


## 기능 요구사항 작성 템플릿 (팀 전체용)

 이 문서는 팀원들이 각자 맡은 기능의 동작 방식을 명확히 정의하고, 협업 중 충돌을 방지하기 위해 작성됩니다.

각자 맡은 기능에 대해 아래 항목을 채워 주세요.

### ☒ 1. 기능 이름

웹 푸시 알림 구독 등록

### ☒ 2. 기능 설명 (한 줄 요약)

클라이언트의 브라우저에서 푸시 알림을 수신할 수 있도록 구독 정보를 서버에 등록한다.

### ☒ 3. 주요 흐름 / 시나리오

1. 사용자가 클라이언트에서 알림 수신을 허용하고, 구독 정보를 생성한다.
2. 클라이언트가 /api/push/subscribe 로 구독 정보를 전송한다.
3. 서버는 사용자의 ID 를 기준으로 구독 정보를 저장한다.
4. 클라이언트에게 등록 성공 메시지를 반환한다.

### ☒ 4. 요청 DTO 예시

```json

{

"userId": 1234,

"subscription": {

"endpoint": "...",

"keys": {

"p256dh": "...",

```
"auth": "..."  
  
}  
  
}  
  
}  
...  

```

#### ☒ 5. 응답 형식 예시

```
```json  
{  
  
  "status": 200,  
  
  "message": "구독 등록 성공",  
  
  "data": {}  
  
}  
...  

```

#### ☒ 6. 예외 케이스 / 실패 상황

- 필드 누락 → 400 에러
- 인증 실패 → 401 에러
- 비정상적인 키 구조 → 400 에러

#### ☒ 7. 엔드포인트 정보

- URL: /api/push/subscribe
- Method: POST
- 인증 필요 여부: 로그인 필요

#### ☒ 8. 담당자

이우창

## 예시 코드 (Controller 예시)

좀 더 공부해보고 작성하겠습니다.

## 예시 코드 (DTO 예시)

좀 더 공부해보고 작성하겠습니다.

### ☑ 1. 기능 이름

웹 푸시 알림 전송

### ☑ 2. 기능 설명 (한 줄 요약)

저장된 구독 정보를 기반으로 사용자에게 브라우저 푸시 알림을 전송한다.

### ☑ 3. 주요 흐름 / 시나리오

1. ChatService 에서 NotificationService 로 알림 전송 메서드를 호출한다.
2. 서버는 DB 에서 해당 사용자 ID 의 구독 정보를 조회한다.
3. webpush-java 의 PushService 를 이용해 푸시 알림을 전송한다.
4. 전송 성공/실패 여부는 내부 로깅 또는 예외 처리로 관리한다.

### ☑ 4. 요청 파라미터 예시

```
```java
public class PushMessage {

    private Long userId;

    private String title;

    private String body;

}
```
```

### ☑ 5. 반환 값 예시

```
Java```

public class PushResult {

    private boolean success;

    private String errorMessage;

}
```

...

## ☑ 6. 예외 케이스 / 실패 상황

- 사용자 구독 정보 없음 → UserSubscriptionNotFoundException
- 푸시 전송 실패 → PushSendException

## ☑ 7. 엔드포인트 정보

- 없음 (내부 서비스 호출로만 작동)

## ☑ 8. 담당자

이우창

## 🔗 예시 코드 (Service 예시)

```
// ChatService

public void sendMessage(ChatMessage message) {

    // 1. 메시지 저장

    chatRepository.save(message);

    // 2. 알림 전송

    PushMessage pushMessage = new PushMessage(

        message.getReceiverId(),

        "새 메시지 알림",

        message.getContent()
```

```

    );

    pushNotificationService.send(pushMessage);
}

// NotificationService

public void send(PushMessage pushMessage) {

    Subscription subscription =
subscriptionRepository.findByUserId(pushMessage.getUserId())

        .orElseThrow(() -> new UserSubscriptionNotFoundException());

    Notification notification = new Notification(

        subscription.getEndpoint(),

        subscription.getKeys().getP256dh(),

        subscription.getKeys().getAuth(),

        createPayload(pushMessage)

    );

    pushService.send(notification);
}

```