**Exercise 9: Employee Management System - Customizing Data Source Configuration**

**application.properties**

# Primary data source

spring.datasource.url=jdbc:h2:mem:primary

spring.datasource.username=sa

spring.datasource.password=

spring.datasource.driver-class-name=org.h2.Driver

# Secondary data source

secondary.datasource.url=jdbc:h2:mem:secondary

secondary.datasource.username=sa

secondary.datasource.password=

secondary.datasource.driver-class-name=org.h2.Driver

**PrimaryDataSourceConfig**

package com.employee.management.system.config;

import org.springframework.boot.autoconfigure.jdbc.DataSourceProperties;

import org.springframework.boot.context.properties.ConfigurationProperties;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Primary;

import org.springframework.stereotype.Component;

import javax.sql.DataSource;

@Component

public class PrimaryDataSourceConfig {

@Bean

@Primary

@ConfigurationProperties("spring.datasource")

public DataSourceProperties primaryDataSourceProperties() {

return new DataSourceProperties();

}

@Bean

@Primary

public DataSource primaryDataSource() {

return primaryDataSourceProperties().initializeDataSourceBuilder().build();

}

}

**SecondaryDataSourceConfig**

package com.employee.management.system.config;

import org.springframework.boot.autoconfigure.jdbc.DataSourceProperties;

import org.springframework.boot.context.properties.ConfigurationProperties;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.stereotype.Component;

import javax.sql.DataSource;

@Component

public class SecondaryDataSourceConfig {

@Bean

@ConfigurationProperties("secondary.datasource")

public DataSourceProperties secondaryDataSourceProperties() {

return new DataSourceProperties();

}

@Bean

public DataSource secondaryDataSource() {

return secondaryDataSourceProperties().initializeDataSourceBuilder().build();

}

}

In the above code, I have customized the data source configuration by leveraging Spring Boot auto-configuration and externalizing the configuration with application.properties.

I have defined two data sources: primary and secondary, and managed them within the application using separate configuration classes: PrimaryDataSourceConfig and SecondaryDataSourceConfig.

Note that I have used @Primary annotation to specify the primary data source, and @ConfigurationProperties to bind the properties to the data source configurations.