بله، حتماً! در اینجا یک راهنمای کامل برای ایجاد یک برنامه Spring Boot است که داده‌ها را از یک API خارجی (https://jsonplaceholder.typicode.com/users) می‌خواند، آنها را در پایگاه داده ذخیره می‌کند، و سپس این داده‌ها را نمایش می‌دهد.

مراحل ایجاد پروژه Spring Boot

1. ایجاد پروژه با Spring Initializr

به وب‌سایت Spring Initializr بروید و پروژه جدیدی با مشخصات زیر ایجاد کنید:

- **Project**: Maven Project
- **Language**: Java
- **Spring Boot**: (آخرین نسخه موجود)
- **Dependencies**:
  - Spring Web
  - Spring Data JPA
  - H2 Database (برای پایگاه داده در حافظه)
  - Spring Boot DevTools (اختیاری)

پس از تنظیمات، پروژه را دانلود کنید و آن را در IDE خود (مانند IntelliJ IDEA یا Eclipse) باز کنید.

2. ایجاد کلاس مدل

یک کلاس مدل برای کاربر ایجاد کنید. این کلاس شامل تمام فیلدهای مورد نیاز از API خواهد بود.

java

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

import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class UserEntity {
    @Id
    private int id;
    private String name;
    private String username;
    private String email;
    private String phone;
    private String website;

    //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 getUsername() {
        return username;
    }

    public void setUsername)String username) {
        this.username = username;
    }

    public String getEmail() {
        return email;
    }

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

    public String getPhone() {
        return phone;
    }

    public void setPhone)String phone) {
        this.phone = phone;
    }

    public String getWebsite() {
        return website;
    }

    public void setWebsite)String website) {
        this.website = website;
    }
}
```

3. ایجاد مخزن ((Repository

کلاسی برای دسترسی به داده‌ها در پایگاه داده ایجاد کنید.

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

import com.example.demo.model.UserEntity;
import org.springframework.data.jpa.repository.JpaRepository;

public interface UserRepository extends JpaRepository<UserEntity, Integer> {
}
```

4. ایجاد سرویس برای خواندن و ذخیره‌سازی داده‌ها

یک سرویس ایجاد کنید که از `RestTemplate` برای خواندن داده‌ها از API و ذخیره‌سازی آنها استفاده کند.

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

import com.example.demo.model.UserEntity;
import com.example.demo.repository.UserRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;

import java.util.List;

@Service
public class UserService {

    @Autowired
    private RestTemplate restTemplate;

    @Autowired
    private UserRepository userRepository;

    public void fetchAndSaveUsers() {
        String url = "https://jsonplaceholder.typicode.com/users";
        UserEntity[] users = restTemplate.getForObject(url, UserEntity[].class);
        for (UserEntity user : users) {
            userRepository.save(user);
        }
    }

    public List<UserEntity> getAllUsers() {
        return userRepository.findAll();
    }
}
```

5. ایجاد کنترلر برای نمایش داده‌ها

یک کنترلر بسازید که داده‌های ذخیره‌شده را به کاربر نمایش دهد.

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

import com.example.demo.model.UserEntity;
import com.example.demo.service.UserService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController
public class UserController {

    @Autowired
    private UserService userService;

    @GetMapping("/fetch-users")
    public String fetchAndSaveUsers() {
        userService.fetchAndSaveUsers();
        return "Users fetched and saved successfully!";
    }

    @GetMapping("/display-users")
    public List<UserEntity> displayUsers() {
        return userService.getAllUsers();
    }
}
```

6. پیکربندی RestTemplate

یک Bean برای `RestTemplate` ایجاد کنید تا بتوانید از آن در سرویس استفاده کنید.

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

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.client.RestTemplate;

@Configuration
public class AppConfig {

    @Bean
    public RestTemplate restTemplate() {
        return new RestTemplate();
```

```
    }
}
```

7. پیکربندی پایگاه داده H2

در فایل `application.properties`، تنظیمات پایگاه داده H2 را اضافه کنید:

```properties
spring.h2.console.enabled=true
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=password
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
```

8. اجرای برنامه

برنامه را اجرا کنید. می‌توانید از طریق مرورگر یا ابزارهای مانند Postman به آدرس‌های زیر بروید:

- برای خواندن و ذخیره‌سازی کاربران:
  - `http://localhost:8080/fetch-users`
- برای نمایش کاربران ذخیره‌شده:
  - `http://localhost:8080/display-users`

9. دسترسی به کنسول H2

برای دسترسی به کنسول H2، به آدرس زیر بروید:

- `http://localhost:8080/h2-console`

در اینجا می‌توانید اطلاعات بیشتری درباره داده‌های ذخیره‌شده در پایگاه داده ببینید.

نتیجه‌گیری

با انجام این مراحل، شما یک برنامه Spring Boot کامل ایجاد کرده‌اید که داده‌ها را از یک API خارجی می‌خواند، آنها را در پایگاه داده ذخیره می‌کند و سپس این داده‌ها را به کاربر نمایش می‌دهد. اگر سوال یا نیاز به کمک بیشتری دارید، لطفاً بفرمایید!