**📘 Scenario**

A software platform needs to deliver notifications to users via different channels. Each channel (Email, SMS, Push) must adhere to a common contract (Notifier), but has its own way of formatting and sending messages.

**✅ Requirements**

**1. Define Interface: Notifier**

* **Method Signature**
  + void sendNotification(String recipient, String message)
* **Purpose**
  + Any class implementing Notifier must provide its own logic for sending a message (e.g., connect to SMTP for email, invoke an SMS gateway for SMS, or call a push‐service for push).

**2. Implementing Class: EmailNotifier**

* **Implements:** Notifier
* **Additional Fields (if needed)**
  + String smtpServer
  + String fromAddress
* **Constructor**
  + Accepts smtpServer and fromAddress (and any other required setup).
* **Implementation of sendNotification(String recipient, String message)**
  + Simulate formatting an email (e.g., subject line “Notification”, body = message).
  + Simulate connecting to the SMTP server (smtpServer).
  + Print a line such as:

Sending Email to [recipient]:

From: [fromAddress]

SMTP Server: [smtpServer]

Message: [message]

**3. Implementing Class: SMSNotifier**

* **Implements:** Notifier
* **Additional Fields (if needed)**
  + String providerName
  + String apiKey
* **Constructor**
  + Accepts providerName and apiKey.
* **Implementation of sendNotification(String recipient, String message)**
  + Simulate formatting an SMS payload.
  + Simulate connecting to the SMS gateway (providerName/apiKey).
  + Print a line such as:

less

Sending SMS to [recipient]:

Provider: [providerName]

API Key: [apiKey]

Message: [message]

**4. Implementing Class: PushNotifier**

* **Implements:** Notifier
* **Additional Fields (if needed)**
  + String appId
  + String serverKey
* **Constructor**
  + Accepts appId and serverKey.
* **Implementation of sendNotification(String recipient, String message)**
  + Simulate formatting a push payload (e.g., title = “Alert”, body = message).
  + Simulate connecting to the push notification service (appId/serverKey).
  + Print a line such as:

less

CopyEdit

Sending Push Notification to [recipient]:

App ID: [appId]

Server Key: [serverKey]

Message: [message]

**5. Utility Class: NotificationService**

* **Methods (static or instance, as you prefer)**
  1. void sendEmail(String recipient, String message)
     + Instantiates or receives an EmailNotifier (with its SMTP settings) and calls its sendNotification(recipient, message).
  2. void sendSMS(String recipient, String message)
     + Instantiates or receives an SMSNotifier (with its provider settings) and calls its sendNotification(recipient, message).
  3. void sendPush(String recipient, String message)
     + Instantiates or receives a PushNotifier (with its app settings) and calls its sendNotification(recipient, message).
* **Purpose**
  1. Encapsulate the logic of choosing which notifier to invoke without using any collection. Each method is responsible for exactly one channel.

**6. Test Class: NotificationTest**

* **main(String[] args) Steps:**
  1. Instantiate an EmailNotifier with realistic SMTP details.
  2. Instantiate an SMSNotifier with realistic provider details.
  3. Instantiate a PushNotifier with realistic push service details.
  4. Call **one by one**: