및 권한 테스트

```
@Test
@DisplayName("유효하지 않은 JWT 토큰 거부")
void invalidJwtToken_Rejected() {
    // Given
    String invalidToken = "invalid.jwt.token";
    HttpHeaders headers = new HttpHeaders();
    headers.setBearerAuth(invalidToken);

    // When
    ResponseEntity<ApiResponse> response = restTemplate.exchange(
        "/api/news/feed",
        HttpMethod.GET,
        new HttpEntity<>(headers),
        ApiResponse.class
    );

    // Then

assertThat(response.getStatusCode()).isEqualTo(HttpStatus.UNAUTHORIZED);
}

@Test
@DisplayName("만료된 JWT 토큰 거부")
void expiredJwtToken_Rejected() {
    // Given
    String expiredToken = generateExpiredJwtToken("123");
    HttpHeaders headers = new HttpHeaders();
```

```
    headers.setBearerAuth(expiredToken);

    // When
    ResponseEntity<ApiResponse> response = restTemplate.exchange(
        "/api/news/feed",
        HttpMethod.GET,
        new HttpEntity<>(headers),
        ApiResponse.class
    );

    // Then

assertThat(response.getStatusCode()).isEqualTo(HttpStatus.UNAUTHORIZED);
}
```

5.2 입력 검증 테스트

```
@Test
@DisplayName("잘못된 페이지 번호 검증")
void invalidPageNumber_Rejected() {
    // Given
    String jwtToken = generateValidJwtToken("123");
    HttpHeaders headers = new HttpHeaders();
    headers.setBearerAuth(jwtToken);

    // When
    ResponseEntity<ApiResponse> response = restTemplate.exchange(
        "/api/news/feed?page=-1&size=21",
        HttpMethod.GET,
        new HttpEntity<>(headers),
        ApiResponse.class
    );

    // Then

assertThat(response.getStatusCode()).isEqualTo(HttpStatus.BAD_REQUEST);
}

@Test
@DisplayName("잘못된 페이지 크기 검증")
void invalidPageSize_Rejected() {
    // Given
    String jwtToken = generateValidJwtToken("123");
    HttpHeaders headers = new HttpHeaders();
    headers.setBearerAuth(jwtToken);

    // When
    ResponseEntity<ApiResponse> response = restTemplate.exchange(
        "/api/news/feed?page=0&size=1000",
        HttpMethod.GET,
        new HttpEntity<>(headers),
```

```
            ApiResponse.class
        );

        // Then

    assertThat(response.getStatusCode()).isEqualTo(HttpStatus.BAD_REQUEST);
    }
```

6. 테스트 데이터 설정

6.1 테스트 헬퍼 메서드

```
@TestConfiguration
class TestDataConfig {

    @Bean
    public List<FeedItemDto> createMockFeedItems() {
        List<FeedItemDto> items = new ArrayList<>();

        // IT_SCIENCE 카테고리 7개
        for (int i = 0; i < 7; i++) {
            items.add(FeedItemDto.builder()
                .newsId(1000L + i)
                .title("IT/과학 뉴스 " + (i + 1))
                .press("테크뉴스")
                .categoryName(RecommendationCategory.IT_SCIENCE)
                .dedupState(DedupState.REPRESENTATIVE)
                .publishedAt(LocalDateTime.now().minusHours(i))
                .build());
        }

        // ECONOMY 카테고리 5개
        for (int i = 0; i < 5; i++) {
            items.add(FeedItemDto.builder()
                .newsId(2000L + i)
                .title("경제 뉴스 " + (i + 1))
                .press("경제일보")
                .categoryName(RecommendationCategory.ECONOMY)
                .dedupState(DedupState.REPRESENTATIVE)
                .publishedAt(LocalDateTime.now().minusHours(i))
                .build());
        }

        // SOCIETY 카테고리 3개
        for (int i = 0; i < 3; i++) {
            items.add(FeedItemDto.builder()
                .newsId(3000L + i)
                .title("사회 뉴스 " + (i + 1))
                .press("사회일보")
                .categoryName(RecommendationCategory.SOCIETY)
                .dedupState(DedupState.REPRESENTATIVE)
```

```
                .publishedAt(LocalDateTime.now().minusHours(i))
                .build());
        }

        return items;
    }

    @Bean
    public List<UserPrefVector> createMockUserPrefVectors() {
        return Arrays.asList(
            UserPrefVector.builder()
                .userId(123L)
                .category(RecommendationCategory.IT_SCIENCE)
                .score(0.4)
                .build(),
            UserPrefVector.builder()
                .userId(123L)
                .category(RecommendationCategory.ECONOMY)
                .score(0.3)
                .build(),
            UserPrefVector.builder()
                .userId(123L)
                .category(RecommendationCategory.SOCIETY)
                .score(0.3)
                .build()
        );
    }
}
```

7. 테스트 실행 가이드

7.1 테스트 실행 명령어

```
# 전체 테스트 실행
./gradlew test

# 특정 테스트 클래스 실행
./gradlew test --tests FeedControllerTest

# 특정 테스트 메서드 실행
./gradlew test --tests FeedControllerTest.getUserFeed_Success

# 통합 테스트만 실행
./gradlew test --tests "*IntegrationTest"

# 성능 테스트만 실행
./gradlew test --tests "*PerformanceTest"
```

7.2 테스트 환경 설정

```yaml
# application-test.yml
spring:
  datasource:
    url: jdbc:h2:mem:testdb
    driver-class-name: org.h2.Driver
  jpa:
    hibernate:
      ddl-auto: create-drop
    show-sql: true
  security:
    jwt:
      secret: test-secret-key
      expiration: 3600000

logging:
  level:
    com.newnormallist.newsservice: DEBUG
```

7.3 테스트 커버리지 확인

```bash
# 테스트 커버리지 리포트 생성
./gradlew test jacocoTestReport

# 커버리지 리포트 확인 (build/reports/jacoco/test/html/index.html)
open build/reports/jacoco/test/html/index.html
```

8. 테스트 결과 검증

8.1 성공 기준

- **단위 테스트**: 90% 이상 커버리지
- **통합 테스트**: 모든 API 엔드포인트 정상 동작
- **성능 테스트**: 응답 시간 요구사항 충족
- **보안 테스트**: 인증 및 권한 검증 통과

8.2 테스트 결과 리포트

- 테스트 실행 결과 요약
- 실패한 테스트 케이스 분석
- 성능 테스트 결과 그래프
- 보안 테스트 결과 보고서