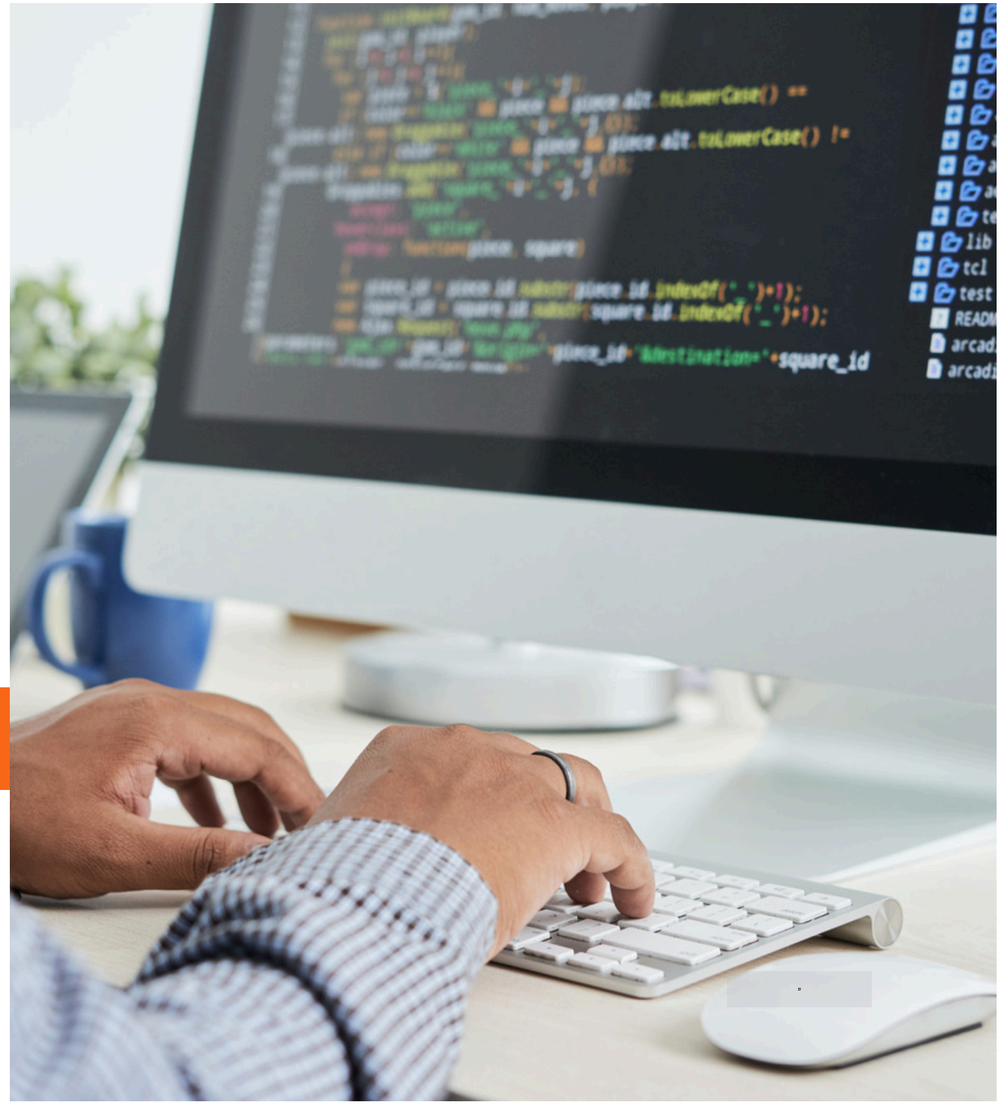
BINUS CENTER

Master Backend Development with Java Spring Boot and Firebase: Build Scalable and Secure Applications

BACKEND



www.binuscenter.com

PENGENALAN

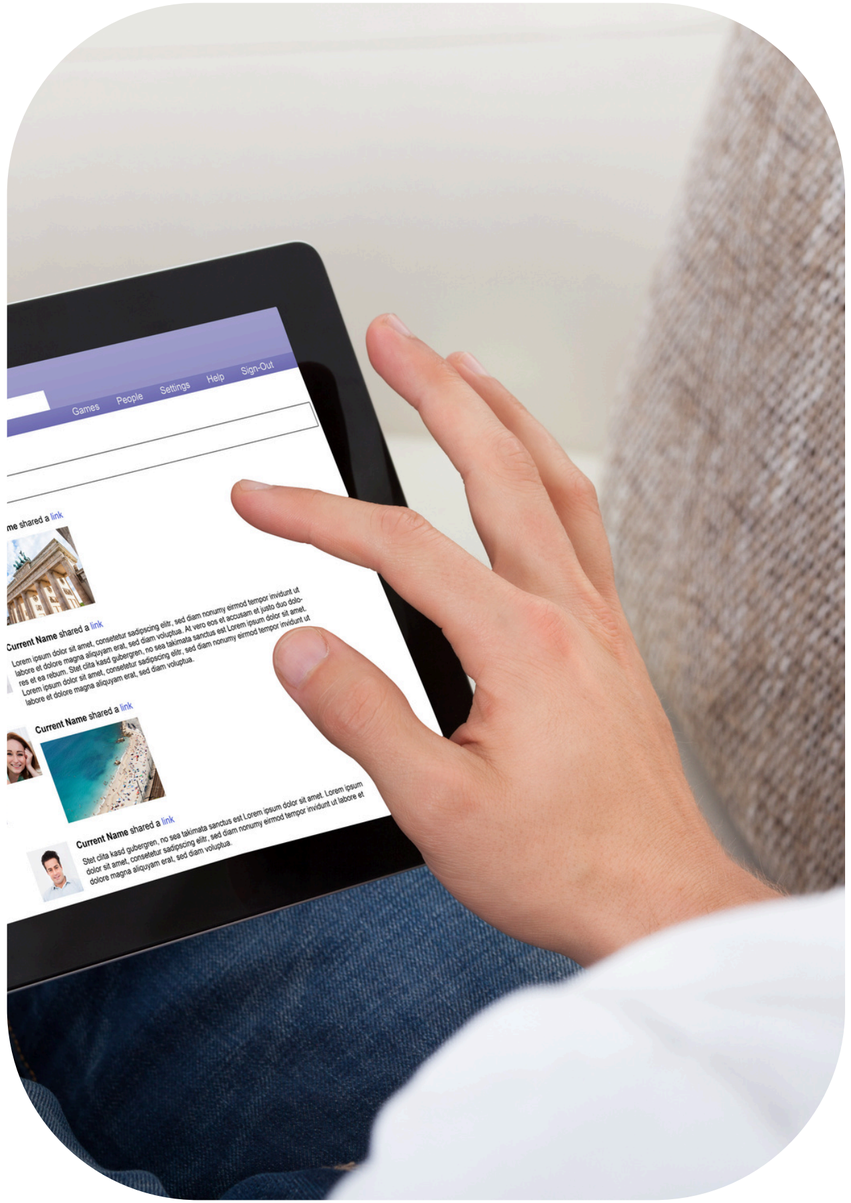
TENAGA PELATIH

Anang Prasetyo, S.Kom., M.Kom.

Pendidikan

- Sarjana Sistem Informasi BINUS
- Magister Teknik Informatika BINUS

https://id.linkedin.com/in/anangp



Profil Tenaga Pelatih

I am now a Lecturer in Computer Science study program at Bina Nusantara University.
My interest study in Machine Learning, Deep Learning, Artificial Intelligence, Mobile Application.
A subject matter in software engineering, artificial intelligence and mobile & game technology.
Some previous research topic in machine learning and IoT.

TOKEN OF APPRECIATION

KOMINFO digitalent DQLab PROA Professional Academy

SERTIFIKAT PELATIHAN

1987291840-862/PROA/BLSDM.Kominfo/2024

Diberikan kepada

ANANG PRASETYO

telah menyelesaikan pelatihan

Data Engineer
Professional Academy
Digital Talent Scholarship 2024

yang diselenggarakan oleh **Pusbang Proserti**

pada tanggal **12 Februari 2024 - 30 Maret 2024** selama **30 Jam Pelatihan**

Jakarta, 1 April 2024

Ditandatangani secara elektronik
KEPALA BADAN PENGEMBANGAN SDM KOMINFO
HARY BUDIARTO

Dokumen ini telah ditandatangani secara elektronik menggunakan sertifikat elektronik yang diterbitkan oleh Balai Sertifikasi Elektronik (BSrE), Badan Siber dan Sandi Negara (BSSN).

Certification of Data Engineer

udemy

Certificate no: UC-8c2593ca-2525-4f69-97a5-228bdb407824
Certificate url: ude.my/UC-8c2593ca-2525-4f69-97a5-228bdb407824
Reference Number: 0004

CERTIFICATE OF COMPLETION

Complete React Native Bootcamp (with Hooks)

Instructors **Andrei Neagoie, Mo Binni**

Anang Prasetyo

Date **Feb. 22, 2024**
Length **27.5 total hours**

Certification of React Native Bootcamp

DESKRIPSI PELATIHAN

This course is designed to equip participants with the skills necessary to build modern backend applications using Java Spring Boot and Firebase for modern web and mobile applications. Participants will develop robust, secure, and scalable RESTful APIs and server-side functionalities for any web or mobile application.

TUJUAN PELATIHAN

1. Participants understand the fundamentals of Java Spring Boot including its architecture, dependency injection, and MVC pattern, laying the foundation for backend development.
2. Participants will learn how to initialize and configure a Spring Boot project, ensuring a proper development environment for backend application development.
3. Participants will integrate databases into their Spring Boot applications, learning to perform CRUD (Create, Read, Update, Delete) operations with Spring Data JPA.
4. Participants will develop RESTful APIs using Spring Boot, understanding how to design endpoints, handle HTTP methods, and manage data.
5. Participants can secure REST APIs with Spring Security and JWT Token authentication.
6. Participants will implement user authentication and authorization using Firebase Authentication.
7. Participants will integrate Firebase Realtime Database and Cloud functions for efficient serverless functionalities.
8. Participants will apply their knowledge to develop a comprehensive backend project using Java Spring Boot and Firebase. They will present their project and incorporating feedback for continuous improvement.

SESI PELATIHAN

Session 1:

- Introduction to Spring and Spring Boot (3 hours)

Session 2:

- Data Persistence with Spring Data JPA (3 hours)

Session 3:

- Spring Security and JWT Token Authentication (3 hours)

Session 4:

- Introduction to Firebase (3 hours)

Session 5:

- Firebase Cloud Messaging with Spring Boot (3 hours)

Session 6:

- Firebase Authentication and Integration with Spring Boot (3 hours)

Session 7:

- Cloud Deployment and Continuous Integration/Continuous Delivery (CI/CD) (3 hours)

Session 8:

- Final Project (3 hours)

Introduction to Spring and Spring Boot

Session 1

Session Overview:
Training Approach and Interaction

Session 1

Presentation of Training Materials

Practical Training

Discussion

Session 1 Objective

Participants understand the fundamentals of Java Spring Boot including its architecture, dependency injection, and MVC pattern, laying the foundation for backend development.

MATERI PELATIHAN

- **Introduction and overview of Java Spring Boot framework.**
- **Spring Boot features and advantages.**
- **Setting up Spring Boot development environment and basic project structure.**
- **Creating simple RESTful APIs and handling requests with Spring MVC annotations.**

What is Spring Boot?

Spring Boot adalah sebuah powerful framework yang dikembangkan diatas Spring framework.

Menyederhanakan pembuatan aplikasi Spring

Menyediakan auto-configuration, opinionated defaults, dan embedded servers.

Key Features of Spring Boot

01 **Auto-configuration: Secara otomatis mengkonfigurasi aplikasi Spring berdasarkan dependensi.**

02 **Starter POMs: Menyederhanakan manajemen dependency dengan pre-configured dependencies.**

03 **Embedded Servers: Menjalankan aplikasi web tanpa server eksternal (e.g., Tomcat, Jetty).**

04 **Spring MVC: Menyederhanakan cara yang flexible dan powerful dalam mengembangkan aplikasi web.**

05 **Spring Data JPA: Menyederhanakan akses data dengan JPA.**

06 **Spring Security: Menyediakan fitur keamanan yang tangguh.**

Setting Up the Development Environment

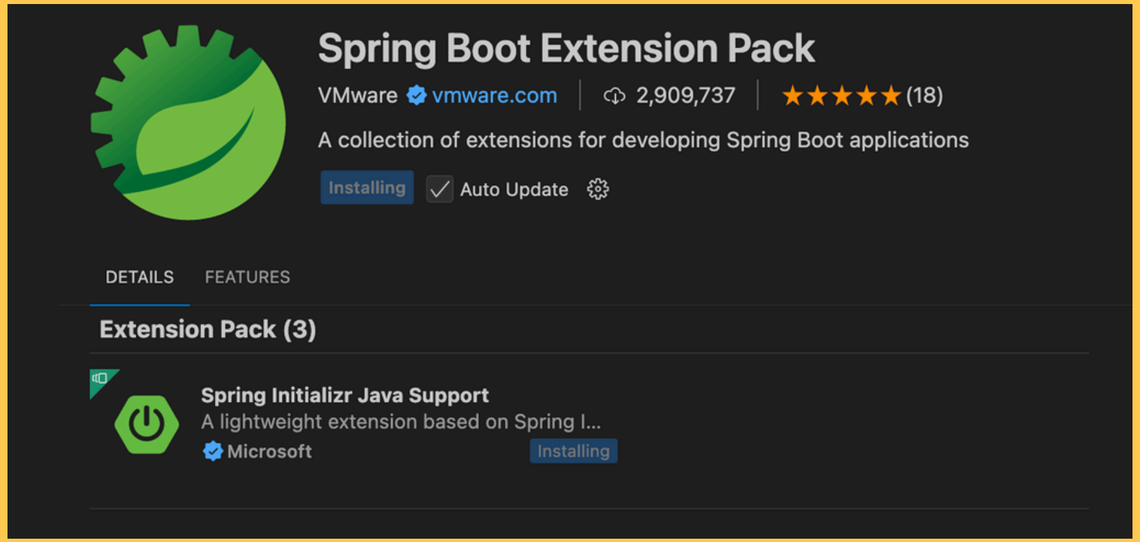
- Java Development Kit (JDK)
- IDE: Choose your preferred IDE (e.g., IntelliJ IDEA, Eclipse, Visual Studio Code).
- Build Tool: Spring Boot projects typically use Maven or Gradle.

Spring Boot with Visual Studio Code

Ekstensi yang digunakan dalam VS Code :

1. Spring Boot Tools
2. Spring Initializr
3. Spring Boot Dashboard

Rekomendasi dengan menginstall **Spring Boot Extension Pack** yang mencakup semua ekstensi di atas.



Spring MVC Annotations

Anotasi Spring MVC (Model VIew Controller) bisa disebut sebagai asisten yang membantu dalam komunikasi yang lebih jelas dalam pengembangan suatu aplikasi. Berikut beberapa anotasi yang penting dalam Spring MVC

1. @Controller
2. @RequestMapping
3. @PathVariable
4. @RequestParam
5. @ModelAttribute
6. @RequestBody and @ResponseBody
7. @RequestHeader and @ResponseHeader

Dokumentasi anotasi Spring MVC

Basic Spring Boot Project Structure

1. Controller : berisikan class untuk menangani HTTP klien
2. Model : berisikan class yang mewakili model data atau entitas data
3. Service : berisikan logika bisnis dari aplikasi seperti CRUD Operation
4. Repository : berisikan interface untuk akes data layer
5. Application : main aplikasi yang akan dijalankan oleh java spring boot

Creating a Simple RESTful API

Model

```java
package com.example.demo.model;

public class User {
    private int id;
    private String name;
    private String email;

    public User() {}

    public User(int id, String name, Stringq email) {
        this.id = id;
        this.name = name;
        this.email = email;
    }

    // Getters and Setters
    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }
}
```

Creating a Simple RESTful API

Repository

```java
package com.example.demo.repository;

import com.example.demo.model.User;
import org.springframework.stereotype.Repository;

import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

@Repository
public class UserRepository {

    private final List<User> userList = new ArrayList<>();

    // Inisialisasi dengan data dummy
    public UserRepository() {
        userList.add(new User(1, "John Doe", "john.doe@example.com"));
        userList.add(new User(2, "Jane Smith", "jane.smith@example.com"));
    }

    // Get user by ID
    public Optional<User> findById(int id) {
        return userList.stream().filter(user -> user.getId() == id).findFirst();
    }
}

```

Creating a Simple RESTful API

Services

```java
package com.example.demo.service;

import com.example.demo.model.User;
import com.example.demo.repository.UserRepository;
import org.springframework.stereotype.Service;

import java.util.Optional;

@Service
public class UserService {

    private final UserRepository userRepository;

    public UserService(UserRepository userRepository) {
        this.userRepository = userRepository;
    }

    // Get user by ID
    public Optional<User> getUserById(int id) {
        return userRepository.findById(id);
    }
}

```

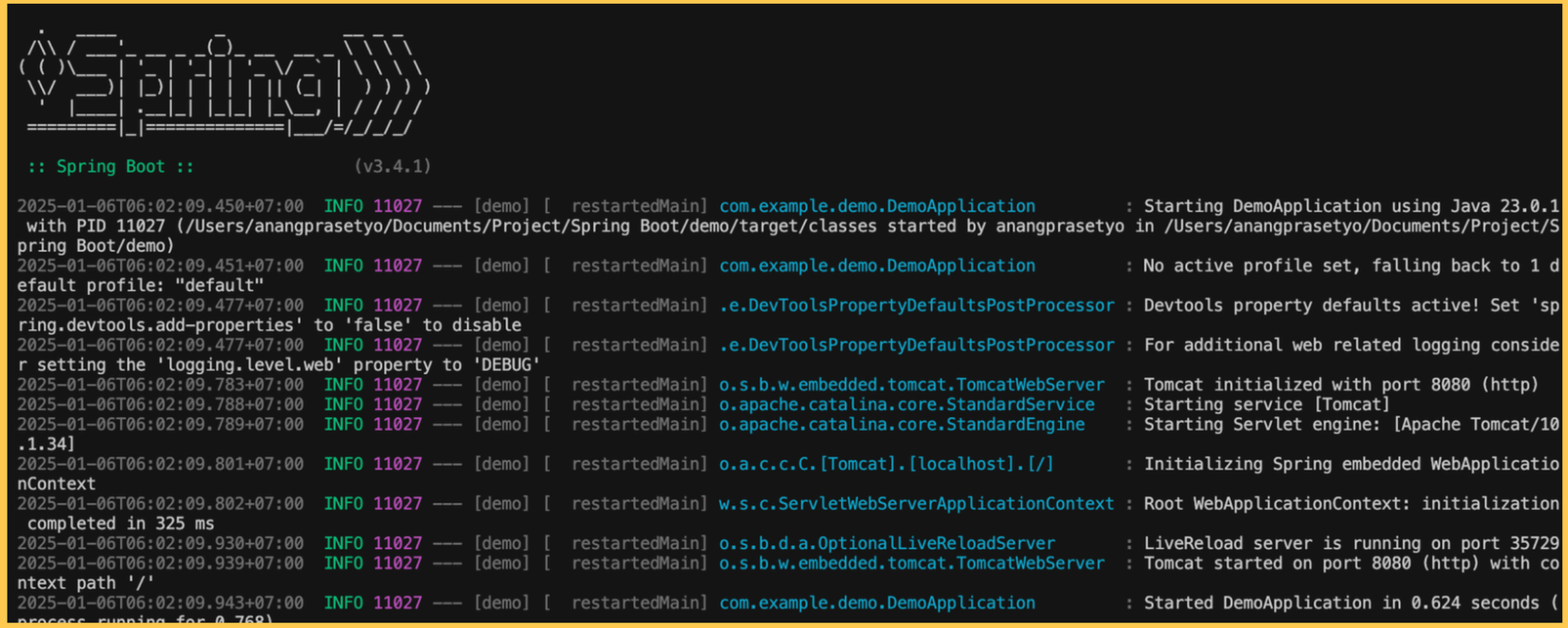
Creating a Simple RESTful API

Controller

```java
package com.example.demo.controller;

import com.example.demo.model.User;
import com.example.demo.service.UserService;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;

@RestController
@RequestMapping("/api/users")
public class UserController {

    private final UserService userService;

    public UserController(UserService userService) {
        this.userService = userService;
    }

    // GET: Retrieve a user by ID
    @GetMapping("/{id}")
    public ResponseEntity<User> getUserById(@PathVariable int id) {
        return userService.getUserById(id)
                .map(ResponseEntity::ok)
                .orElse(ResponseEntity.notFound().build());
    }
}

```

Running API and Testing

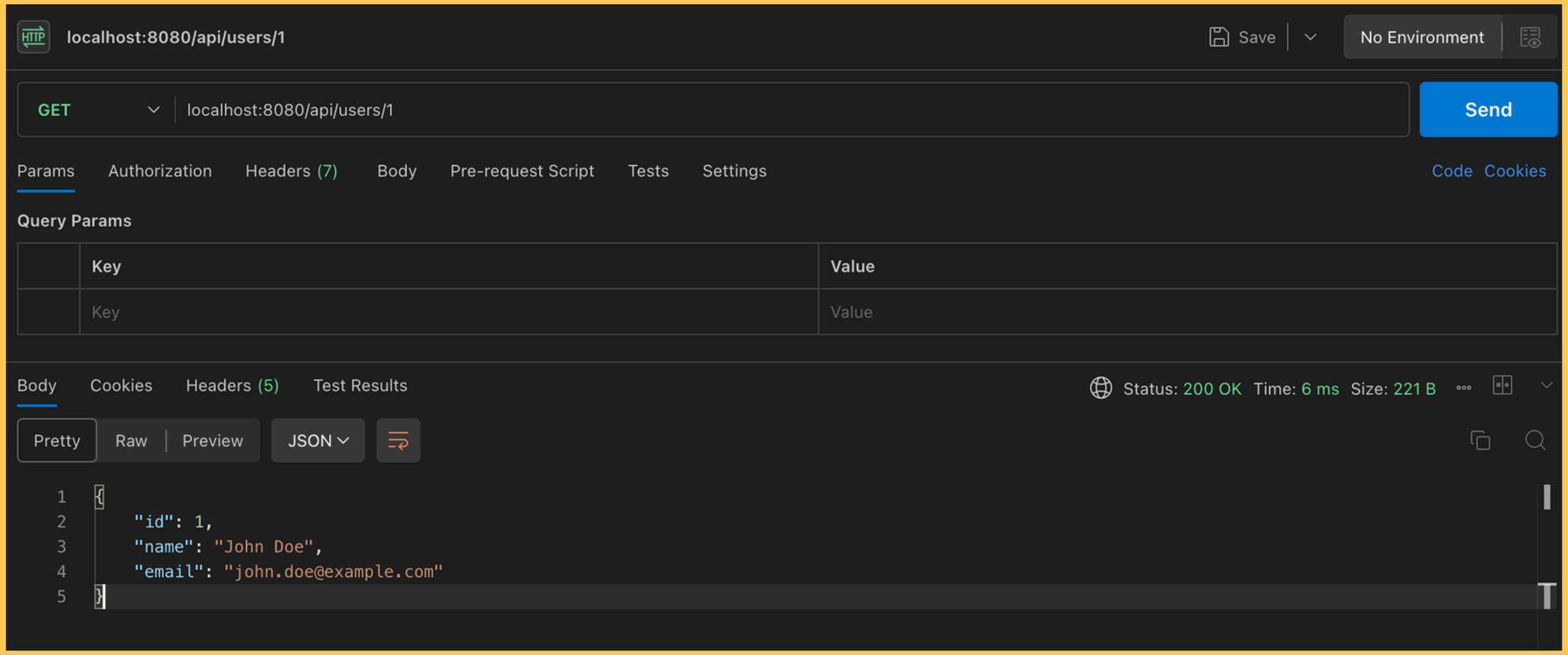
Started Application



```
  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/

 :: Spring Boot ::                (v3.4.1)

2025-01-06T06:02:09.450+07:00  INFO 11027 --- [demo] [  restartedMain] com.example.demo.DemoApplication         : Starting DemoApplication using Java 23.0.1
 with PID 11027 (/Users/anangprasetyo/Documents/Project/Spring Boot/demo/target/classes started by anangprasetyo in /Users/anangprasetyo/Documents/Project/S
pring Boot/demo)
2025-01-06T06:02:09.451+07:00  INFO 11027 --- [demo] [  restartedMain] com.example.demo.DemoApplication         : No active profile set, falling back to 1 d
efault profile: "default"
2025-01-06T06:02:09.477+07:00  INFO 11027 --- [demo] [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : Devtools property defaults active! Set 'sp
ring.devtools.add-properties' to 'false' to disable
2025-01-06T06:02:09.477+07:00  INFO 11027 --- [demo] [  restartedMain] .e.DevToolsPropertyDefaultsPostProcessor : For additional web related logging conside
r setting the 'logging.level.web' property to 'DEBUG'
2025-01-06T06:02:09.783+07:00  INFO 11027 --- [demo] [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port 8080 (http)
2025-01-06T06:02:09.788+07:00  INFO 11027 --- [demo] [  restartedMain] o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2025-01-06T06:02:09.789+07:00  INFO 11027 --- [demo] [  restartedMain] o.apache.catalina.core.StandardEngine    : Starting Servlet engine: [Apache Tomcat/10
.1.34]
2025-01-06T06:02:09.801+07:00  INFO 11027 --- [demo] [  restartedMain] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Spring embedded WebApplicatio
nContext
2025-01-06T06:02:09.802+07:00  INFO 11027 --- [demo] [  restartedMain] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization
 completed in 325 ms
2025-01-06T06:02:09.930+07:00  INFO 11027 --- [demo] [  restartedMain] o.s.b.d.a.OptionalLiveReloadServer       : LiveReload server is running on port 35729
2025-01-06T06:02:09.939+07:00  INFO 11027 --- [demo] [  restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port 8080 (http) with co
ntext path '/'
2025-01-06T06:02:09.943+07:00  INFO 11027 --- [demo] [  restartedMain] com.example.demo.DemoApplication         : Started DemoApplication in 0.624 seconds (
process running for 0.768)
```

Running API and Testing

Test Get User by Id



REFERENCES

- Spring Framework. (n.d.). Spring Framework. https://spring.io/projects/spring-framework
- Why Spring. (n.d.). Why Spring. https://spring.io/why-spring
- Annotated Controllers :: Spring Framework. (n.d.). https://docs.spring.io/spring-framework/reference/web/webmvc/mvc-controller.html

BINUS CENTER

THANK YOU

Thank you for following session 1 to the end.
See you at the next meeting