

# Firestore Cloud Messaging with Spring Boot

| Session 5



# Session Overview:

## Training Approach and Interaction




### Session 5

Presentation of Training Materials

Practical Training

Discussion



# Session 5 Objective

Participants understand Firebase Cloud Messaging (FCM), send push notifications using a Spring Boot backend, and seamlessly integrate push notifications into a Spring Boot application.

---

# MATERI PELATIHAN

---

- **Understanding Firebase Cloud Messaging (FCM).**
- **Sending push notifications from a Spring Boot backend using FCM.**
- **Integrating push notifications into a Spring Boot application.**

# Firebase Cloud Messaging

---

- FCM digunakan untuk mengirim pesan notifikasi
- FCM mendukung ke berbagai platform seperti Android, iOS, dan web.
- Cocok untuk berbagai skenario, seperti pemberitahuan, pengingat, atau pembaruan data secara real-time

# Firebase Cloud Messaging

---

- FCM digunakan untuk mengirim pesan notifikasi
- FCM mendukung ke berbagai platform seperti Android, iOS, dan web.
- Cocok untuk berbagai skenario, seperti pemberitahuan, pengingat, atau pembaruan data secara real-time

# Setup Firebase Cloud Messaging

---

- Membuat firebase project:  
<https://console.firebase.google.com/>
- Menambahkan dependency firebase pada project Spring Boot
- Download secret key dari firebase project
- Menambahkan Firebase Admin SDK untuk project

# Implementation FCM into Spring Boot

## Penambahan FCM Service

```
1 package com.example.demo.service;
2
3 import com.google.firebase.messaging.FirebaseMessaging;
4 import com.google.firebase.messaging.Message;
5 import com.google.firebase.messaging.Notification;
6 import org.springframework.stereotype.Service;
7
8 @Service
9 public class FCMService {
10
11     public String sendNotification(String token, String title, String body) throws Exception {
12         Notification notification = Notification.builder()
13             .setTitle(title)
14             .setBody(body)
15             .build();
16
17         Message message = Message.builder()
18             .setToken(token)
19             .setNotification(notification)
20             .build();
21
22         return FirebaseMessaging.getInstance().send(message);
23     }
24 }
```

FCM Service digunakan sebagai class service untuk proses operasi send notification sesuai yang dikirimkan.



# Firebase Cloud Messaging

## Penambahan Notification Controller

```
1 package com.example.demo.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.http.ResponseEntity;
5 import org.springframework.web.bind.annotation.PostMapping;
6 import org.springframework.web.bind.annotation.RequestBody;
7 import org.springframework.web.bind.annotation.RequestMapping;
8 import org.springframework.web.bind.annotation.RestController;
9
10 import com.example.demo.model.NotificationRequest;
11 import com.example.demo.service.FCMService;
12
13 @RestController
14 @RequestMapping("/api/notifications")
15 public class NotificationController {
16
17     @Autowired
18     private FCMService fcmService;
19
20     @PostMapping("/send")
21     public ResponseEntity<String> sendNotification(@RequestBody NotificationRequest notificationRequest) {
22         try {
23             String response = fcmService.sendNotification(notificationRequest.getToken(),
24                 notificationRequest.getTitle(),
25                 notificationRequest.getBody());
26             return ResponseEntity.ok("Successfully sent message: " + response);
27         } catch (Exception e) {
28             return ResponseEntity.status(500).body("Failed to send message: " + e.getMessage());
29         }
30     }
31 }
```

Notification Controller digunakan sebagai controller untuk menangani request POST send notification.

# Config

## Edit Security Config

```
1  @Bean
2  public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {
3      http
4          .authorizeRequests(authorizeRequests -> authorizeRequests
5              .requestMatchers("/api/auth/**").permitAll()
6              .requestMatchers("/api/users/**").permitAll()
7              .requestMatchers("/api/notifications/**").authenticated()
8              .requestMatchers("/api/task/**").authenticated()
9              .anyRequest().permitAll())
10         .csrf(csrf -> csrf.disable())
11         .sessionManagement(
12             sessionManagement -> sessionManagement.sessionCreationPolicy(SessionCreationPolicy.STATELESS));
13
14         // JWT Token Filter
15         // http.addFilterBefore(jwtAuthenticationFilter,
16         // UsernamePasswordAuthenticationFilter.class);
17
18         // Firebase Authentication Filter
19         http.addFilterBefore(firebaseTokenFilter, UsernamePasswordAuthenticationFilter.class);
20         return http.build();
21     }
```

Menambahkan filter untuk api notification harus melalulai autentikasi terlebih dahulu

# REFERENCES

---

FlutterFire. (n.d.).  
<https://firebase.flutter.dev/docs/messaging/overview/>



# THANK YOU

Thank you for following session 5 to the end.  
See you at the next meeting