LAPORAN PRAKTIKUM PEMROGRAMAN BERORIENTASI OBJEK (PBO)

PRAKTIKUM 12

2411102441249

Hervino Islami Fasha

FAKULTAS SAINS DAN TEKNOLOGI

PROGRAM STUDI S1 TEKNIK INFORMATIKA

UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR

EMAIL : 2411102441249@umkt.ac.id

Link Github: https://github.com/Rainzy21/Praktikum_PBO/tree/main/P12

a. Screenshoot kode latihan

```python
from abc import ABC, abstractmethod
from dataclasses import dataclass
import logging

# Konfigurasi dasar: Semua Log Level INFO ke atas akan ditampilkan
# Format: Waktu - Level - Nama Kelas/Fungsi - Pesan
logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(name)s - %(message)s'
)
# Tambahkan logger untuk kelas yang akan kita gunakan
LOGGER = logging.getLogger('Checkout')

@dataclass
class Order:
    customer_name: str
    total_price: float
    status: str = "open"

class IPaymentProcessor(ABC):
    """Kontrak: Semua prosesor pembayaran hrus punya method 'process'."""
    @abstractmethod
    def process(self, order: Order) -> bool:
        pass

class INotificationService(ABC):
    """Kontrak: Semua layanan notifikasi harus punya method 'send'."""
    @abstractmethod
    def send(self, order: Order):
        pass

class CreditCardProcessor(IPaymentProcessor):
    def process(self, order: Order) -> bool:
        print("Payment: Memproses Kartu Kredit.")
        return True

class EmailNotifier(INotificationService):
    def send(self, order: Order):
        print(f"Notif: Mengirim email konfirmasi ke {order.customer_name}.")

class CheckoutService: # Tanggung jawab tunggal: Mengkoordinasikan Checkout
    """
    Kelas high-level untuk mengkoordinasi proses transaksi pembayaran.

    Kelas ini memisahkan logika pembayaran dan notifikasi (memenuhi SRP).
    """
    def __init__(self, payment_processor: IPaymentProcessor, notifier: INotificationService):
        """
        Menginisialisasi CheckOutService dengan depedensi yang diperlukan.

         Args:
            payment_processort (IPaymentProcessor): Implementasi interface pembayaran.
            notifier (INotificationService): Implementsi interface notifikasi.
         """
        self.payment_processor = payment_processor
        self.notifier = notifier

    def run_checkout(self, order: Order):
        """
        Menjalankan proses checkout dan memvalidasi pembayaran.

        Args:
            order (Order): Objek pesanan yang akan diproses.

        Returns:
            bool: True jika checkout sukses, False jika gagal.
        """
        # Logging alih-alih print()
        LOGGER.info(f"Memulai checkout untuk {order.customer_name}. Total: {order.total_price}")
        payment_success = self.payment_processor.process(order) # Delegasi 1

        if payment_success:
            order.status = "paid"
            self.notifier.send(order) # Delegasi 2
            LOGGER.info("Checkout Sukses. Status pesanan: PAID.")
            return True
        else:
            # Gunakan level ERROR/WARNING untuk masalah
            LOGGER.error("Pembayaran gagal. Transaksi dibatalkan.")
            return False

        pass

andi_order = Order("Andi", 500000)
email_service = EmailNotifier()

cc_processor = CreditCardProcessor()
checkout_cc = CheckoutService(payment_processor=cc_processor, notifier=email_service)
print("--- Skenario 1: Credit Card ---")
checkout_cc.run_checkout(andi_order)

class QrisProcessor(IPaymentProcessor):
    def process(self, order: Order) -> bool:
        print("Payment: Memproses Qris.")
        return True

budi_order = Order("Budi", 100000)
qris_processor = QrisProcessor()

checkout_qris = CheckoutService(payment_processor=qris_processor, notifier=email_service)
print("\n--- Skenario 2: Pembuktian OCP (QRIS) ---")
checkout_qris.run_checkout(budi_order)
```

Output

```
C:\Users\Lenovo\Documents\GitHub\Praktikum_PBO>C:/Python313/python.exe c:/Users/Lenovo/Documents/GitHub/Praktikum_PBO/P12/refactor_solid.py
--- Skenario 1: Credit Card ---
2025-12-19 00:33:32,769 - INFO - Checkout - Memulai checkout untuk Andi. Total: 500000
Payment: Memproses Kartu Kredit.
Notif: Mengirim email konfirmasi ke Andi.
2025-12-19 00:33:32,769 - INFO - Checkout - Checkout Sukses. Status pesanan: PAID.

--- Skenario 2: Pembuktian OCP (QRIS) ---
2025-12-19 00:33:32,770 - INFO - Checkout - Memulai checkout untuk Budi. Total: 100000
Payment: Memproses Qris.
Notif: Mengirim email konfirmasi ke Budi.
2025-12-19 00:33:32,770 - INFO - Checkout - Checkout Sukses. Status pesanan: PAID.
```

2411102441249 – Hervino Islami Fasha

Screenshoot kode latihan mandiri

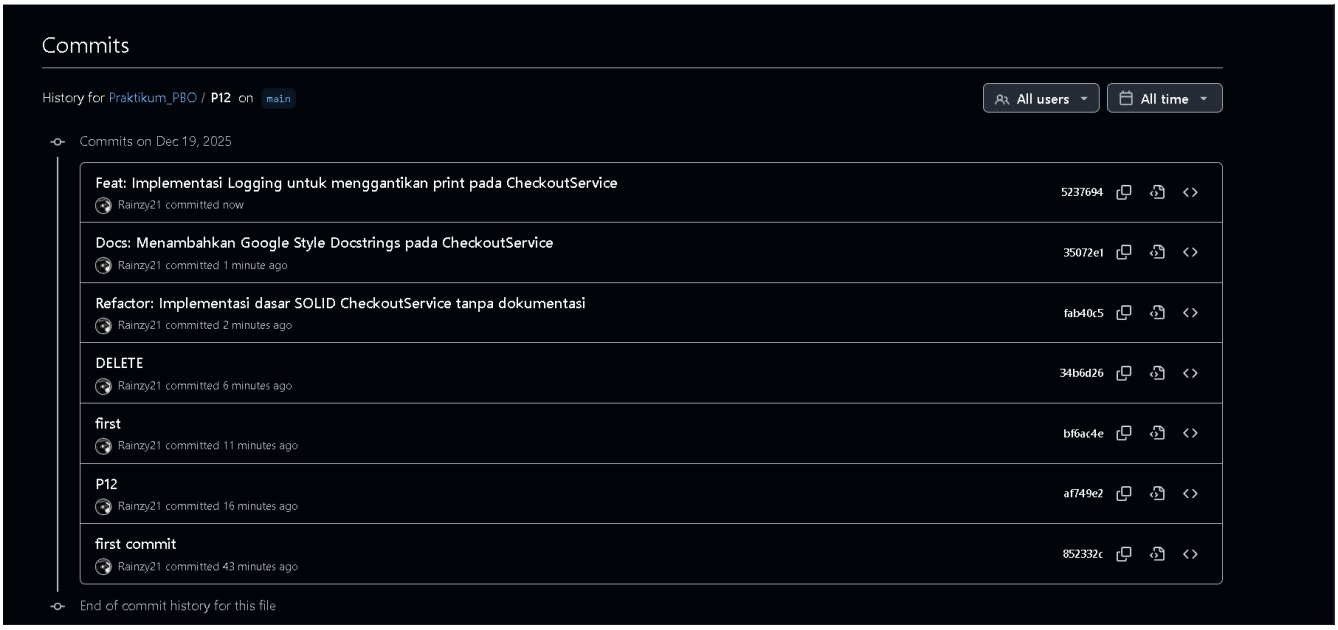
```python
import logging
from abc import ABC, abstractmethod
from dataclasses import dataclass
from typing import List

logging.basicConfig(
    level=logging.INFO,
    format="%(levelname)s | %(asctime)s | %(name)s | %(message)s",
    datefmt="%Y-%m-%d %H:%M:%S",
)
logger = logging.getLogger("checkout")

@dataclass
class Order:
    customer_name: str
    total_price: float
    status: str = "open"

class IPaymentProcessor(ABC):
    @property
    @abstractmethod
    def payment_name(self) -> str:
        pass

    @abstractmethod
    def process(self, order: Order) -> bool:
        pass

class INotificationService(ABC):
    @property
    @abstractmethod
    def service_name(self) -> str:
        pass

    @abstractmethod
    def send(self, order: Order) -> bool:
        pass

class CreditCardProcessor(IPaymentProcessor):
    @property
    def payment_name(self) -> str:
        return "Credit Card"

    def process(self, order: Order) -> bool:
        logger.info("Memproses kartu kredit (gateway, CVV, verifikasi)")
        return True

class BankTransferProcessor(IPaymentProcessor):
    @property
    def payment_name(self) -> str:
        return "Bank Transfer"

    def process(self, order:  Order) -> bool:
        logger.info("Memproses Transfer Bank (cek rekening, konfirmasi)")
        return True

class EWalletProcessor(IPaymentProcessor):
    @property
    def payment_name(self) -> str:
        return "E-Wallet"

    def process(self, order: Order) -> bool:
        logger.info("Memproses E-Wallet (request ke provider)")
        return True

class EmailNotifier(INotificationService):
    @property
    def service_name(self) -> str:
        return "Email"

    def send(self, order: Order) -> bool:
        logger.info("Mengirim email konfirmasi ke %s", order.customer_name)
        return True

class SmsNotifier(INotificationService):
    @property
    def service_name(self) -> str:
        return "SMS"

    def send(self, order:  Order) -> bool:
        logger.info("Mengirim SMS ke %s", order.customer_name)
        return True

class CheckoutService:
    def __init__(self, payment_processor: IPaymentProcessor,
                 notifiers: List[INotificationService] = None):
        self._payment_processor = payment_processor
        self._notifiers = notifiers or []

    def run_checkout(self, order: Order) -> bool:
        logger.info("PROSES CHECKOUT: %s | Total: Rp %s | Metode: %s",
                    order.customer_name, f"{order.total_price:,.0f}", self._payment_processor.payment_name)

        logger.info("Memproses Pembayaran (%s)", self._payment_processor.payment_name)
        payment_success = self._payment_processor.process(order)

        if not payment_success:
            logger.warning("CHECKOUT GAGAL - Pembayaran ditolak untuk %s", order.customer_name)
            return False

        order.status = "paid"

        if self._notifiers:
            logger.info("Mengirim Notifikasi (%d notifier)", len(self._notifiers))
            for notifier in self._notifiers:
                ok = notifier.send(order)
                if not ok:
                    logger.warning("Notifikasi %s gagal untuk %s", notifier.service_name, order.customer_name)

        logger.info("CHECKOUT BERHASIL! Status Order: %s", order.status)
        return True

if __name__ == "__main__":
    logger.info("SISTEM CHECKOUT - SETELAH REFACTORING (SRP, OCP, DIP + Logging)")
    email_notifier = EmailNotifier()
    sms_notifier = SmsNotifier()

    order1 = Order("Andi Pratama", 500000)
    checkout_cc = CheckoutService(CreditCardProcessor(), [email_notifier])
    checkout_cc.run_checkout(order1)

    order2 = Order("Budi Santoso", 750000)
    checkout_bank = CheckoutService(BankTransferProcessor(), [email_notifier, sms_notifier])
    checkout_bank.run_checkout(order2)

    order3 = Order("Citra Dewi", 250000)
    checkout_ewallet = CheckoutService(EWalletProcessor(), [sms_notifier])
    checkout_ewallet.run_checkout(order3)

    class QrisProcessor(IPaymentProcessor):
        @property
        def payment_name(self) -> str:
            return "QRIS"
        def process(self, order:  Order) -> bool:
            logger.info("Memproses QRIS (QR Code, scan, verifikasi)")
            return True

    order4 = Order("Dani Wijaya", 100000)
    checkout_qris = CheckoutService(QrisProcessor(), [email_notifier, sms_notifier])
    checkout_qris.run_checkout(order4)

    class WhatsAppNotifier(INotificationService):
        @property
        def service_name(self) -> str:
            return "WhatsApp"
        def send(self, order: Order) -> bool:
            logger.info("Mengirim WhatsApp ke %s", order.customer_name)
            return True

    order5 = Order("Eva Marlina", 350000)
    wa_notifier = WhatsAppNotifier()
    checkout_full = CheckoutService(CreditCardProcessor(), [email_notifier, sms_notifier, wa_notifier])
    checkout_full.run_checkout(order5)
```

EMAIL : 2411102441249@umkt.ac.id

Output

```
C:\Users\Lenovo\Documents\GitHub\Praktikum_PBO>C:/Python313/python.exe c:/Users/Lenovo/Documents/GitHub/Praktikum_PBO/P12/main.py
INFO | 2025-12-19 00:37:14 | checkout | SISTEM CHECKOUT - SETELAH REFACTORING (SRP, OCP, DIP + Logging)
INFO | 2025-12-19 00:37:14 | checkout | PROSES CHECKOUT: Andi Pratama | Total: Rp 500,000 | Metode: Credit Card
INFO | 2025-12-19 00:37:14 | checkout | Memproses Pembayaran (Credit Card)
INFO | 2025-12-19 00:37:14 | checkout | Memproses Kartu Kredit (gateway, CVV, verifikasi)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim Notifikasi (1 notifier)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim email konfirmasi ke Andi Pratama
INFO | 2025-12-19 00:37:14 | checkout | CHECKOUT BERHASIL! Status Order: paid
INFO | 2025-12-19 00:37:14 | checkout | PROSES CHECKOUT: Budi Santoso | Total: Rp 750,000 | Metode: Bank Transfer
INFO | 2025-12-19 00:37:14 | checkout | Memproses Pembayaran (Bank Transfer)
INFO | 2025-12-19 00:37:14 | checkout | Memproses Transfer Bank (cek rekening, konfirmasi)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim Notifikasi (2 notifier)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim email konfirmasi ke Budi Santoso
INFO | 2025-12-19 00:37:14 | checkout | Mengirim SMS ke Budi Santoso
INFO | 2025-12-19 00:37:14 | checkout | CHECKOUT BERHASIL! Status Order: paid
INFO | 2025-12-19 00:37:14 | checkout | PROSES CHECKOUT: Citra Dewi | Total: Rp 250,000 | Metode: E-Wallet
INFO | 2025-12-19 00:37:14 | checkout | Memproses Pembayaran (E-Wallet)
INFO | 2025-12-19 00:37:14 | checkout | Memproses E-Wallet (request ke provider)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim Notifikasi (1 notifier)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim SMS ke Citra Dewi
INFO | 2025-12-19 00:37:14 | checkout | CHECKOUT BERHASIL! Status Order: paid
INFO | 2025-12-19 00:37:14 | checkout | PROSES CHECKOUT: Dani Wijaya | Total: Rp 100,000 | Metode: QRIS
INFO | 2025-12-19 00:37:14 | checkout | Memproses Pembayaran (QRIS)
INFO | 2025-12-19 00:37:14 | checkout | Memproses QRIS (QR Code, scan, verifikasi)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim Notifikasi (2 notifier)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim email konfirmasi ke Dani Wijaya
INFO | 2025-12-19 00:37:14 | checkout | Mengirim SMS ke Dani Wijaya
INFO | 2025-12-19 00:37:14 | checkout | CHECKOUT BERHASIL! Status Order: paid
INFO | 2025-12-19 00:37:14 | checkout | PROSES CHECKOUT: Eva Marlina | Total: Rp 350,000 | Metode: Credit Card
INFO | 2025-12-19 00:37:14 | checkout | Memproses Pembayaran (Credit Card)
INFO | 2025-12-19 00:37:14 | checkout | Memproses Kartu Kredit (gateway, CVV, verifikasi)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim Notifikasi (3 notifier)
INFO | 2025-12-19 00:37:14 | checkout | Mengirim email konfirmasi ke Eva Marlina
INFO | 2025-12-19 00:37:14 | checkout | Mengirim SMS ke Eva Marlina
INFO | 2025-12-19 00:37:14 | checkout | Mengirim WhatsApp ke Eva Marlina
INFO | 2025-12-19 00:37:14 | checkout | CHECKOUT BERHASIL! Status Order: paid
```

Screenshoot history commit



f. Refleksi singkat:

- Docstring memperjelas kontrak dan maksud setiap class/method sehingga memudahkan kolaborasi dan onboarding.
- Logging menggantikan print untuk observabilitas; level INFO/WARNING membantu memisahkan informasi normal dan anomali saat debugging.
- Kombinasi Docstring + Logging meningkatkan traceability serta mempercepat identifikasi masalah di pipeline dan saat integrasi tim.