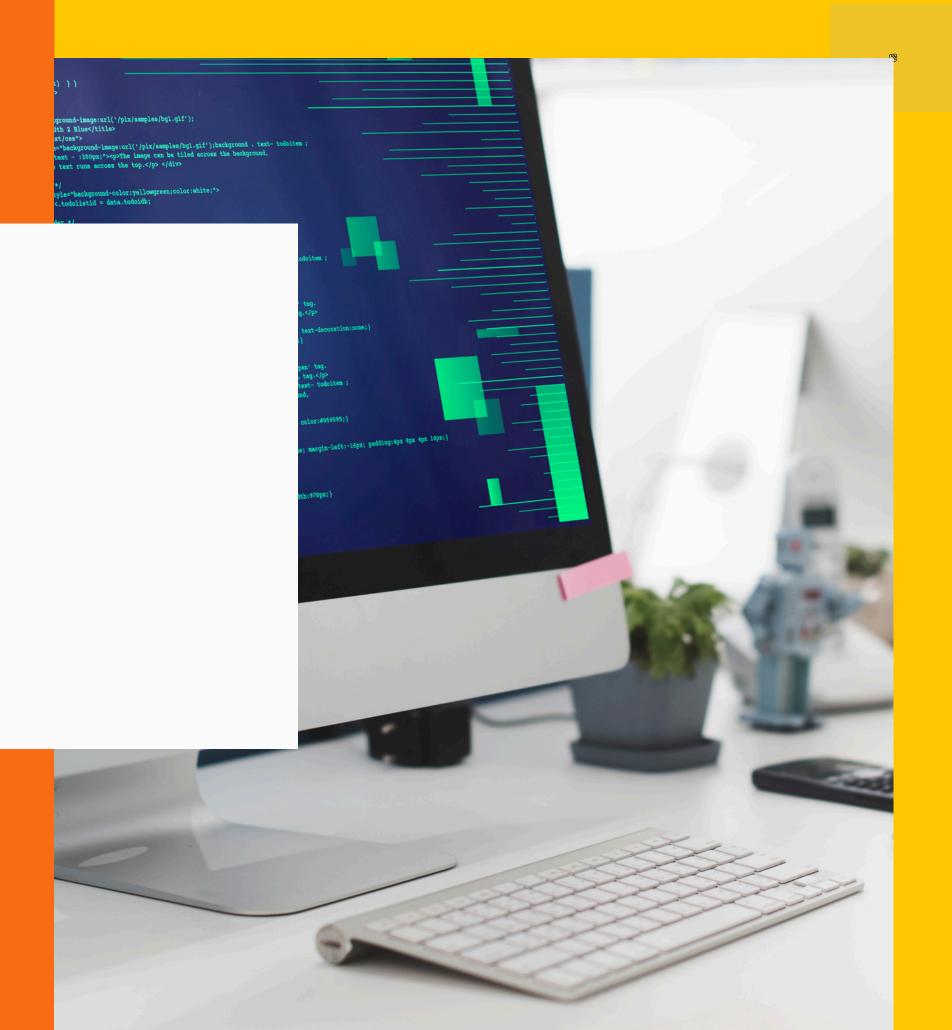
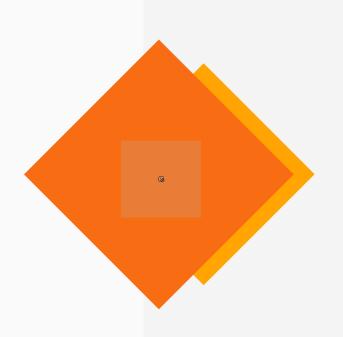
Firebase Cloud Messaging with Spring Boot

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: <u>https://console.firebase.google.com/</u>
- 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

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

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

Firebase Cloud Messaging

Penambahan Notification Controller

```
package com.example.demo.controller;
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;
10 import com.example.demo.model.NotificationRequest;
import com.example.demo.service.FCMService;
13 @RestController
14  @RequestMapping("/api/notifications")
15  public class NotificationController {
       @Autowired
       private FCMService fcmService;
        @PostMapping("/send")
        public ResponseEntity<String> sendNotification(@RequestBody NotificationRequest notificationRequest) {
                String response = fcmService.sendNotification(notificationRequest.getToken(),
                        notificationRequest.getTitle(),
                        notificationRequest.getBody());
               return ResponseEntity.ok("Successfully sent message: " + response);
            } catch (Exception e) {
                return ResponseEntity.status(500).body("Failed to send message: " + e.getMessage());
31 }
```

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

Config

Edit Security Config

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

Menambahkan filter untuk api notification harus melaluai autentikasi terlebih dahulu

REFERENCES

Firebase Cloud Messaging | 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

